LilyPond

Das Notensatzprogramm

Notationsreferenz

Das LilyPond-Entwicklerteam

Dieses Handbuch stellt eine Referenz aller Notationsformen zur Verfügung, die mit LilyPond Version 2.27.4 erstellt werden können. Es wird vorausgesetzt, dass der Leser mit dem Abschnitt "Handbuch zum Lernen" in *Handbuch zum Lernen* vertraut ist.

Zu mehr Information, wie dieses Handbuch unter den anderen Handbüchern positioniert, oder um dieses Handbuch in einem anderen Format zu lesen, besuchen Sie bitte Abschnitt "Handbücher" in *Allgemeine Information*.

Wenn Ihnen Handbücher fehlen, finden Sie die gesamte Dokumentation unter `https://lilypond.org/`.

Copyright © 1999–2026 bei den Autoren.

The translation of the following copyright notice is provided for courtesy to non-English speakers, but only the notice in English legally counts.

Die Übersetzung der folgenden Lizenzanmerkung ist zur Orientierung für Leser, die nicht Englisch sprechen. Im rechtlichen Sinne ist aber nur die englische Version gültig.

> Es ist erlaubt, dieses Dokument unter den Bedingungen der GNU Free Documentation Lizenz (Version 1.1 oder spätere, von der Free Software Foundation publizierte Versionen, ohne invariante Abschnitte), zu kopieren, zu verbreiten und/oder zu verändern. Eine Kopie der Lizenz ist im Abschnitt "GNU Free Documentation License" angefügt.
>
> Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with no Invariant Sections. A copy of the license is included in the section entitled "GNU Free Documentation License".

Für LilyPond Version 2.27.4

Inhaltsverzeichnis

Allgemeine Notation

1 Tonhöhen ... 3

1.1 Tonhöhen setzen ... 3
1.1.1 Absolute Oktavenbezeichnung ... 3
1.1.2 Relative Oktavenbezeichnung ... 4
1.1.3 Versetzungszeichen ... 7
1.1.4 Notenbezeichnungen in anderen Sprachen ... 9
1.2 Viele Tonhöhen gleichzeitig verändern ... 11
1.2.1 Oktavenüberprüfung ... 11
1.2.2 Transponieren ... 12
1.2.3 Umkehrung ... 15
1.2.4 Krebs ... 16
1.2.5 Modale Transformierungen ... 16
1.3 Tonhöhen anzeigen lassen ... 19
1.3.1 Notenschlüssel ... 19
1.3.2 Tonartbezeichnung ... 22
1.3.3 Oktavierungsklammern ... 24
1.3.4 Transposition von Instrumenten ... 25
1.3.5 Automatische Versetzungszeichen ... 26
1.3.6 Tonumfang ... 33
1.4 Notenköpfe ... 36
1.4.1 Besondere Notenköpfe ... 36
1.4.2 Easy-Notation-Notenköpfe ... 38
1.4.3 Notenköpfe mit besonderen Formen ... 39
1.4.4 Improvisation ... 42

2 Rhythmus ... 44

2.1 Rhythmen eingeben ... 44
2.1.1 Tondauern ... 44
2.1.2 Andere rhythmische Aufteilungen ... 47
2.1.3 Tondauern skalieren ... 52
2.1.4 Bindebögen ... 53
2.2 Pausen eingeben ... 57
2.2.1 Pausen ... 57
2.2.2 Unsichtbare Pausen ... 59
2.2.3 Ganztaktpausen ... 60
2.3 Rhythmen anzeigen lassen ... 65
2.3.1 Taktangabe ... 65
2.3.2 Metronomangabe ... 69
2.3.3 Auftakte ... 72
2.3.4 Musik ohne Metrum ... 73
2.3.5 Polymetrische Notation ... 75
2.3.6 Automatische Aufteilung von Noten ... 78
2.3.7 Melodierhythmus anzeigen ... 79
2.4 Balken ... 81
2.4.1 Automatische Balken ... 81
2.4.2 Einstellung von automatischen Balken ... 84

2.4.3 Manuelle Balken ... 94
2.4.4 Gespreizte Balken ... 96
2.5 Takte ... 97
2.5.1 Taktstriche ... 97
2.5.2 Taktzahlen ... 103
2.5.3 Takt- und Taktzahlüberprüfung ... 109
2.5.4 Übungszeichen ... 110
2.6 Besondere rhythmische Fragen ... 112
2.6.1 Verzierungen ... 112
2.6.2 An Kadenzen ausrichten ... 118
2.6.3 Verwaltung der Zeiteinheiten ... 118

3 Ausdrucksbezeichnungen ... 120
3.1 Ausdrucksbezeichnungen an Noten angehängt ... 120
3.1.1 Artikulationszeichen und Verzierungen ... 120
3.1.2 Dynamik ... 123
3.1.3 Neue Lautstärkezeichen ... 128
3.2 Ausdrucksbezeichnungen als Bögen ... 130
3.2.1 Legatobögen ... 130
3.2.2 Phrasierungsbögen ... 133
3.2.3 Atemzeichen ... 134
3.2.4 Glissando zu unbestimmter Tonhöhe ... 136
3.3 Ausdrucksbezeichnungen als Linien ... 136
3.3.1 Glissando ... 136
3.3.2 Arpeggio ... 138
3.3.3 Triller ... 141

4 Wiederholungszeichen ... 144
4.1 Lange Wiederholungen ... 144
4.1.1 Normale Wiederholungen ... 144
4.1.2 Manuelle Wiederholungszeichen ... 152
4.1.3 Ausgeschriebene Wiederholungen ... 154
4.2 Kurze Wiederholungen ... 156
4.2.1 Prozent-Wiederholungen ... 156
4.2.2 Tremolo-Wiederholung ... 158

5 Gleichzeitig erscheinende Noten ... 160
5.1 Eine einzelne Stimme ... 160
5.1.1 Noten mit Akkorden ... 160
5.1.2 Akkord-Wiederholungen ... 162
5.1.3 Gleichzeitige Ausdrücke ... 164
5.1.4 Cluster ... 165
5.2 Mehrere Stimmen ... 166
5.2.1 Mehrstimmigkeit in einem System ... 166
5.2.2 Stimmenstile ... 169
5.2.3 Auflösung von Zusammenstößen ... 170
5.2.4 Automatische Kombination von Stimmen ... 175
5.2.5 Musik parallel notieren ... 179

6 Notation auf Systemen ... 183

6.1 Systeme anzeigen lassen ... 183
6.1.1 Neue Notensysteme erstellen ... 183
6.1.2 Systeme gruppieren ... 185
6.1.3 Verschachtelte Notensysteme ... 188
6.1.4 Systeme trennen ... 190
6.2 Einzelne Systeme verändern ... 191
6.2.1 Das Notensystem ... 191
6.2.2 Ossia-Systeme ... 194
6.2.3 Systeme verstecken ... 198
6.3 Orchesterstimmen erstellen ... 201
6.3.1 Instrumentenbezeichnungen ... 201
6.3.2 Andere Stimmen zitieren ... 205
6.3.3 Stichnoten formatieren ... 208

7 Anmerkungen ... 214

7.1 Innerhalb des Systems ... 214
7.1.1 Auswahl der Notations-Schriftgröße ... 214
7.1.2 Fingersatzanweisungen ... 215
7.1.3 Unsichtbare Noten ... 217
7.1.4 Farbige Objekte ... 218
7.1.5 Klammern ... 219
7.1.6 Hälse ... 220
7.2 Außerhalb des Notensystems ... 221
7.2.1 Erklärungen in Ballonform ... 221
7.2.2 Gitternetzlinien ... 222
7.2.3 Analyseklammern ... 224

8 Text ... 226

8.1 Text eingeben ... 226
8.1.1 Textarten ... 226
8.1.2 Text mit Verbindungslinien ... 228
8.1.3 Textartige Zeichen ... 229
8.1.4 Separater Text ... 231
8.2 Text formatieren ... 233
8.2.1 Textbeschriftung (Einleitung) ... 233
8.2.2 Überblick über die wichtigsten Textbeschriftungsbefehle ... 234
8.2.3 Textausrichtung ... 237
8.2.4 Graphische Notation innerhalb einer Textbeschriftung ... 241
8.2.5 Musikalische Notation innerhalb einer Textbeschriftung ... 243
8.2.6 Textbeschriftung über mehrere Seiten ... 246
8.3 Schriftarten ... 247
8.3.1 Was sind Schriftarten ... 247
8.3.2 Schriftarten für einen Eintrag ... 249
8.3.3 Schriftart des gesamten Dokuments ... 249

Notation für spezielle Zwecke

9 Notation von Gesang 253

9.1 Übliche Notation für Vokalmusik 253
9.1.1 Referenz für Vokalmusik 253
9.1.2 Eingabe von Text 254
9.1.3 Text an einer Melodie ausrichten 255
9.1.4 Automatische Silbendauern 257
9.1.5 Manuelle Silbendauern 260
9.1.6 Mehrere Silben zu einer Note 261
9.1.7 Mehrere Noten zu einer Silbe 262
9.1.8 Fülllinien und Trennstriche 265
9.2 Techniken für die Gesangstextnotation 266
9.2.1 Mit Gesangstexten und Bezeichnern arbeiten 266
9.2.2 Gesangstext vertikal verschieben 267
9.2.3 Silben horizontal verschieben 270
9.2.4 Gesangstext und Wiederholungen 271
9.2.5 Getrennte Texte 280
9.3 Strophen 281
9.3.1 Strophennummern hinzufügen 281
9.3.2 Lautstärkebezeichnung zu Strophen hinzufügen 282
9.3.3 Sängernamen zu Strophen hinzufügen 282
9.3.4 Strophen mit unterschiedlichem Rhythmus 282
9.3.5 Die Strophen am Ende ausdrucken 285
9.3.6 Die Strophen am Ende in mehreren Spalten drucken 286
9.4 Lieder 288
9.4.1 Verweise für Lieder 288
9.4.2 Liedblätter 289
9.5 Chormusik 289
9.5.1 Verweise für Chormusik 289
9.5.2 Partiturbeispiele für Chormusik 290
9.5.3 Geteilte Stimmen 291
9.6 Oper und Musical 292
9.6.1 Verweise für Oper und Musical 293
9.6.2 Namen von Figuren 293
9.6.3 Musikalische Stichnoten 295
9.6.4 Gesprochene Musik 299
9.6.5 Dialog zur Musik 299
9.7 Psalmengesänge und Hymnen 301
9.7.1 Verweise für Psalmen und Hymnen 301
9.7.2 Kirchengesang notieren 301
9.7.3 Einen Psalm notieren 307
9.7.4 Unvollständige Takte in Hymnen 310
9.8 Alte Vokalmusik 313

10 Tasteninstrumente und andere Instrumente mit mehreren Systemen 314

10.1 Übliche Notation für Tasteninstrumente 314
10.1.1 Referenz für Tasteninstrumente 315
10.1.2 Notensysteme manuell verändern 315

10.1.3 Automatischer Systemwechsel ... 317
10.1.4 Stimmführungslinien ... 319
10.1.5 Hälse über beide Systeme ... 319
10.2 Klavier ... 321
10.2.1 Klavierpedal ... 321
10.3 Akkordeon ... 322
10.3.1 Diskant-Symbole ... 322
10.4 Harfe ... 322
10.4.1 Referenzen für Harfe ... 322
10.4.2 Harfenpedal ... 323

11 Bundlose Saiteninstrumente ... 324
11.1 Übliche Notation für bundlose Saiteninstrumente ... 324
11.1.1 Hinweise für bundlose Saiteninstrumente ... 324
11.1.2 Bezeichnung des Bogens ... 325
11.1.3 Flageolett ... 325
11.1.4 Bartók-Pizzicato ... 326

12 Saiteninstrumente mit Bünden ... 327
12.1 Übliche Notation für Saiteninstrumente mit Bünden ... 327
12.1.1 Referenz für Saiteninstrumente mit Bünden ... 327
12.1.2 Seitennummerbezeichnung ... 328
12.1.3 Standardtabulaturen ... 330
12.1.4 Angepasste Tabulaturen ... 343
12.1.5 Bund-Diagramm-Beschriftung ... 346
12.1.6 Vordefinierte Bund-Diagramme ... 355
12.1.7 Automatische Bund-Diagramme ... 365
12.1.8 Fingersatz der rechten Hand ... 368
12.2 Gitarre ... 370
12.2.1 Position und Barré anzeigen ... 370
12.2.2 Flageolett und gedämpfte Noten ... 370
12.2.3 Powerakkorde anzeigen ... 372
12.3 Banjo ... 373
12.3.1 Banjo-Tabulaturen ... 373

13 Schlagzeug ... 375
13.1 Übliche Notation für Schlagzeug ... 375
13.1.1 Referenz für Schlagzeug ... 375
13.1.2 Grundlagen der Schlagzeugnotation ... 375
13.1.3 Trommelwirbel ... 376
13.1.4 Schlagzeug mit Tonhöhe ... 377
13.1.5 Schlagzeugsysteme ... 377
13.1.6 Eigene Schlagzeugsysteme ... 379
13.1.7 Geisternoten ... 383

14 Blasinstrumente ... 384
14.1 Übliche Notation für Bläser ... 384
14.1.1 Referenz für Blasinstrumente ... 384
14.1.2 Fingersatz ... 385
14.2 Dudelsack ... 387

14.2.1 Dudelsack-Definitionen ... 387
14.2.2 Dudelsack-Beispiele ... 388
14.3 Holzbläser ... 389
14.3.1 Holzbläserdiagramme ... 389

15 Notation von Akkorden ... 397
15.1 Akkord-Modus ... 397
15.1.1 Überblick über den Akkord-Modus ... 397
15.1.2 Übliche Akkorde ... 398
15.1.3 Erweiterte und modifizierte Akkorde ... 400
15.2 Akkorde anzeigen ... 402
15.2.1 Akkordbezeichnungen drucken ... 403
15.2.2 Akkordbezeichnungen anpassen ... 405
15.3 Generalbass ... 411
15.3.1 Grundlagen des Bezifferten Basses ... 412
15.3.2 Eingabe des Generalbass' ... 413
15.3.3 Generalbass anzeigen ... 415

16 Zeitgenössische Musik ... 418
16.1 Tonhöhe und Harmonie in zeitgenössischer Musik ... 418
16.1.1 Verweise zu Tonhöhe und Harmonie in zeitgenössischer Musik ... 418
16.1.2 Mikrotonale Notation ... 418
16.1.3 Zeitgenössische Tonartvorzeichnung und Harmonie ... 418
16.2 Zeitgenössische Notation von Rhythmen ... 418
16.2.1 Verweise für zeitgenössische Benutzung von Rhythmus ... 418
16.2.2 N-tolen in zeitgenössischer Musik ... 418
16.2.3 Zeitgenössische Taktarten ... 418
16.2.4 Erweiterte polymetrische Notation ... 418
16.2.5 Balken in zeitgenössischer Musik ... 418
16.2.6 Taktstriche in zeitgenössischer Musik ... 418
16.3 Graphische Notation ... 418
16.4 Zeitgenössische Partiturtechniken ... 418
16.5 Neue Instrumententechniken ... 419
16.6 Leseliste und interessante Referenzpartituren ... 419
16.6.1 Bücher und Artikel über zeitgenössische Notation ... 419
16.6.2 Partituren und Musikbeispiele ... 419

17 Notation von alter Musik ... 420
17.1 Überblick über die unterstützten Stile ... 421
17.2 Alte Notation – Allgemeines ... 422
17.2.1 Vordefinierte Umgebungen ... 422
17.2.2 Ligaturen ... 422
17.2.3 Custodes ... 423
17.2.4 Unterstützung für Generalbass ... 423
17.3 Mensurale Musik setzen ... 424
17.3.1 Mensural-Kontexte ... 424
17.3.2 Mensurale Schlüssel ... 424
17.3.3 Mensurale Taktartenbezeichnungen ... 425
17.3.4 Mensurale Notenköpfe ... 426
17.3.5 Mensurale Fähnchen ... 427

17.3.6 Mensurale Pausen ... 428
17.3.7 Mensurale Versetzungszeichen und Tonartbezeichnung ... 428
17.3.8 Vorgeschlagene Versetzungszeichen (*musica ficta*) ... 429
17.3.9 Weiße Mensuralligaturen ... 429
17.4 Gregorianischen Choral setzen ... 431
17.4.1 Gregorianische Gesangs-Kontexte ... 431
17.4.2 Gregorianische Schlüssel ... 432
17.4.3 Gregorianische Versetzungszeichen und Tonartbezeichnung ... 432
17.4.4 Divisiones ... 433
17.4.5 Artikulationszeichen des Gregorianischen Chorals ... 434
17.4.6 Augmentationspunkte (*morae*) ... 434
17.4.7 Ligaturen der gregorianischen Quadratnotation ... 435
17.5 Kiever Quadratnotation setzen ... 441
17.5.1 Kiever Kontexte ... 441
17.5.2 Kiever Schlüssel ... 442
17.5.3 Kiever Notenköpfe ... 442
17.5.4 Kiever Versetzungszeichen ... 442
17.5.5 Kiever Taktstriche ... 443
17.6 Musiksatz Alter Musik in der Praxis – Szenarien und Lösungen ... 443
17.6.1 Incipite ... 443
17.6.2 Mensurstriche ... 443
17.6.3 Gregorianischen Choral transkribieren ... 444
17.6.4 Alte und moderne Edition aus einer Quelldatei ... 447

18 Weltmusik ... 448
18.1 Übliche Notation für nichteuropäische Musik ... 448
18.1.1 Erweiterung von Notation und Stimmungssystemen ... 448
18.2 Arabische Musik ... 449
18.2.1 Referenz für arabische Musik ... 449
18.2.2 Arabische Notenbezeichnungen ... 449
18.2.3 Arabische Tonarten ... 450
18.2.4 Arabische Taktarten ... 452
18.2.5 Arabische Notenbeispiele ... 453
18.2.6 Weitere Literatur zur arabischen Musik ... 453
18.3 Türkische klassische Musik ... 454
18.3.1 Verweise für türkische klassische Musik ... 454
18.3.2 Türkische Notenbezeichnungen ... 454

Allgemeine Eingabe und Ausgabe

19 Eingabestruktur ... 457
19.1 Struktur einer Partitur ... 457
19.2 Mehrere Partituren in einem Buch ... 458
19.3 Mehrere Ausgabedateien aus einer Eingabedatei ... 459
19.4 Dateinamen der Ausgabedateien ... 460
19.5 Die Dateistruktur ... 461

20 Titel ... 464
20.1 Titel, Kopf- und Fußzeilen erstellen ... 464
20.1.1 Wie funktioniert die Titel-Umgebung? ... 464
20.1.2 Standardlayout von book- und Partitur-Titelumgebungen ... 466
20.1.3 Standardlayout von Kopf- und Fußzeilen ... 469
20.2 Eigene Kopf- und Fußzeilen sowie Titel ... 470
20.2.1 Angepasste Textformatierung für Titelumgebungen ... 470
20.2.2 Angepasstes Layout für Titelumgebungen ... 471
20.2.3 Angepasstes Layout für Kopf- und Fußzeilen ... 474
20.3 Fußnoten erstellen ... 476
20.3.1 Übersicht über Fußnoten ... 476
20.3.2 Automatische Fußnoten ... 476
20.3.3 Manuelle Fußnoten ... 479
20.4 Verweis auf die Seitenzahlen ... 483
20.5 Inhaltsverzeichnis ... 485

21 Arbeiten an Eingabe-Dateien ... 488
21.1 LilyPond-Dateien einfügen ... 488
21.2 Verschiedene Editionen aus einer Quelldatei ... 489
21.2.1 Variablen benutzen ... 489
21.2.2 Marken benutzen ... 491
21.2.3 Globale Einstellungen benutzen ... 500
21.3 Sonderzeichen ... 501
21.3.1 Zeichenkodierung ... 501
21.3.2 Unicode ... 501
21.3.3 ASCII-Aliase ... 502

22 Ausgabe kontrollieren ... 504
22.1 Notationsfragmente extrahieren ... 504
22.2 Korrigierte Musik überspringen ... 504
22.3 Alternative Ausgabeformate ... 505
22.4 Die Notationsschriftart verändern ... 505

23 MIDI-Ausgabe ... 506
23.1 MIDI-Dateien erstellen ... 506
23.2 Der MIDI-Block ... 508
23.3 Was geht in die MIDI-Ausgabe ... 509
23.4 Wiederholungen im MIDI ... 510
23.5 MIDI-Lautstärke kontrollieren ... 510
23.6 Schlagzeug in MIDI ... 514
23.7 Artikuliere-Skript ... 515

24 Musikalische Information extrahieren ... 516
24.1 LilyPond-Notation anzeigen ... 516
24.2 Musikalische Scheme-Ausdrücke anzeigen ... 516
24.3 Musikalische Ereignisse in einer Datei speichern ... 516

Layout-Kontrolle

25 Seitenlayout ... 521

25.1 Die \paper-Umgebung ... 521
25.2 Papierformat und automatische Skalierung ... 522
25.2.1 Das Papierformat einstellen ... 522
25.2.2 Automatische Skalierung auf ein Papierformat ... 523
25.3 Vertikale \paper-Variablen mit festen Abständen ... 523
25.4 Vertikale \paper-Variablen mit flexiblen Abständen ... 524
25.4.1 Struktur der Alisten für flexible vertikale Abstände ... 524
25.4.2 Liste der flexiblen vertikalen Abstandsvariablen in \paper ... 525
25.5 \paper-Variablen für horizontale Abstände ... 526
25.5.1 \paper-Variablen für Breite und Ränder ... 527
25.5.2 \paper-Variablen für zweiseitigen Satz ... 528
25.5.3 \paper-Variablen für Verschiebungen und Einrückungen ... 528
25.6 Andere \paper-Variablen ... 529
25.6.1 \paper-Variablen für den Zeilenumbruch ... 529
25.6.2 \paper-Variablen für den Seitenumbruch ... 529
25.6.3 \paper-Variablen für Seitenzahlen ... 530
25.6.4 Verschiedene \paper-Variablen ... 531

26 Partiturlayout ... 532

26.1 Die \layout-Umgebung ... 532
26.2 Die Notensystemgröße einstellen ... 534

27 Umbrüche ... 536

27.1 Zeilenumbrüche ... 536
27.2 Seitenumbrüche ... 538
27.3 Optimale Seitenumbrüche ... 539
27.4 Optimale Umbrüche zum Blättern ... 539
27.5 Minimale Seitenumbrüche ... 540
27.6 Eine-Seite-Seitenumbrüche ... 540
27.7 Ausdrückliche Umbrüche ... 541
27.8 Eine zusätzliche Stimme für Umbrüche benutzen ... 542

28 Vertikale Abstände ... 545

28.1 Flexible vertikale Abstände in Systemgruppen ... 545
28.1.1 Eigenschaften für Abstände innerhalb von Systemgruppen ... 545
28.1.2 Abstände von nicht gruppierten Notensystemen ... 548
28.1.3 Abstände von gruppierten Notensystemen ... 550
28.1.4 Abstände von nicht-Notensystemzeilen ... 551
28.2 Explizite Positionierung von Systemen ... 552
28.3 Vermeidung von vertikalen Zusammenstößen ... 559

29 Horizontale Abstände ... 561

29.1 Überblick über horizontale Abstände ... 561
29.2 Eine neuer Bereich mit anderen Abständen ... 562
29.3 Horizontale Abstände verändern ... 563
29.4 Zeilenlänge ... 565
29.5 Proportionale Notation ... 565

30 Die Musik auf weniger Seiten zwingen ... 573
30.1 Abstände anzeigen lassen ... 573
30.2 Abstände verändern ... 574

Feinadjustierung des Notenbildes

31 Standardeinstellungen verändern ... 579

32 Interpretationskontexte ... 580
32.1 Was sind Kontexte? ... 580
32.1.1 Score – der Vater aller Kontexte ... 580
32.1.2 Oberste Kontexte – Container für Systeme ... 580
32.1.3 Mittlere Kontexte – Systeme ... 580
32.1.4 Unterste Kontexte – Stimmen ... 581
32.2 Kontexte erstellen und referenzieren ... 582
32.3 Kontexte am Leben halten ... 583
32.4 Umgebungs-Plugins verändern ... 586
32.5 Die Standardeinstellungen von Kontexten ändern ... 588
32.5.1 Alle Kontexte des gleichen Typs verändern ... 588
32.5.2 Nur einen bestimmten Kontext verändern ... 591
32.5.3 Rangfolge von Kontextwerten ... 593
32.6 Neue Kontexte definieren ... 593
32.7 Reihenfolge des Kontextlayouts ... 595

33 Die Referenz der Programminterna erklärt ... 597
33.1 Zurechtfinden in der Programmreferenz ... 597
33.2 Layout-Schnittstellen ... 598
33.3 Die Grob-Eigenschaften ... 599
33.4 Benennungskonventionen ... 600

34 Eigenschaften verändern ... 601
34.1 Grundlagen zum Verändern von Eigenschaften ... 601
34.2 Der `\set`-Befehl ... 601
34.3 Der `\\override`-Befehl ... 603
34.4 Der `\tweak`-Befehl ... 605
34.5 `\\set` versus `\\override` ... 607
34.6 Alisten verändern ... 607

35 Nützliche Konzepte und Eigenschaften ... 610
35.1 Eingabe-Modi ... 610
35.2 Richtung und Platzierung ... 611
35.3 Abstände und Maße ... 613
35.4 Eigenschaften des Staff-Symbols ... 613
35.5 Strecker ... 614
35.6 Sichtbarkeit von Objekten ... 619
35.6.1 Einen `stencil` entfernen ... 619
35.6.2 Objekten unsichtbar machen ... 620
35.6.3 Objekte weiß malen ... 620

35.6.4 `break-visibility` (unsichtbar machen) benutzen ... 620
35.6.5 Besonderheiten ... 622
35.7 Linienstile ... 624
35.8 Drehen von Objekten ... 625
35.8.1 Drehen von Layout-Objekten ... 625
35.8.2 Textbeschriftung drehen ... 626

36 Fortgeschrittene Optimierungen ... 627
36.1 Objekte ausrichten ... 627
36.1.1 `X-offset` und `Y-offset` direkt setzen ... 628
36.1.2 Das `side-position-interface` benutzen ... 628
36.1.3 Das `self-alignment-interface` benutzen ... 629
36.1.4 Benutzung des `break-alignable-interface` ... 630
36.2 Vertikale Gruppierung der grafischen Objekte („grob“s) ... 632
36.3 stencils verändern ... 632
36.4 Formen verändern ... 633
36.4.1 Bögen verändern ... 633
36.5 Reine und unreine Container ... 634

37 Musikfunktionen benutzen ... 637
37.1 Syntax der Ersetzungsfunktion ... 637
37.2 Beispiele der Ersetzungsfunktionen ... 638

Anhänge

Anhang A Notationsübersicht ... 643
A.1 Liste der Akkordbezeichnungen ... 643
A.2 Übliche Akkord-Variablen ... 643
A.3 Vordefinierte Saitenstimmungen ... 646
A.4 Die vordefinierten Bund-Diagramme ... 647
A.5 Vordefinierte Papierformate ... 652
A.6 MIDI-Instrumente ... 655
A.7 Liste der Farben ... 656
A.8 Die Emmentaler-Schriftart ... 658
A.8.1 Notenschlüssel-Glyphen ... 658
A.8.2 Taktart-Glyphen ... 658
A.8.3 Zahlen-Glyphen ... 659
A.8.4 Versetzungszeichen-Glyphen ... 660
A.8.5 Standard-Notenkopf-Glyphen ... 661
A.8.6 Spezielle Notenkopf-Glyphen ... 662
A.8.7 Geformte Notenkopf-Glyphen ... 662
A.8.8 Pausen-Glyphen ... 665
A.8.9 Fähnchen-Glyphen ... 666
A.8.10 Punkt-Glyphen ... 667
A.8.11 Dynamik-Glyphen ... 667
A.8.12 Schrift-Glyphen ... 667
A.8.13 Pfeilkopf-Glyphen ... 670
A.8.14 Klammerspitzen-Glyphen ... 670
A.8.15 Pedal-Glyphen ... 670

A.8.16 Akkordeon-Glyphen . . . 670
A.8.17 Bogen-Glyphen . . . 670
A.8.18 Vaticana-Glyphen . . . 671
A.8.19 Medicaea-Glyphen . . . 672
A.8.20 Hufnagel-Glyphen . . . 672
A.8.21 Mensural-Glyphen . . . 673
A.8.22 Neomensural-Glyphen . . . 675
A.8.23 Petrucci-Glyphen . . . 676
A.8.24 Solesmes-Glyphen . . . 677
A.8.25 Glyphen der Kiever Notation . . . 677
A.9 Notenkopfstile . . . 678
A.10 Textbeschriftungsbefehle . . . 678
A.10.1 Font markup . . . 678
A.10.2 Markup for text alignment . . . 691
A.10.3 Graphical markup . . . 710
A.10.4 Markup for music and musical symbols . . . 721
A.10.5 Conditional markup . . . 734
A.10.6 Instrument-specific markup . . . 734
A.10.7 Accordion registers . . . 740
A.10.8 Other markup commands . . . 744
A.11 Textbeschriftungslistenbefehle . . . 756
A.12 Liste der Sonderzeichen . . . 759
A.13 Liste der Artikulationszeichen . . . 760
A.13.1 Artikulationsskripte . . . 760
A.13.2 Ornamentale Skripte . . . 761
A.13.3 Fermatenskripte . . . 761
A.13.4 Instrumentenspezifische Skripte . . . 761
A.13.5 Wiederholungszeichenskripte . . . 762
A.13.6 Skripte der alten Notation . . . 762
A.14 Schlagzeugnoten . . . 763
A.15 Technisches Glossar . . . 764
A.15.1 alist . . . 764
A.15.2 callback . . . 764
A.15.3 closure . . . 764
A.15.4 glyph . . . 764
A.15.5 grob . . . 765
A.15.6 immutable . . . 765
A.15.7 interface . . . 765
A.15.8 lexer . . . 765
A.15.9 mutable . . . 765
A.15.10 output-def . . . 766
A.15.11 parser . . . 766
A.15.12 parser variable . . . 766
A.15.13 prob . . . 766
A.15.14 simple closure . . . 766
A.15.15 smob . . . 767
A.15.16 stencil . . . 767
A.16 Erhältliche Musikfunktionen . . . 767
A.17 Bezeichner zur Kontextveränderung . . . 787
A.18 Vordefinierte Typprädikate . . . 788
A.18.1 R5RS primary predicates . . . 788
A.18.2 R5RS secondary predicates . . . 788

A.18.3 Guile predicates ... 789
A.18.4 LilyPond scheme predicates ... 789
A.18.5 LilyPond exported predicates ... 790

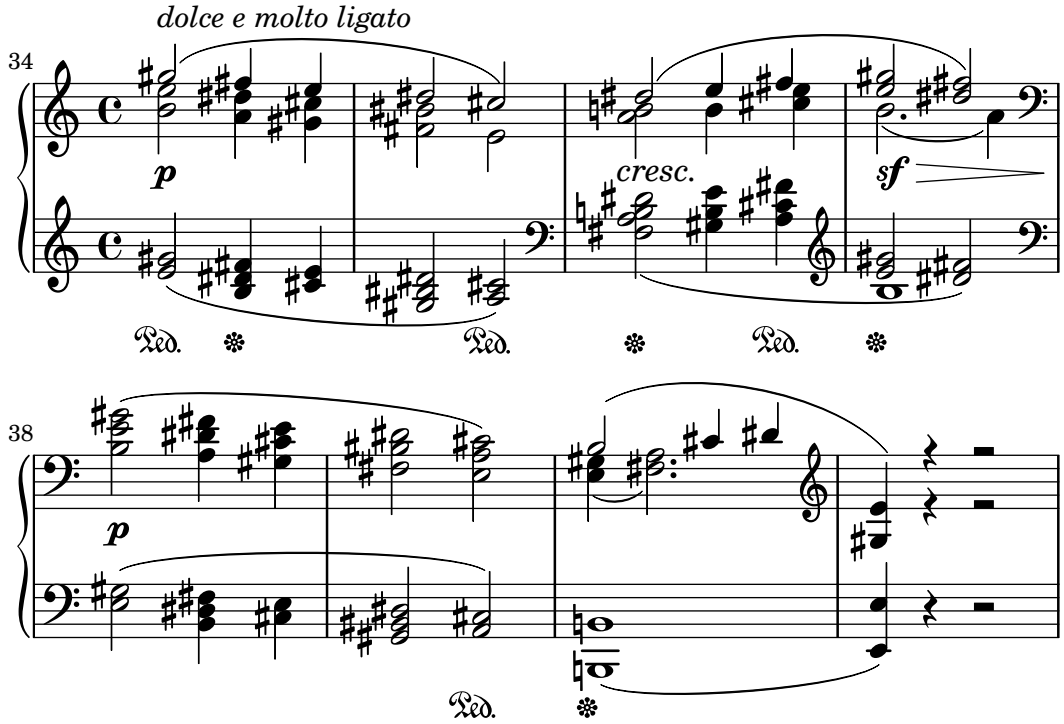
Anhang B Befehlsübersicht ... 792

Anhang C GNU Free Documentation License ... 795

Anhang D Register ... 802

Allgemeine Notation

1 Tonhöhen



Dieser Abschnitt zeigt, wie man die Tonhöhe notieren kann. Es gibt drei Stufen in diesem Prozess: Eingabe, Veränderung und Ausgabe.
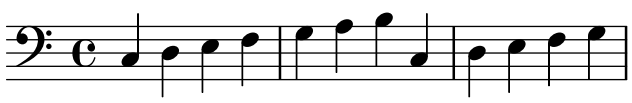
1.1 Tonhöhen setzen

Dieser Abschnitt zeigt, wie man Tonhöhen notiert. Es gibt zwei verschiedene Möglichkeiten, Noten in bestimmten Oktaven zu notieren: den absoluten und den relativen Modus. In den meisten Fällen eignet sich der relative Modus besser.

1.1.1 Absolute Oktavenbezeichnung
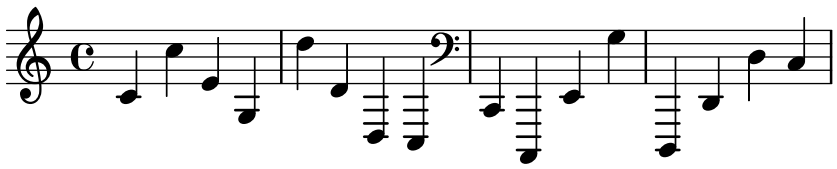
Tonhöhenbezeichnungen werden durch Kleinbuchstaben von `a` bis `g` angegeben. Dabei wird ein aus dem Englischen entlehntes Modell benutzt, das sich vom Deutschen dadurch unterscheidet, dass `b` für die Note „H“ steht. Die Benutzung deutscher Notenbezeichnungen mit der Unterscheidung von `b` und `h` ist auch möglich, siehe Abschnitt 1.1.4 [Notenbezeichnungen in anderen Sprachen], Seite 9. Die Notenbezeichnungen `c` bis `b` werden in der Oktave unter dem eingestrichenen C gesetzt.

```
{
  \clef bass
  c4 d e f
  g4 a b c
  d4 e f g
}
```



Andere Oktaven können erreicht werden, indem man ein Apostroph (`'`) oder ein Komma (`,`) benutzt. Jedes `'` erhöht die Tonhöhe um eine Oktave, jedes `,` erniedrigt sie um eine Oktave.

```
{
  \clef treble
  c'4 c'' e' g
  d''4 d' d c
  \clef bass
  c,4 c,, e, g
  d,,4 d, d c
}
```



Siehe auch

Glossar: Abschnitt “Pitch names” in *Glossar*.

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

1.1.2 Relative Oktavenbezeichnung

Wenn Oktaven im absoluten Modus notiert, passiert es schnell, eine Note auf der falschen Oktave zu notieren. Mit dem relativen Modus kommen solche Fehler seltener vor, weil man die Oktave nur noch sehr selten spezifizieren muss. Hinzu kommt, dass im absoluten Modus ein einzelner Fehler schwer zu finden ist, während er im relativen Modus den ganzen Rest des Stückes um eine Oktave verschiebt.
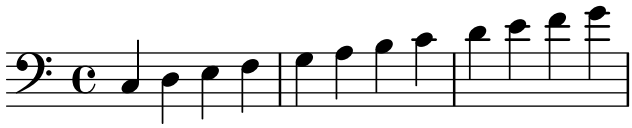
`\relative` *`Anfangstonhöhe musikalischer Ausdruck`*

Im relativen Modus wird angenommen, dass sich jede folgende Note so dicht wie möglich bei der nächsten befindet. Das bedeutet, dass die Oktave jeder Tonhöhe innerhalb eines *musikalischen Ausdrucks* wie folgt errechnet wird:

- Wenn kein Oktaväänderungszeichen an einer Tonhöhe benutzt wird, wird ihre Oktave so errechnet, dass das Intervall zur vorigen Noten weniger als eine Quinte ist. Das Intervall wird errechnet, ohne Versetzungszeichen zu berücksichtigen.
- Ein Oktaväänderungszeichen `'` oder `,` kann hinzugefügt werden, um eine Tonhöhe explizit um eine Oktave zu erhöhen bzw. zu erniedrigen, relativ zu der Tonhöhe, die ohne das Oktaväänderungszeichen errechnet wurde.
- Mehrfache Oktaväänderungszeichen können benutzt werden. Die Zeichen`''` und `,,` ändern zum Beispiel die Tonhöhe um zwei Oktaven.
- Die Tonhöhe der ersten Note ist relativ zu *`Anfangstonhöhe`*. Die *Anfangstonhöhe* wird im absoluten Modus gesetzt, und als Empfehlung gilt, eine Oktave von C zu nehmen.
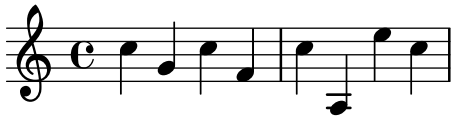
So funktioniert der relative Modus:

```
\relative {
  \clef bass
  c d e f
  g a b c
  d e f g
}
```
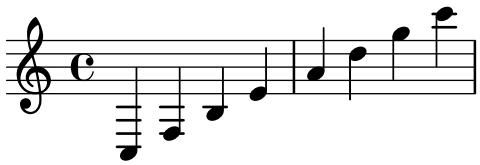


Oktavversetzungen müssen für alle Intervalle angezeigt werden, die größer als eine Quarte sind.

```
\relative {
  c'' g c f,
  c' a, e'' c
}
```
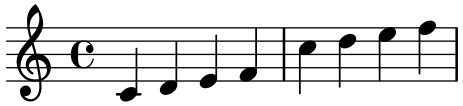


Eine Sequenz ohne ein einziges Oktaväänderungszeichen kann aber trotzdem weite Intervalle umfassen:

```
\relative {
  c f b e
  a d g c
}
```



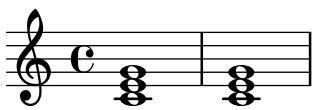
Wenn \relative-Umgebungen geschachtelt werden, gilt der innerste \relative-Abschnitt.

```
\relative {
  c' d e f
  \relative {
    c'' d e f
  }
}
```



\relative hat keine Auswirkung auf \chordmode-Abschnitte.
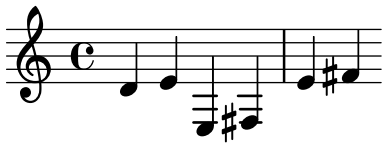
```
\new Staff {
  \relative c''' {
    \chordmode { c1 }
  }
  \chordmode { c1 }
}
```



\relative darf nicht innerhalb von \chordmode notiert werden.
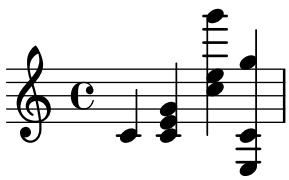
Tonhöhen innerhalb eines \transpose-Abschnitts sind absolut, es sei denn ein \relativ wird eingefügt.

```
\relative {
  d' e
  \transpose f g {
    d e
    \relative {
      d' e
    }
  }
}
```
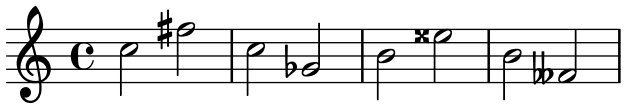


Wenn der vorherige Ausdruck ein Akkord ist, wird die erste Note des Akkordes benutzt, um die erste Note des nächsten Akkordes zu bestimmen. Innerhalb von Akkorden ist die nächste Note immer relativ zur vorherigen. Betrachten Sie das folgende Beispiel aufmerksam, insbesondere die c-Noten.

```
\relative {
  c'
  <c e g>
  <c' e g'>
  <c, e, g''>
}
```



Wie oben erklärt wurde, wird die Oktave einer Tonhöhe nur nach ihrer Notenbezeichnung errechnet, unabhängig von allen Versetzungszeichen. Darum wird ein Eisis auf ein H (notiert als b) folgend höher gesetzt, während ein Feses tiefer gesetzt wird. Anders gesagt wird eine doppelterhöhte Quarte wird als kleineres Intervall angesehen als eine doppelterniedrigte Quinte, unabhängig von der Anzahl an Halbtönen, die jedes Intervall enthält.

```
\relative {
  c''2 fis
  c2 ges
  b2 eisis
  b2 feses
}
```



Eine Konsequenz dieser Regeln ist, dass die erste Note innerhalb von `\relative f` auf die selbe Art interpretiert wird als wenn sie im absoluten Tonhöhenmodus geschrieben worden wäre.

Siehe auch

Musikgossar: Abschnitt “fifth” in *Glossar*, Abschnitt “interval” in *Glossar*, Abschnitt “Pitch names” in *Glossar*.

Notationsreferenz: Abschnitt 1.2.1 [Oktavenüberprüfung], Seite 11.

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

Referenz der Interna: Abschnitt “RelativeOctaveMusic” in *Referenz der Interna*.
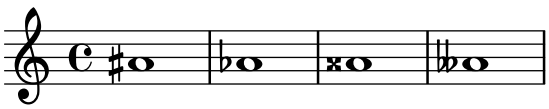
Wenn keine *Anfangstonhöhe* für `\relative` angegeben wird, wird `c'` angenommen. Das ist aber eine veraltete Option, die in späteren Programmversionen verschwinden kann. Darum wird von der Benutzung abgeraten.

1.1.3 Versetzungszeichen

> **Achtung:** Neue Benutzer sind manchmal verwirrt, wie Versetzungszeichen und Vorzeichen/Tonarten funktionieren. In LilyPond sind Notenbezeichnungen die wirkliche Tonhöhe, erst durch Vorzeichen wird bestimmt, wie diese Tonhöhe dann im Notenbild dargestellt wird. Eine einfache Tonhöhe wie etwa `c` bedeutet also immer das eingestrichene C ohne Versetzungszeichen, egal was für Vorzeichen/Tonart oder Schlüssel gesetzt sind. Mehr Information dazu in Abschnitt “Tonhöhen und Tonartbezeichnungen (Vorzeichen)” in *Handbuch zum Lernen*.

Ein *Kreuz* wird eingegeben, indem man `-is` an die Notenbezeichnung hängt, ein b durch `-es`. Doppelkreuze und Doppel-Bs werden durch Hinzufügen von `-isis` und `-eses` hinter die Notenbezeichnung erzeugt. Diese Syntax leitet sich von den holländischen Notenbezeichnungen ab. Um andere Bezeichnungen für Versetzungszeichen zu benutzen, siehe Abschnitt 1.1.4 [Notenbezeichnungen in anderen Sprachen], Seite 9.
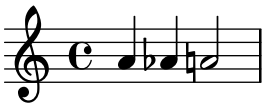
```
\relative c'' { ais1 aes aisis aeses }
```



Auch die deutschen Varianten `as` für `aes` und `es` für `ees` sind erlaubt. Im Unterschied zum Deutschen ist aber `bes` die einzige Version für den Ton B, während his als `bis` geschrieben werden muss. Das kann aber auch verändert werden, siehe Abschnitt 1.1.4 [Notenbezeichnungen in anderen Sprachen], Seite 9.

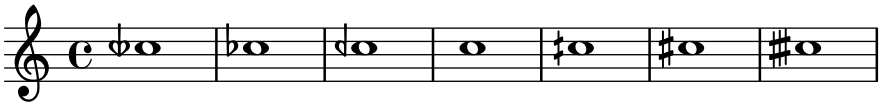
Ein Auflösungszeichen macht die Wirkung eines Kreuzes oder Bs rückgängig. Diese Auflösungszeichen werden jedoch nicht als Suffix einer Tonhöhenbezeichnung eingegeben, sondern sie ergeben sich (automatisch) aus dem Kontext, wenn die nicht alterierte Notenbezeichnung eingegeben wird.

```
a4 aes a2
```
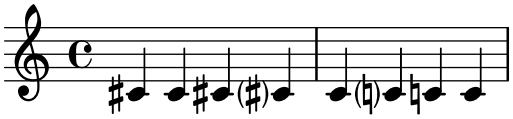


Versetzungszeichen für Vierteltöne werden durch Anhängen der Endungen `-eh` (Erniedrigung) und `-ih` (Erhöhung) an den Tonhöhenbuchstaben erstellt. Das Beispiel zeigt eine in Vierteltönen aufsteigende Serie vom eingestrichenen C.

```
\relative c'' { ceseh1 ces ceh c cih cis cisih }
```



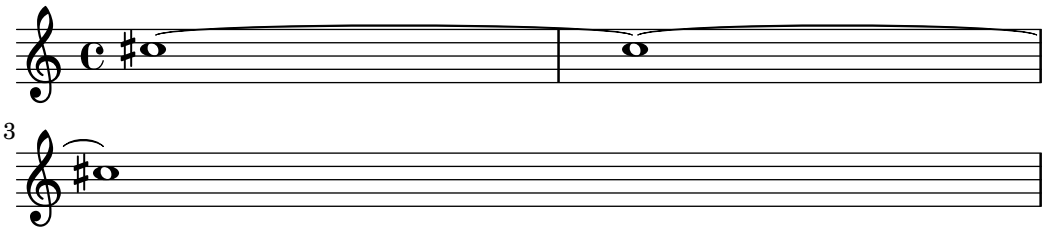
Normalerweise werden Versetzungszeichen automatisch gesetzt, aber sie können auch manuell hinzugefügt werden. Ein erinnerndes Versetzungszeichen kann erzwungen werden, indem man ein Ausrufungszeichen (`!`) hinter die Notenbezeichnung schreibt. Ein warnendes Versetzungszeichen (also ein Vorzeichen in Klammern) wird durch Anfügen eines Fragezeichens (`?`) erstellt. Mit diesen zusätzlichen Zeichen kann man sich auch Auflösungszeichen ausgeben lassen.

```
cis cis cis! cis? c c? c! c
```



Versetzungzeichen von übergebundenen Noten werden nur dann gesetzt, wenn ein neues System begonnen wird:

```
cis1~ 1~
\break
cis
```

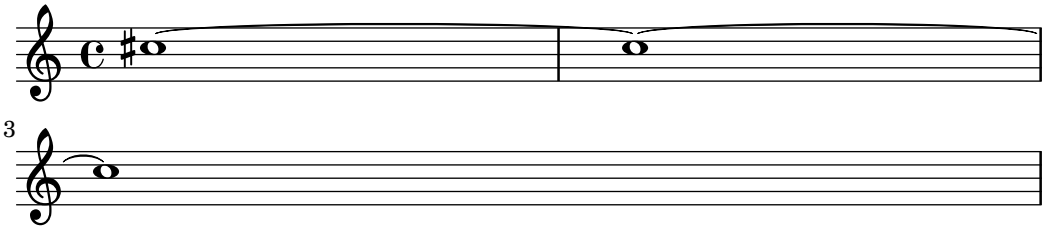


Ausgewählte Schnipsel

Hiding accidentals on tied notes at the start of a new system

This shows how to hide accidentals on tied notes at the start of a new system.
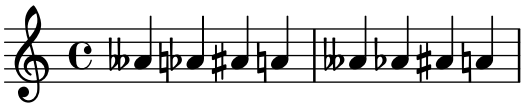
```
\relative c'' {
  \override Accidental.hide-tied-accidental-after-break = ##t
  cis1~ cis~
  \break
  cis
}
```



Verhindern, dass zusätzliche Auflösungszeichen automatisch hinzugefügt werden

Den traditionellen Notensatzregeln zufolge wird ein Auflösungszeichen immer dann vor einem Kreuz oder B gesetzt, wenn ein vorheriges Versetzungszeichen der gleichen Note aufgehoben werden soll. Um dieses Verhalten zu ändern, muss die Eigenschaft `extraNatural` im `Staff`-Kontext auf "false" gesetzt werden.

```
\relative c'' {
  aeses4 aes ais a
  \set Staff.extraNatural = ##f
  aeses4 aes ais a
}
```



Siehe auch

Glossar: Abschnitt “sharp” in *Glossar*, Abschnitt “flat” in *Glossar*, Abschnitt “double sharp” in *Glossar*, Abschnitt “double flat” in *Glossar*, Abschnitt “Pitch names” in *Glossar*, Abschnitt “quarter tone” in *Glossar*.

Handbuch zum Lernen: Abschnitt “Tonhöhen und Tonartbezeichnungen (Vorzeichen)” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 1.3.5 [Automatische Versetzungszeichen], Seite 26, Abschnitt 17.3.8 [Vorgeschlagene Versetzungszeichen (*musica ficta*)], Seite 429, Abschnitt 1.1.4 [Notenbezeichnungen in anderen Sprachen], Seite 9.

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

Referenz der Interna: Abschnitt “Accidental_engraver” in *Referenz der Interna*, Abschnitt “Accidental” in *Referenz der Interna*, Abschnitt “AccidentalCautionary” in *Referenz der Interna*, Abschnitt “accidental-interface” in *Referenz der Interna*.

Bekannte Probleme und Warnungen
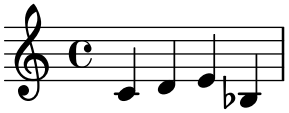
Es gibt keine allgemeinen Regeln für die Notation von Vierteltönen, die Symbole von LilyPond folgen also keinem Standard.

1.1.4 Notenbezeichnungen in anderen Sprachen

Es gibt vordefinierte Bezeichnungen für die Notenbezeichnungen in anderen Sprachen als Englisch. Die Sprache für die Notenbezeichnungen wird normalerweise zu Beginn einer Datei ausgewählt: das folgende Beispiel zeigt die Verwendung von italienischen Notenbezeichnungen:

```
\language "italiano"

\relative {
  do' re mi sib
}
```



In der Tabelle sind die existierenden Sprachdefinitionen mit den dazugehörigen Notenbezeichnungen dargestellt.

Sprache	**Notenbezeichnungen**
nederlands	c d e f g a bes b
català oder catalan	do re mi fa sol la sib si
deutsch	c d e f g a b h
english	c d e f g a bf/b-flat b
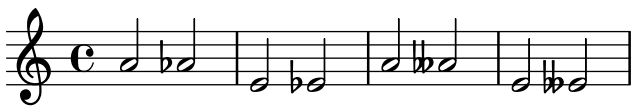
español oder espanol	do re mi fa sol la sib si
français	do ré/re mi fa sol la sib si
italiano	do re mi fa sol la sib si
norsk	c d e f g a b h
português oder portugues	do re mi fa sol la sib si
suomi	c d e f g a b h
svenska	c d e f g a b h
vlaams	do re mi fa sol la sib si

und die dazugehörigen Versetzungszeichen-Endungen:

Sprache	Kreuz	B	Doppelkreuz	Doppel-B
nederlands	is	es	isis	eses
català oder catalan	d/s	b	dd/ss	bb
deutsch	is	es	isis	eses
english	s/-sharp	f/-flat	ss/x/-sharpsharp	ff/-flatflat
español oder espanol	s	b	ss/x	bb
français	d	b	dd/x	bb
italiano	d	b	dd	bb
norsk	iss/is	ess/es	ississ/isis	essess/eses
português oder portugues	s	b	ss	bb
suomi	is	es	isis	eses
svenska	iss	ess	ississ	essess
vlaams	k	b	kk	bb

Auf Holländisch, Deutsch, Norwegisch und Schwedisch (u. a.) wird die Erniedrigungen von ‚a' – aes – zu as zusammengezogen. Beide Formen werden jedoch akzeptiert. Genauso gelten auch es und ees. Das gilt auch für aeses / ases und eeses / eses. In manchen Sprachen sind nur diese Kurzformen definiert.

```
\relative c'' { a2 as e es a ases e eses }
```



Bestimmte Musik verwendet Alterationen, die Bruchteile von den „normalen" Kreuzen oder Bs sind. Die Notenbezeichnungen für Vierteltöne für die verschiedenen Sprachen sind in der folgenden Tabelle aufgeführt. Die Präfixe *semi-* und *sesqui-* bedeuten „halb" bzw. „eineinhalb".

Sprache	Vierteltonkreuz	Viertelton-B	3/4-Tonkreuz	3/4-Ton-B
nederlands	ih	eh	isih	eseh
català oder catalan	qd/qs	qb	tqd/tqs	tqb
deutsch	ih	eh	isih	eseh
english	qs	qf	tqs	tqf
español oder espanol	cs	cb	tcs	tcb
français	sd	sb	dsd	bsb
italiano	sd	sb	dsd	bsb
norsk	ih	eh	issih/isih	esseh/eseh

português oder portugues	sqt	bqt	stqt	btqt
suomi	ih	eh	isih	eseh
svenska	ih	eh	issih	esseh
vlaams	hk	hb	khk	bhb

Die meisten Sprachen, die hier vorkommen, werden normalerweise mit der klassischen westlichen Musik assoziiert. Alternative Tonhöhen und Stimmungen sind aber auch unterstützt: siehe Abschnitt 18.1 [Übliche Notation für nichteuropäische Musik], Seite 448,

Siehe auch

Glossar: Abschnitt “Pitch names” in *Glossar*, Abschnitt “Common Practice Period” in *Glossar*.

Notationsreferenz: Abschnitt 18.1 [Übliche Notation für nichteuropäische Musik], Seite 448.

Installierte Dateien: `scm/define-note-names.scm`.

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

1.2 Viele Tonhöhen gleichzeitig verändern
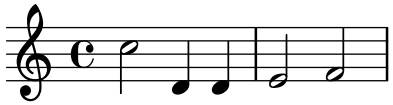
Dieser Abschnitt zeigt, wie man Tonhöhen beeinflusst.

1.2.1 Oktavenüberprüfung

Im relativen Modus geschieht es recht häufig, dass ein Oktaväderungszeichen vergessen wird. Oktavenüberprüfungen machen es einfacher, solche Fehler zu entdecken und zu korrigieren. Sie geben eine Warnung aus und korrigieren die Oktave, wenn eine Note in einer unerwarteten Oktave gefunden wird.
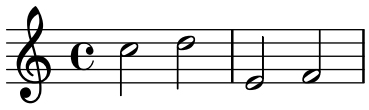
Um die Oktave einer Note zu überprüfen, muss die absolute Oktave nach dem =-Symbol angegeben werden. Im folgenden Beispiel wird eine Warnung (und eine Tonhöhenänderung) generiert, weil die zweite Note als absolute Oktave ein `d''` anstelle von `d'` notiert ist, wie es die Oktavierungskorrektur markiert.

```
\relative {
  c''2 d='4 d
  e2 f
}
```



Die Oktave von einer Note kann auch mit dem `\octaveCheck` *Kontrolltonhöhe*-Befehl überprüft werden. *Kontrollhöhe* wird im absoluten Modus eingegeben. Dabei wird überprüft, ob das Intervall zwischen der vorherigen Note und der *Kontrolltonhöhe* nicht größer als eine Quarte ist (die normale Berechnung im relativen Modus). Wenn diese Überprüfung einen Fehler ausgibt, wird eine Warnung gemeldet, aber die vorigen Note wird nicht verändert. Folgende Noten sind dann relativ zur *Kontrolltonhöhe*.
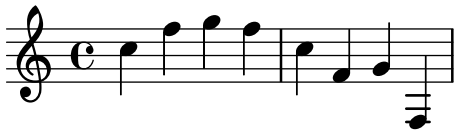
```
\relative {
  c''2 d
  \octaveCheck c'
  e2 f
}
```



Vergleichen Sie die zwei Takte im nächsten Beispiel. Die erste und dritte `\octaveCheck`-Überprüfung gibt einen Fehler aus, die zweite dagegen ist erfolgreich:

```
\relative {
  c''4 f g f

  c4
  \octaveCheck c'
  f
  \octaveCheck c'
  g
  \octaveCheck c'
  f
}
```



Siehe auch

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

Referenz der Interna: Abschnitt “RelativeOctaveCheck” in *Referenz der Interna*.

1.2.2 Transponieren

Ein musikalischer Ausdruck kann mit dem Befehl `\transpose` transponiert werden. Die Syntax lautet:
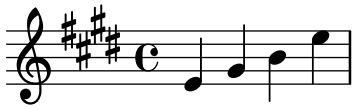
```
\transpose vonTonhöhe nachTonhöhe mus. Ausdruck
```

Das bedeutet, dass der *mus. Ausdruck* um das Intervall zwischen den Tonhöhen *vonTonhöhe* und *nachTonhöhe* transponiert wird: Jede Note, die die Tonhöhe *vonTonhöhe* hat, wird in die Tonhöhe *nachTonhöhe* umgewandelt, und alle anderen Noten um das gleiche Intervall. Beide Tonhöhen werden im absoluten Modus eingegeben.

> **Achtung:** Tonhöhen innerhalb einer `\transpose`-Umgebung sind absolut, es sei denn, ein `\relative` wird eingefügt.
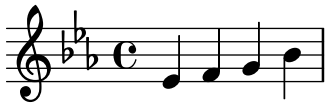
So kann z. B. ein Stück in D-Dur, wenn es für den Sänger etwas zu tief ist, nach E-Dur transponiert werden. Dabei werden auch die Vorzeichen entsprechend angepasst:

```
\transpose d e {
  \relative {
    \key d \major
    d'4 fis a d
  }
}
```



Wenn eine Stimme, die in C notiert ist, von einer A-Klarinette gespielt werden soll (für die A als C notiert wird, aber eine kleine Terz tiefer erklingt als es notiert ist), kann die entsprechende Stimme wie folgt erstellt werden:
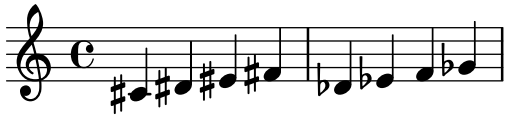
```
\transpose a c' {
  \relative {
    \key c \major
    c'4 d e g
  }
}
```



Beachten Sie, dass `\key c \major` explizit angegeben werden muss. Wenn hier keine Tonart angemerkt würde, würde die Noten zwar transponiert, aber keine Vorzeichen angezeigt werden.

`\transpose` unterscheidet enharmonische Verwechslungen: sowohl `\transpose c cis` als auch `\transpose c des` transponieren die Musik einen Halbton nach oben. Aber die erste Version gibt als Versetzungszeichen Kreuze aus, die zweite dagegen B-Versetzungszeichen.

```
music = \relative { c' d e f }
\new Staff {
  \transpose c cis { \music }
  \transpose c des { \music }
}
```



`\transpose` kann auch benutzt werden, um die geschriebenen Noten eines transponierenden Instruments zu notieren. Im vorigen Beispiel wurde die Tonhöhen so eingegeben, wie sie erklingen (also in C), aber man kann genauso gut auch andersherum aus einer Stimme, die für ein transponierendes Instrument in einem anderen Ton als C geschrieben wurde, eine Partitur in C erstellen. Die Noten einer B-Trompete, die mit einem notierten E (also einem klingenden D) anfangen, könnte man also auch so eingeben:

```
musicInBflat = { e4 ... }
\transpose c bes, \musicInBflat
```

Um die Noten dann in F zu setzen (um sie etwa für ein Horn zu arrangieren), könnte man die schon geschriebenen Noten wieder mit einem weiteren `\transpose` umgeben:

```
musicInBflat = { e4 ... }
\transpose f c' { \transpose c bes, \musicInBflat }
```

Für mehr Information zu transponierenden Instrumenten siehe auch Abschnitt 1.3.4 [Transposition von Instrumenten], Seite 25.

Ausgewählte Schnipsel

Transposing pitches with minimum accidentals („smart“ transpose)

This example uses some Scheme code to enforce enharmonic modifications for notes in order to have the minimum number of accidentals. In this case, the following rules apply:

- double accidentals should be removed
- b sharp → c
- e sharp → f
- c flat → b
- f flat → e

In this manner, the most natural enharmonic notes are chosen.

```
#(define (naturalize-pitch p)
   (let ((o (ly:pitch-octave p))
         ;; `ly:pitch-alteration` returns quarter tone steps.
         (a (* 4 (ly:pitch-alteration p)))
         (n (ly:pitch-notename p)))
     (cond
      ((and (> a 1)
            (or (eqv? n 6) (eqv? n 2)))
       (set! a (- a 2))
       (set! n (+ n 1)))
      ((and (< a -1)
            (or (eqv? n 0) (eqv? n 3)))
       (set! a (+ a 2))
       (set! n (- n 1))))
     (cond
      ((> a 2)
       (set! a (- a 4))
       (set! n (+ n 1)))
      ((< a -2)
       (set! a (+ a 4))
       (set! n (- n 1))))
     (when (< n 0)
       (set! o (- o 1))
       (set! n (+ n 7)))
     (when (> n 6)
       (set! o (+ o 1))
       (set! n (- n 7)))
     (ly:make-pitch o n (/ a 4))))
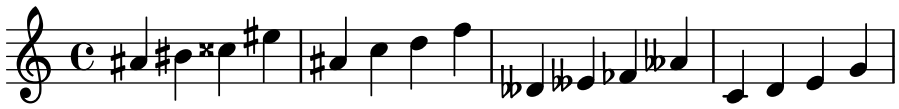
#(define (naturalize music)
   (let ((es (ly:music-property music 'elements))
         (e (ly:music-property music 'element))
         (p (ly:music-property music 'pitch)))
     (when (pair? es)
       (ly:music-set-property! music 'elements
                               (map naturalize es)))
```

```
      (when (ly:music? e)
        (ly:music-set-property! music 'element
                                (naturalize e)))
      (when (ly:pitch? p)
        (set! p (naturalize-pitch p))
        (ly:music-set-property! music 'pitch p))
      music))

naturalizeMusic =
#(define-music-function (m) (ly:music?)
   (naturalize m))

music = \relative c' { c4 d e g }

\new Staff {
  \transpose c ais { \music }
  \naturalizeMusic \transpose c ais { \music }
  \transpose c deses { \music }
  \naturalizeMusic \transpose c deses { \music }
}
```



Siehe auch

Notationsreferenz: Abschnitt 1.1.2 [Relative Oktavenbezeichnung], Seite 4, Abschnitt 1.3.4 [Transposition von Instrumenten], Seite 25, Abschnitt 1.2.3 [Umkehrung], Seite 15, Abschnitt 1.2.4 [Krebs], Seite 16, Abschnitt 1.2.5 [Modale Transformierungen], Seite 16.

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

Referenz der Interna: Abschnitt “TransposedMusic” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Der relative Modus wirkt nicht in `\transpose`, `\chordmode` oder `\relative`. Um auch im relativen Modus transponieren zu können, muss ein `\relative` innerhalb des `\transpose` zusätzlich gesetzt werden.

Dreifache Versetzungszeichen werden nicht ausgegeben, wenn `\transpose` benutzt wird. Eine enharmonisch entsprechende Tonhöhe wird anstatt dessen gesetzt (z. B. Des anstelle von e-dreifach-b).
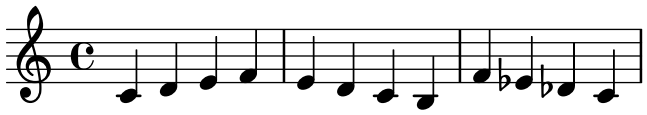
1.2.3 Umkehrung

Ein musikalischer Ausdruck kann in seine Umkehrung transformiert und gleichzeitig transponiert werden:

`\inversion` *`umTonhöhe nachTonhöhe mus. Ausdruck`*

Der *`mus. Ausdruck`* wird Intervall für Intervall um v*`UmTonhöhe`* umgekehrt und dann von v*`UmTonhöhe`* nach *`nachTonhöhe`* transponiert.

```
music = \relative { c' d e f }
\new Staff {
  \music
  \inversion d' d' \music
  \inversion d' ees' \music
}
```



Achtung: Motive, die umgekehrt werden, sollen in absoluten Tonhöhen notiert werden oder zuerst in absolute Tonhöhen umgewandelt werden, indem sie in eine `\relative`-Umgebung eingeschlossen werden.

Siehe auch
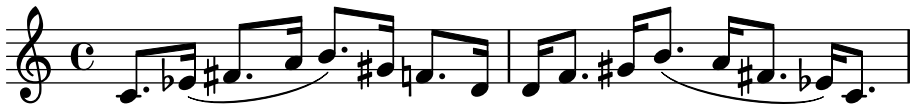
Notationsreferenz: Abschnitt 1.2.5 [Modale Transformierungen], Seite 16, Abschnitt 1.2.4 [Krebs], Seite 16, Abschnitt 1.2.2 [Transponieren], Seite 12.

1.2.4 Krebs

Ein musikalischer Ausdruck kann umgekehrt werden, um im Krebsgang dargestellt zu werden:

```
music = \relative { c'8. ees16( fis8. a16 b8.) gis16 f8. d16 }

\new Staff {
  \music
  \retrograde \music
}
```



Bekannte Probleme und Warnungen

Manuell gesetzte Bindebögen innerhalb des Krebsgangs `\retrograde` werden aufgebrochen und erzeugen Warnungen. Einige Bindebögen können automatisch erstellt werden, indem man Abschnitt 2.3.6 [Automatische Aufteilung von Noten], Seite 78, aktiviert.

Siehe auch

Notationsreferenz: Abschnitt 1.2.5 [Modale Transformierungen], Seite 16, Abschnitt 1.2.3 [Umkehrung], Seite 15, Abschnitt 1.2.2 [Transponieren], Seite 12.

1.2.5 Modale Transformierungen

In einer Komposition, die auf einer Skala basiert, wird ein Motiv oft in unterschiedlicher Weise transformiert. Es kann transponiert werden, um von einer anderen Tonhöhe in der Skala zu beginnen, oder beginnend von einer bestimmten Tonhöhe der Skala umgekehrt werden. Es kann auch im Krebsgang, also rückwärts erscheinen, siehe Abschnitt 1.2.4 [Krebs], Seite 16.

Achtung: Alle Noten, die nicht zu der vorgegebenen Skala gehören, werden nicht transformiert.

Modale Transposition
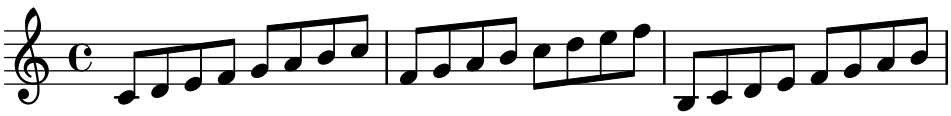
Ein Motiv kann innerhalb einer Skala transponiert werden:

`\modalTranspose` *vonTonhöhe nachTonhöhe Skala mus. Ausdruck*

Die Noten von *mus. Ausdruck* werden innerhalb von *Skala* um die Anzahl von Skalenstufen transponiert, die dem Intervall zwischen *vonTonhöhe* und *nachTonhöhe* entspricht.

```
diatonicScale = \relative { c' d e f g a b }
motif = \relative { c'8 d e f g a b c }
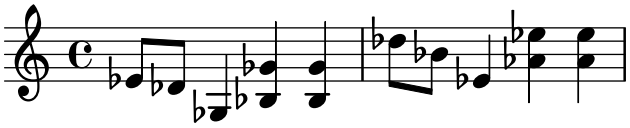
\new Staff {
  \motif
  \modalTranspose c f \diatonicScale \motif
  \modalTranspose c b, \diatonicScale \motif
}
```



Eine absteigende Skala beliebiger Länge und mit beliebigen Intervallen kann definiert werden:

```
pentatonicScale = \relative { ges aes bes des ees }
motif = \relative { ees'8 des ges,4 <ges' bes,> <ges bes,> }
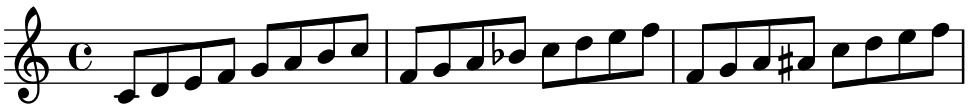
\new Staff {
  \motif
  \modalTranspose ges ees' \pentatonicScale \motif
}
```



Wenn `\modalTranspose` mit einer chromatischen Skala benutzt wird, entspricht es der Funktion von `\transpose` mit dem Unterschied, dass die Bezeichnungen der eingesetzten Noten definiert werden können:

```
chromaticScale = \relative { c' cis d dis e f fis g gis a ais b }
motif = \relative { c'8 d e f g a b c }

\new Staff {
  \motif
  \transpose c f \motif
  \modalTranspose c f \chromaticScale \motif
}
```



Modale Umkehrung

Ein musikalischer Ausdruck kann innerhalb einer definierten Skala um eine Referenznote umgekehrt werden und transponiert werden:
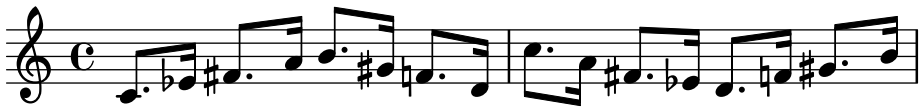
`\modalInversion` *Umkehrungstonhöhe nachTonhöhe Skala mus. Ausdruck*

Die Noten von *mus. Ausdruck* werden mit den gleichen Intervallabständen voneinander gesetzt, ausgehend von *Umkehrungstonhöhe*, allerdings in die entgegengesetzte Richtung. Das Result wird dann innerhalb der *Skala* um das Intervall transponiert, das sich aus dem Unterschied von *Umkehrungstonhöhe* und *nachTonhöhe* ergibt.

Um also einfach eeine Umkehrung ohne zusätzliche Transposition zu machen, sollten *Umkehrungstonhöhe* und *nachTonhöhe* gleich sein.

```
octatonicScale = \relative { ees' f fis gis a b c d }
motif = \relative { c'8. ees16 fis8. a16 b8. gis16 f8. d16 }
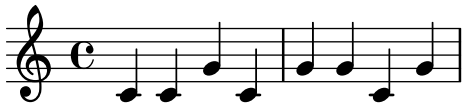
\new Staff {
  \motif
  \modalInversion fis' fis' \octatonicScale \motif
}
```



Um den Drehpunkt der Umkehrung zwischen zwei Noten der Skala zu definieren, wird um eine der Noten die Umkehrung vorgenommen und dann eine Skalenstufe transponiert. Die beiden Noten klammern sozusagen den Drehpunkt ein.

```
scale = \relative { c' g' }
motive = \relative { c' c g' c, }
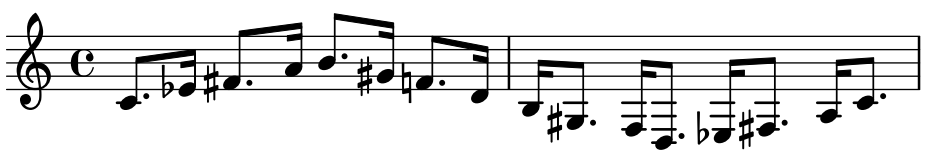
\new Staff {
  \motive
  \modalInversion c' g' \scale \motive
}
```



Die kombinierte Operation von Umkehrung und Krebs erzeugt die Krebsumkehrung:

```
octatonicScale = \relative { ees' f fis gis a b c d }
motif = \relative { c'8. ees16 fis8. a16 b8. gis16 f8. d16 }

\new Staff {
  \motif
  \retrograde \modalInversion c' c' \octatonicScale \motif
}
```



Siehe auch

Notationsreferenz: Abschnitt 1.2.3 [Umkehrung], Seite 15, Abschnitt 1.2.4 [Krebs], Seite 16, Abschnitt 1.2.2 [Transponieren], Seite 12.
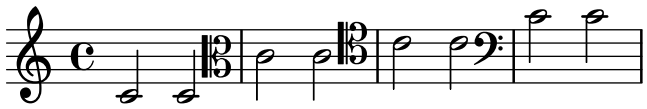
1.3 Tonhöhen anzeigen lassen

Dieser Abschnitt zeigt, wie die Ausgabe von Tonhöhen verändern werden kann.

1.3.1 Notenschlüssel

Der Schlüssel kann verändert werden. Das eingestrichene C wird in jedem Beispiel gezeigt:

```
\clef treble
c'2 c'
\clef alto
c'2 c'
\clef tenor
c'2 c'
\clef bass
c'2 c'
```



Andere Schlüssel sind u. A.:

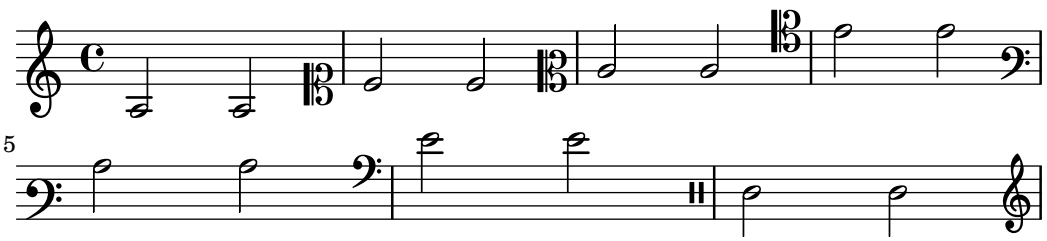
```
\clef french
c2 c
\clef soprano
c2 c
\clef mezzosoprano
c2 c
\clef baritone
c2 c
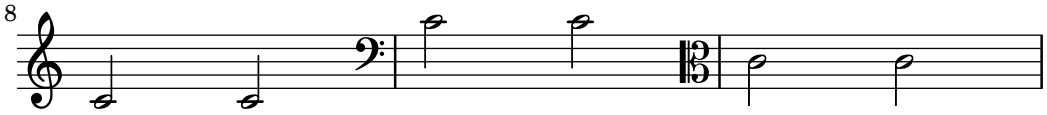
\break

\clef varbaritone
c2 c
\clef subbass
c2 c
\clef percussion
c2 c

\break
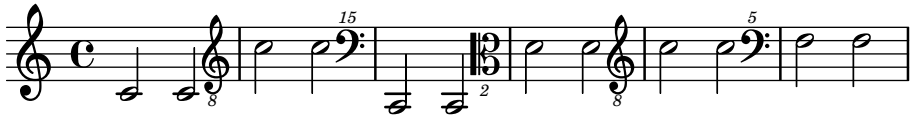
\clef G   % synonym for treble
c2 c
\clef F   % synonym for bass
c2 c
\clef C   % synonym for alto
c2 c
```





Indem `_8` oder `^8` an die jeweilige Schlüsselbezeichnung angehängt wird, wird der Schlüssel um eine Oktave nach oben oder unten transponiert, mit `_15` oder `^15` um zwei Oktaven. Auch andere Ganzzahlen können verwendet werden, wenn es gewünscht wird. Die Schlüsselbezeichnung muss in Anführungszeichen gesetzt werden, wenn nicht-alphabetische Zeichen enthält, siehe Beispiel:

```
\clef treble
c'2 c'
\clef "treble_8"
c'2 c'
\clef "bass^15"
c'2 c'
\clef "alto_2"
c'2 c'
\clef "G_8"
c'2 c'
\clef "F^5"
c'2 c'
```



Weitere unterstützte Schlüssel sind beschrieben in Abschnitt 17.3.2 [Mensurale Schlüssel], Seite 424, Abschnitt 17.4.2 [Gregorianische Schlüssel], Seite 432, Abschnitt 12.1.3 [Standardtabulaturen], Seite 330, und Abschnitt 12.1.4 [Angepasste Tabulaturen], Seite 343. Zur Benutzung unterschiedlicher Schlüssel bei Stichnoten siehe die Funktionen `\cueClef` und `\cueDuringWithClef` in

Ausgewählte Schnipsel

Eigenschaften des Schlüssels optimieren

Der Befehl `\clef "treble_8"` ist gleichbedeutend mit einem expliziten Setzen der Eigenschaften von `clefGlyph`, `clefPosition` (welche die vertikale Position des Schlüssels bestimmt), `middleCPosition` und `clefTransposition`. Ein Schlüssel wird ausgegeben, wenn eine der Eigenschaften außer `middleCPosition` sich ändert.

Eine Änderung des Schriftzeichens (Glyph), der Schlüsselposition oder der Oktavierung selber ändert noch nicht die Position der darauf folgenden Noten auf dem System: das geschieht nur, wenn auch die Position des eingestrichenen C (middleCPosition) angegeben wird. Die Positionsparameter sind relativ zur Mittellinie des Systems, dabei versetzen positive Zahlen die Position nach oben, jeweils eine Zahl für jede Linie plus Zwischenraum. Der `clefTransposition`-Wert ist normalerweise auf 7, -7, 15 oder -15 gesetzt, aber auch andere Werte sind gültig.

Wenn ein Schlüsselwechsel an einem Zeilenwechsel geschieht, wird das neue Symbol sowohl am Ende der alten Zeilen als auch am Anfang der neuen Zeile ausgegeben. Wenn der Warnungs-Schlüssel am Ende der alten Zeile nicht erforderlich ist, kann er unterdrückt werden, indem die `explicitClefVisibility`-Eigenschaft des `Staff`-Kontextes auf den Wert `end-of-line-invisible` gesetzt wird. Das Standardverhalten kann mit `\unset Staff.explicitClefVisibility` wieder hergestellt werden.

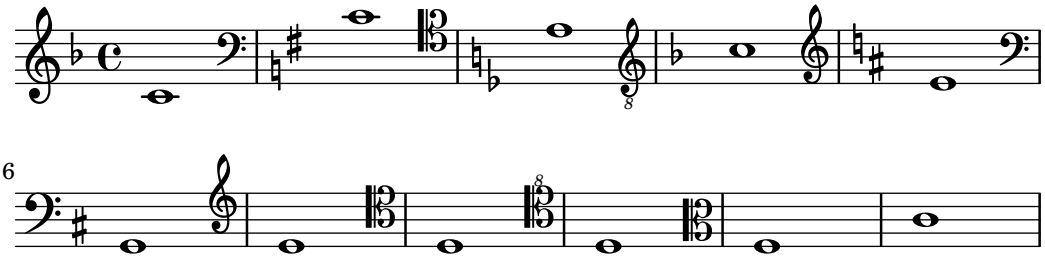
Die folgenden Beispiele zeigen die Möglichkeiten, wenn man diese Eigenschaften manuell setzt. Auf der ersten Zeile erhalten die manuellen Änderungen die ursprüngliche relative Positionierung von Schlüssel und Noten, auf der zweiten Zeile nicht.

```
{
  % The default treble clef.
  \key f \major
  c'1
  % The standard bass clef
  \set Staff.clefGlyph = "clefs.F"
  \set Staff.clefPosition = 2
  \set Staff.middleCPosition = 6
  \set Staff.middleCClefPosition = 6
  \key g \major
  c'1
  % The baritone clef.
  \set Staff.clefGlyph = "clefs.C"
  \set Staff.clefPosition = 4
  \set Staff.middleCPosition = 4
  \set Staff.middleCClefPosition = 4
  \key f \major
  c'1
  % The standard choral tenor clef.
  \set Staff.clefGlyph = "clefs.G"
  \set Staff.clefPosition = -2
  \set Staff.clefTransposition = -7
  \set Staff.middleCPosition = 1
  \set Staff.middleCClefPosition = 1
  \key f \major
  c'1
  % A non-standard clef.
  \set Staff.clefPosition = 0
  \set Staff.clefTransposition = 0
  \set Staff.middleCPosition = -4
  \set Staff.middleCClefPosition = -4
  \key g \major
  c'1 \break

  % The following clef changes do not preserve
  % the normal relationship between notes, key signatures
  % and clefs.
  \set Staff.clefGlyph = "clefs.F"
  \set Staff.clefPosition = 2
  c'1
  \set Staff.clefGlyph = "clefs.G"
  c'1
  \set Staff.clefGlyph = "clefs.C"
  c'1
  \set Staff.clefTransposition = 7
  c'1
  \set Staff.clefTransposition = 0
  \set Staff.clefPosition = 0
```

```
  c'1

  % Return to the normal clef.
  \set Staff.middleCPosition = 0
  c'1
}
```



Siehe auch

Notationsreferenz: Abschnitt 17.3.2 [Mensurale Schlüssel], Seite 424, Abschnitt 17.4.2 [Gregorianische Schlüssel], Seite 432, Abschnitt 12.1.3 [Standardtabulaturen], Seite 330, Abschnitt 12.1.4 [Angepasste Tabulaturen], Seite 343, Abschnitt 6.3.3 [Stichnoten formatieren], Seite 208.

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

Referenz der Interna: Abschnitt “Clef_engraver” in *Referenz der Interna*, Abschnitt “Clef” in *Referenz der Interna*, Abschnitt “ClefModifier” in *Referenz der Interna*, Abschnitt “clef-interface” in *Referenz der Interna*.

1.3.2 Tonartbezeichnung

> **Achtung:** Neue Benutzer sind manchmal verwirrt, wie Versetzungszeichen und Vorzeichen/Tonarten funktionieren. In LilyPond sind Notenbezeichnungen die wirkliche Tonhöhe, erst durch Vorzeichen wird bestimmt, wie diese Tonhöhe dann im Notenbild dargestellt wird. Eine einfache Tonhöhe wie etwa `c` bedeutet also immer das eingestrichene C ohne Versetzungszeichen, egal was für Vorzeichen/Tonart oder Schlüssel gesetzt sind. Mehr Information dazu in Abschnitt “Tonhöhen und Tonartbezeichnungen (Vorzeichen)” in *Handbuch zum Lernen*.
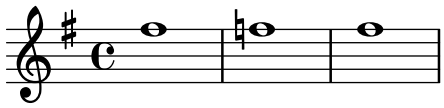
Die Vorzeichen zeigen die Tonart an, in welcher ein Stück notiert ist. Es handelt sich um eine Anzahl von Alterationszeichen (Kreuzen oder Bs) am Beginn jedes Notensystems.

Die Tonart kann geändert werden:

```
\key Tonhöhe Modus
```

Der Wert *Modus* sollte entweder `\major` oder `\minor` sein, um Moll oder Dur der *Tonhöhe* zu erhalten. Es können auch Modusbezeichnungen für Kirchentonarten verwendet werden: `\ionian` (Ionisch), `\locrian` (Lokrisch), `\aeolian` (Aeolisch), `\mixolydian` (Mixolydisch), `\lydian` (Lydisch), `\phrygian` (Phrygisch) und `\dorian` (Dorisch).

```
\relative {
  \key g \major
  fis''1
  f
  fis
}
```
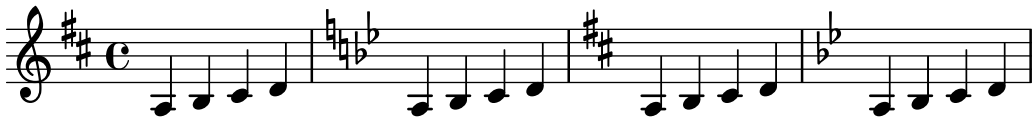


Ausgewählte Schnipsel

Auflösungzeichen nicht setzen, wenn die Tonart wechselt

Wenn die Tonart wechselt, werden automatisch Auflösungszeichen ausgegeben, um Versetzungszeichen der vorherigen Tonart aufzulösen. Das kann verhindert werden, indem die `printKeyCancellation`-Eigenschaft im `Staff`-Kontext auf "false" gesetzt wird.

```
\relative c' {
  \key d \major
  a4 b cis d
  \key g \minor
  a4 bes c d
  \set Staff.printKeyCancellation = ##f
  \key d \major
  a4 b cis d
  \key g \minor
  a4 bes c d
}
```



Untypische Tonarten

Der üblicherweise benutzte `\key`-Befehl setzt die `keySignature`-Eigenschaft im `Staff`-Kontext.

Um untypische Tonartenvorzeichen zu erstellen, muss man diese Eigenschaft direkt setzen. Das Format für den Befehl ist eine Liste: `\set Staff.keySignature = #`(((Oktave . Schritt) . Alteration) ((Oktave . Schritt) . Alteration) ...)` wobei für jedes Element in der Liste `Oktave` die Oktave angibt (0 ist die Oktave vom eingestrichenen C bis zum eingestrichenen H), `Schritt` gibt die Note innerhalb der Oktave an (0 heißt C und 6 heißt H), und `Alteration` ist `,SHARP` `,FLAT` `,DOUBLE-SHARP` usw. (Beachte das beginnende Komma.)
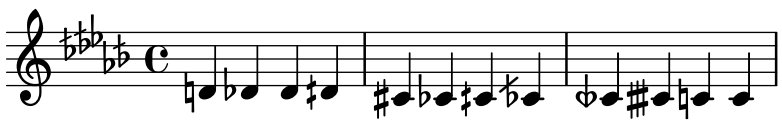
Alternativ kann auch jedes Element der Liste mit dem allgemeineren Format `(Schritt . Alteration)` gesetzt werden, wobei dann die Einstellungen für alle Oktaven gelten.

Hier ein Beispiel einer möglichen Tonart für eine Ganztonleiter:

```
\include "arabic.ly"

\relative do' {
  \set Staff.keyAlterations = #`((0 . ,SEMI-FLAT)
                                 (1 . ,SEMI-FLAT)
                                 (2 . ,FLAT)
                                 (5 . ,FLAT)
```

```
                                (6 . ,SEMI-FLAT))
  % \set Staff.extraNatural = ##f
  re reb \dwn reb resd
  dod dob dosd \dwn dob |
  dobsb dodsd do do |
}
```



Siehe auch

Glossar: Abschnitt “church mode” in *Glossar*, Abschnitt “scordatura” in *Glossar*.

Handbuch zum Lernen: Abschnitt “Tonhöhen und Tonartbezeichnungen (Vorzeichen)” in *Handbuch zum Lernen*.

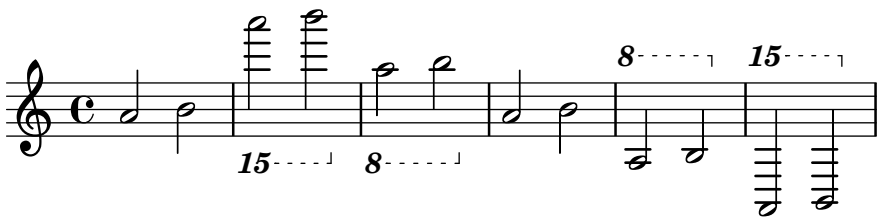
Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

Referenz der Interna: Abschnitt “KeyChangeEvent” in *Referenz der Interna*, Abschnitt “Key_engraver” in *Referenz der Interna*, Abschnitt “Key_performer” in *Referenz der Interna*, Abschnitt “KeyCancellation” in *Referenz der Interna*, Abschnitt “KeySignature” in *Referenz der Interna*, Abschnitt “key-cancellation-interface” in *Referenz der Interna*, Abschnitt “key-signature-interface” in *Referenz der Interna*.

1.3.3 Oktavierungsklammern

Oktavierungsklammern zeigen eine zusätzliche Transposition von einer Oktave an:

```
\relative a' {
  a2 b
  \ottava #-2
  a2 b
  \ottava #-1
  a2 b
  \ottava #0
  a2 b
  \ottava #1
  a2 b
  \ottava #2
  a2 b
}
```



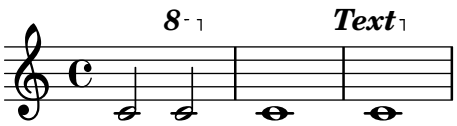
Ausgewählte Schnipsel

Changing ottava text

Internally, `\ottava` sets the properties `ottavation` (for example, to `8va` or `8vb`) and `middleCPosition`. To override the text of the bracket, set `ottavation` after invoking `\ottava`.

Short text is especially useful when a brief ottava is used.

```
{
  c'2
  \ottava 1
  \set Staff.ottavation = "8"
  c''2
  \ottava 0
  c'1
  \ottava 1
  \set Staff.ottavation = "Text"
  c''1
}
```



Siehe auch

Glossar: Abschnitt “octavation” in *Glossar*.

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

Referenz der Interna: Abschnitt “Ottava_spanner_engraver” in *Referenz der Interna*, Abschnitt “OttavaBracket” in *Referenz der Interna*, Abschnitt “ottava-bracket-interface” in *Referenz der Interna*.

1.3.4 Transposition von Instrumenten

Wenn man Noten setzt, die von transponierenden Instrumenten gespielt werden, sind oft einige Stimmen auf einer anderen Tonhöhe notiert als dem Kammerton. In diesem Fall muss die Tonart des transponierenden Instruments gekennzeichnet werden, weil sonst die MIDI-Ausgabe und Stichnoten in anderen Stimmen falsche Tonhöhen produzieren. Mehr Information zu Stichnoten in Abschnitt 6.3.2 [Andere Stimmen zitieren], Seite 205.

`\transposition` *Tonhöhe*

Die Tonhöhe, die für `\transposition` benutzt wird, muss mit dem wirklichen Ton übereinstimmen, der erklingt, wenn das Instrument ein `c'` in seiner Stimme spielt. Die Tonhöhe wird im absoluten Modus angegeben, ein Instrument also, dass einen Ton höher erklingt als es notiert wird, muss folgenden Befehl benutzen: `\transposition d'`. `\transposition` sollte *nur* dann benutzt werden, wenn sie nicht *nicht* in C notiert werden.
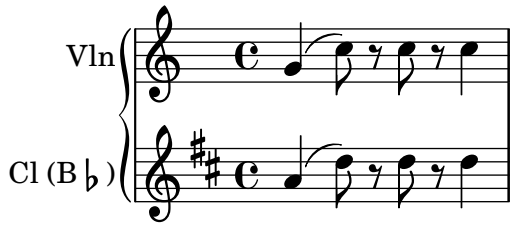
Hier einige Noten für Geige und B-Klarinette: die Stimmen (Noten und Vorzeichen) sind so notiert, wie sie in der Partitur erscheinen. Die zwei Instrumente spielen unisono.

```
\new GrandStaff <<
  \new Staff = "violin" {
    \relative c'' {
      \set Staff.instrumentName = "Vln"
      \set Staff.midiInstrument = "violin"
      % not strictly necessary, but a good reminder
      \transposition c'

      \key c \major
      g4( c8) r c r c4
    }
  }
  \new Staff = "clarinet" {
```

```
    \relative c'' {
      \set Staff.instrumentName = \markup { Cl (B\flat) }
      \set Staff.midiInstrument = "clarinet"
      \transposition bes

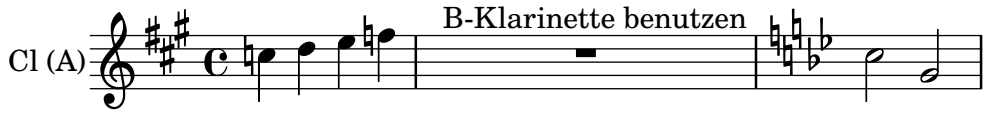
      \key d \major
      a4( d8) r d r d4
    }
  }
>>
```



Die \transposition kann während eines Stückes geändert werden. Ein Klarinettist zum Beispiel kann zwischen B- und A-Klarinette wechseln.

```
\relative c'' {
  \set Staff.instrumentName = "Cl (A)"
  \key a \major
  \transposition a
  c d e f
  \textLengthOn
  <>^\markup { B-Klarinette benutzen }
  R1

  \key bes \major
  \transposition bes
  c2 g
}
```



Siehe auch

Glossar: Abschnitt “concert pitch” in *Glossar*, Abschnitt “transposing instrument” in *Glossar*.

Notationsreferenz: Abschnitt 6.3.2 [Andere Stimmen zitieren], Seite 205, Abschnitt 1.2.2 [Transponieren], Seite 12.

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

1.3.5 Automatische Versetzungszeichen

Es gibt viele unterschiedliche Regeln, wie Versetzungszeichen notiert werden. LilyPond hat eine Funktion, mit der spezifiziert werden kann, welcher Stil benutzt werden soll. Diese Funktion kann man wie folgt benutzen:

```
\new Staff <<
  \accidentalStyle voice
```

```
  { ... }
>>
```

Der Versetzungszeichenstil bezieht sich auf das aktuelle Notensystem in der Standardeinstellung (eine Ausnahme bilden die Stile `piano` und `piano-cautionary`, die weiter unten erklärt werden). Die Funktion kann aber auch ein zweites Argument erhalten, mit der spezifiziert wird, auf welchen Bereich sich der neue Stil erstreckt. Um etwa den neuen Stil in allen Systemen einer Stimmgruppe (`StaffGroup`) zu benutzen, müsste der Befehl so aussehen:

```
\accidentalStyle StaffGroup.voice
```

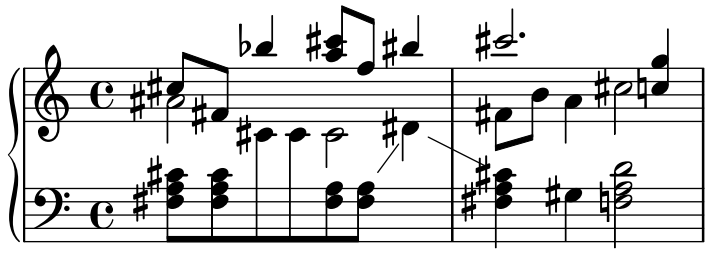
Folgende Versetzungszeichenstile sind unterstützt. Um jeden Stil zu erklären, wird folgendes Beispiel benützt:

```
musicA = {
  <<
    \relative {
      cis''8 fis, bes'4 <a cis>8 f bis4 |
      cis2. <c, g'>4 |
    }
    \\
    \relative {
      ais'2 cis, |
      fis8 b a4 cis2 |
    }
  >>
}

musicB = {
  \clef bass
  \new Voice {
    \voiceTwo \relative {
      <fis a cis>8[ <fis a cis>
      \change Staff = up
      cis' cis
      \change Staff = down
      <fis, a> <fis a>]
      \showStaffSwitch
      \change Staff = up
      dis'4 |
      \change Staff = down
      <fis, a cis>4 gis <f a d>2 |
    }
  }
}

\new PianoStaff {
  <<
    \context Staff = "up" {
      \accidentalStyle default
      \musicA
    }
    \context Staff = "down" {
      \accidentalStyle default
      \musicB
```

```
    }
  >>
}
```

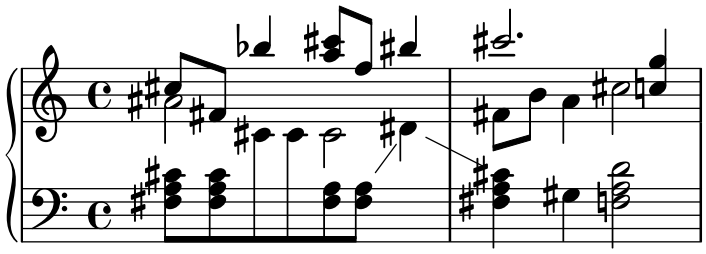


Die letzten Zeilen des Beispiels könnten auch mit folgendem Code ersetzt werden, solange der gleiche Versetzungszeichenstil in beiden Systemen benutzt werden soll:

```
\new PianoStaff {
  <<
    \context Staff = "up" {
      %%% nächste Zeile wie gewünscht ändern:
      \accidentalStyle Score.default
      \musicA
    }
    \context Staff = "down" {
      \musicB
    }
  >>
}
```

`default (Standard)`

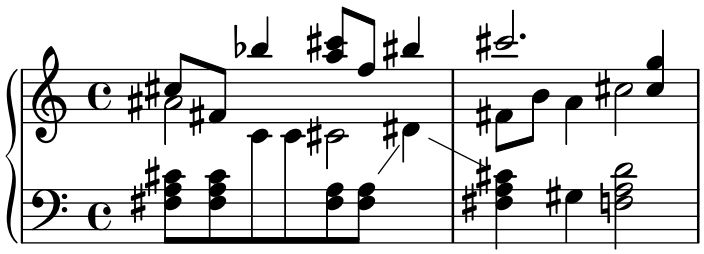
Das ist das Standardverhalten. Es entspricht der Konvention für Notation von Musik des 18. Jahrhunderts: Versetzungszeichen werden bis zum Taktende erinnert, in dem sie gesetzt wurden, und nur in ihrer eigenen Oktave. Im nächsten Beispiel wird also kein Auslösungszeichen vor dem `b` (H) im zweiten Takt oder dem letzen `c` gesetzt:



`voice (Stimme)`

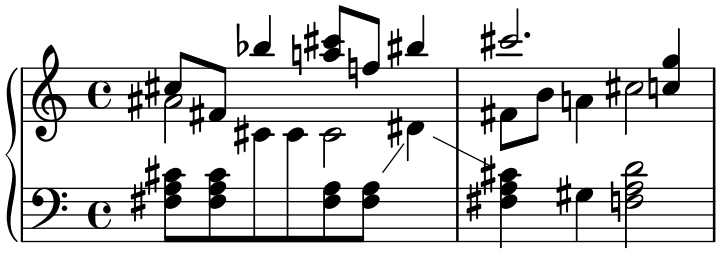
Das normale Verhalten ist es, die Versetzungszeichen auf der Notensystemebene zu erinnern. In diesem Stil aber werden Versetzungszeichen individuell für jede Stimme errechnet. Abgesehen davon gelten die Regeln des Standardstiles (`default`).

Das hat zur Folge, dass Versetzungszeichen von einer Stimme in der anderen nicht aufgelöst werden, was oft ein unerwünschtes Ergebnis ist: im folgenden Beispiel kann man schwer sagen, ob das zweite `a` unalteriert oder erhöht gespielt werden soll. Die `voice`-Option sollte also nur benutzt werden, wenn die Stimmen separat von unterschiedlichen Musikern gelesen werden. Wenn das System nur von einem Musiker benutzt wird (etwa der Dirigent oder ein Klavierspieler), dann sind die Stile `modern` oder `modern-cautionary` besser.



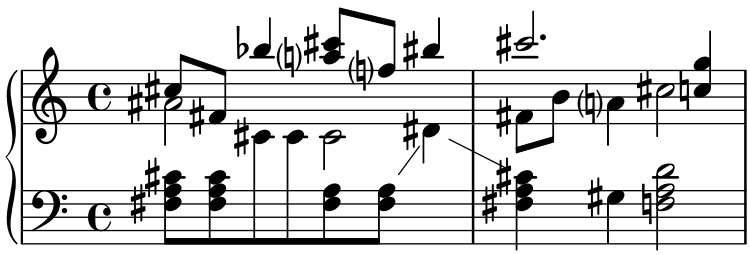
`modern (Modern)`

Dieser Stil orientiert sich an den üblichen Regeln für das 20. Jahrhundert. Hierbei werden einige zusätzliche Auflösungszeichen ausgelassen, die traditionell für ein Kreuz nach einem Doppelkreuz und ein b nach einem Doppel-b gesetzt werden. Die gleichen Versetzungszeichen wie im Standardstil werden gesetzt, allerdings mit zwei Zusätzen, die Uneindeutigkeiten verhindern sollen: nach vorübergehenden Versetzungszeichen werden Auflösungszeichen auch im folgenden Takt gesetzt (für Noten innerhalb der selben Oktave) und im gleichen Takt für Noten in unterschiedlichen Oktaven. Daher kommen also die Auflösungszeichen vor dem H und dem C im zweiten Takt des oberen Systems:



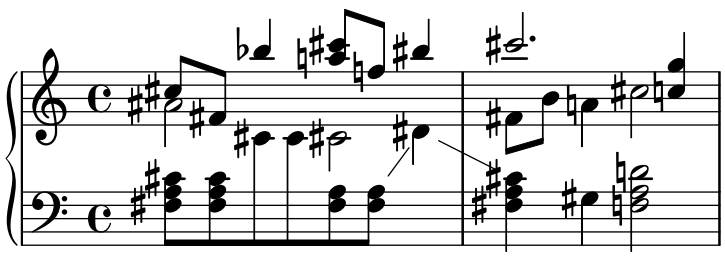
`modern-cautionary (Modern mit Warnungen)`

Dieser Stil ähnelt `modern`, aber die „zusätzlichen“ Versetzungszeichen (die normalerweise nicht gesetzt werden) werden als Warnungen gesetzt. In der Standardeinstellung werden sie in Klammern gesetzt, aber sie können auch in kleinerer Größe gesetzt werden, wenn man die `cautionary-style`-Eigenschaft von `AccidentalSuggestion` definiert.
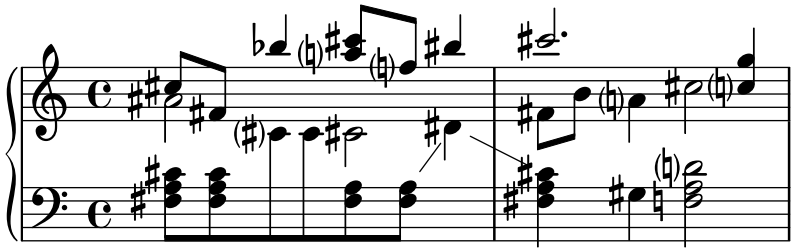


`modern-voice (Modern für Stimmen)`

Diese Regel wird für vielstimmige Noten benutzt, die sowohl von unterschiedlichen Spielern für jede Stimme als auch von einem Spieler für alle Stimmen benutzt. Versetzungszeichen werden für jede Stimme gesetzt, aber sie *werden* über die Stimme hinweg aufgelöst innerhalb des selben Notensystems. Das `a` im letzten Takt ist also aufgelöst, weil die vorigen Auflösung in einer anderen Stimme stattgefunden hatte, und das `d` im unteren System ist aufgelöst wegen eines Versetzungszeichens in einer anderen Stimme im vorigen Takt:



`modern-voice-cautionary` (`modern` mit `Warnungen` für `einzelne` `Stimmen`)

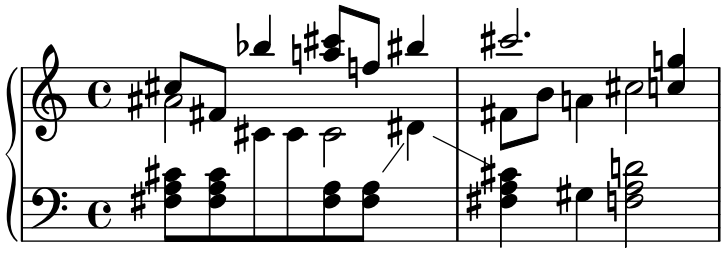
Dieser Stil ist der gleiche wie `modern-voice`, nur dass hier die zusätzlichen Versetzungszeichen (die nicht vom `voice`-Stil gesetzt werden) als Warnungsversetzungszeichen gesetzt werden. Obwohl alle Versetzungszeichen, die mit `default` gesetzt werden, auch mit diesem Stil gesetzt werden, sind manche Warnungsversetzungszeichen.



`piano` (`Klavier`)

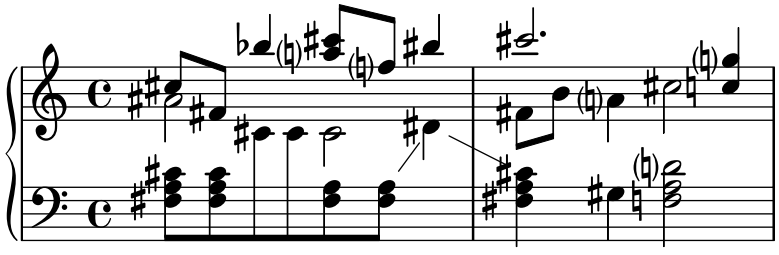
Dieser Stil orientiert sich an den Regeln im 20. Jahrhundert für die Notation von Klaviermusik. Er ist sehr ähnlich mit dem modernen Stil, aber Versetzungszeichen werden auch über Notensysteme hinweg für die selbe Akkolade (`GrandStaff` oder `PianoStaff`) aufgelöst.

Dieser Versetzungszeichenstil wirkt sich standardmäßig auf die gesamte Akkolade (`GrandStaff` oder `PianoStaff`) aus.



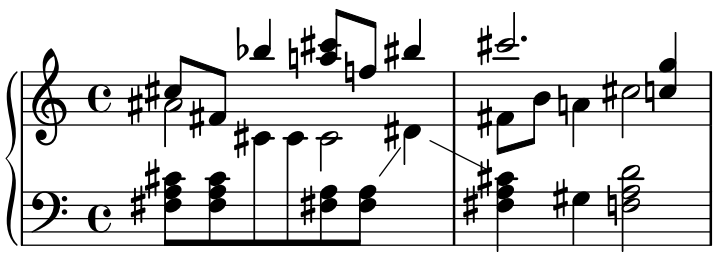
`piano-cautionary` (`Klavier` mit `Warnungen`)

Dieser Stil verhält sich wie `piano`, aber die zusätzlichen Versetzungszeichen werden als Warnungen ausgegeben:

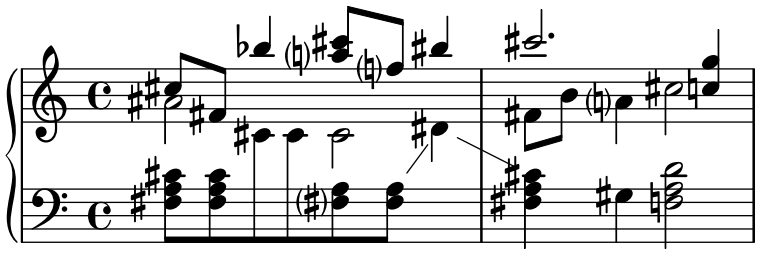


`neo-modern`

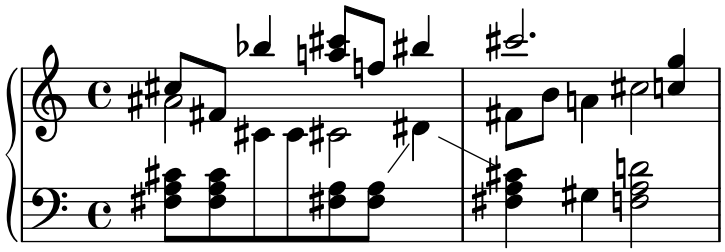
Dieser Stil richtet sich nach den Regeln für moderne Musik: Versetzungszeichen werden mit im `modern`-Stil gesetzt, aber sie werden nochmal gesetzt, wenn die gleiche Note später im selben Takt auftritt – außer die Note wird unmittelbar wiederholt.



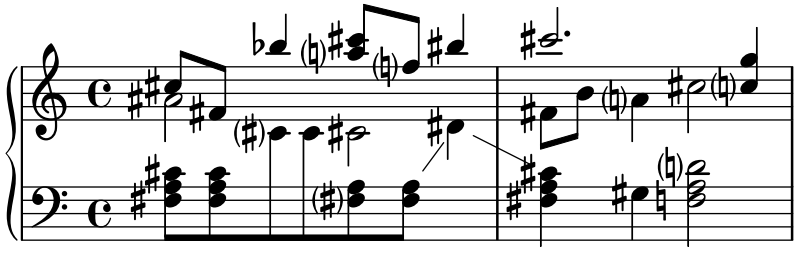
`neo-modern-cautionary` (`neo-modern` mit `Warnungen`)
: Dieser Stil ähnelt `neo-modern`, aber die zusätzlichen Versetzungszeichen werden als Warnungen gesetzt.



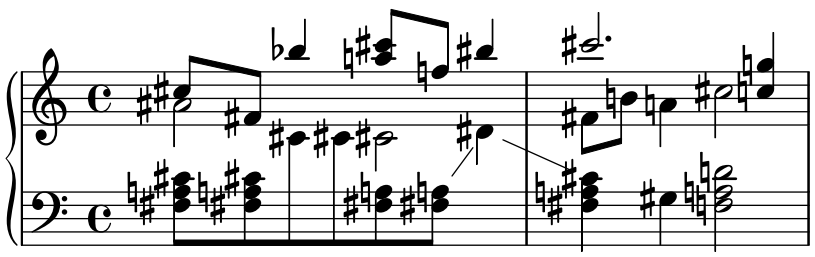
`neo-modern-voice` (`neo-modern` für `Stimmen`)
: Diese Regel wird für für Versetzungszeichen in mehreren Stimmen eingesetzt, wenn die Noten sowohl von Musikern gelesen werden, die nur eine Stimme lesen, als auch von Musikern, die alle Stimmen lesen. Versetzungszeichen werden für jede Stimme so wie mit der `neo-modern`-Regel gesetzt, aber innerhalb des gesamten Notensystems mit Auflösungszeichen versehen.



`neo-modern-voice-cautionary`
: Diese Regel ähnelt `neo-modern-voice`, aber die zusätzlichen Versetzungszeichen werden hier als warnende Versetzungszeichen gesetzt.
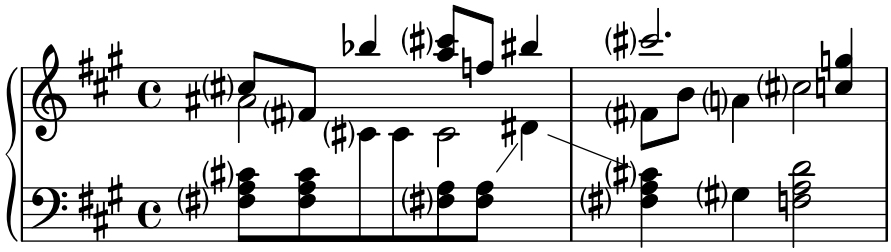


`dodecaphonic` (`Zwölftonmusik`)
: Dieser Stil orientiert sich an der Notation von sog. Zwölftonmusik, der Stil wurde Anfang des 20. Jahrhunderts in Gebrauch genommen. In diesem Stil erhält *jede* Note ein Versetzungszeichen, wozu auch Auflösungszeichen zählen.



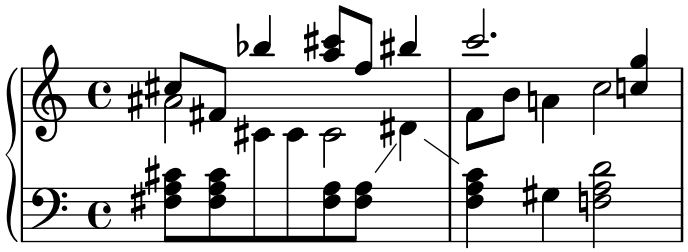
`teaching (didaktisch)`

Dieser Stil ist für Lernende bestimmt: der Stil orientiert sich am `modern`-Stil, aber die Alterationen, die sich durch die Tonart ergeben, werden zusätzlich als Warnungsversetzungszeichen gesetzt. Eine Ausnahme sind direkt wiederholte Noten.



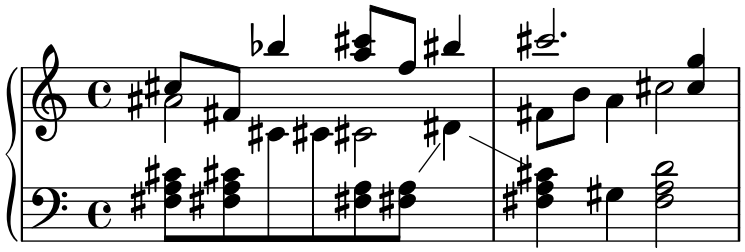
`no-reset (nicht zurücksetzen)`

Das ist der gleiche Stil wie `default`, aber die Versetzungszeichen dauern für „immer“ an, nicht nur im aktuellen Takt:



`forget (vergessen)`

Das ist das Gegenteil von `no-reset`: Versetzungszeichen werden überhaupt nicht erinnert und folgerichtig werden alle Versetzungszeichen entsprechend der Tonart gesetzt, unabhängig vom Kontext der Noten.



Siehe auch

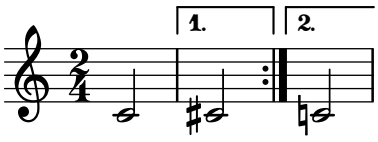
Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

Referenz der Interna: Abschnitt “Accidental” in *Referenz der Interna*, Abschnitt “Accidental_engraver” in *Referenz der Interna*, Abschnitt “GrandStaff” in *Referenz der Interna*, Abschnitt “PianoStaff” in *Referenz der Interna*, Abschnitt “Staff” in *Referenz der Interna*, Abschnitt “AccidentalSuggestion” in *Referenz der Interna*, Abschnitt “AccidentalPlacement” in *Referenz der Interna*, Abschnitt “accidental-suggestion-interface” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

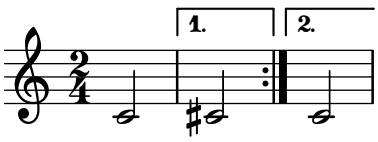
Gleichzeitig erklingende Noten werden bei der automatischen Bestimmung der Versetzungszeichen nicht berücksichtigt: nur die vorige Note und die Vorzeichen werden einbezogen. Man muss die Versetzungszeichen mit `!` oder `?` schreiben, wenn gleichzeitig unterschiedliche Alterationen vorkommen, wie etwa für '`<f! fis!>`'.

Die warndenden Auflösungzeichen werden gesetzt, indem die vorangegangenen Takte betrachtet werden. In der zweiten oder einer weiteren Wiederholungsklammer erwartet man jedoch, dass die Auflösungszeichen sich aus dem letzten *gespielten* und nicht dem letzten *gesetzten* Takt ergeben. Im folgenden Beispiel bräuchte das `c` in der zweiten Klammer kein Auflösungszeichen:



Die folgende Notlösung kann benutzt werden: Man definiert eine Funktion, die den Versetzungszeichenstil kurzzeitig auf `forget` umschaltet:

```
forget = #(define-music-function (music) (ly:music?) #{
  \accidentalStyle forget
  #music
  \accidentalStyle modern
#})
{
  \accidentalStyle modern
  \time 2/4
  \repeat volta 2 {
    c'2
  }
  \alternative {
     cis'
     \forget c'
  }
}
```
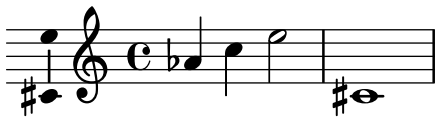


1.3.6 Tonumfang

Der Begriff *ambitus* (Pl. ambitus) beschreibt den Stimmumfang einer Stimme. Er kann auch die Töne bedeuten, die ein Musikinstrument zu spielen in der Lage ist. Ambitus werden in Chorpartituren gesetzt, damit die Sänger schnell wissen, ob sie die Stimme meistern können.

Ambitus werden zu Beginn des Stückes nahe des ersten Schlüssels notiert. Der Stimmumfang wird durch zwei Notenköpfe dargestellt, die die tiefste und höchste Note der Stimme repräsentieren. Versetzungszeichen werden nur gesetzt, wenn sie nicht durch die Tonart definiert werden.

```
\layout {
  \context {
    \Voice
    \consists Ambitus_engraver
  }
}

\relative {
  aes' c e2
  cis,1
}
```
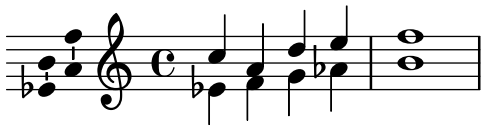


Ausgewählte Schnipsel

Ambitus pro Stimme hinzufügen

Ambitus können pro Stimme gesetzt werden. In diesem Fall müssen sie manuell verschoben werden, um Zusammenstöße zu verhindern.
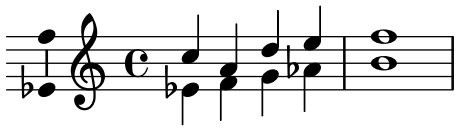
```
\new Staff <<
  \new Voice \with {
    \consists "Ambitus_engraver"
  } \relative c'' {
    \override Ambitus.X-offset = 2.0
    \voiceOne
    c4 a d e
    f1
  }
  \new Voice \with {
    \consists "Ambitus_engraver"
  } \relative c' {
    \voiceTwo
    es4 f g as
    b1
  }
>>
```



Ambitus mit vielen Stimmen

Indem man den `Ambitus_engraver` im `Staff`-Kontext hinzufügt, erhält man einen einzigen Ambitus pro System, auch in dem Fall, dass mehrere Stimmen sich im gleichen System befinden.

```
\new Staff \with {
  \consists "Ambitus_engraver"
  }
<<
  \new Voice \relative c'' {
    \voiceOne
    c4 a d e
    f1
  }
  \new Voice \relative c' {
    \voiceTwo
    es4 f g as
    b1
  }
>>
```



Changing the ambitus gap

It is possible to change the default gap between the ambitus noteheads and the line joining them.

```
\layout {
  \context {
    \Voice
    \consists "Ambitus_engraver"
  }
}

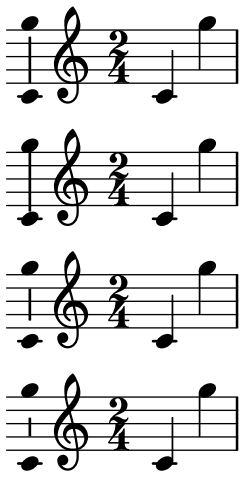
\new Staff {
  \time 2/4
  % Default setting
  c'4 g''
}

\new Staff {
  \time 2/4
  \override AmbitusLine.gap = 0
  c'4 g''
}

\new Staff {
  \time 2/4
  \override AmbitusLine.gap = 1
  c'4 g''
}

\new Staff {
  \time 2/4
  \override AmbitusLine.gap = 1.5
```

```
  c'4 g''
}
```



Siehe auch

Glossar: Abschnitt “ambitus” in *Glossar*.

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

Referenz der Interna: Abschnitt “Ambitus_engraver” in *Referenz der Interna*, Abschnitt “Voice” in *Referenz der Interna*, Abschnitt “Staff” in *Referenz der Interna*, Abschnitt “Ambitus” in *Referenz der Interna*, Abschnitt “AmbitusAccidental” in *Referenz der Interna*, Abschnitt “AmbitusLine” in *Referenz der Interna*, Abschnitt “AmbitusNoteHead” in *Referenz der Interna*, Abschnitt “ambitus-interface” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Es gibt keine Kollisionskontrolle bei mehreren Ambitus in einem System.
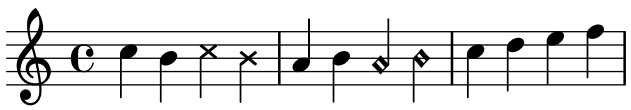
1.4 Notenköpfe

Dieser Abschnitt zeigt, wie man Notenköpfe ändern kann.

1.4.1 Besondere Notenköpfe

Notenköpfe können verändert werden:

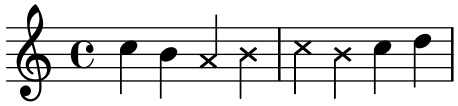
```
\relative c'' {
  c4 b
  \override NoteHead.style = #'cross
  c4 b
  \revert NoteHead.style
  a b
  \override NoteHead.style = #'harmonic
  a b
  \revert NoteHead.style
  c4 d e f
}
```



Für alle Notenkopfstile siehe Abschnitt A.9 [Notenkopfstile], Seite 678,

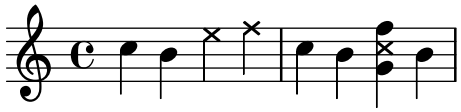
Der Kreuz-(cross) Stil wird mit unterschiedlichen musikalischen Absichten eingesetzt. Die folgenden vordefinierten Befehle verändern die Notenköpfe sowohl in Notensystemen als auch in Tabulaturen und können benutzt werden, um alle musikalischen Bedeutungen zu notieren:

```
\relative {
  c''4 b
  \xNotesOn
   a b c4 b
  \xNotesOff
  c4 d
}
```



Die Form als musikalische Funktion dieses Befehls kann innerhalb und außerhalb von Akkorden benutzt werden, um Notenköpfe mit Kreuzen in normalen und Tabulatursystemen zu erstellen:
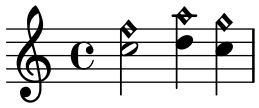
```
\relative {
  c''4 b
  \xNote { e f }
  c b < g \xNote c f > b
}
```



Als Synonym für \xNote, \xNotesOn und \xNotesOff kann \deadNote, \deadNotesOn und \deadNotesOff benutzt werden. Der Begriff *dean note* (engl. tote Note) wird regelmäßig von Gitaristen benutzt.

Es gibt auch einen Kurzbefehl für die Rautenform, der nur innerhalb von Akkorden benutzt werden kann:

```
<c f\harmonic>2 <d a'\harmonic>4 <c g'\harmonic>
```



Vordefinierte Befehle

\harmonic, \xNotesOn, \xNotesOff, \xNote.

Siehe auch

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

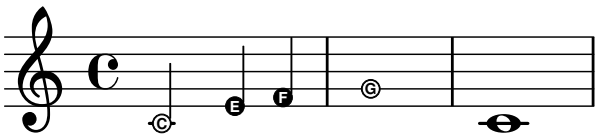
Notationsreferenz: Abschnitt A.9 [Notenkopfstile], Seite 678, Abschnitt 5.1.1 [Noten mit Akkorden], Seite 160, Abschnitt 12.2.2 [Flageolett und gedämpfte Noten], Seite 370.

Referenz der Interna: Abschnitt “note-event” in *Referenz der Interna*, Abschnitt “Note_heads_engraver” in *Referenz der Interna*, Abschnitt “Ledger_line_engraver” in *Referenz der Interna*, Abschnitt “NoteHead” in *Referenz der Interna*, Abschnitt “LedgerLineSpanner” in *Referenz der Interna*, Abschnitt “note-head-interface” in *Referenz der Interna*, Abschnitt “ledger-line-spanner-interface” in *Referenz der Interna*.

1.4.2 Easy-Notation-Notenköpfe

Die „einfachen Notenköpfe“ haben die Bezeichnung der Note im Kopf gedruckt. Das wird eingesetzt, um die Notation beizubringen. Damit die Buchstaben noch lesbar sind, müssen sie sehr groß gesetzt werden. Wie man eine größere Schriftart einstellt, findet sich in Abschnitt 26.2 [Die Notensystemgröße einstellen], Seite 534.

```
#(set-global-staff-size 26)
\relative c' {
  \easyHeadsOn
  c2 e4 f
  g1
  \easyHeadsOff
  c,1
}
```



Vordefinierte Befehle

`\easyHeadsOn`, `\easyHeadsOff`.

Ausgewählte Schnipsel

Numbers as easy note heads

Easy notation note heads use the `note-names` property of the `NoteHead` object to determine what appears inside the note head. By overriding this property, it is possible to print numbers representing the scale-degree.

A simple engraver can be created to do this for every note head object it sees.

```
#(define Ez_numbers_engraver
   (make-engraver
    (acknowledgers
     ((note-head-interface engraver grob source-engraver)
      (let* ((context (ly:translator-context engraver))
             (tonic-pitch (ly:context-property context 'tonic))
             (tonic-name (ly:pitch-notename tonic-pitch))
             (grob-pitch
              (ly:event-property (event-cause grob) 'pitch))
             (grob-name (ly:pitch-notename grob-pitch))
             (delta (modulo (- grob-name tonic-name) 7))
             (note-names
              (make-vector 7 (number->string (1+ delta)))))
        (ly:grob-set-property! grob 'note-names note-names))))))

#(set-global-staff-size 30)

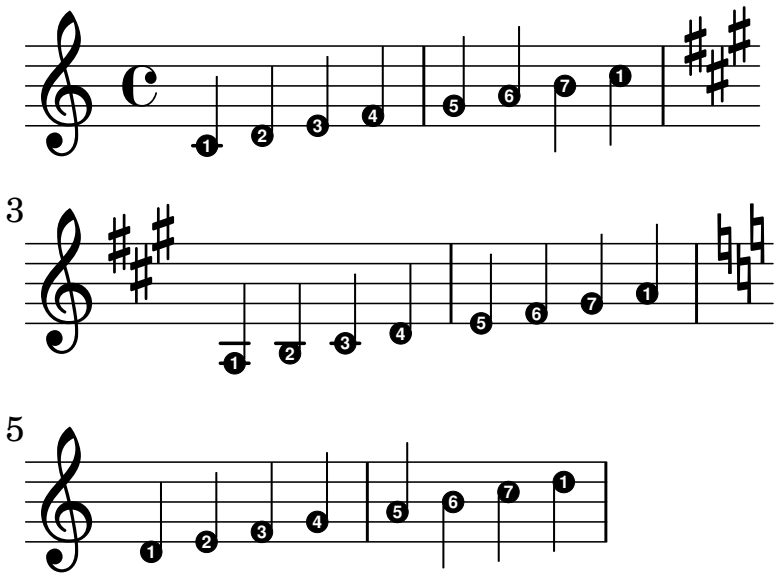
\layout {
  ragged-right = ##t
  \context {
    \Voice
    \consists \Ez_numbers_engraver
```

```
  }
}

\relative c' {
  \easyHeadsOn
  c4 d e f
  g4 a b c \break

  \key a \major
  a,4 b cis d
  e4 fis gis a \break

  \key d \dorian
  d,4 e f g
  a4 b c d
}
```



Siehe auch

Notationsreferenz: Abschnitt 26.2 [Die Notensystemgröße einstellen], Seite 534.
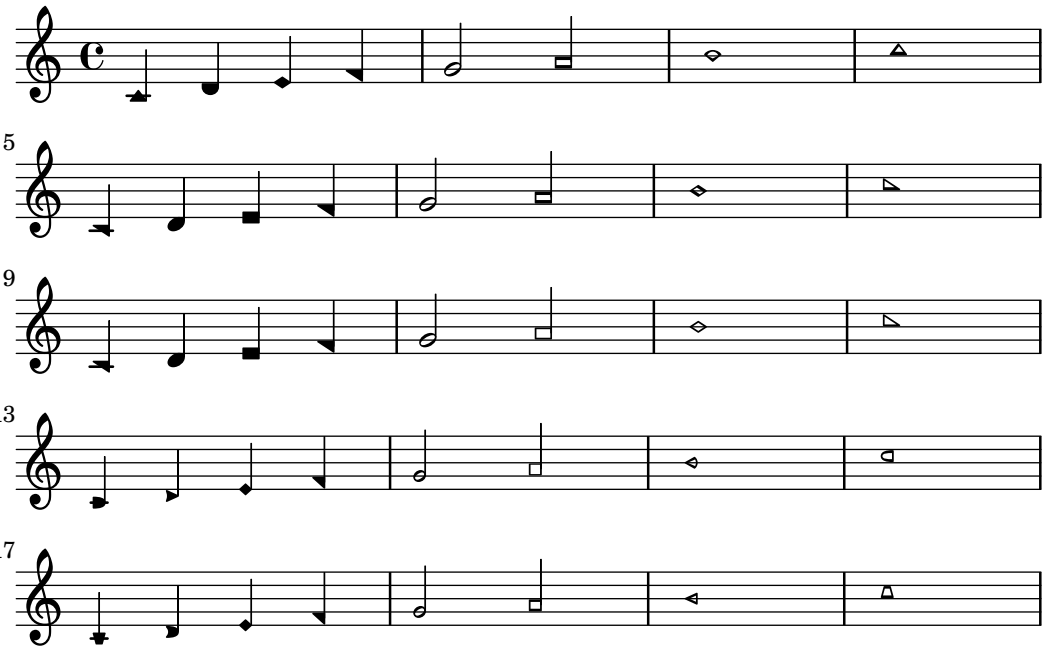
Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

Referenz der Interna: Abschnitt “note-event” in *Referenz der Interna*, Abschnitt “Note_heads_engraver” in *Referenz der Interna*, Abschnitt “NoteHead” in *Referenz der Interna*, Abschnitt “note-head-interface” in *Referenz der Interna*.

1.4.3 Notenköpfe mit besonderen Formen
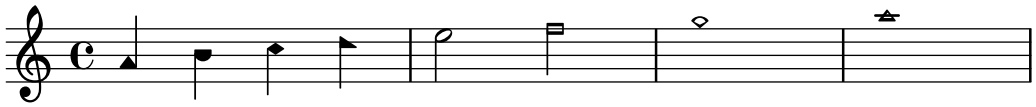
In dieser Notation haben die Notenköpfe eine Form, die ihrer harmonischen Funktion innerhalb der Tonleiter entspricht. Die Notation war sehr beliebt in amerikanischen Liederbüchern des 19. Jahrhunderts. Auf diese Weise können die Formen Sacred Harp, Southern Harmony, Funk (Harmonia Sacra), Walker und Aiken (Christian Harmony) benutzt werden:
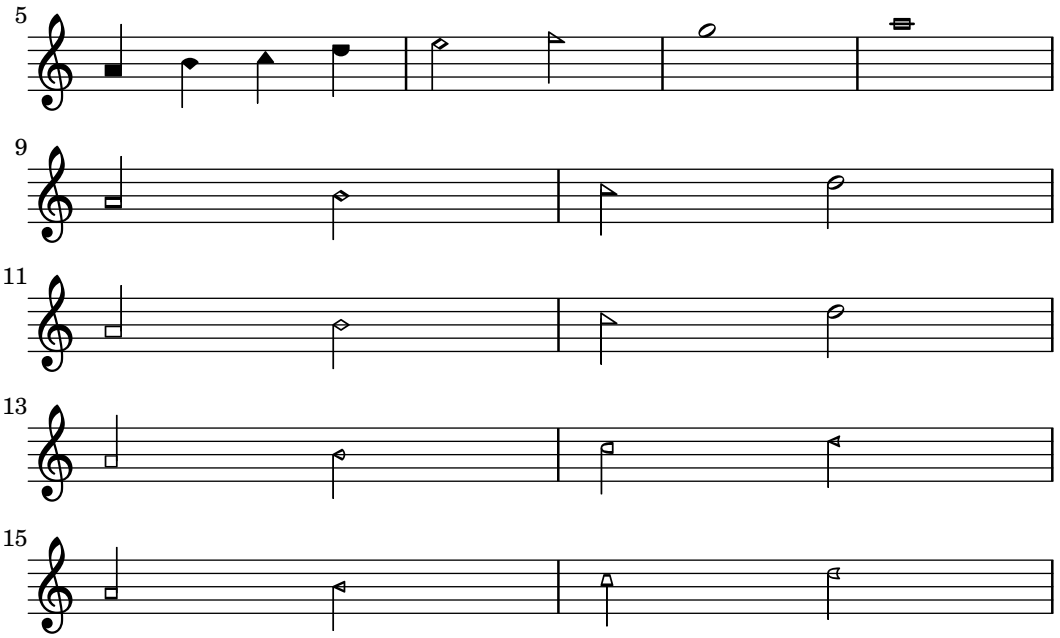
```
\relative c'' {
  \aikenHeads
  c, d e f g2 a b1 c \break
  \sacredHarpHeads
  c,4 d e f g2 a b1 c \break
  \southernHarmonyHeads
  c,4 d e f g2 a b1 c \break
  \funkHeads
  c,4 d e f g2 a b1 c \break
  \walkerHeads
  c,4 d e f g2 a b1 c \break
}
```



Die unterschiedlichen Formen richten sich nach der Stufe in der Skala, wobei der Grundton der Skala aus dem \key-Befehl entnommen wird. Wenn man eine Moll-Skala benutzt, ergibt sich die Form der Notenköpfe aus der parallelen Dur-Tonleiter:

```
\key a \minor
\aikenHeads
a b c d e2 f g1 a \break
\aikenHeadsMinor
a,4 b c d e2 f g1 a \break
\sacredHarpHeadsMinor
a,2 b c d \break
\southernHarmonyHeadsMinor
a2 b c d \break
\funkHeadsMinor
a2 b c d \break
\walkerHeadsMinor
a2 b c d \break
```





Vordefinierte Befehle

`\aikenHeads`, `\aikenHeadsMinor`, `\funkHeads`, `\funkHeadsMinor`, `\sacredHarpHeads`, `\sacredHarpHeadsMinor`, `\southernHarmonyHeads`, `\southernHarmonyHeadsMinor`, `\walkerHeads`, `\walkerHeadsMinor`.

Ausgewählte Schnipsel

Notenkopfstile basierend auf der Tonleiterstufe erstellen

Die `shapeNoteStyles`-(NotenFormenStile)-Eigenschaft kann benutzt werden, um verschiedene Notenstile für jeden Schritt der Tonleiter zu definieren (vorgegeben von der Tonart oder der „tonic“ (Tonika)-Eigenschaft. Diese Eigenschaft braucht eine Anzahl von Symbolen, welche beliebig sein können (geometrische Ausdrücke wie `triangle` (Dreieck), `cross` (Kreuz) und `xcircle` (X-Kreis) sind erlaubt) oder basierend auf einer alten amerikanischen Notensatztradition (einige lateinische Notenbezeichnungen sind auch erlaubt).

Um alte amerikanische Liederbücher zu imitieren, gibt es einige vordefinierte Notenstile wie etwa `\aikenHeads` (im Stil von Aiken) oder `\sacredHarpHeads` (im Stil der Sacred Harp-Tradition).

Dieses Beispiel zeigt, wie man unterschiedlich geformte Noten erhält und eine Melodie transponieren kann, ohne dass das Verhältnis zwischen den harmonischen Funktionen und dem Notenstil verloren geht.

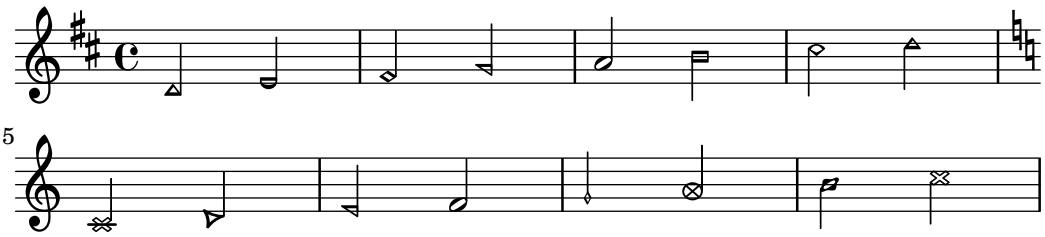
```
fragment = {
  \key c \major
  c2 d
  e2 f
  g2 a
  b2 c
}

\new Staff {
  \transpose c d
  \relative c' {
    \set shapeNoteStyles = ##(do re mi fa
```

```
                                #f la ti)
    \fragment
  }

  \break

  \relative c' {
    \set shapeNoteStyles = ##(cross triangle fa #f
                              mensural xcircle diamond)
    \fragment
  }
}
```



Alle Notenkopfstile finden sich in Abschnitt A.9 [Notenkopfstile], Seite 678.

Siehe auch

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

Notationsreferenz: Abschnitt A.9 [Notenkopfstile], Seite 678.

Referenz der Interna: Abschnitt “note-event” in *Referenz der Interna*, Abschnitt “Note_heads_engraver” in *Referenz der Interna*, Abschnitt “NoteHead” in *Referenz der Interna*, Abschnitt “note-head-interface” in *Referenz der Interna*.

1.4.4 Improvisation

Improvisation wird manchmal angezeigt, indem schräge Notenköpfe gesetzt werden, wenn der Spieler eine beliebige Tonhöhe wählen kann aber den vorgegebenen Rhythmus spielen soll. Sie können wie folgt benutzt werden:

```
\new Voice \with {
  \consists Pitch_squash_engraver
} \relative {
  e''8 e g a a16( bes) a8 g
  \improvisationOn
  e8 ~
  2 ~ 8 f4 f8 ~
  2
  \improvisationOff
  a16( bes) a8 g e
}
```



Vordefinierte Befehle
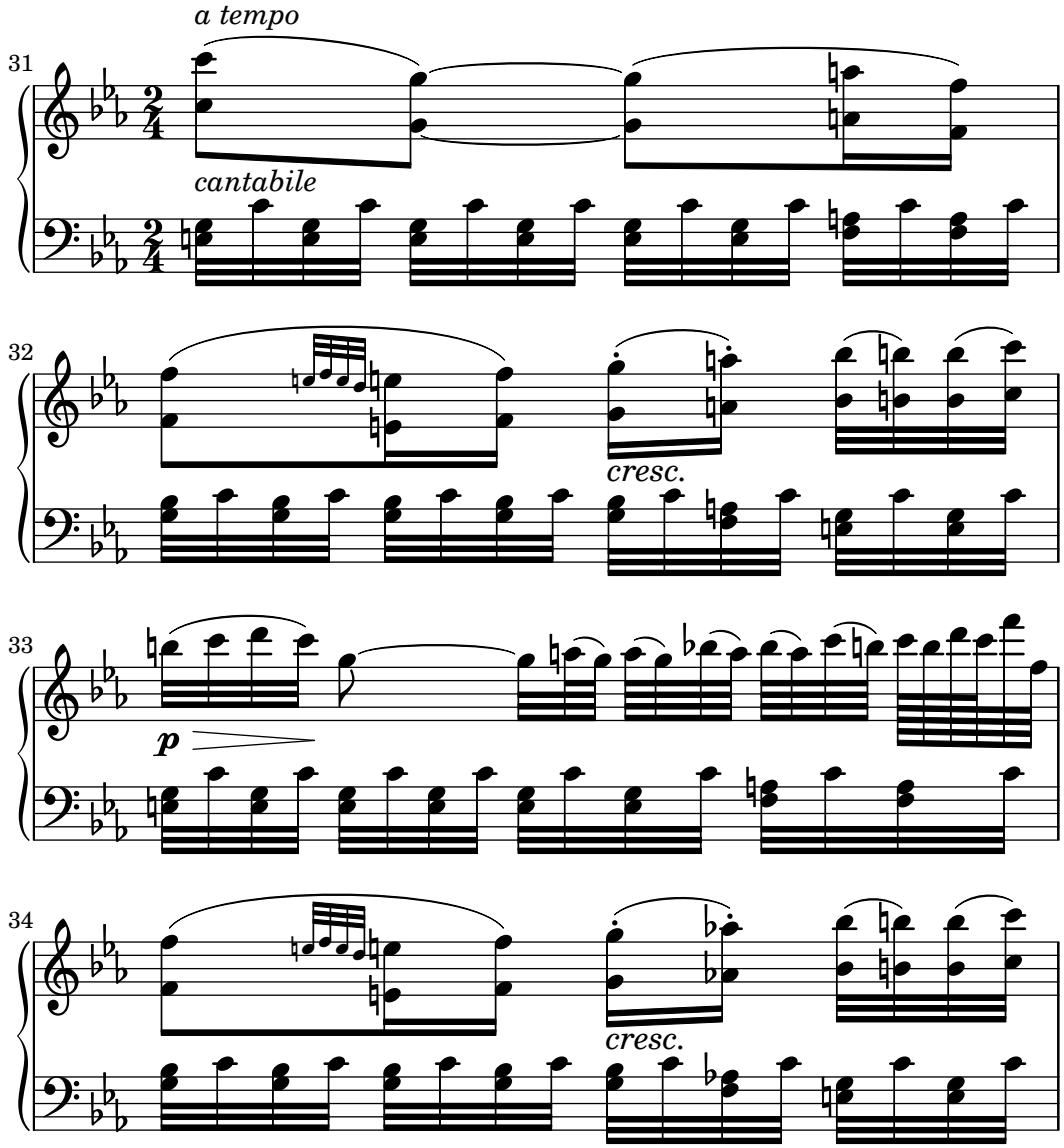
`\improvisationOn`, `\improvisationOff`.

Siehe auch

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

Referenz der Interna: Abschnitt “Pitch_squash_engraver” in *Referenz der Interna*, Abschnitt “Voice” in *Referenz der Interna*, Abschnitt “RhythmicStaff” in *Referenz der Interna*.

2 Rhythmus
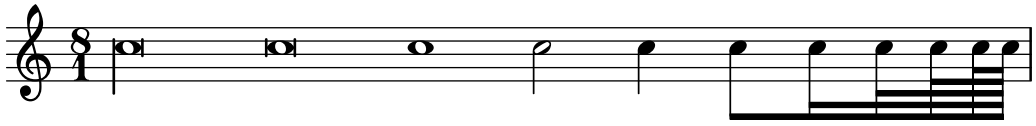


Dieser Abschnitt erklärt die Eingabe von Rhythmen, Pausen, Dauern, Bebalkung und Takten.

2.1 Rhythmen eingeben

2.1.1 Tondauern
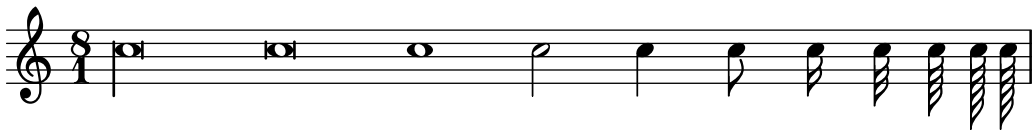
Notenlängen (Dauern) werden durch Zahlen und Punkte notiert: Dauern werden als reziproke Werte geschrieben. Zum Beispiel wird eine Viertelnote mit `4` notiert (weil sie eine 1/4-Note ist), eine halbe Note mit `2` (weil sie eine 1/2-Note ist). Noten, die länger als eine Ganze sind, müssen mit `\longa` (für die Longa, also vier Ganze) und `\breve` (für die Brevis, auch Doppelganze genannt) notiert werden. Notendauern bis hin zu 128steln sind unterstützt. Kürzere Notenwerte können auch notiert werden, können allerdings nur als Noten mit Balken auftreten.

```
\relative {
  \time 8/1
  c''\longa c\breve c1 c2
  c4 c8 c16 c32 c64 c128 c128
}
```



Hier die selben Notendauern ohne die Balken.
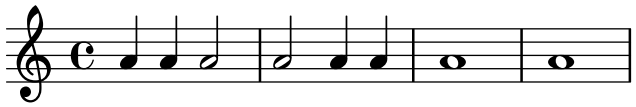
```
\relative {
  \time 8/1
  \autoBeamOff
  c''\longa c\breve c1 c2
  c4 c8 c16 c32 c64 c128 c128
}
```



Eine Note mit der vierfachen Dauer einer Brevis kann mit dem Befehl `\maxima` eingegeben werden, aber ihre Darstellung ist nur für die Alte Musiknotation unterstützt. Zu Einzelheiten siehe Kapitel 17 [Notation von alter Musik], Seite 420.
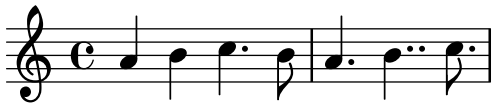
Wenn die Dauer hinter einer Notenbezeichnung nicht angegeben ist, wird die Dauer der vorhergehenden Note eingesetzt. Der Standardwert für die erste Note ist eine Viertel.

```
\relative { a' a a2 a a4 a a1 a }
```



Um punktierte Notendauern zu erhalten, muss einfach nur ein Punkt (`.`) hinter die Zahl der Dauer gesetzt werden. Zwei Punkte ergeben eine doppelte Punktierung, usw.

```
\relative { a'4 b c4. b8 a4. b4.. c8. }
```



Manche Notenlängen können nicht mit binären Dauern und Punkten dargestellt werden, sie können nur erreicht werden, indem man Noten überbindet. Für Einzelheiten siehe Abschnitt 2.1.4 [Bindebögen], Seite 53.

Wie den Silben von Gesangstext eigene Dauern zugewiesen werden können und wie man sie an den Noten ausrichtet ist erklärt in Kapitel 9 [Notation von Gesang], Seite 253.

Optional können Noten streng proportional nach ihrer exakten Dauer gesetzt werden. Zu Einzelheiten hierzu und weiteren Einstellungen für proportionale Notation siehe Abschnitt 29.5 [Proportionale Notation], Seite 565.

Punkte werden normalerweise nach oben verschoben, damit sie die Notenlinien nicht berühren. Punkte können manuelle über oder unter dem Notensystem gesetzt werden, zu Einzelheiten siehe Abschnitt 35.2 [Richtung und Platzierung], Seite 611.

Vordefinierte Befehle

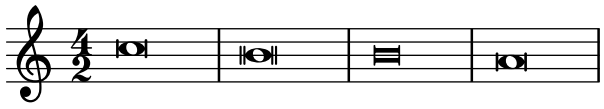
`\autoBeamOn`, `\autoBeamOff`, `\dotsUp`, `\dotsDown`, `\dotsNeutral`.

Ausgewählte Schnipsel

Alternative breve notes

Breve notes are also available with two vertical lines on each side of the notehead instead of one line and in baroque style.
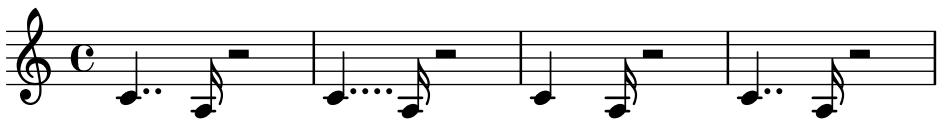
```
\relative c'' {
  \time 4/2
  c\breve |
  \override Staff.NoteHead.style = #'altdefault
  b\breve
  \override Staff.NoteHead.style = #'baroque
  b\breve
  \revert Staff.NoteHead.style
  a\breve
}
```



Changing the number of augmentation dots per note

The number of augmentation dots on a single note can be overridden by setting the `dot-count` property of the `Dots` grob.

```
\relative c' {
  c4.. a16 r2 |
  \override Dots.dot-count = 4
  c4.. a16 r2 |
  \override Dots.dot-count = 0
  c4.. a16 r2 |
  \revert Dots.dot-count
  c4.. a16 r2 |
}
```



Siehe auch

Glossar: Abschnitt “breve” in *Glossar*, Abschnitt “longa” in *Glossar*, Abschnitt “maxima” in *Glossar*, Abschnitt “note value” in *Glossar*, Abschnitt “Duration names notes and rests” in *Glossar*.

Notationsreferenz: Abschnitt 2.4.1 [Automatische Balken], Seite 81, Abschnitt 2.1.4 [Bindebögen], Seite 53, Abschnitt 7.1.6 [Hälse], Seite 220, Abschnitt 2.1 [Rhythmen eingeben], Seite 44, Abschnitt 2.2 [Pausen eingeben], Seite 57, Kapitel 9 [Notation von Gesang], Seite 253, Kapitel 17 [Notation von alter Musik], Seite 420, Abschnitt 29.5 [Proportionale Notation], Seite 565.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Referenz der Interna: Abschnitt “Dots” in *Referenz der Interna*, Abschnitt “DotColumn” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Es gibt keine grundlegende Grenze für die Dauer von Pausen (sowohl kürzer als auch länger), aber die Anzahl an Symbolen ist begrenzt: Einzelne Pausen können von 128stel bis zur Maxima (8 Ganze) gesetzt werden.

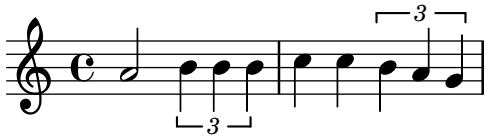
2.1.2 Andere rhythmische Aufteilungen

Triolen und andere rhythmische Aufteilungen werden aus einem musikalischen Ausdruck erstellt, indem dessen Tondauern mit einem Bruch multipliziert werden.

`\times` *`Bruch musikalischer Ausdruck`*

Die Dauer eines *musikalischen Ausdrucks* wird mit dem Bruch multipliziert. Der Nenner des Bruchs wird über (oder unter) den Noten ausgegeben, optional mit einer eckigen Klammer, die die Noten einfasst. Die üblichste Aufteilung ist die Triole, in welcher drei Noten die Länge von zwei haben, der Wert jeder einzelnen Note ist also 2/3 der notierten Länge.
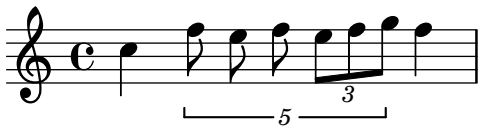
```
a2 \tuplet 3/2 { b4 b b }
c4 c \tuplet 3/2 { b4 a g }
```



Triolenklammern können manuell über oder unter dem Notensystem ausgegeben werden, siehe Abschnitt 35.2 [Richtung und Platzierung], Seite 611.

N-tolen können ineinander geschachtelt werden:

```
\relative {
  \autoBeamOff
  c''4 \tuplet 5/4 { f8 e f \tuplet 3/2 { e[ f g] } } f4
}
```



Wenn man die Eigenschaften von N-tolen verändern will, die zum selben musikalischen Zeitpunkt beginnen, muss `\tweak` eingesetzt werden.

Um die Dauern von Noten zu ändern, ohne die N-tolen-Klammern zu setzen, siehe Abschnitt 2.1.3 [Tondauern skalieren], Seite 52.

Vordefinierte Befehle

`\tupletUp`, `\tupletDown`, `\tupletNeutral`.

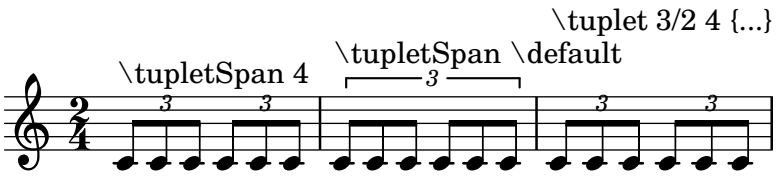
Ausgewählte Schnipsel

Entering several tuplets using only one \tuplet command

The property `tupletSpannerDuration` sets how long each of the tuplets contained within the brackets after `\tuplet` should last. Many consecutive tuplets can then be placed within a single `\tuplet` expression, thus saving typing.

There are ways to set tupletSpannerDuration besides using a \set command. The command \tupletSpan sets it to a given duration, or clears it when instead of a duration \default is specified. Another way is to use an optional argument with \tuplet.

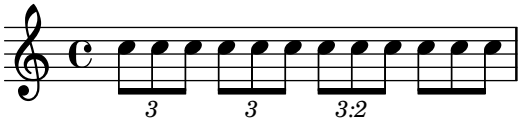
```
\relative c' {
  \time 2/4
  \tupletSpan 4
  \tuplet 3/2 { c8^"\\tupletSpan 4" c c c c c }
  \tupletSpan \default
  \tuplet 3/2 { c8^"\\tupletSpan \\default" c c c c c }
  \tuplet 3/2 4 { c8^"\\tuplet 3/2 4 {...}" c c c c c }
}
```



Die Zahl der N-tole verändern

Standardmäßig wird nur der Zähler des N-tolen-Bruchs über der Klammer dargestellt, wie er dem \times-Befehl übergeben wird. Man kann aber auch Zähler/Nenner ausgeben lassen, oder die Zahl vollständig unterdrücken.

```
\relative c'' {
  \tuplet 3/2 { c8 c c }
  \tuplet 3/2 { c8 c c }
  \override TupletNumber.text = #tuplet-number::calc-fraction-text
  \tuplet 3/2 { c8 c c }
  \omit TupletNumber
  \tuplet 3/2 { c8 c c }
}
```

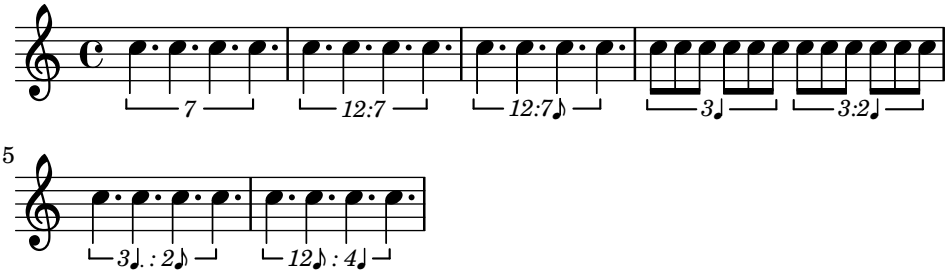


Nicht-standard-N-tolennummern

LilyPond stellt auch Formatierungsfunktionen zur Verfügung, mit denen N-tolennummern gesetzt werden können, die sich von dem eigentlichen Bruch unterscheiden. Auch ein Notenwert kann zu Nenner oder Zähler des Bruchs hinzugefügt werden.

```
\relative c'' {
  \once \override TupletNumber.text =
    #(tuplet-number::non-default-tuplet-denominator-text 7)
  \tuplet 3/2  { c4. c4. c4. c4. }
  \once \override TupletNumber.text =
    #(tuplet-number::non-default-tuplet-fraction-text 12 7)
  \tuplet 3/2  { c4. c4. c4. c4. }
  \once \override TupletNumber.text =
    #(tuplet-number::append-note-wrapper
      (tuplet-number::non-default-tuplet-fraction-text 12 7)
      (ly:make-duration 3 0))
  \tuplet 3/2  { c4. c4. c4. c4. }
```

```
  \once \override TupletNumber.text =
    #(tuplet-number::append-note-wrapper
      tuplet-number::calc-denominator-text
      (ly:make-duration 2 0))
  \tuplet 3/2  { c8 c8 c8 c8 c8 c8 }
  \once \override TupletNumber.text =
    #(tuplet-number::append-note-wrapper
      tuplet-number::calc-fraction-text
      (ly:make-duration 2 0))
  \tuplet 3/2  { c8 c8 c8 c8 c8 c8 }
  \once \override TupletNumber.text =
    #(tuplet-number::fraction-with-notes
      (ly:make-duration 2 1) (ly:make-duration 3 0))
  \tuplet 3/2  { c4. c4. c4. c4. }
  \once \override TupletNumber.text =
    #(tuplet-number::non-default-fraction-with-notes 12
      (ly:make-duration 3 0) 4 (ly:make-duration 2 0))
  \tuplet 3/2  { c4. c4. c4. c4. }
}
```



Controlling tuplet bracket visibility

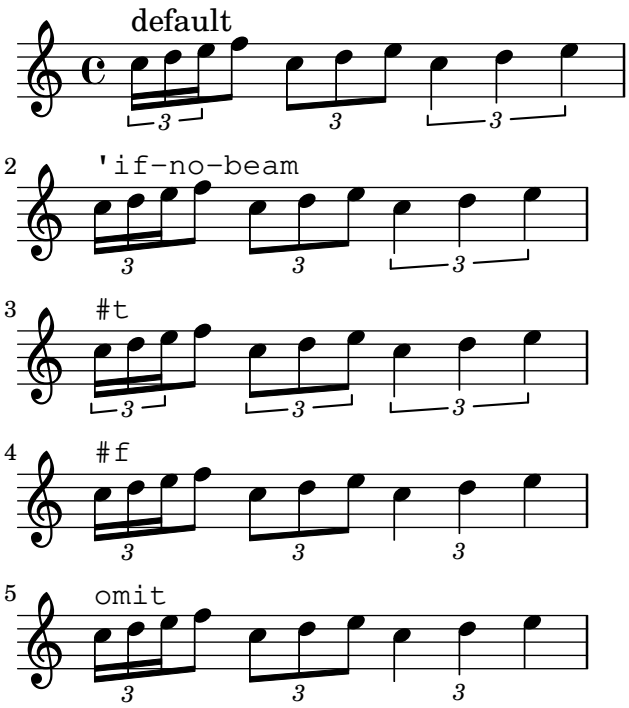
The default behavior of tuplet-bracket visibility is to print a bracket unless there is a beam of the same length as the tuplet.

To control the visibility of tuplet brackets, set the property `bracket-visibility` to either `#t` (always print a bracket), `if-no-beam` (only print a bracket if there is no beam) or `#f` (never print a bracket). The latter is in fact equivalent to omitting the `TupletBracket` object altogether from the printed output.

```
music = \relative c'' {
  \tuplet 3/2 { c16[ d e } f8]
  \tuplet 3/2 { c8 d e }
  \tuplet 3/2 { c4 d e }
}

\new Voice {
  \relative c' {
    \override Score.TextMark.non-musical = ##f
    \textMark "default" \music
    \override TupletBracket.bracket-visibility = #'if-no-beam
    \textMark \markup \typewriter "'if-no-beam" \music
    \override TupletBracket.bracket-visibility = ##t
    \textMark \markup \typewriter "#t" \music
    \override TupletBracket.bracket-visibility = ##f
    \textMark \markup \typewriter "#f" \music
```

```
    \omit TupletBracket
    \textMark \markup \typewriter "omit" \music
  }
}
```



Zeilenumbrüche bei N-tolen mit Balken erlauben

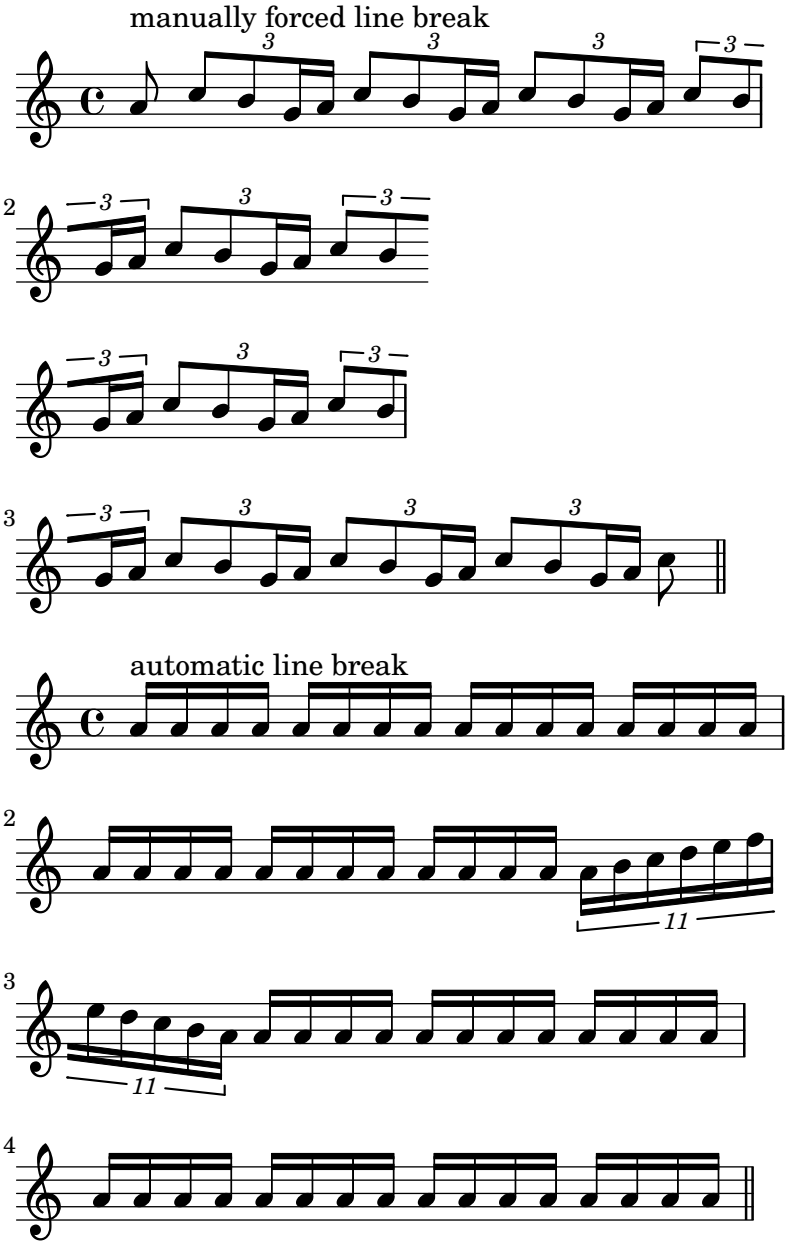
Dieses künstliche Beispiel zeigt, wie sowohl automatische als auch manuelle Zeilenumbrüche innerhalb einer N-tole mit Balken erlaubt werden können. Diese unregelmäßige Bebalkung muss allerdings manuell gesetzt werden.

```
\layout {
  \context {
    \Voice
    % Permit automatic line breaks within tuplets.
    \remove "Forbid_line_break_engraver"
    % Allow beams to be broken at line breaks.
    \override Beam.breakable = ##t
  }
}

\relative c'' {
  <>^"manually forced line break"
  a8
  \*5 { \tuplet 3/2 { c8[ b g16 a] } }
  \tuplet 3/2 { c8[ b \break g16 a] }
  \*5 { \tuplet 3/2 { c8[ b g16 a] } }
  c8 \bar "||"
}

\relative c'' {
  <>^"automatic line break"
  \*28 a16
```

```
  \tuplet 11/8 { a16[ b c d e f e d c b a] }
  \*28 a16 \bar "||"
}
```



Siehe auch

Glossar: Abschnitt “triplet” in *Glossar*, Abschnitt “tuplet” in *Glossar*, Abschnitt “polymetric” in *Glossar*.

Handbuch zum Lernen: Abschnitt “Optimierungsmethoden” in *Handbuch zum Lernen*.

Notationrreferenz: Abschnitt 2.6.3 [Verwaltung der Zeiteinheiten], Seite 118, Abschnitt 2.1.3 [Tondauern skalieren], Seite 52, Abschnitt 34.4 [Der `\tweak`-Befehl], Seite 605, Abschnitt 2.3.5 [Polymetrische Notation], Seite 75.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Referenz der Interna: Abschnitt “TupletBracket” in *Referenz der Interna*, Abschnitt “TupletNumber” in *Referenz der Interna*, Abschnitt “TimeScaledMusic” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Verzierungen können innerhalb von Triolenklammern gesetzt werden, *außer* wenn ein System mit einer Verzierung beginnt, die von einer N-tole gefolgt wird. In diesem besonderen Fall müssen die Verzierungen vor dem `\times`-Befehl gesetzt werden, damit sich keine Fehler ergeben.
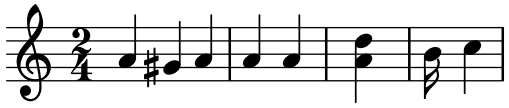
Wenn man eine N-tole zu Beginn eines Stückes notiert, das eine Tempobezeichnung mit `\tempo` enthält, müssen die Noten in einer explizit begonnenen Stimme notiert werden. Siehe auch Abschnitt "Voice enthält Noten" in *Handbuch zum Lernen*.

2.1.3 Tondauern skalieren

Die Dauer von einzelnen Noten, Pausen oder Akkorden kann mit einem Bruch multipliziert werden, indem hinter die Notendauer „*N/M“ (oder „*N“ wenn M 1 ist) geschrieben wird. Die Erscheinung der Noten oder Pausen wird dadurch nicht beeinflusst, die neue Dauer wird aber dazu benutzt, ihre Position im Takt zu errechnen und die neue Dauer in der MIDI-Ausgabe einzusetzen. Die Faktoren, mit denen multipliziert wird, können auch kombiniert werden, etwa „*L*M*/N“. Die Faktoren sind Teil der Dauer: wenn man keine Dauer für die nächste Note angibt, wird die Dauer der vorigen Note mit allen Skalierungsfaktoren übernommen.

Im nächsten Beispiel nehmen die drei ersten Noten genau zwei Schläge ein, aber es wird keine Triolenklammer über ihnen ausgegeben.

```
\relative {
  \time 2/4
  % Alter durations to triplets
  a'4*2/3 gis a
  % Normal durations
  a4 a
  % Double the duration of chord
  <a d>4*2
  % Duration of quarter, appears like sixteenth
  b16*4 c4
}
```

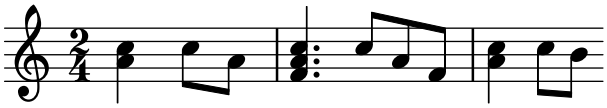


Die Dauer von unsichtbaren Pausen kann auch mit einem Faktor beeinflusst werden. Das ist sinnvoll, wenn man viele Takte überspringen muss, etwa `s1*23`.

Längere Notenabschnitte können auf die gleiche Art durch Multiplikation mit einem Bruch komprimiert werden, als ob jede Note, jeder Akkord oder jede Pause mit dem Bruch multipliziert würde. Damit bleibt das Aussehen der Musik unverändert, aber die interne Dauer der Noten wird mit dem Bruch multipliziert. Hier ein Beispiel, das zeigt, wie Noten komprimiert und ausgedehnt werden kann:

```
\time 2/4
% Normal durations
<c a>4 c8 a
% Scale music by *2/3
\scaleDurations 2/3 {
  <c a f>4. c8 a f
}
% Scale music by *2
\scaleDurations 1/2 {
  <c' a>4 c8 b
```

```
}
```



Eine Anwendung für diesen Befehl ist polymetrische Notation, siehe Abschnitt 2.3.5 [Polymetrische Notation], Seite 75.

Siehe auch

Notationsreferenz: Abschnitt 2.1.2 [Andere rhythmische Aufteilungen], Seite 47, Abschnitt 2.2.2 [Unsichtbare Pausen], Seite 59, Abschnitt 2.3.5 [Polymetrische Notation], Seite 75.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

2.1.4 Bindebögen

Ein Bindebogen verbindet zwei benachbarte Noten der selben Tonhöhe. Als Resultat wird die Dauer der Notenlänge verlängert.
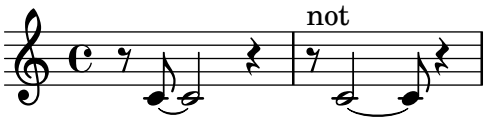
> **Achtung:** Bindebögen (engl. tie) dürfen nicht mit Legatobögen (engl. slur) verwechselt werden, durch die die Vortragsart bezeichnet wird, noch mit Phrasierungsbögen (engl. phrasing slur), die musikalische Phrasen anzeigen. Ein Bindebogen ist nur eine Art, die Tondauer zu verlängern, ähnlich etwa wie die Punktierung.

Ein Bindebogen wird mit der Tilde ~ (AltGr++) notiert.

```
a2 ~ 2
```



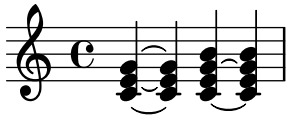
Bindebögen werden eingesetzt, wenn die Note entweder über eine Taktlinie hinüberreicht, oder wenn die entsprechende Dauer der Note nicht mit Punktierung erreicht werden kann. Bindebögen sollten auch benutzt werden, wenn Notenwerte über die inneren Unterteilungen von Takten hinüberreichen:



Wenn viele Noten über Taktlinien gebunden werden müssen, kann es einfacher sein, automatische Notenaufteilung einzustellen, wie beschrieben in Abschnitt 2.3.6 [Automatische Aufteilung von Noten], Seite 78. Mit diesem Mechanismus werden lange Noten automatisch aufgeteilt, wenn sie über Taktgrenzen reichen.

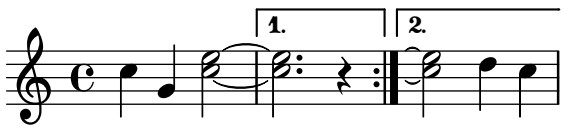
Wenn ein Bindebogen an einen Akkord gehängt wird, werden alle Noten dieses Akkordes übergebunden. Wenn kein Notenkopf passt, wird auch kein Bogen erzeugt. Noten in Akkorden können auch einzeln übergebunden werden, indem sie innerhalb des Akkordes hinter die entsprechende Note geschrieben werden.
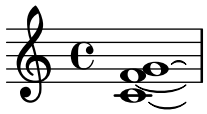
```
<c e g> ~ <c e g>
<c~ e g~ b> <c e g b>
```



Wenn die zweite Variante einer Wiederholung mit einer übergebundenen Note anfängt, muss der Bindebogen wie folgt notiert werden:

```
\repeat volta 2 { c g <c e>2 ~ }
\alternative {
  % First alternative: following note is tied normally
  { <c e>2. r4 }
  % Second alternative: following note has a repeated tie
  { <c e>2\repeatTie d4 c } }
```



So genannte *laissez vibrer*-Bögen werden verwendet um anzuzeigen, dass man die Musik ausklingen lassen soll. Sie werden in der Klavier-, Harfen-, anderer Saiteninstrument- und Schlagzeugnotation verwendet. Sie können folgendermaßen notiert werden:
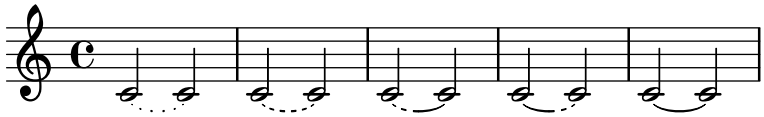
```
<c' f' g'>1\laissezVibrer
```



Bindebögen können manuell über oder unter dem Notensystem gesetzt werden. Zu Einzelheiten siehe Abschnitt 35.2 [Richtung und Platzierung], Seite 611.
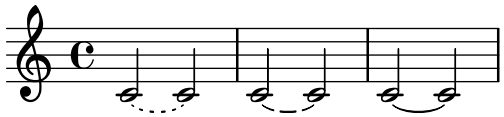
Bindebögen können durchgehend, gestrichelt, gepunktet oder in einer Kombination von Strichen und durchgehender Linie definiert werden.

```
\tieDotted
c2 ~ 2
\tieDashed
c2 ~ 2
\tieHalfDashed
c2 ~ 2
\tieHalfSolid
c2 ~ 2
\tieSolid
c2 ~ 2
```



Eigene Strichelungsmuster können definiert werden:
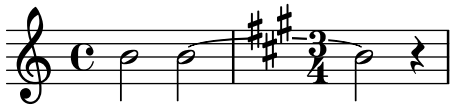
```
\tieDashPattern #0.3 #0.75
c2 ~ 2
\tieDashPattern #0.7 #1.5
c2 ~ 2
\tieSolid
c2 ~ 2
```



Die Definition von Muster für die Strichelung der Bindebögen hat die gleiche Struktur wie die Definition für Legatobögen. Zu weiterer Information zu komplizierten Strichelungsmustern, siehe die Schnipsel im Abschnitt Abschnitt 3.2.1 [Legatobögen], Seite 130.

Durch Veränderung der Eigenschaften *whiteout* (weiß malen) und *layer* (Ebene) kann verhindert werden, dass Bindebögen mit anderen Objekten im Notensystem zusammenstoßen.

```
\relative {
  \override Tie.layer = #-2
  \override Staff.TimeSignature.layer = #-1
  \override Staff.KeySignature.layer = #-1
  \override Staff.TimeSignature.whiteout = ##t
  \override Staff.KeySignature.whiteout = ##t
  b'2 b~
  \time 3/4
  \key a \major
  b r4
}
```



Vordefinierte Befehle

\tieUp, \tieDown, \tieNeutral, \tieDotted, \tieDashed, \tieDashPattern, \tieHalfDashed, \tieHalfSolid, \tieSolid.
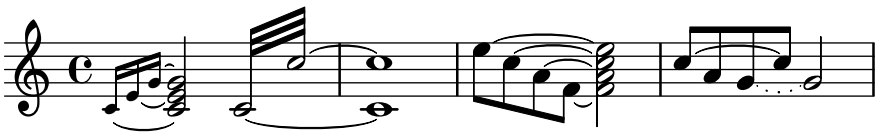
Ausgewählte Schnipsel

Überbindungen für Arpeggio benutzen

Überbindungen werden teilweise benutzt, um Arpeggios zu notieren. In diesem Fall stehen die übergebundenen Noten nicht unbedingt hintereinander. Das Verhalten kann erreicht werden, indem die tieWaitForNote-Eigenschaft auf #t gesetzt wird. Diese Funktion ist auch sinnvoll, um etwa ein Tremolo mit einem Akkord zu überbinden, kann aber prinzipiell auch für normale Überbindungen eingesetzt werden

```
\relative c' {
  \set tieWaitForNote = ##t
  \grace { c16[ ~ e ~ g] ~ } <c, e g>2
  \repeat tremolo 8 { c32 ~ c' ~ } <c c,>1
  e8 ~ c ~ a ~ f ~ <e' c a f>2
  \tieUp
  c8 ~ a
  \tieDown
```

```
  \tieDotted
  g8 ~ c g2
}
```



Bindebögen manuell setzen

Überbindungen können manuell gesetzt werden, indem man die `tie-configuration`-Eigenschaft des `TieColumn`-Objekts beeinflusst. Die erste Zahl zeigt den Abstand von der Mitte in Notensystemabständen an, die zweite Zahl zeigt die Richtung an (1 = nach oben, -1 = nach unten).

```
\relative c' {
  <>^"default"
  g'1 ^~ g

  <>^"0"
  \once \override Tie.staff-position = 0
  g1 ^~ g

  <>^"0.0"
  \once \override Tie.staff-position = 0.0
  g1 ^~ g

  <>^"reset"
  \revert Tie.staff-position
  g1 ^~ g
}

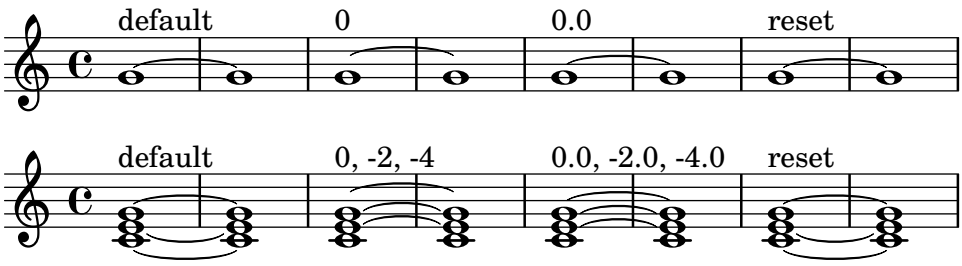
\relative c' {
  \override TextScript.outside-staff-priority = ##f
  \override TextScript.padding = 0

  <>^"default"
  <c e g>1~ <c e g>

  <>^"0, -2, -4"
  \override TieColumn.tie-configuration =
    #'((0 . 1) (-2 . 1) (-4 . 1))
  <c e g>1~ <c e g>

  <>^"0.0, -2.0, -4.0"
  \override TieColumn.tie-configuration =
    #'((0.0 . 1) (-2.0 . 1) (-4.0 . 1))
  <c e g>1~ <c e g>

  <>^"reset"
  \override TieColumn.tie-configuration = ##f
  <c e g>1~ <c e g>
}
```



Siehe auch

Glossar: Abschnitt “tie” in *Glossar*, Abschnitt “laissez vibrer” in *Glossar*.

Notationsreferenz: Abschnitt 3.2.1 [Legatobögen], Seite 130, Abschnitt 2.3.6 [Automatische Aufteilung von Noten], Seite 78.

Schnipsel: Abschnitt “Expressive marks” in *Schnipsel*, Abschnitt “Rhythms” in *Schnipsel*.

Referenz der Interna: Abschnitt “LaissezVibrerTie” in *Referenz der Interna*, Abschnitt “LaissezVibrerTieColumn” in *Referenz der Interna*, Abschnitt “TieColumn” in *Referenz der Interna*, Abschnitt “Tie” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Der Wechsel zwischen Systemen bei aktiver Überbindung produziert keinen gekrümmten Bogen.

Änderung von Schlüssel oder Oktavierung zwischen übergebundenen Noten ist nicht richtig definiert. In diesen Fällen kann es besser sein, einen Legatobogen zu verwenden.

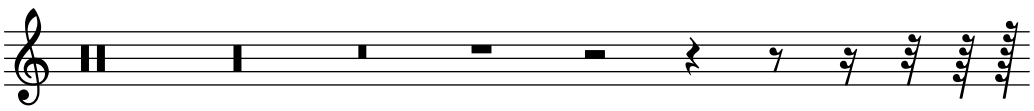
2.2 Pausen eingeben

Pausen werden als Teil der musikalischen Ausdrücke zusammen mit den Noten notiert.

2.2.1 Pausen

Pausen werden wie Noten eingegeben, ihre Bezeichnung ist `r`. Dauern, die länger als eine Ganze sind, haben die vordefinierten Befehle:

```
\new Staff {
  % These two lines are just to prettify this example
  \time 16/1
  \omit Staff.TimeSignature
  % Print a maxima rest, equal to four breves
  r\maxima
  % Print a longa rest, equal to two breves
  r\longa
  % Print a breve rest
  r\breve
  r1 r2 r4 r8 r16 r32 r64 r128
}
```



Pausen, die ganze Takte ausfüllen und in der Taktmitte zentriert werden sollen, müssen als mehrtaktige Pausen eingegeben werden. Sie können sowohl für einen einzigen Takt als auch für mehrere Takte verwendet werden, Näheres im Abschnitt Abschnitt 2.2.3 [Ganztaktpausen], Seite 60.

Um die vertikale Position einer Pause explizit festzulegen, kann eine Note eingegeben werden, gefolgt vom Befehl `\rest`. Die Pause wird dann an die Stelle gesetzt, wo sich sonst die Note befinden würde. Damit wird die manuelle Formatierung von mehrstimmiger Musik sehr viel einfacher, da die Formatierungsfunktion zur automatischen Auflösung von Zusammenstößen diese Pausen nicht mit einbezieht.

```
\relative { a'4\rest d4\rest }
```



Ausgewählte Schnipsel

Pausenstile

Pausen können in verschiedenen Stilen dargestellt werden.

```
restsA = {
  r\maxima r\longa r\breve r1 r2 r4 r8 r16 s32
  s64 s128 s256 s512 s1024 s1024
}
restsB = {
  r\maxima r\longa r\breve r1 r2 r4 r8 r16 r32
  r64 r128 r256 r512 r1024 s1024
}

\new Staff \relative c {
  \omit Score.TimeSignature
  \cadenzaOn

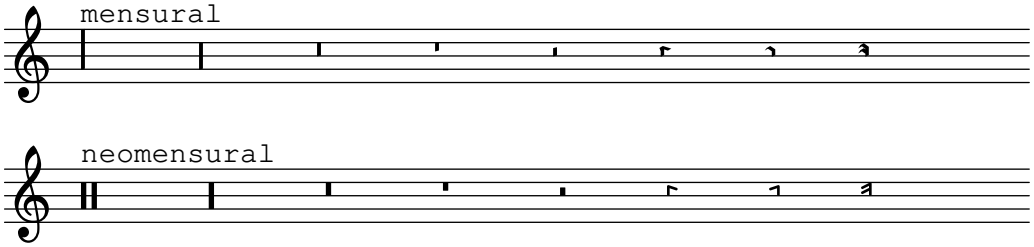
  \override Staff.Rest.style = #'mensural
  <>^\markup \typewriter { mensural } \restsA \bar "" \break

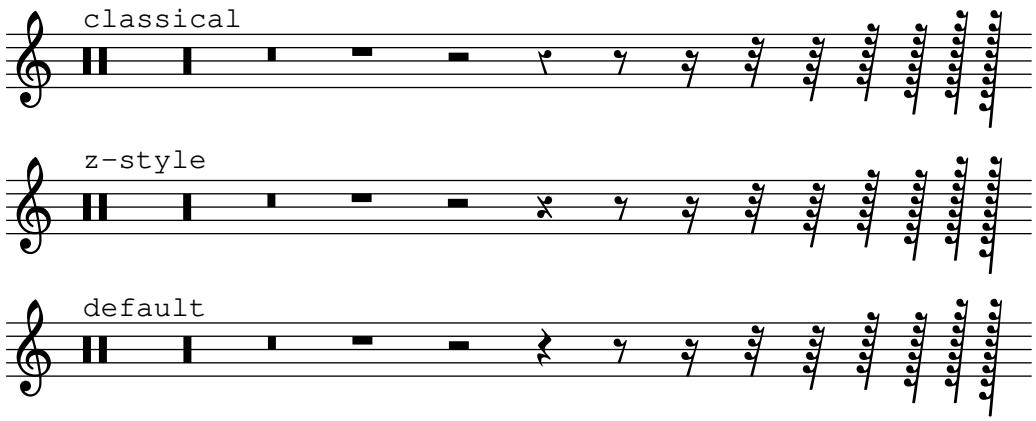
  \override Staff.Rest.style = #'neomensural
  <>^\markup \typewriter { neomensural } \restsA \bar "" \break

  \override Staff.Rest.style = #'classical
  <>^\markup \typewriter { classical } \restsB \bar "" \break

  \override Staff.Rest.style = #'z
  <>^\markup \typewriter { z-style } \restsB \bar "" \break

  \override Staff.Rest.style = #'default
  <>^\markup \typewriter { default } \restsB \bar "" \break
}
```





Siehe auch

Glossar: Abschnitt “breve” in *Glossar*, Abschnitt “longa” in *Glossar*, Abschnitt “maxima” in *Glossar*.

Notationsreferenz: Abschnitt 2.2.3 [Ganztaktpausen], Seite 60.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Referenz der Interna: Abschnitt “Rest” in *Referenz der Interna*.
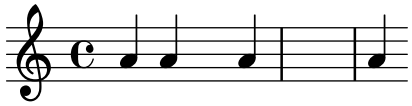
Bekannte Probleme und Warnungen

Es gibt keine grundlegende Grenze für die Dauer von Pausen (sowohl kürzer als auch länger), aber die Anzahl von Symbolen ist begrenzt: Es gibt Zeichen für Pausen von einer 128 bis zu einer Maxima (8 Ganze).

2.2.2 Unsichtbare Pausen
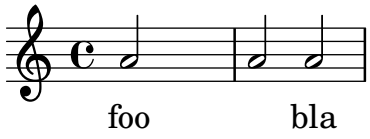
Eine unsichtbare Pause (auch als „skip“ oder Überspringung bezeichnet) kann wie eine Note eingegeben werden, die Notationsbezeichnung ist `s`.

```
a4 a4 s4 a4 \skip 1 a4
```



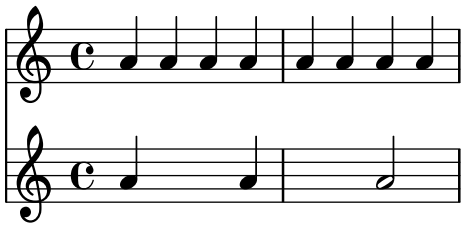
Die `s`-Syntax steht nur im Noten- oder Akkordmodus zur Verfügung. In anderen Situationen, z. B. innerhalb eines Liedtextes, muss der Befehl `\skip` benutzt werden. `\skip` benötigt eine explizite Dauerangabe.

```
<<
  {
    a'2 \skip2 a'2 a'2
  }
  \new Lyrics {
    \lyricmode {
      foo2 \skip 1 bla2
    }
  }
>>
```



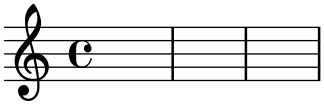
Weil \skip ein Befehl ist, wirkt er sich nicht auf die Dauer der folgenden Noten aus, anders als s.

```
<<
  {
    \repeat unfold 8 { a'4 }
  }
  {
    a'4 \skip 2 a' |
    s2 a'
  }
>>
```



Die Platzhalterpause mit s erstellt Staff- und Voice-Kontext, wenn es erforderlich ist, genauso wie Noten und Pausen.

```
{ s1 s s }
```



Der Überspringungsbefehl (\skip) ist einfach ein leerer Platzhalter. Durch ihn wird überhaupt nichts gesetzt, auch keine transparenten Objekte.

```
% This is valid input, but does nothing
{ \skip 1 \skip1 \skip 1 }
```

Siehe auch

Handbuch zum lernen: Abschnitt “Sichtbarkeit und Farbe von Objekten” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 7.1.3 [Unsichtbare Noten], Seite 217, Abschnitt 35.6 [Sichtbarkeit von Objekten], Seite 619.

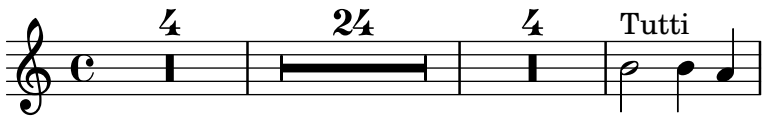
Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Referenz der Interna: Abschnitt “SkipMusic” in *Referenz der Interna*

2.2.3 Ganztaktpausen

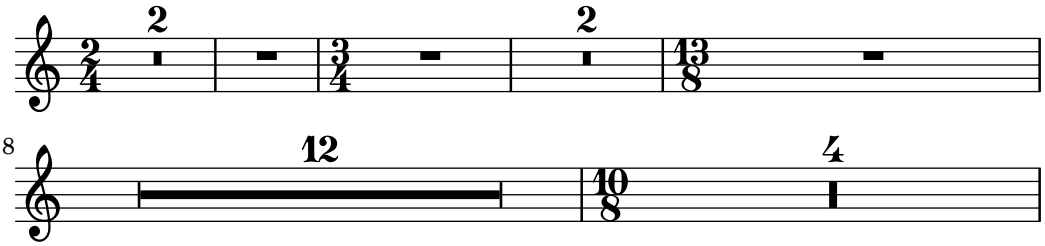
Pausen für einen oder mehrere ganze Takte werden wie Noten eingegeben, wobei die Bezeichnung ein Großbuchstabe R ist:

```
% Rest measures contracted to single measure
\compressEmptyMeasures
R1*4
R1*24
R1*4
b2^"Tutti" b4 a4
```
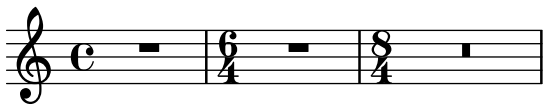


Die Dauer von Ganztaktpausen wird genauso angegeben wie die Dauer von Noten. Die Dauer einer Ganztaktpause muss immer eine ganze Anzahl an Taktlängen sein, weshalb Punktierungen und Brüche recht häufig eingesetzt werden müssen.

```
\compressEmptyMeasures
\time 2/4
R1 | R2 |
\time 3/4
R2. | R2.*2 |
\time 13/8
R1*13/8 | R1*13/8*12 |
\time 10/8
R4*5*4 |
```



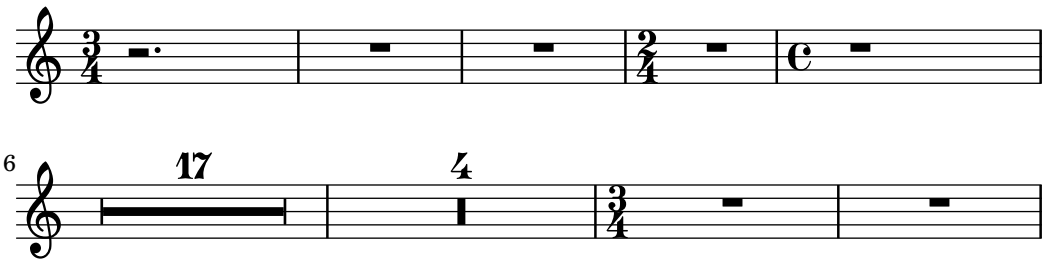
Eine Ganztaktpause wird abhängig von der Taktart entweder als Ganze oder Brevis-Pause gesetzt, zentriert im Takt.

```
\time 4/4
R1 |
\time 6/4
R1*3/2 |
\time 8/4
R1*2 |
```



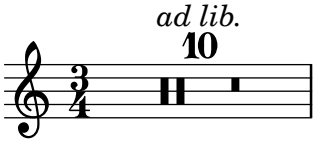
In den Standardeinstellungen werden mehrtaktige Pausen ausgeschrieben gesetzt, sodass sie die entsprechende Anzahl von Takten einnehmen. Alternativ kann die mehrtaktige Pause aber auch nur in einem Takt angezeigt werden, der ein Mehrtaktpausensymbol beinhaltet, wobei die Anzahl der Takte der Pausendauer über dem Pausenzeichen ausgegeben wird:

```
% Default behavior
\time 3/4 r2. | R2.*2 |
\time 2/4 R2 |
\time 4/4
% Rest measures contracted to single measure
\compressEmptyMeasures
r1 | R1*17 | R1*4 |
% Rest measures expanded
\expandEmptyMeasures
\time 3/4
R2.*2 |
```



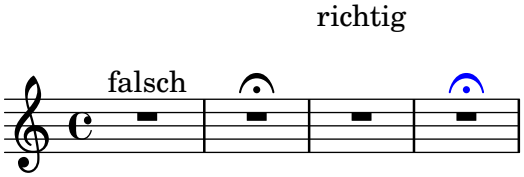
Textbeschriftung kann Mehrtaktpausen mit `\markup` hinzugefügt werden.

```
\compressEmptyMeasures
\time 3/4
R2.*10^\markup { \italic "ad lib." }
```



Achtung: Beschriftungen und Artikulationen, die an Mehrtaktpausen gehängt werden, sind Objekte vom Typ `MultiMeasureRestText` bzw. `MultiMeasureRestScript`, nicht vom Typ `TextScript` bzw. `Script`. Änderungen etwa mit `\override` müssen auf das richtige Objekt gerichtet werden, damit sie nicht ignoriert werden. Siehe auch das folgende Beispiel.

```
% Dies hat keine Auswirkungen wegen der falschen Objektnamen
\override TextScript.padding = #5
\override Script.color = #blue
R1^"falsch"
R1\fermata
% Dies sind die richtigen Objektnamen
\override MultiMeasureRestText.padding = #5
\override MultiMeasureRestScript.color = #blue
R1^"richtig"
R1\fermata
```



Wenn eine Mehrtaktpause direkt auf einen Auftakt mit `\partial` folgt, werden möglicherweise daraus resultierende Taktprüfungswarnungen nicht angezeigt.

Vordefinierte Befehle

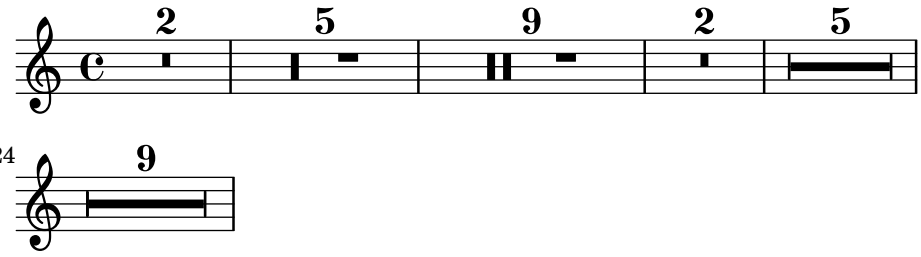
`\textLengthOn`, `\textLengthOff`, `\compressEmptyMeasures`, `\expandEmptyMeasures`.

Ausgewählte Schnipsel

Die Erscheinung von Pausentakten ändern

Wenn zehn oder weniger Pausentakte vorkommen, wird eine Reihe von Longa- und Brevispausen (auch Kirchenpausen genannt) gesetzt, bei mehr Takten wird eine Line mit der Taktanzahl ausgegeben. Der vorgegebene Wert von zehn kann geändert werden, indem man die `expand-limit`-Eigenschaft setzt:

```
\relative c'' {
  \compressMMRests {
    R1*2 | R1*5 | R1*9
    \override MultiMeasureRest.expand-limit = 3
    R1*2 | R1*5 | R1*9
  }
}
```



Positionierung von Ganztaktpausen

Anders als bei normalen Pausen gibt es keinen direkten Befehl, um die vertikale Position von Ganztaktpausen zu beeinflussen, indem man sie an eine Tonhöhe anhängt. In polyphoner Notation wird aber dennoch die Position der Pausen von geraden und ungeraden Stimmen voneinander unterschieden. Die Position von Ganztaktpausen kann wie folgt verändert werden:

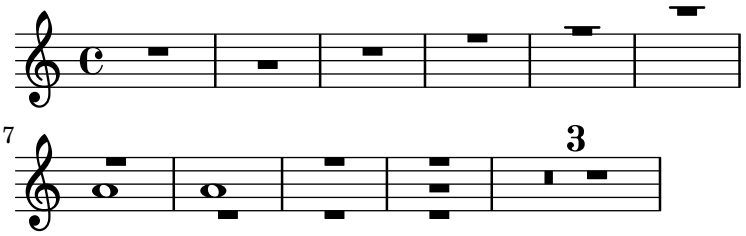
```
\relative c'' {
  % Multi-measure rests by default are set under the fourth line.
  R1
  % They can be moved using an override or tweak.
  \tweak staff-position -2 R1
  \tweak staff-position 0 R1
  \tweak staff-position 2 R1
  \override MultiMeasureRest.staff-position = 3 R1
  \override MultiMeasureRest.staff-position = 6 R1
  \revert MultiMeasureRest.staff-position
  \break

  % Odd-numbered voices are under the top line.
  << { R1 } \\ { a1 } >>
  % Even-numbered voices are under the bottom line.
  << { a1 } \\ { R1 } >>
  % Multi-measure rests in both voices remain separate.
  << { R1 } \\ { R1 } >>

  % Separating multi-measure rests in more than two voices
  % requires an override or tweak.
  << { R1 } \\ { R1 } \\ { \tweak staff-position -2 R1 } >>

  % Using compressed bars in multiple voices requires another override
  % in all voices to avoid multiple instances being printed.
```

```
  \compressMMRests
  <<
    \revert MultiMeasureRest.direction
    { R1*3 } \\
    \revert MultiMeasureRest.direction
    { R1*3 }
  >>
}
```

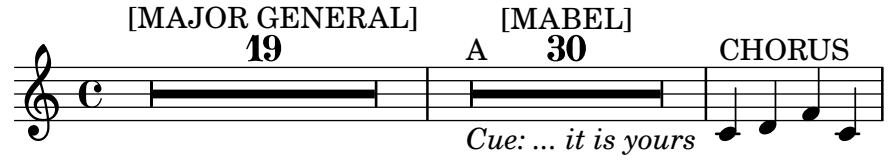


Textbeschriftung und Mehrtaktpausen

Textbeschriftungen, die an Mehrtaktpausen gehängt wird, wird über oder unter der Pause zentriert. Lange Beschriftungen lassen den Takt nicht breiter werden. Um eine Mehrtaktpause einer Beschriftung anzupassen, muss eine unsichtbare Pause mit der Beschriftung direkt vor der Mehrtaktpause eingesetzt werden.

Man sollte beachten, dass unsichtbare Pausen automatische Taktstriche nach sich ziehen. Text, der an eine unsichtbare Pause gehängt wird, ist links ausgerichtet an der Position, wo die Pause erscheinen würde. Wenn aber die Länge des Taktes durch die Länge des Textes bestimmt wird, sieht es so aus, als ob der Text zentriert gesetzt ist.

```
\relative c' {
  \compressMMRests {
    \textLengthOn
    <>^\markup { [MAJOR GENERAL] }
    R1*19
    <>_\markup { \italic { Cue: ... it is yours } }
    <>^\markup { A }
    R1*30^\markup { [MABEL] }
    \textLengthOff
    c4^\markup { CHORUS } d f c
  }
}
```



Siehe auch

Glossar: Abschnitt “multi-measure rest” in *Glossar*.

Notationsreferenz: Abschnitt 2.1.1 [Tondauern], Seite 44, Kapitel 8 [Text], Seite 226, Abschnitt 8.2 [Text formatieren], Seite 233, Abschnitt 8.1.1 [Textarten], Seite 226.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Referenz der Interna: Abschnitt “MultiMeasureRest” in *Referenz der Interna*, Abschnitt “MultiMeasureRestNumber” in *Referenz der Interna*, Abschnitt “MultiMeasureRestScript” in *Referenz der Interna*, Abschnitt “MultiMeasureRestText” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Wenn man versucht, mit Fingersatz (etwa `R1*10-4` Zahlen über Ganztaktpausen zu setzen, kann die Zahl des Fingersatzes (4) mit der Taktanzahl (10) zusammenstoßen.

Es gibt keine Möglichkeit, normale Pausen automatisch zu Ganztaktpausen zu reduzieren.

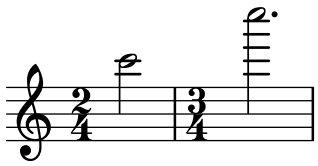
Ganztaktpausen werden bei der Vermeidung von Zusammenstößen nicht berücksichtigt.

2.3 Rhythmen anzeigen lassen

2.3.1 Taktangabe

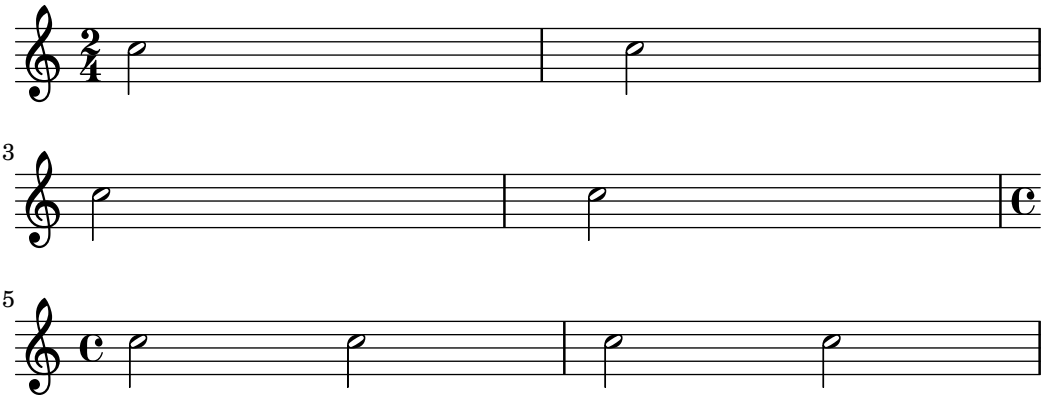
Taktangaben werden wie folgt erstellt.

```
\time 2/4 c'2
\time 3/4 c'2.
```
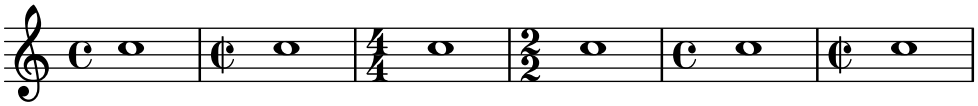


Taktangaben werden zu Beginn eines Stückes gesetzt und immer dann, wenn sich die Taktart ändert. Wenn eine Änderung am Ende einer Zeile geschieht, wird eine warnende Taktangabe am Ende der Zeile ausgegeben. Dieses Verhalten kann verändert werden, siehe Abschnitt 35.6 [Sichtbarkeit von Objekten], Seite 619.

```
\relative c'' {
  \time 2/4
  c2 c
  \break
  c c
  \break
  \time 4/4
  c c c c
}
```



Das Symbol für die Taktarten 2/2 und 4/4 kann in ein Zahlensymbol umgewandelt werden:
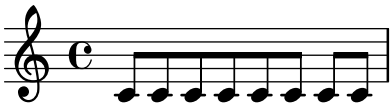
```
\relative c'' {
  % Default style
  \time 4/4 c1
  \time 2/2 c1
  % Change to numeric style
  \numericTimeSignature
  \time 4/4 c1
  \time 2/2 c1
  % Revert to default style
  \defaultTimeSignature
  \time 4/4 c1
  \time 2/2 c1
}
```



Symbole für Modus und Proprietas der mensuralen Notation werden behandelt unter Abschnitt 17.3.3 [Mensurale Taktartenbezeichnungen], Seite 425.

Zusätzlich zu der gedruckten Taktart werden mit der Definition des Befehls `\time` auch die Standardwerte fürt die Eigenschaften `beatBase`, `beatStructure` und `beamExtensions` gesetzt. Die vordefinierten Standardwerte für diese Eigenschaften finden sich in `scm/time-signature-settings.scm`. Die existierenden Standardwerte können verändert oder neue Standardwerte hinzugefügt werden.

```
\score {
  \new Staff {
    \relative c' {
      \overrideTimeSignatureSettings
        4/4        % timeSignature
        #1/4       % beatBase
        3,1        % beatStructure
        #'()       % beamExceptions
      \time 4/4
      \repeat unfold 8 { c8 } |
    }
  }
}
```
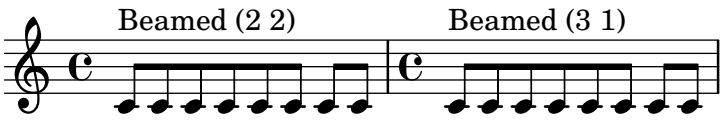


`\overrideTimeSignatureSettings` braucht fünf Argumente:

1. *`timeSignature`* (Taktart), ein Bruch, der den Takt beschreibt.
2. *`beatBase`* (Grundmoment-Bruch), ein Bruch, der den Zähler und Nenner der Grundschlageinheit der Taktart enhält.
3. *`beatStructure`* (Taktzeit-Struktur), eine Scheme-Liste, die die Struktur der Taktschläge anzeigt, in Einheiten des Grundmoments.
4. *`beamExceptions`* (Balken-Ausnahmen), eine Aliste, die alle Bebalkungsregeln für die Taktart enthält, außer dem Balken, der zum Taktende endet, wie beschrieben in Abschnitt 2.4.2 [Einstellung von automatischen Balken], Seite 84.
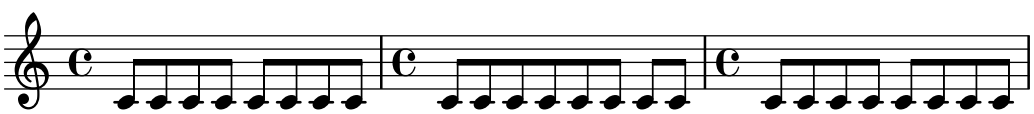
Der Kontext, der \overrideTimeSignatureSettings enthält, muss begonnen sein, bevor \overrideTimeSignatureSettings aufgerufen wird. Das heißt, dass er entweder explizit begonnen wird oder sich Noten in dem Kontext befinen müssen, bevor \overrideTimeSignatureSettings aufgerufen wird:

```
\score {
  \relative c' {
    % This call will fail because the context isn't yet instantiated
    \overrideTimeSignatureSettings
      4/4        % timeSignature
      #1/4       % beatBase
      3,1        % beatStructure
      #'()       % beamExceptions
    \time 4/4
    c8^\markup {"Beamed (2 2)"}
    \repeat unfold 7 { c8 } |
    % This call will succeed
    \overrideTimeSignatureSettings
      4/4        % timeSignature
      #1/4       % beatBase
      3,1        % beatStructure
      #'()       % beamExceptions
    \time 4/4
    c8^\markup {"Beamed (3 1)"}
    \repeat unfold 7 { c8 } |
  }
}
```



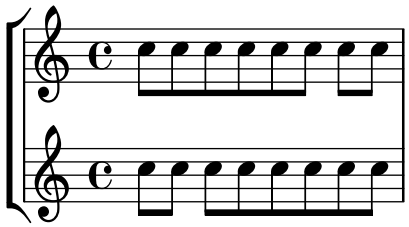
Veränderte Werte der Taktart-Eigenschaften können wieder auf den Standard zurückgesetzt werden:

```
\score{
  \relative {
    \repeat unfold 8 { c'8 } |
    \overrideTimeSignatureSettings
      4/4        % timeSignature
      #1/4       % beatBase
      3,1        % beatStructure
      #'()       % beamExceptions
    \time 4/4
    \repeat unfold 8 { c8 } |
    \revertTimeSignatureSettings 4/4
    \time 4/4
    \repeat unfold 8 { c8 } |
  }
}
```



Unterschiedliche Werte der Standard-Taktarteigenschaften für unterschiedliche Notensysteme können eingerichtet werden, indem man den `Timing_translator` und den `Default_bar_line_engraver` aus dem `Score`-Kontext in den `Staff`-Kontext verschiebt.

```
\score {
  \new StaffGroup <<
    \new Staff {
      \overrideTimeSignatureSettings
        4/4        % timeSignature
        #1/4       % beatBase
        3,1        % beatStructure
        #'()       % beamExceptions
      \time 4/4
      \repeat unfold 8 {c''8}
    }
    \new Staff {
      \overrideTimeSignatureSettings
        4/4        % timeSignature
        #1/4       % beatBase
        1,3        % beatStructure
        #'()       % beamExceptions
      \time 4/4
      \repeat unfold 8 {c''8}
    }
  >>
  \layout {
    \context {
      \Score
      \remove Timing_translator
    }
    \context {
      \Staff
      \consists Timing_translator
    }
  }
}
```



Vordefinierte Befehle

`\numericTimeSignature`, `\defaultTimeSignature`.
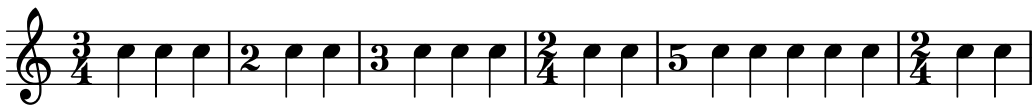
Ausgewählte Schnipsel

Time signature printing only the numerator as a number (instead of the fraction)

Sometimes, a time signature should not print the whole fraction (for example, 7/4), but only the numerator (digit 7 in this case). This can be easily done by using `\override`

`Staff.TimeSignature.style = #'single-number` to change the style permanently. By using `\revert Staff.TimeSignature.style`, this setting can be reversed. To apply the single-number style to only one time signature, use `\tweak`.

```
\relative c'' {
  \time 3/4
  c4 c c
  % Change the style permanently
  \override Staff.TimeSignature.style = #'single-number
  \time 2/4
  c4 c
  \time 3/4
  c4 c c
  % Revert to default style:
  \revert Staff.TimeSignature.style
  \time 2/4
  c4 c
  % single-number style only for the next time signature
  \tweak style #'single-number \time 5/4
  c4 c c c c
  \time 2/4
  c4 c
}
```



Siehe auch

Glossar: Abschnitt “time signature” in *Glossar*

Notationsreferenz: Abschnitt 17.3.3 [Mensurale Taktartenbezeichnungen], Seite 425, Abschnitt 2.6.3 [Verwaltung der Zeiteinheiten], Seite 118.
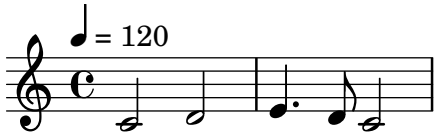
Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Referenz der Interna: Abschnitt “TimeSignature” in *Referenz der Interna*, Abschnitt “Timing_translator” in *Referenz der Interna*.

2.3.2 Metronomangabe
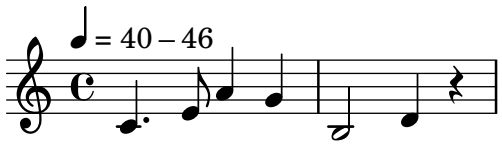
Eine Metronomanweisung wird wie folgt erstellt:

```
\relative {
  \tempo 4 = 120
  c'2 d
  e4. d8 c2
}
```
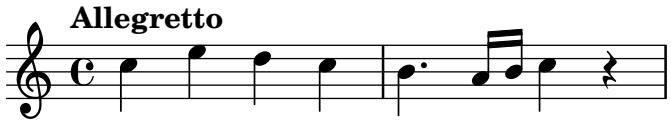


Metronombezeichnungen können auch für einen Zahlenbereich notiert werden:

```
\relative {
  \tempo 4 = 40 - 46
  c'4. e8 a4 g
  b,2 d4 r
}
```
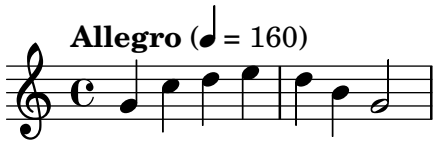


Anstelle dessen kann auch Text als Argument angegeben werden:

```
\relative {
  \tempo "Allegretto"
    c''4 e d c
    b4. a16 b c4 r4
}
```
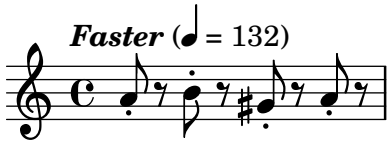


Wenn eine Metronombezeichnung und Text kombiniert wird, wird die Metronombezeichnung automatisch in Klammern gesetzt:

```
\relative {
  \tempo "Allegro" 4 = 160
  g'4 c d e
  d4 b g2
}
```
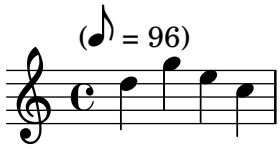


Der Text kann ein beliebiges Textbeschriftungsobjekt sein:

```
\relative {
  \tempo \markup { \italic Faster } 4 = 132
  a'8-. r8 b-. r gis-. r a-. r
}
```



Eine Metronombezeichnung in Klammern ohne Text kann erstellt werden, indem eine leere Zeichenkette hinzugefügt wird:

```
\relative {
  \tempo "" 8 = 96
  d''4 g e c
}
```



Ausgewählte Schnipsel

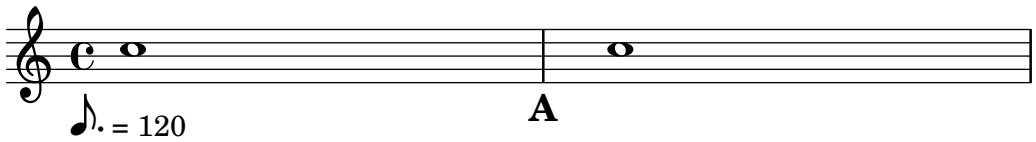
Metronom- und Übungszeichen unter das System setzen

Normalerweise werden Metronom- und Übungszeichen über dem Notensystem ausgegeben. Um sie unter das System zu setzen, muss die `direction`-Eigenschaft von `MetronomeMark` oder `RehearsalMark` entsprechend verändert werden.

```
\layout {
  ragged-right = ##f
}

{
  % Metronome marks below the staff
  \override Score.MetronomeMark.direction = #DOWN
  \tempo 8. = 120
  c''1

  % Rehearsal marks below the staff
  \override Score.RehearsalMark.direction = #DOWN
  \mark \default
  c''1
}
```
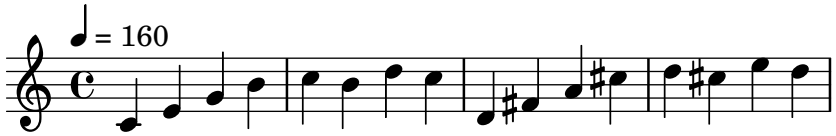


Das Tempo ohne Metronom-Angabe verändern

Um das Tempo für die MIDI-Ausgabe zu ändern, ohne eine Tempoangabe in den Noten auszugeben, kann die Metronombezeichnung unsichtbar gemacht werden:

```
\score {
  \new Staff \relative c' {
    \tempo 4 = 160
    c4 e g b
    c4 b d c
    \set Score.tempoHideNote = ##t
    \tempo 4 = 96
    d,4 fis a cis
    d4 cis e d
  }
  \layout { }
  \midi { }
```
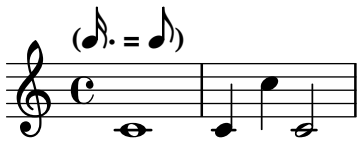
```
}
```



Eine Metronombezeichnung als Textbeschriftung erstellen

Neue Metronombezeichnungen können als Textbeschriftung erstellt werden, aber sie ändern nicht das Tempo für die MIDI-Ausgabe.

```
\relative c' {
  \tempo \markup {
    \concat {
      (
      \smaller \general-align #Y #DOWN \note { 16. } #UP
      " = "
      \smaller \general-align #Y #DOWN \note { 8 } #UP
      )
    }
  }
  c1
  c4 c' c,2
}
```



Zu Einzelheiten siehe Abschnitt 8.2 [Text formatieren], Seite 233.

Siehe auch

Glossar: Abschnitt “metronome” in *Glossar*, Abschnitt “metronomic indication” in *Glossar*, Abschnitt “tempo indication” in *Glossar*, Abschnitt “metronome mark” in *Glossar*.

Notationsreferenz: Abschnitt 8.2 [Text formatieren], Seite 233, Kapitel 23 [MIDI-Ausgabe], Seite 506.

Schnipsel: Abschnitt “Staff notation” in *Schnipsel*.

Referenz der Interna: Abschnitt “MetronomeMark” in *Referenz der Interna*.
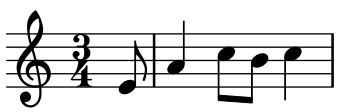
2.3.3 Auftakte

Verkleinerte Takte, wie etwa ein Auftakt, werden mit dem Befehl `\partial` notiert, dessen Syntax lautet:

```
\partial Dauer
```
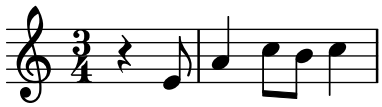
wobei *Dauer* die Länger der Noten darstellt, bevor der nächste vollständige Takt beginnt:

```
\time 3/4
\partial 8
e8 | a4 c8 b c4 |
```



Die *Dauer* kan ein beliebiger Wert kleiner als der vollständige Takt sein.

```
\relative {
  \time 3/4
  \partial 4.
  r4 e'8 | a4 c8 b c4 |
}
```
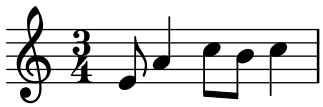


\partial *Dauer* kann auch folgendermaßen geschrieben werden:

```
\set Timing.measurePosition -Länge der Dauer
```

So wird etwa aus \partial 8:

```
\time 3/4
\set Timing.measurePosition = #(ly:make-moment -1/8)
e8 | a4 c8 b c4 |
```



Die Eigenschaft measurePosition (Takt-Position) enthält eine rationale Zahl, die anzeigt, wie groß der Abstand zum Taktanfang ist. Deshalb ist sie eine negative Zahl; \partial 4 wird also intern übersetzt zu -4 was soviel bedeutet wie: „Eine Viertel bleibt übrig vom ganzen Takt.“

Siehe auch

Glossar: Abschnitt “anacrusis” in *Glossar*.

Notationsreferenz: Abschnitt 2.6.1 [Verzierungen], Seite 112.
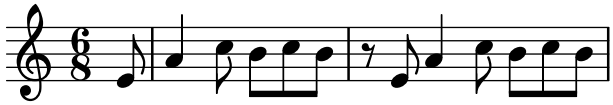
Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Referenz der Interna: Abschnitt “Timing_translator” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

\partial ist nur für den Anfang eines Stückes vorgesehen. Wenn der Befehl nach dem Anfang verwendet wird, können Warnungen oder Probleme auftreten. In solchem Fall sollten Sie \set Timing.measurePosition benutzen.
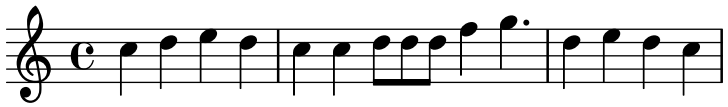
```
\time 6/8
\partial 8
e8 | a4 c8 b[ c b] |
\set Timing.measurePosition = #(ly:make-moment -1/4)
r8 e,8 | a4 c8 b[ c b] |
```



2.3.4 Musik ohne Metrum
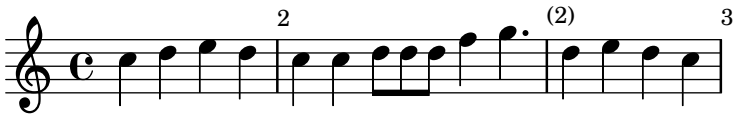
In Musik mit Metrum werden Taktstriche automatisch eingefügt und Taktzahlen automatisch berechnet. In Musik ohne Metrum hingegen (etwa Kadenzen) ist das nicht gewollt und kann „ausgeschaltet“ werden mit dem Befehl \cadenzaOn, um dann wieder zum normalen Verhalten mit \cadenzaOff zurückzukehren.

```
\relative c'' {
  c4 d e d
  \cadenzaOn
  c4 c d8[ d d] f4 g4.
  \cadenzaOff
  \bar "|"
  d4 e d c
}
```
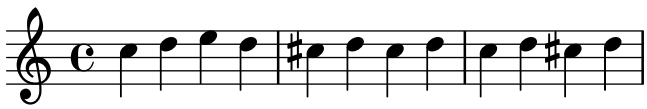


Taktnummerierung wird am Ende der Kadenz wieder aufgenommen:

```
% Show all bar numbers
\override Score.BarNumber.break-visibility = #all-visible
c4 d e d
\cadenzaOn
c4 c d8[ d d] f4 g4.
\cadenzaOff
\bar "|"
d4 e d c
```
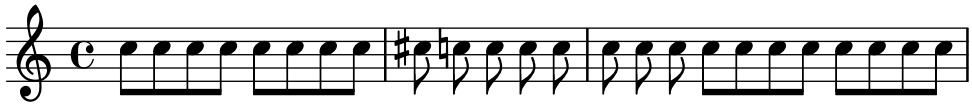


Ein neuer Takt wird nie innerhalb der Kadenz begonnen, auch wenn einer oder mehrere \bar-Befehle eingefügt ewrden. Darum müssen auch Erinnerungsversetzungszeichen manuell eingefügt werden. Siehe Abschnitt 1.1.3 [Versetzungszeichen], Seite 7.

```
c4 d e d
\cadenzaOn
cis4 d cis d
\bar "|"
cis4 d cis! d
\cadenzaOff
\bar "|"
```



Automatische Bebalkung wird durch \cadenzeOn ausgestellt. Darum müssen alle Balken in Kadenzen manuell eingegeben werden (siehe Abschnitt 2.4.3 [Manuelle Balken], Seite 94).

```
\relative {
  \repeat unfold 8 { c''8 }
  \cadenzaOn
  cis8 c c c c
  \bar"|"
  c8 c c
  \cadenzaOff
  \repeat unfold 8 { c8 }
}
```



Diese vordefinierten Befehle wirken sich auf alle Systeme in der Partitur aus, auch wenn sie nur in einer einzigen Stimme notiert werden. Um dieses Verhalten zu ändern, müsen Sie `Timing_translator` aus dem `Score`-Kontext in den `Staff`-Kontext verschieben, wie gezeigt in Abschnitt 2.3.5 [Polymetrische Notation], Seite 75.

Vordefinierte Befehle

`\cadenzaOn`, `\cadenzaOff`.

Siehe auch

Glossar: Abschnitt “cadenza” in *Glossar*.

Notationsreferenz: Abschnitt 35.6 [Sichtbarkeit von Objekten], Seite 619, Abschnitt 2.3.5 [Polymetrische Notation], Seite 75, Abschnitt 2.4.3 [Manuelle Balken], Seite 94, Abschnitt 1.1.3 [Versetzungszeichen], Seite 7.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Bekannte Probleme und Warnungen

Automatische Zeilen- und Seitenumbrüche werden nur an Taktstrichen vorgenommen, sodass „unsichtbare“ Taktstriche manuell eingefügt werden müssen, damit Umbrüche erlaubt werden:

```
\bar ""
```

Man muss explizit einen `Voice`-Kontext erstellen, wenn man ein Stück mit `cadenzaOn` beginnen will, weil sonst ein seltsamer Fehler auftreten kann.

```
\new Voice {
  \relative c' {
    \cadenzaOn
    c16[^"Solo Free Time" d e f] g2.
    \bar "||"
    \cadenzaOff
  }
}
```

2.3.5 Polymetrische Notation

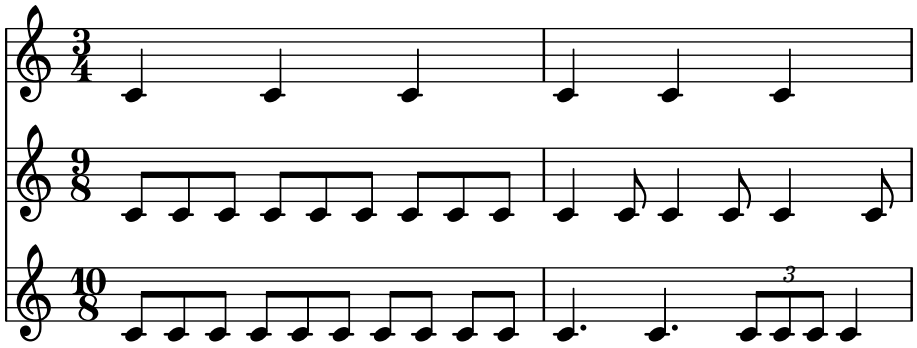
Polymetrische Notation ist direkt unterstützt, oder indem man das sichtbare Taktart-Symbol verändert und zusätzlich die Notendauern skaliert.

Unterschiedliche Taktarten mit gleicher Taktlänge

Für jedes System wird eine gemeinsame Taktart gesetzt und dann die Einstellung von `timeSignature` auf den gewünschten Bruch geändert. Mit dem Befehl `\scaleDurations` werden die Dauern der Noten auf jedem System dann auf die gemeinsame Taktart skaliert. Siehe auch Abschnitt 2.1.3 [Tondauern skalieren], Seite 52.

In folgendem Beispiel werden Noten mit den Taktarten 3/4, 9/8 und 10/8 parallel notiert. Im zweiten System werden die gezeigten Dauern mit 2/3 multipliziert, da 2/3 x 9/8 = 3/4, und im dritten System werden die gezeigten Dauern mit 3/5 multipliziert, da 3/5 x 10/8 = 3/4. Oft wird es nötig sein, Balken manuell zu setzen, weil die Skalierung sich auch auf die automatische Bebalkung auswirkt.

```
\relative <<
  \new Staff {
    \time 3/4
    c'4 c c |
    c4 c c |
  }
  \new Staff {
    \time 3/4
    \set Staff.timeSignature = 9/8
    \scaleDurations 2/3 {
      \repeat unfold 3 { c8[ c c] }
      \repeat unfold 3 { c4 c8 }
    }
  }
  \new Staff {
    \time 3/4
    \set Staff.timeSignature = 10/8
    \scaleDurations 3/5 {
      \repeat unfold 2 { c8[ c c] }
      \repeat unfold 2 { c8[ c] } |
      c4. c4. \tuplet 3/2 { c8[ c c] } c4
    }
  }
>>
```



Unterschiedlichen Taktarten mit unterschiedlicher

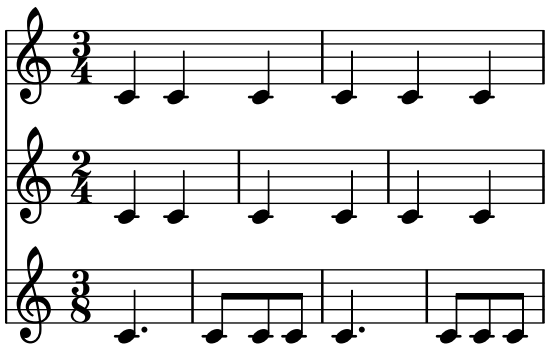
Taktlänge

Jedes System kann auch eine eigene unabhängige Taktart erhalten. Dazu muss der `Timing_translator` und der `Default_bar_line_engraver` in den `Staff`-Kontext verschoben werden.

```
\layout {
  \context {
    \Score
    \remove Timing_translator
  }
  \context {
    \Staff
    \consists Timing_translator
  }
}
```

```
% Now each staff has its own time signature.

\relative <<
  \new Staff {
    \time 3/4
    c'4 c c |
    c4 c c |
  }
  \new Staff {
    \time 2/4
    c4 c |
    c4 c |
    c4 c |
  }
  \new Staff {
    \time 3/8
    c4. |
    c8 c c |
    c4. |
    c8 c c |
  }
>>
```
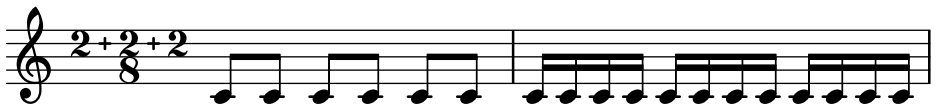


Zusammengesetzte Taktarten

Taktarten aus mehreren Teilen werden mit der Funktion `\timeAbbrev` erstellt. Die Syntax ist folgende:

```
\timeAbbrev #'(Liste aus Listen)
```

Die einfachste Konstruktion ist eine einzige Liste, wobei die letzte Zahl den Nenner des Bruches darstellt, während die vorherkommenden Zahlen die Zähler sind.

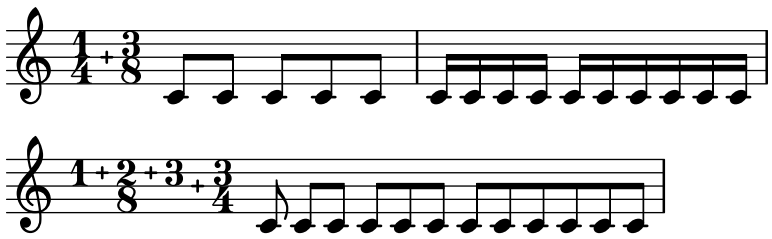
```
\relative {
  \timeAbbrev #'((2 2 2 8))
  \repeat unfold 6 c'8 \repeat unfold 12 c16
}
```



Kompliziertere Taktarten können durch zusätzliche Listen erstellt werden (von Klammern abgegrenzt). Automatische Balken werden entsprechend der Werte angepasst.

```
\relative {
  \timeAbbrev #'((1 4) (3 8))
  \repeat unfold 5 c'8 \repeat unfold 10 c16
}

\relative {
  \timeAbbrev #'((1 2 3 8) (3 4))
  \repeat unfold 12 c'8
}
```



Siehe auch

Glossar: Abschnitt “polymetric” in *Glossar*, Abschnitt “polymetric time signature” in *Glossar*, Abschnitt “meter” in *Glossar*.

Notationsreferenz: Abschnitt 2.3.1 [Taktangabe], Seite 65, Abschnitt 2.4.1 [Automatische Balken], Seite 81, Abschnitt 2.4.3 [Manuelle Balken], Seite 94, Abschnitt 2.1.3 [Tondauern skalieren], Seite 52.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Referenz der Interna: Abschnitt “TimeSignature” in *Referenz der Interna*, Abschnitt “Timing_translator” in *Referenz der Interna*, Abschnitt “Staff” in *Referenz der Interna*.
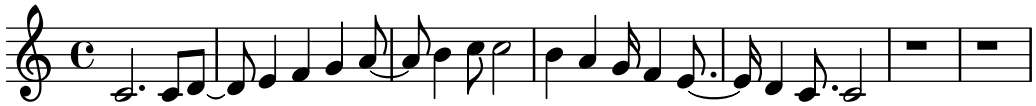
Bekannte Probleme und Warnungen

Wenn unterschiedliche Taktarten parallel benutzt werden, werden Noten auf demselben musikalischen Moment horizontal auf die gleiche Position gesetzt. Die unterschiedlichen Taktlinien führen allerdings dazu, dass die Noten nicht ganz so regelmäßig gesetzt werden, wie es ohne unterschiedliche Taktarten der Fall wäre.

2.3.6 Automatische Aufteilung von Noten

Lange Noten, die über Taktlinien hinüberreichen, können automatisch in übergebundene Noten aufgeteilt werden. Dieses Verhalten erreicht man, indem der Abschnitt “Note_heads_engraver” in *Referenz der Interna* mit dem Abschnitt “Completion_heads_engraver” in *Referenz der Interna* ausgetauscht wird. Auf gleiche Art können lange Pausen, die über Taktgrenzen reichen, automatisch aufgeteilt werden, indem man den `Rest_engraver` mit dem `Completion_rest_engraver` ersetzt. Im nächsten Beispiel werden Noten und Pausen, die über die Taktlinie dauern, aufgeteilt; Noten werden auch übergebunden.

```
\new Voice \with {
  \remove Note_heads_engraver
  \consists Completion_heads_engraver
  \remove Rest_engraver
  \consists Completion_rest_engraver
}
\relative {
  c'2. c8 d4 e f g a b c8 c2 b4 a g16 f4 e d c8. c2 r1*2
}
```



Dieser Engraver teilt alle Noten und Pausen auf, die über eine Taktlinie dauern und fügt für Noten Bindebögen hinzu. Er kann unter Anderem dann nützlich sein, wenn man komplexe Partituren auf Fehler überprüfen möchte: Wenn die Takte nicht vollständig gefüllt sind, zeigt die Überbindung genau an, wie viele Notenwerte noch in dem jeweiligen Takt fehlen.

Siehe auch

Glossar: Abschnitt “tie” in *Glossar*

Handbuch zum Lernen: Abschnitt “Was sind Engraver?” in *Handbuch zum Lernen*, Abschnitt “Engraver hinzufügen und entfernen” in *Handbuch zum Lernen*.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Referenz der Interna: Abschnitt “Note_heads_engraver” in *Referenz der Interna*, Abschnitt “Completion_heads_engraver” in *Referenz der Interna*, Abschnitt “Rest_engraver” in *Referenz der Interna*, Abschnitt “Completion_rest_engraver” in *Referenz der Interna*, Abschnitt “Forbid_line_break_engraver” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

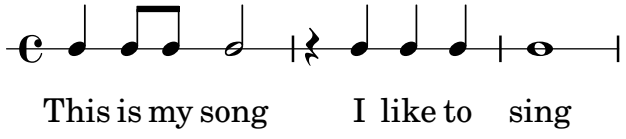
Nicht alle Notenwerte (besonders wenn sie andere rhythmische Aufteilungen beinhalten) können exakt durch normale Noten und Punktierungen wiedergegeben werden. Der Engraver setzt aber trotzdem keine Triolen etc.

`Completion_heads_engraver` wirkt sich nur auf Noten aus; Pausen werden nicht aufgeteilt.

2.3.7 Melodierhythmus anzeigen

Manchmal soll nur der Rhythmus einer Melodie dargestellt werden. Das erreicht man mit einem Rhythmus-Notensystem. Alle Tonhöhen werden auf eine Linie reduziert und das System hat auch nur eine einzige Linie.

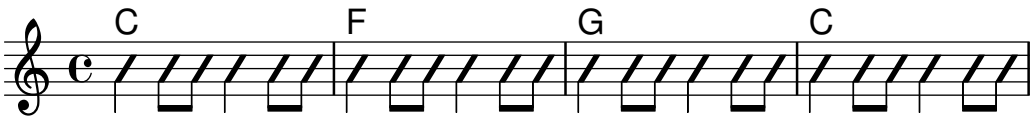
```
<<
  \new RhythmicStaff {
    \new Voice = "myRhythm" \relative {
      \time 4/4
      c'4 e8 f g2
      r4 g g f
      g1
    }
  }
  \new Lyrics {
    \lyricsto "myRhythm" {
      This is my song
      I like to sing
    }
  }
>>
```



Akkordnotation für Gitarren bezeichnet auch oft zusätzlich den geschlagenen Rhythmus. Das kann notiert werden unter Verwendung des `Pitch_squash_engraver` und indem Tonhöhenimprovisation eingeschaltet wird mit `\improvisationOn`.

```
<<
  \new ChordNames {
    \chordmode {
      c1 f g c
    }
  }

  \new Voice \with {
    \consists Pitch_squash_engraver
  } \relative c'' {
    \improvisationOn
    c4 c8 c c4 c8 c
    f4 f8 f f4 f8 f
    g4 g8 g g4 g8 g
    c4 c8 c c4 c8 c
  }
>>
```



Vordefinierte Befehle

`\improvisationOn`, `\improvisationOff`.

Ausgewählte Schnipsel

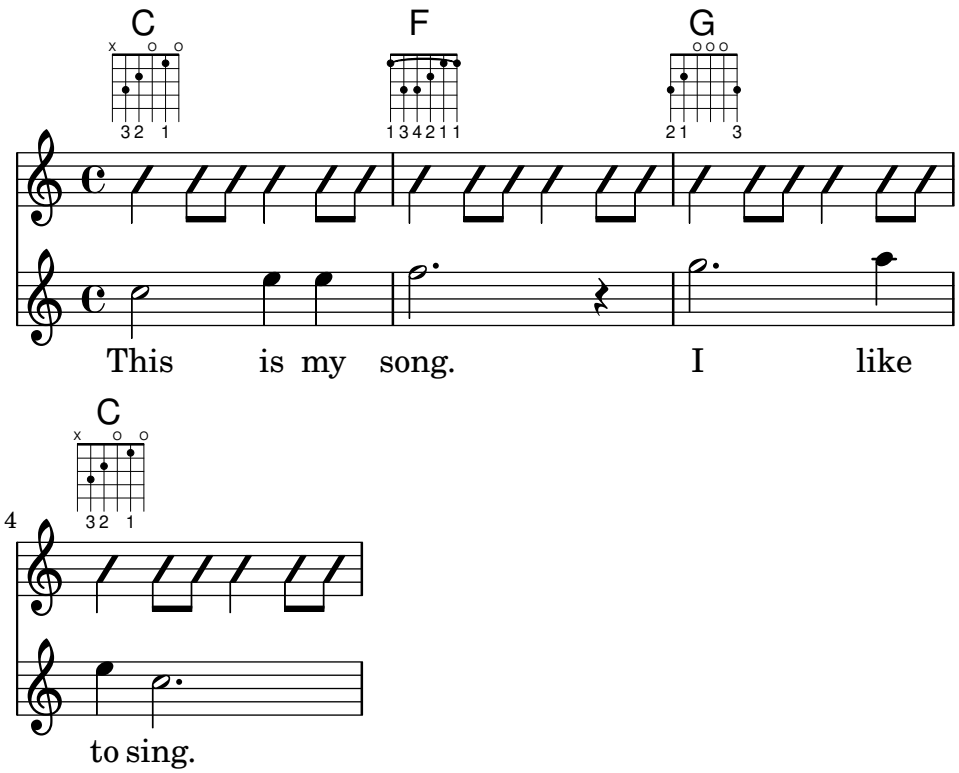
Schlagrhythmus für Gitarren

In Gitarrennotation kann neben Melodie, Akkordbezeichnungen und Bunddiagrammen auch der Schlagrhythmus angegeben werden.

```
\include "predefined-guitar-fretboards.ly"

<<
  \new ChordNames \chordmode {
    c1 | f | g | c
  }
  \new FretBoards \chordmode {
    c1 | f | g | c
  }
  \new Voice \with {
    \consists "Pitch_squash_engraver"
  } \relative c'' {
    \improvisationOn
    c4 c8 c c4 c8 c
    f4 f8 f f4 f8 f
    g4 g8 g g4 g8 g
    c4 c8 c c4 c8 c
  }
```

```
  \new Voice = "melody" \relative c'' {
    c2 e4 e4
    f2. r4
    g2. a4
    e4 c2.
  }
  \new Lyrics \lyricsto "melody" {
    This is my song.
    I like to sing.
  }
>>
```



Siehe auch

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

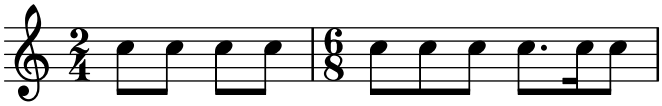
Referenz der Interna: Abschnitt “RhythmicStaff” in *Referenz der Interna*, Abschnitt “Pitch_squash_engraver” in *Referenz der Interna*.

2.4 Balken

2.4.1 Automatische Balken

LilyPond setzt Balken (engl. beam) automatisch.
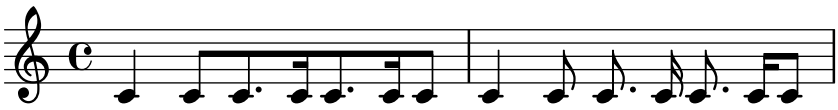
```
\relative c'' {
  \time 2/4 c8 c c c
  \time 6/8 c8 c c c8. c16 c8
}
```



Wenn diese automatischen Entscheidungen nicht gut genug sind, können die Balken auch explizit eingegeben werden, siehe Abschnitt 2.4.3 [Manuelle Balken], Seite 94. Balken *müssen* auch auf diese Weise eingegeben werden, wenn sie über Pausen hinwegreichen sollen.

Wenn automatische Bebalkung nicht benötigt wird, kann sie mit dem Befehl `\autoBeamOff` aufgehoben werden und mit dem Befehl `\autoBeamOn` wieder eingeschaltet werden.

```
\relative c' {
  c4 c8 c8. c16 c8. c16 c8
  \autoBeamOff
  c4 c8 c8. c16 c8.
  \autoBeamOn
  c16 c8
}
```



Achtung: Wenn Balken eingesetzt werden, um Melismen in Gesang zu notieren, sollte die automatische Bebalkung mit `\autoBeamOff` ausgeschaltet werden und die Balken manuell notiert werden. Die Benutzung von `\partCombine` zusammen mit `\autoBeamOff` kann zu unbeabsichtigten Ergebnissen führen. Siehe die Schnipsel für mehr Information.

Balkenmuster, die sich von den automatisch erstellen unterscheiden, können erstellt werden, siehe Abschnitt 2.4.2 [Einstellung von automatischen Balken], Seite 84.

Vordefinierte Befehle

`\autoBeamOff`, `\autoBeamOn`.

Ausgewählte Schnipsel

Balken über Zeilenumbrüche

Zeilenumbrüche sind normalerweise während Balken verboten. Das kann geändert werden.

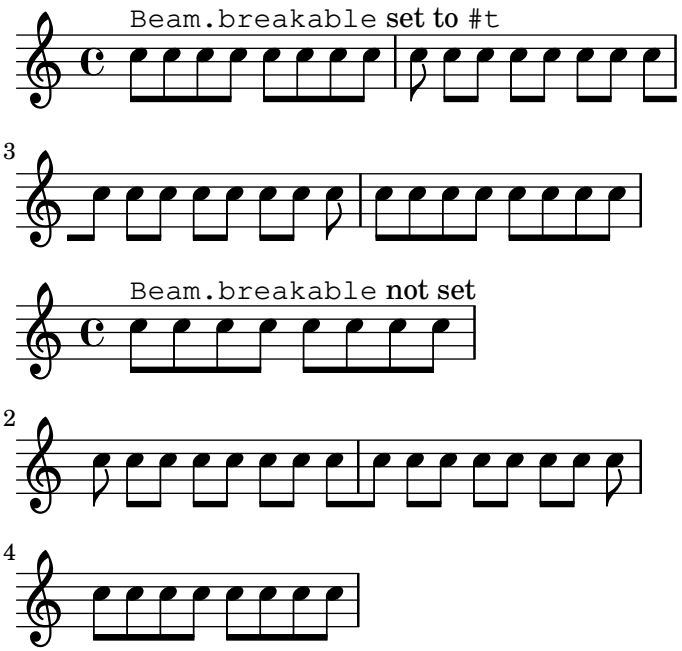
```
music = {
  \*8 c8
  c8 \*7 { c[ c] }  c
  \*8 c8
}

\relative c'' {
  <>^\markup { \typewriter Beam.breakable set to \typewriter "#t" }
  \override Beam.breakable = ##t
  \music
}

\relative c'' {
  <>^\markup { \typewriter Beam.breakable not set }
  \music
}

\paper {
```
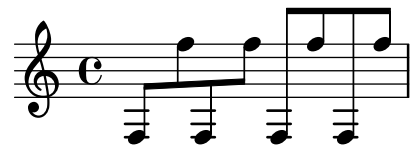
```
  line-width = 100\mm
}
```



Balken für weit auseinander liegende Noten ändern

Balken mit Hälsen in unterschiedliche Richtungen werden automatisch erstellt, wenn ein großer Sprung zwischen Tonhöhen gefunden wird. Dieses Verhalten kann durch die auto-knee-gap-Eigenschaft beeinflusst werden. Ein derartiger Knie-Balken wird erstellt, wenn der Abstand größer ist als der Wert von auto-knee-gap plus der Dicke des Balkens (was von der Notendauer und der Neigung des Balkens abhängt). Der Standardwert von auto-knee-gap ist 5.5 Notensystemabstände.

```
{
  f8 f''8 f8 f''8
  \override Beam.auto-knee-gap = #6
  f8 f''8 f8 f''8
}
```



Partcombine and \autoBeamOff

The function of \autoBeamOff when used with \partCombine can be difficult to understand. It may be preferable to use
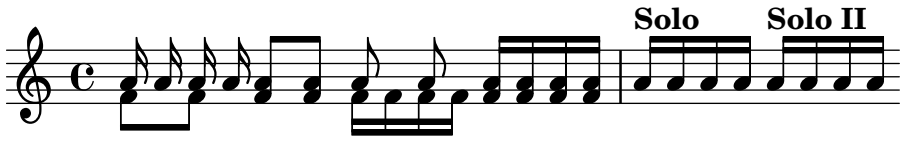
```
\set Staff.autoBeaming = ##f
```

instead to ensure that auto-beaming is turned off for the entire staff. Use this at a spot in your score where no beam generated by the auto-beamer is still active.

Internally, \partCombine works with four voices – up-stem single, down-stem single, combined, and solo. In order to use \autoBeamOff to stop all auto-beaming when used with \partCombine, it is necessary to use *four* calls to \autoBeamOff.

```
{
  % \set Staff.autoBeaming = ##f % turns off all auto-beaming

  \partCombine {
    \autoBeamOff   % applies to split up-stems
    \*4 a'16
    % \autoBeamOff % applies to combined stems
    \*4 a'8
    \*4 a'16
    % \autoBeamOff % applies to solo
    \*4 a'16
    r4
  } {
    % \autoBeamOff % applies to split down-stems
    \*4 f'8
    \*8 f'16 |
    r4
    \*4 a'16
  }
}
```



Siehe auch

Notationsreferenz: Abschnitt 2.4.3 [Manuelle Balken], Seite 94, Abschnitt 2.4.2 [Einstellung von automatischen Balken], Seite 84.

Installierte Dateien: `scm/auto-beam.scm`.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Referenz der Interna: Abschnitt “Auto_beam_engraver” in *Referenz der Interna*, Abschnitt “Beam_engraver” in *Referenz der Interna*, Abschnitt “Beam” in *Referenz der Interna*, Abschnitt “BeamEvent” in *Referenz der Interna*, Abschnitt “BeamForbidEvent” in *Referenz der Interna*, Abschnitt “beam-interface” in *Referenz der Interna*, Abschnitt “unbreakable-spanner-interface” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Die Eigenschaften eines Balkens werden am *Beginn* seiner Konstruktion bestimmt. Alle zusätzlichen Änderungen der Balkeneigenschaften, die auftreten, bevor der Balken zuende ist, werden nicht ausgewertet, bevor nicht der *nächste* Balken beginnt.

2.4.2 Einstellung von automatischen Balken

Wenn die automatischen Balken angeschaltet sind, wird die Platzierung der automatischen Balken von drei Kontexteigenschaften bestimmt: `beatBase`, `beatStructure` und `beamExceptions`. Wenn eine `beamExceptions`-Regel für die aktuelle Taktart definiert ist, wird diese Regel zur Bestimmmung der Balkenplatzierung herangezogen. Wenn keine `beamExceptions`-Regel für die aktuelle Taktart vorhanden ist, wird die Platzierung der Balken durch die Einstellungen von `beatBase` und `beatStructure` bestimmt.

Standardmäßig sind `beamExceptions`-Regel für alle häufigen Taktarten vordefiniert, sodass die `beamExceptions`-Regeln deaktiviert werden müssen, wenn die Balkenplatzierung aufgrund
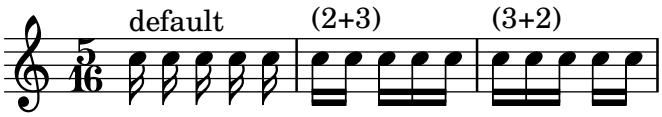
von beatBase und beatStructure stattfinden soll. Die beamExceptions-Regeln werden deaktiviert durch

```
\set Timing.beamExceptions = #'()
```

Bebalkung basierend auf beatBase *und* beatStructure

In den meisten Fällen enden automatische Balken am Ende eines Taktes. Die Endpunkte für Schläge werden durch die Kontexteigenschaften beatBase und beatStructure bestimmt. beatStructure ist eine Scheme-Liste, die die Länge jedes Schlages im Takt in Einheiten von beatBase angibt. Der Standard von beatBase ist Eins durch den Numerator der Taktangabe. Der Standardwert jeder Längeneinheit beatBase ist ein einzelner Taktschlag.

```
\time 5/16
c16^"default" c c c c |
\set Timing.beatStructure = 2,3
c16^"(2+3)" c c c c |
\set Timing.beatStructure = 3,2
c16^"(3+2)" c c c c |
```

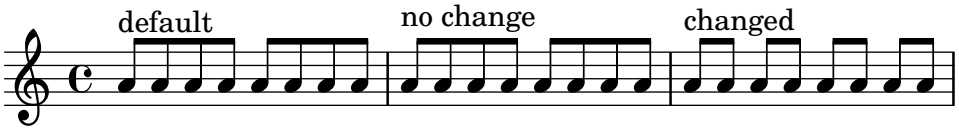


Wenn eine häufige Taktart benützt wird, *muss* beamExceptions deaktiviert werden, damit beatStructure funktionieren kann. Der \set Timing.beamExceptions = #'()-Befehl kann immer eingefügt werden, wenn die Bebalkung von beatStructure bestimmt werden soll.

```
\time 4/4
a8^"default" a a a a a a a

\set Timing.beatBase = #1/4
\set Timing.beatStructure = 1,1,1,1
a8^"no change" a a a a a a a

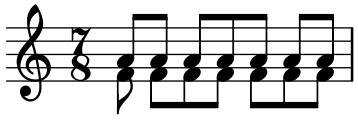
\set Timing.beamExceptions = #'()
\set Timing.beatBase = #1/4
\set Timing.beatStructure = 1,1,1,1
a8^"changed" a a a a a a a
```



Balkenregelveränderungen können auf bestimmte Kontexte beschränkt werden. Wenn keine Regeln in einen unteren Kontext definiert sind, gelten die Regeln des höheren Kontext, in dem sich der niedrigere befindet.

```
\new Staff <<
  \time 7/8
  \set Staff.beatStructure = 2,3,2
  \new Voice = one {
    \relative {
        a'8 a a a a a a
    }
  }
```
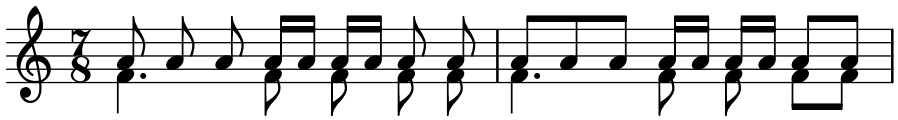
```
  \new Voice  = two {
    \relative {
      \voiceTwo
      \set Voice.beatStructure = 1,3,3
      f'8 f f f f f f
    }
  }
>>
```



Wenn mehrere Stimmen eingesetzt werden, muss der `Staff`-Kontext definiert werden, wenn die Balkenregeln auf alle Stimmen des Systems angewendet werden sollen:

```
\time 7/8
% rhythm 3-1-1-2
% Context applied to Voice by dafault --  does not work correctly
% Because of autogenerated voices, all beating will
% be at beatBase #1/8
\set beatStructure = 3,1,1,2
<< {a8 a a a16 a a a a8 a} \\ {f4. f8 f f f} >>

% Works correctly with context Staff specified
\set Staff.beatStructure = 3,1,1,2
<< {a8 a a a16 a a a a8 a} \\ {f4. f8 f f f} >>
```



Der Wert von `beatBase` kann angepasst werden, um das Bebalkungsverhalten zu ändern, wenn gewünscht. In diesem Fall muss der Wert von `beatStructure` so gesetzt werden, dass er kompatibel mit dem neuen Wert von `beatBase` ist.

```
\time 5/8
\set Timing.beatBase = #1/16
\set Timing.beatStructure = 7,3
\repeat unfold 10 { a16 }
```
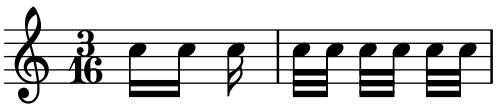


Der Standardwert von `beatBase` ist Eins durch den Denominator der Taktangabe. Alle Ausnahmen dieses Standards finden sich in der Datei `scm/time-signature-settings.scm`.

Bebalkung basierend auf `beamExceptions`

Besondere automatische Bebalkungsregeln (außer dass ein Balken auf einem Taktschlag aufhört) sind in der `beamExceptions`-Eigenschaft definiert.

```
\time 3/16
\set Timing.beatStructure = 2,1
\set Timing.beamExceptions =
  #'(                          ;start of alist
     (end .                    ;entry for end of beams
      (                        ;start of alist of end points
       ((1 . 32) . (2 2 2))    ;rule for 1/32 beams -- end each 1/16
      )))                      %close all entries
c16 c c |
\repeat unfold 6 { c32 } |
```



beamExceptions ist eine Aliste mit einem Schlüssel der Regeltypen (rule-type) und einem Wert der Bebalkungsregeln (beaming-rules).

Im Moment ist der einzige mögliche rule-type 'end für ein Balkenende.

Beaming-rules ist eine Scheme-Aliste (oder eine paarige Liste), die den Balkentyp und die Gruppierung anzeigt, die auf Balken angewendet werden, welche Noten mit einer kürzesten Dauer des Balkentyps enthalten.

```
#'((beam-type1 . grouping-1)
   (beam-type2 . grouping-2)
   (beam-type3 . grouping-3))
```

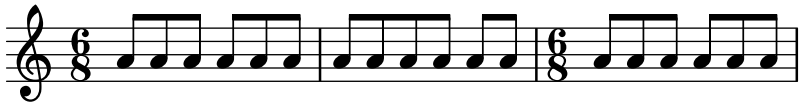
Beam-type ist ein Scheme-Paar, das die Dauer eines Balkens anzeigt, etwa (1 . 16) für ein Sechszehntel.

Grouping ist eine Scheme-Liste, die die auf den Balken anzuwendene Gruppierung anzeigt. Die Gruppierung wird in Einheiten des Balkentyps angegeben.

> **Achtung:** Ein beamExceptions-Wert muss eine *vollständige* Ausnahme-Liste sein. Das heißt, dass jede Ausnahme, die angewendet werden soll, auch in die Einstellungen mit aufgenommen werden muss. Es ist nicht möglich, nur eine der Einstellungen zu ändern, zu entfernen oder hinzuzufügen. Das mag seltsam erscheinen, bedeutet aber, dass die aktuellen Balkenregeln bekann sein müssen, um ein neues Bebalkungsmuster definieren zu können.

Wenn die Taktart geändert wird, werden neue Standardwerte für Timing.beatBase, Timing.beatStructure und Timing.beamExceptions definiert. Wenn die Taktart definiert wird, werden die automatischen Bebalkungsregeln für den Timing-Kontext auf den Standard zurückgesetzt.
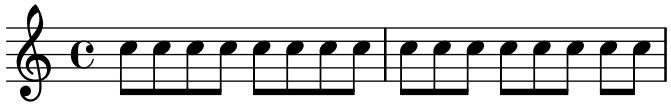
```
\relative a' {
  \time 6/8
  \repeat unfold 6 { a8 }
  % group (4 + 2)
  \set Timing.beatStructure = 4,2
  \repeat unfold 6 { a8 }
  % go back to default behavior
  \time 6/8
  \repeat unfold 6 { a8 }
}
```



Die automatischen Standardeinstellungen für die Bebalkung einer Taktart werden in der Datei `scm/time-signature-settings.scm` bestimmt. Änderungen der automatischen Bebalkungsregeln für eine Taktart werden in Abschnitt 2.3.1 [Taktangabe], Seite 65, beschrieben.
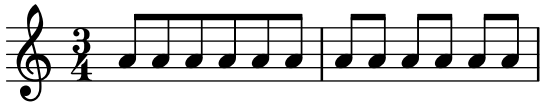
Die meisten automatischen Bebalkungsregeln für eine Taktart enthalten einen Eintrag für `beamExceptions`. Beispielsweise wird in einem 4/4-Takt versucht, den Takt in zwei Hälften zu teilen, wenn nur Achtelnoten vorkommen. Die `beamExceptions`-Regel kann die `beatStructure`-Einstellung überschreiben, wenn `beamExceptions` nicht zurückgesetzt wird:

```
\time 4/4
\set Timing.beatBase = #1/8
\set Timing.beatStructure = 3,3,2
% This won't beam (3 3 2) because of beamExceptions
\repeat unfold 8 {c''8} |
% This will beam (3 3 2) because we clear beamExceptions
\set Timing.beamExceptions = #'()
\repeat unfold 8 {c''8}
```
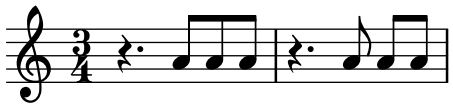


Auf gleiche Art werden Achtelnoten im 3/4-Takt über den ganzen Takt hin mit Balken versehen. Damit Achtelnoten im 3/4-Takt auf jedem Schlag einen neuen Balken erhalten, muss `beamExceptions` verändert werden:

```
\time 3/4
% by default we beam in (6) due to beamExceptions
\repeat unfold 6 {a8} |
% This will beam (1 1 1) due to beatLength
\set Timing.beamExceptions = #'()
\repeat unfold 6 {a8}
```



In Notenstichen der romantischen und klassischen Periode wird teilweise auch ein halber Takt Achtelnoten im 3/4-Takt mit einem Balken versehen, moderner Notenstich vermeidet dies jedoch, um nicht den falschen Eindruck eines 6/8-Taktes entstehen zu lassen (siehe Gould, S. 153). Eine ähnliche Situation entsteht im 3/8-Takt. Dieses Verhalten wird durch die Eigenschaft `beamHalfMeasure` bestimmt, welche sich nur auf Takte mit einer 3 im Zähler auswirkt:

```
\relative a' {
  \time 3/4
  r4. a8 a a |
  \set Timing.beamHalfMeasure = ##f
  r4. a8 a a |
}
```



Wie die automatische Bebalkung funktioniert

Wenn die automatische Bebalkung aktiviert ist, wird die Platzierung der automatischen Balken durch die Kontexteigenschaften `beatBase`, `beatStructure` und `beamExceptions` bestimmt.

Die folgenden Regeln, in der Reihenfolge ihrer Priorität, gelten, wenn das Aussehen der Balken bestimmt wird:

- Wenn ein manueller Balken mit `[...]` definiert ist, wird er gesetzt, andernfalls
- wenn eine Balkenendung-Regel für den Balkentyp in `beamExceptions` definiert ist, wird sie verwendet, um die gültigen Plätze für Balkenenden zu berechnen, andernfalls
- wenn eine Balkenendung-Regel für einen größeren Balkentyp in `beamExceptions` definiert ist, wird sie verwendet, um die gültigen Plätze für Balkenenden zu berechnen, andernfalls
- benutze die Werte von `beatBase` und `beatStructure`, um die Enden der Balken im Takt zu definieren und beende Balken am Ende jedes Taktes.

In den oben genannten Regeln ist der Balkentyp die Dauer der kürzesten Note der bebalkten Gruppe.

Die Standardbebalkungsregeln finden sich in der Datei `scm/time-signature-settings.scm`.

Ausgewählte Schnipsel

Subdividing beams

The beams of consecutive 16th (or shorter) notes are, by default, not subdivided. That is, the beams of more than two stems stretch over the entire group of notes without a break. This behavior can be modified to subdivide the beams into sub-groups by setting the property `subdivideBeams` to `#t`. When set, beams are subdivided at (rhythmic) intervals to match the metric value of the subdivision.

Using the properties `beamMinimumSubdivision` and `beamMaximumSubdivision` it is possible to configure the limits of automatic beam subdivision, namely the minimum and maximum rhythmic lengths at which beamlets are removed. The default values are `0` for the former and `+inf.0` for the latter, making LilyPond subdivide beams as much as possible.

There are two special cases to consider.

- If the numerator of `beamMaximumSubdivision` is not a power of 2, the rhythmic lengths considered for subdivision are `beamMaximumSubdivision` divided by powers of 2 that stay greater than or equal to `beamMinimumSubdivision`.
- If `beamMaximumSubdivision` is smaller than `beamMinimumSubdivision`, the depth of beam subdivisions is limited by `beamMaximumSubdivision`, but not the frequency and rhythmic intervals, therefore possibly deviating from the correct, expected metric value.

If `respectIncompleteBeams` is set to `#t`, incomplete subdivisions with more than two stems are treated as an ‚extension‘ of the previous subdivision group, i.e., the length of the previous subdivision group gets extended to also cover the incomplete subdivision. If set to `#f` (which is the default), a new subdivision group gets started instead.

```
\relative c'' {
  \time 1/4

  <>^"default"
  c32 c c c c c c c

  <>^"with subdivision"
  \set subdivideBeams = ##t
  c32 c c c c c c c
```

```
  <>^"min 1/8"
  \once \set beamMinimumSubdivision = #1/8
  c32 c c c c c c c

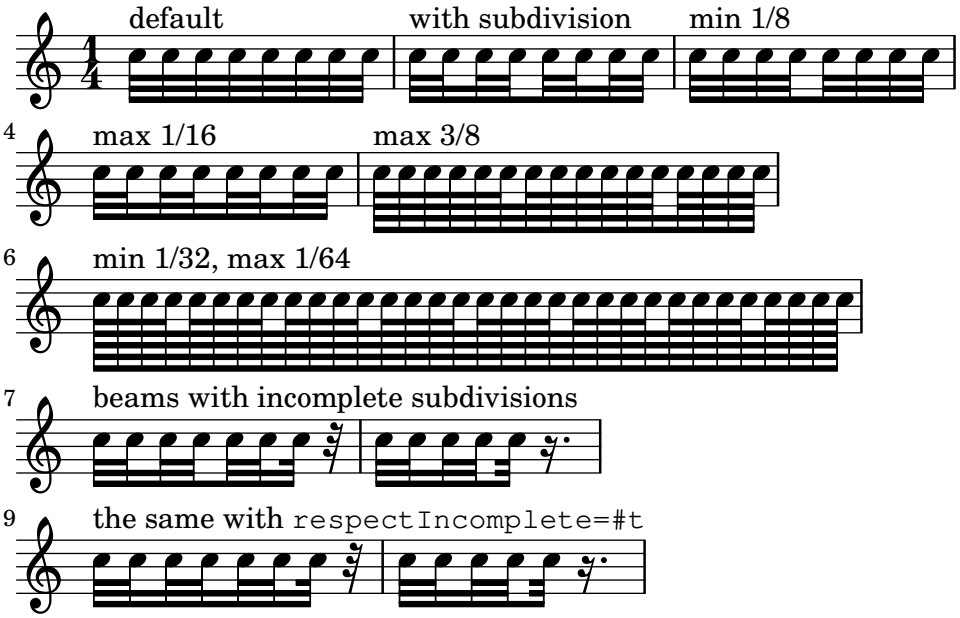
  <>^"max 1/16"
  \once \set beamMaximumSubdivision = #1/16
  c32 c c c c c c c

  <>^"max 3/8"
  \once \set beamMaximumSubdivision = #3/8
  \*16 c64

  <>^"min 1/32, max 1/64"
  % Set maximum beam subdivision interval to 1/64 to limit
  % subdivision depth, despite not being metrically correct.
  \once \set beamMinimumSubdivision = #1/32
  \once \set beamMaximumSubdivision = #1/64
  \*32 c128
  \break

  <>^"beams with incomplete subdivisions"
  c32 c c c c c c r32
  c32 c c c c r16.
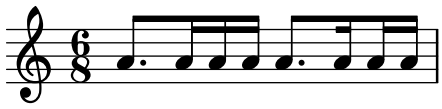
  <>^\markup { "the same with"
               \typewriter { "respectIncomplete=#t" } }
  \set respectIncompleteBeams = ##t
  % The incomplete subgroup extends the completed subgroup.
  c32 c c c c c c r32
  % No visual change since we have only two stems in the
  % incomplete subgroup.
  c32 c c c c r16.
}
```



Bebalkung nach Taktschlag

Sekundäre Balken können in die Richtung gesetzt werden, die ihrer rhythmischen Zugehörigkeit entspricht. Der erste Balken ist zusammengefasst (Standard), der zweite Sechszehntelbalken zeigt den Taktschlag an.

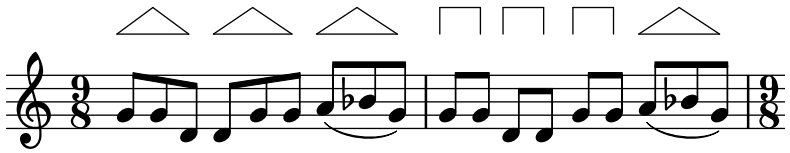
```
\relative c'' {
  \time 6/8
  a8. a16 a a
  \set strictBeatBeaming = ##t
  a8. a16 a a
}
```

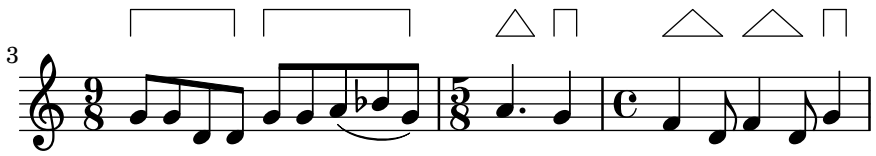


Dirigierzeichen, Taktgruppenzeichen

Optionen, mit denen die Balken in einem Takt gruppiert werden, sind durch die Scheme-Funktion `set-time-signature` erhältlich, die drei Argumente braucht: Die Zahl der Taktschläge, die Länge des Schlages und die interne gruppieren von Balken in dem Takt. Wenn der `Measure_grouping_engraver` hinzugefügt worden ist, erstellt diese Funktion auch `MeasureGrouping`-(Taktgruppen)-Zeichen. Derartige Zeichen erleichtern das Lesen von rhythmisch komplexer Musik. In dem Beispiel ist der 9/8-Takt in 2, 2, 2 und 3 aufgeteilt. Das wird der `set-time-signature`-Funktion als das dritte Argument mitgegeben: `'(2 2 2 3)`:

```
\score {
  \new Voice \relative c'' {
    \time 9/8
    g8 g d d g g a( bes g) |
    \set Timing.beatStructure = 2,2,2,3
    g8 g d d g g a( bes g) |
    \time 4,5 9/8
    g8 g d d g g a( bes g) |
    \time 5/8
    a4. g4 |
    \time 3,3,2 4/4
    \set Timing.beatBase = #1/8
    f4 d8 f4 d8 g4
  }
  \layout {
    \context {
      \Staff
      \consists "Measure_grouping_engraver"
    }
  }
}
```
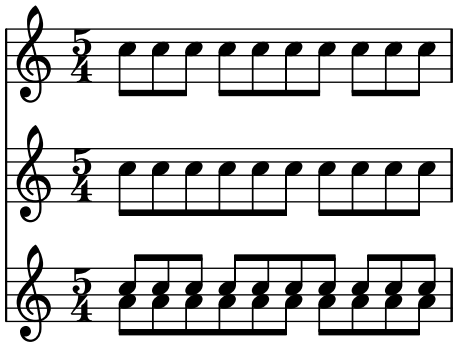




Balkenenden auf Score-Ebene

Balkenenderegeln, die im `Score`-Kontext definiert werden, wirken sich auf alle Systeme aus, können aber auf `Staff`- und `Voice`-Ebene neu verändert werden:

```
\relative c'' {
  \time 5/4
  % Set default beaming for all staves
  \set Score.beatBase = #1/8
  \set Score.beatStructure = 3,4,3
  <<
    \new Staff {
      c8 c c c c c c c c c
    }
    \new Staff {
      % Modify beaming for just this staff
      \set Staff.beatStructure = 6,4
      c8 c c c c c c c c c
    }
    \new Staff {
      % Inherit beaming from Score context
      <<
        {
          \voiceOne
          c8 c c c c c c c c c
        }
        % Modify beaming for this voice only
        \new Voice {
          \voiceTwo
          \set Voice.beatStructure = 6,4
          a8 a a a a a a a a a
        }
      >>
    }
  >>
}
```



Siehe auch

Installierte Dateien: `scm/beam-settings.scm`.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

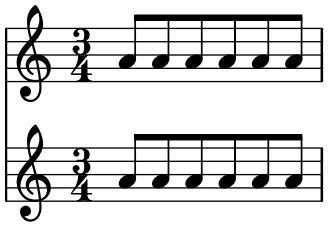
Referenz der Interna: Abschnitt “Auto_beam_engraver” in *Referenz der Interna*, Abschnitt “Beam” in *Referenz der Interna*, Abschnitt “BeamForbidEvent” in *Referenz der Interna*, Abschnitt “beam-interface” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Wenn eine Partitur endet, während ein automatischer Balken noch nicht beendet wurde und weiterhin Noten erwartet, wird dieser letzte Balken nicht ausgegeben. Das Gleiche gilt auch für polyphone Stimmen, die mit der << ... \\ ... >>-Konstruktion notiert wurden. Wenn eine polyphone Stimme endet, während ein Balken noch weitere Noten erwartet, wird der Balken nicht gesetzt. Eine Notlösung für dieses Probelm ist, den letzten Balken in der Stimme oder Partitur manuell zu setzen.

Die Standardeinstellungen weisen den dem `Score`-Kontext zu. Das bedeutet, dass das Setzen der Taktart (time signature) in einem System sich auch auf die Bebalkung der anderen Systeme auswirkt. Darum verändert eine neue Taktart in einem späteren System auch alle früher vorgenommenen eigenen Einstellungen der Bebalkung eines anderen Systems. Eine Möglichkeit, dieses Problem zu vermeiden, ist es, die Taktart nur in einem System zu setzen.

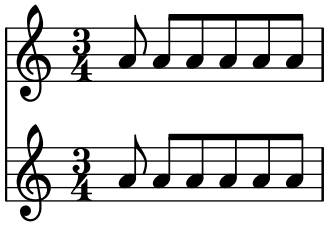
```
<<
  \new Staff {
    \time 3/4
    \set Timing.beatBase = #1/8
    \set Timing.beatStructure = 1,5
    \repeat unfold 6 { a8 }
  }
  \new Staff {
    \repeat unfold 6 { a8 }
  }
>>
```



Die Standardbebalkungsregeln für die Taktart können auch verändert werden, sodass die gewünschte Bebalkung immer benützt wird. Veränderungen der automatischen Bebalkungsregeln für eine Taktart sind in Abschnitt 2.3.1 [Taktangabe], Seite 65, beschrieben.

```
<<
  \new Staff {
    \overrideTimeSignatureSettings
      3/4               % timeSignature
      #1/8              % beatBase
      1,5               % beatStructure
      #'() % beamExceptions
    \time 3/4
    \repeat unfold 6 { a'8 }
```
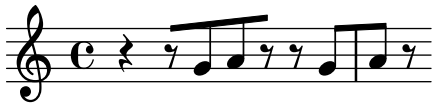
```
  }
  \new Staff {
    \time 3/4
    \repeat unfold 6 { a'8 }
  }
>>
```



2.4.3 Manuelle Balken
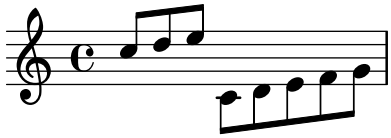
In einigen Fällen kann es nötig sein, den automatischen Algorithmus für die Balken zu überschreiben. Die automatischen Balken werden beispielsweise nicht über Pausen oder Taktlinien hinweg gesetzt, und in Gesang werden die Balken oft nach dem Rhythmus des Textes und nicht dem der Musik gesetzt. Manuell definierte Balken werden mit den Zeichen `[` und `]` (AltGr+8 bzw. 9) markiert.

```
\relative { r4 r8[ g' a r] r g[ | a] r }
```
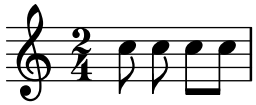


Die Richtung von Balken kann mit den Richtungszeichen verändert werden:

```
\relative { c''8^[ d e] c,_[ d e f g] }
```



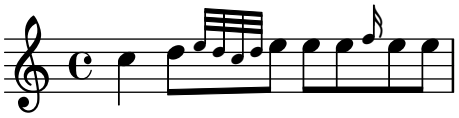
Einzelne Noten können mit dem Befehl `\noBeam` markiert werden, damit sie nicht mit einem Balken versehen werden.

```
\time 2/4 c8 c\noBeam c c
```
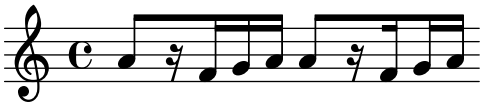


Balken von Verzierungsnoten und normale Balken können gleichzeitig gesetzt werden. Unbebalkte Verzierungen werden nicht innerhalb von normalen Balken gesetzt.

```
\relative {
  c''4 d8[
  \grace { e32 d c d }
  e8] e[ e
  \grace { f16 }
  e8 e]
}
```



Noch bessere manuelle Kontrolle über die Balken kann durch Setzen der Eigenschaften `stemLeftBeamCount` und `stemRightBeamCount` erreicht werden. Sie bestimmen die Anzahl von Balken, die rechts und links vom Hals der nächsten Note gesetzt werden sollen. Wenn eine Eigenschaften gesetzt ist, wird ihr Wert nur einmal eingesetzt und dann wieder auf Null gesetzt. Im folgenden Beispiel hat das letzte `f` nur einen Balken an seiner linken Seite (der als Achtelbalken der gesamten Gruppe gewertet wird).

```
\relative a' {
  a8[ r16 f g a]
  a8[ r16
  \set stemLeftBeamCount = #2
  \set stemRightBeamCount = #1
  f16
  \set stemLeftBeamCount = #1
  g16 a]
}
```



Ausgewählte Schnipsel

Beam nibs

Beam nibs at the start and end of beams together with beams attached to solitary notes that look like flat flags are possible with a combination of `stemLeftBeamCount`, `stemRightBeamCount`, and paired `[]` beam indicators.

For imitating right-pointing flat flags on lone notes, use paired `[]` beam indicators and set `stemLeftBeamCount` to zero. For imitating left-pointing flat flags on lone notes, set `stemRightBeamCount` to zero instead (line one).

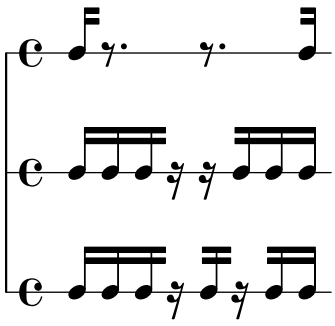
For right-pointing nibs at the end of a run of beamed notes, set `stemRightBeamCount` to a positive value. For left-pointing nibs at the start of a run of beamed notes, set `stemLeftBeamCount` instead (line two).

Sometimes it may make sense for a lone note surrounded by rests to carry both a left- and right-pointing nib. Do this with paired `[]` beam indicators alone (line three).

Note that `\set stemLeftBeamCount` is always equivalent to `\once \set`. In other words, the beam count settings are not „sticky“, so the pair of nibs attached to the lone 16th note in the last example has nothing to do with the `\set` command for the beam before.

```
\score {
  <<
    \new RhythmicStaff {
      \set stemLeftBeamCount = 0
      c16[] r8.
      r8.
      \set stemRightBeamCount = 0
      16[]
    }
    \new RhythmicStaff {
```

```
        16 16
        \set stemRightBeamCount = 2
        16 r r
        \set stemLeftBeamCount = 2
        16 16 16
      }
      \new RhythmicStaff {
        16 16
        \set stemRightBeamCount = 2
        16 r16
        16[] r16
        \set stemLeftBeamCount = 2
        16 16
      }
    >>
}
```



Siehe auch

Notationsreferenz: Abschnitt 35.2 [Richtung und Platzierung], Seite 611, Abschnitt 2.6.1 [Verzierungen], Seite 112.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Referenz der Interna: Abschnitt “Beam” in *Referenz der Interna*, Abschnitt “BeamEvent” in *Referenz der Interna*, Abschnitt “Beam_engraver” in *Referenz der Interna*, Abschnitt “beam-interface” in *Referenz der Interna*, Abschnitt “Stem_engraver” in *Referenz der Interna*.

2.4.4 Gespreizte Balken
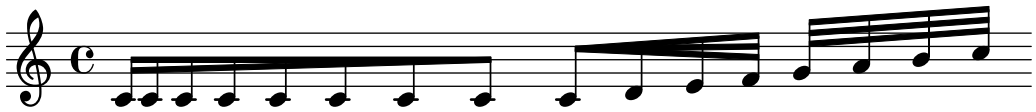
Gespreizte Balken werden teilweise eingesetzt um anzuzeigen, dass kleine Notengruppen in beschleunigendem oder verlangsamendem Tempo gespielt werden sollen, ohne dass sich das Tempo des Stückes verändert. Die Reichweite der gespreizten Balken muss manuell mit `[` und `]` angegeben werden und die Spreizung wird kontrolliert, indem der Balken-Eigenschaft `grow-direction` eine Richtung zugewiesen wird.

Wenn die Anordnung der Noten und die MIDI-Ausgabe das *Ritardando* oder *Accelerando*, wie es die Spreizung angibt, reflektieren soll, müssen die Noten als ein musikalischer Ausdruck notiert werden, der von geschweiften Klammern umgeben ist und dem ein `featheredDurations`-(gespreizteDauern)-Befehl vorangestellt ist, der das Verhältnis der ersten und letzten Dauer definiert.

Die eckigen Klammern geben die Reichweite des Balkens an und die geschweiften Klammern zeigen, auf welche Noten sich die Veränderung der Dauern auswirkt. Normalerweise bezieht sich das auf die selbe Notengruppe, aber das ist nicht unbedingt erforderlich: beide Befehle sind unabhängig voneinander.

Im folgenden Beispiel nehmen die acht 16-Noten exakt die gleiche Zeit ein wie eine halbe Note, aber die erste Note ist halb so lang wie die letzte der Gruppe, und die Noten dazwischen werden stufenweise verlängert. Die ersten vier 32-Noten beschleunigen stufenweise das Tempo, während die darauffolgenden vier 32-Noten ein gleichmäßiges Tempo haben.

```
\relative c' {
  \override Beam.grow-direction = #LEFT
  \featherDurations 2/1
  { c16[ c c c c c c c] }
  \override Beam.grow-direction = #RIGHT
  \featherDurations 2/3
  { c32[ d e f] }
  % revert to non-feathered beams
  \override Beam.grow-direction = #'()
  { g32[ a b c] }
}
```



Die Platzierung der Noten im Druckbild entspricht den Notendauern nur annähernd, aber die MIDI-Ausgabe ist exakt.

Vordefinierte Befehle

`\featherDurations`.

Siehe auch

Snippets: Abschnitt “Rhythms” in *Schnipsel*.

Bekannte Probleme und Warnungen

Der `\featherDurations`-Befehl funktioniert nur mit kurzen Notenabschnitten, und wenn die Zahlen in den Brüchen klein sind.
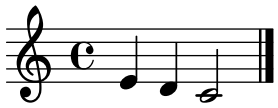
2.5 Takte

2.5.1 Taktstriche

Taktstriche trennen Takte voneinander, werden aber auch verwendet, um Wiederholungen anzuzeigen. Normalerweise werden sie automatisch nach Vorgabe der aktuellen Taktart eingefügt.

Die einfachen, automatisch eingefügten Taktstriche können mit dem `\bar`-Befehl geändert werden. Eine doppelter Taktstrich etwa wird normalerweise am Ende eines Stückes gesetzt:

```
\relative { e'4 d c2 \bar "|." }
```



Es ist kein Fehler, wenn die letzte Note in einem Takt nicht zum automatisch eingefügten Taktstrich aufhört: es wird angenommen, dass die Note im nächsten Takt weitergeht. Wenn aber eine ganze Reihe solcher überlappenden Takte auftritt, können die Noten gedrungen aussehen oder sogar über den Seitenrand hinausragen. Das kommt daher, dass Zeilenumbrüche nur dann vorgenommen werden, wenn ein vollständiger Takt auftritt, also ein Takt, an dem alle Noten vor dem Taktstrich zu Ende sind.

Achtung: Eine falsche Dauer kann bewirken, dass Zeilenumbrüche verhindert werden, woraus resultiert, dass die Noten entweder sehr stark gedrängt auf der Zeile notiert werden, oder die Zeile über den Seitenrand hinausragt.

Zeilenumbrüche werden erlaubt, wenn ein Taktstrich manuell eingefügt wird, auch, wenn es sich um keinen vollständigen Takt handelt. Um einen Zeilenumbruch zu erlauben, ohne den Taktstrich auszugeben, kann

```
\bar ""
```
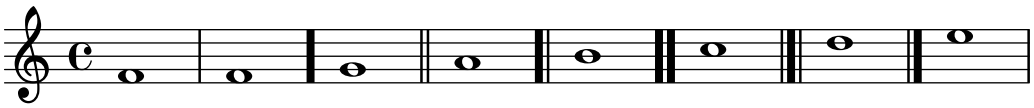
benutzt werden. Damit wird ein unsichtbarer Taktstrich an dieser Stelle eingefügt und damit ein Zeilenumbruch erlaubt (aber nicht erzwungen), ohne dass sich die Anzahl der Takte erhöhen würde. Um einen Zeilenumbruch zu erzwingen, siehe Abschnitt 27.1 [Zeilenumbrüche], Seite 536.

Diese Taktstrichart und auch andere besondere Taktstriche können manuell an jeder Stelle in der Partitur eingefügt werden. Wenn sie mit dem Ende eines Taktes übereinstimmen, wird der automatische Taktstrich durch den manuellen ersetzt. Diese manuellen Einfügungen haben keine Auswirkung auf die Zählung und Position der folgenden automatischen Taktstriche.

Dabe gilt zu beachten, dass manuell gesetzten Taktstriche nur visuell sichtbar sind. Sie wirken sich auf keine der Eigenschaften aus, die ein normaler Taktstrich beeinflussen würde, wie etwa Taktzahlen, Versetzungszeichen, Zeilenumbrüche usw. Sie beeinflussen auch nicht die Berechnung und Platzierung von weiteren automatischen Taktstrichen. Wenn ein manueller Taktstrich dort gesetzt wird, wo ein automatischer Taktstrich sowieso wäre, werden die Auswirkungen des originalen Taktstriches nicht verändert.

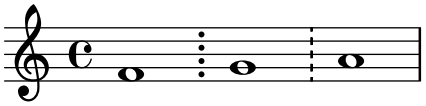
Manuell können zwei einfache Taktstriche und zusätzlich fünf Arten eines doppelten Taktstriches gesetzt werden:

```
\relative {
  f'1 \bar "|"
  f1 \bar "."
  g1 \bar "||"
  a1 \bar ".|"
  b1 \bar ".."
  c1 \bar "|.|"
  d1 \bar "|."
  e1
}
```



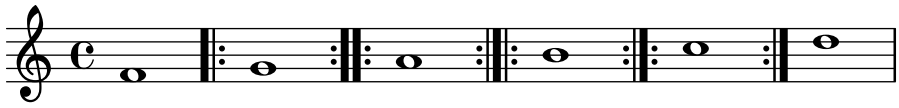
Zusätzlich gibt es noch punktierte und gestrichelte Taktstriche:

```
\relative {
  f'1 \bar ";"
  g1 \bar "!"
  a1
}
```



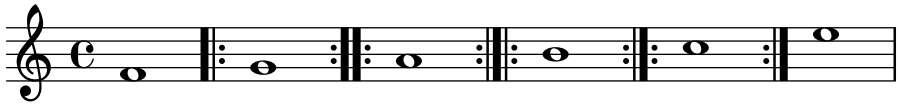
und fünf unterschiedliche Wiederholungstaktstriche:

```
f1 \bar ".|:" g \bar ":..:" a \bar ":|.|:" b \bar ":|.:" c \bar ":|." d
```



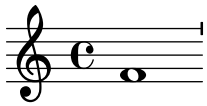
Zusätzlich kann eine Taktlinie mit einem einfachen Apostroph gesetzt werden:

```
f1 \bar ".|:"
g1 \bar ":..:"
a1 \bar ":|.|:"
b1 \bar ":|.:"
c1 \bar ":|."
e1
```



Zusätzliche kann ein Taktstrich auch nur als kleines Komma gesetzt werden:
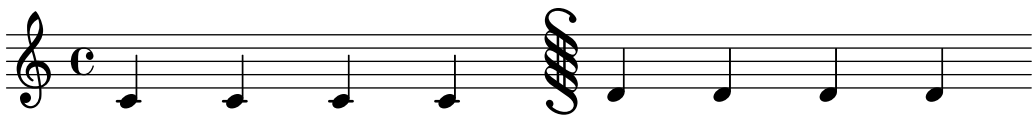
```
f1 \bar "'"
```

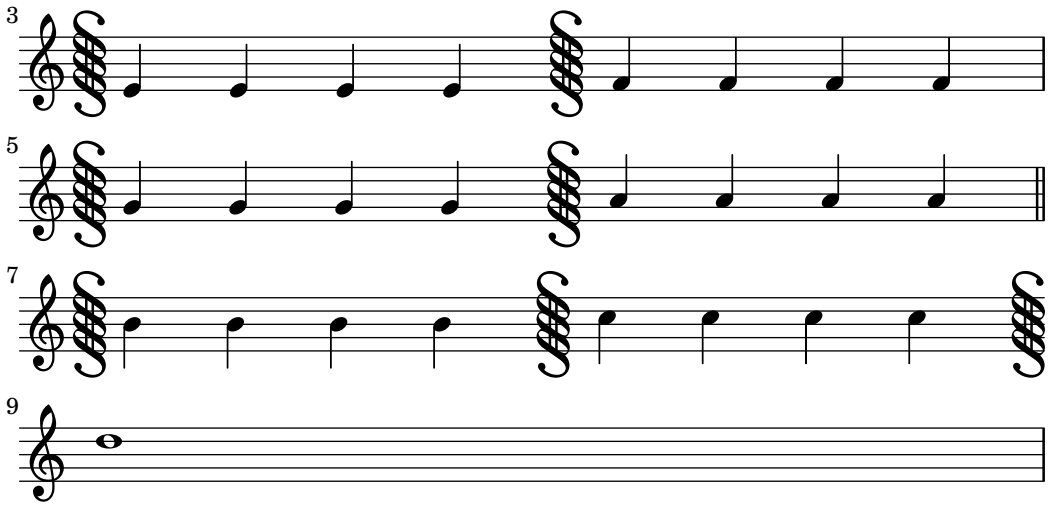


Derartige Apostrophe werden allerdings vor allem im gregorianischen Choral eingesetzt, und es wird empfohlen, anstatt dessen `\divisioMinima` zu benutzen, wie beschrieben im Abschnitt 17.4.4 [Divisiones], Seite 433.

Für *segno*-Zeichen innerhalb des Systems gibt es drei Taktstricharten, die sich in ihrem Verhalten an Zeilenumbrüchen unterscheiden:

```
\fixed c' {
  c4 4 4 4
  \bar "S"
  d4 4 4 4 \break
  \bar "S"
  e4 4 4 4
  \bar "S-|"
  f4 4 4 4 \break
  \bar "S-|"
  g4 4 4 4
  \bar "S-||"
  a4 4 4 4 \break
  \bar "S-||"
  b4 4 4 4
  \bar "S-S"
  c'4 4 4 4 \break
  \bar "S-S"
  d'1
}
```

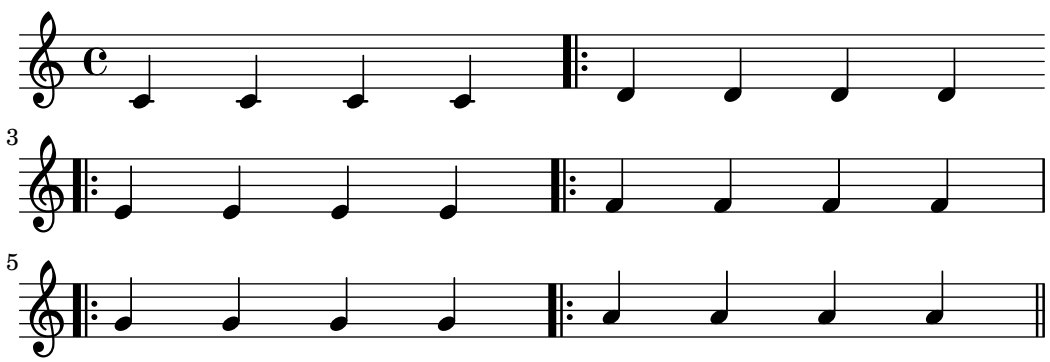


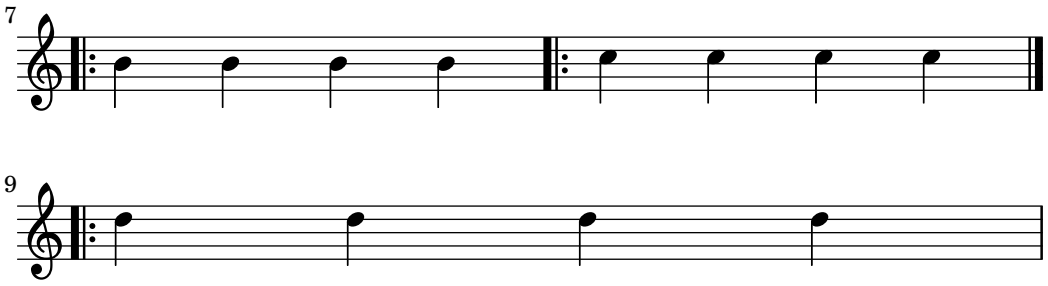


Auch wenn die Taktlinien, die Wiederholungen angeben, manuell eingefügt werden können, wird die Wiederholung dadurch nicht von LilyPond erkannt. Wiederholte Stellen werden besser notiert, indem man die Wiederholungs-Befehle einsetzt, die automatische die richtigen Taktlinien setzen. Das ist beschrieben inKapitel 4 [Wiederholungszeichen], Seite 144.

Zusätzlich kann noch `"||:"` verwendet werden, dass sich genauso wie `"|:"` verhält, außer bei Zeilenumbrüchen, wo ein doppelter Taktstrich am Ende der Zeile ausgegeben wird und ein öffnender Wiederholungsstrich am Anfang der nächsten Zeile.

```
\fixed c' {
  c4 4 4 4
  \bar ".|:"
  d4 4 4 4 \break
  \bar ".|:"
  e4 4 4 4
  \bar ".|:-|"
  f4 4 4 4 \break
  \bar ".|:-|"
  g4 4 4 4
  \bar ".|:-||"
  a4 4 4 4 \break
  \bar ".|:-||"
  b4 4 4 4
  \bar ".|:-|."
  c'4 4 4 4 \break
  \bar ".|:-|."
  d'4 4 4 4
}
```
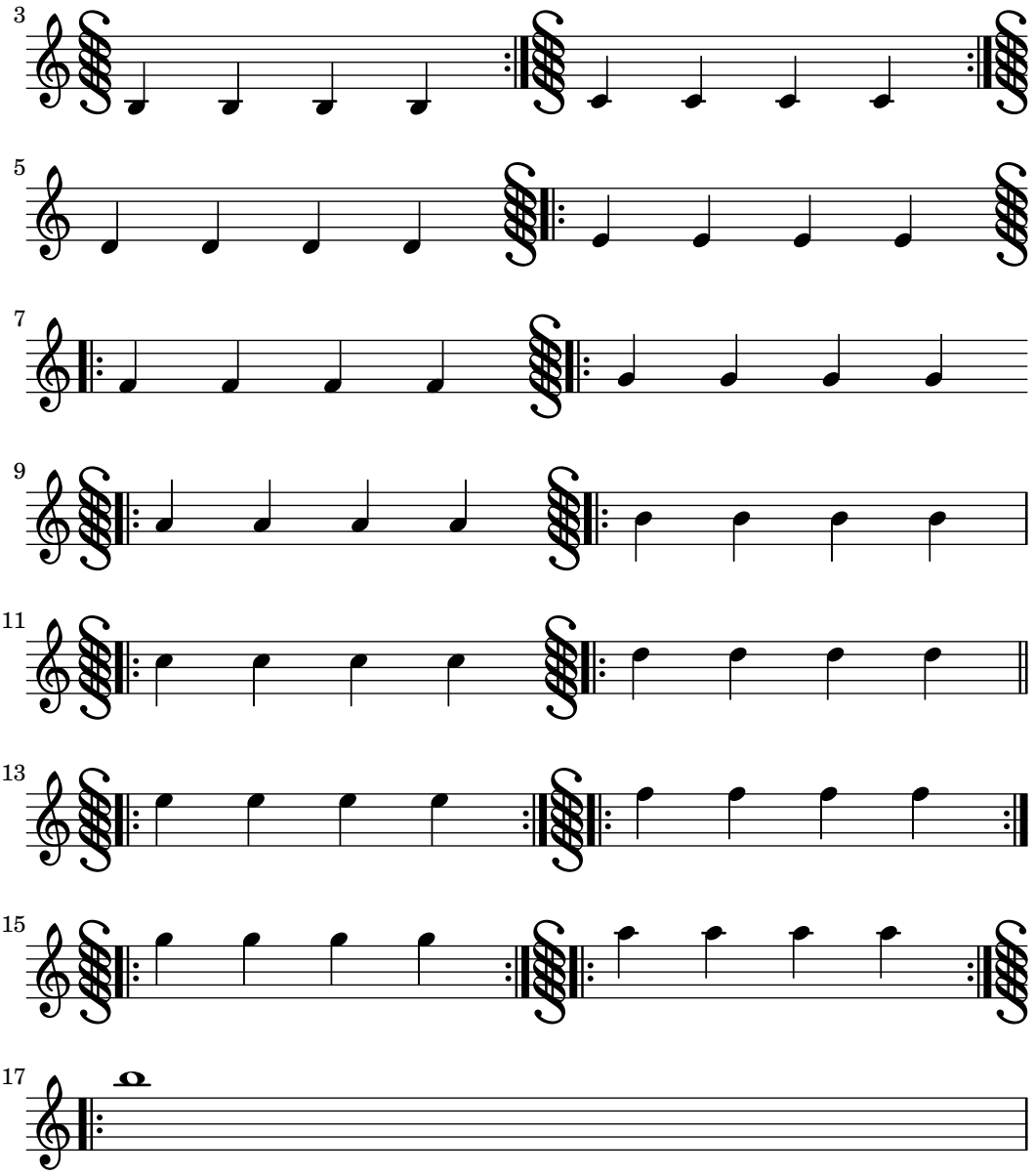




Für Kombinationen von Wiederholungen mit dem segno-Zeichen gibt es sechs verschiedene Variationen:

```
\fixed c' {
  g,4 4 4 4
  \bar ":|.S"
  a,4 4 4 4 \break
  \bar ":|.S"
  b,4 4 4 4
  \bar ":|.S-S"
  c4 4 4 4 \break
  \bar ":|.S-S"
  d4 4 4 4
  \bar "S.|:-S"
  e4 4 4 4 \break
  \bar "S.|:-S"
  f4 4 4 4
  \bar "S.|:"
  g4 4 4 4 \break
  \bar "S.|:"
  a4 4 4 4
  \bar "S.|:-|"
  b4 4 4 4 \break
  \bar "S.|:-|"
  c'4 4 4 4
  \bar "S.|:-||"
  d'4 4 4 4 \break
  \bar "S.|:-||"
  e'4 4 4 4
  \bar ":|.S.|:"
  f'4 4 4 4 \break
  \bar ":|.S.|:"
  g'4 4 4 4
  \bar ":|.S.|:-S"
  a'4 4 4 4 \break
  \bar ":|.S.|:-S"
  b'1
}
```
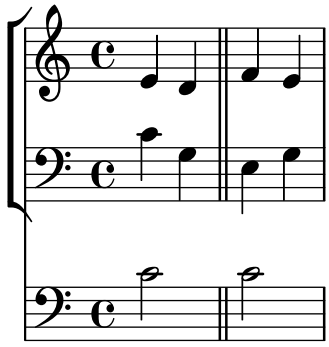




Darüber hinaus wählt der `\inStaffSegno`-Befehl eines dieser Segno-Taktstriche aus, in Zusammenarbeit mit dem `\repeat volta`-Befehl.

In Partituren mit vielen Systemen wird ein `\bar`-Befehl in einem System automatisch auf alle anderen Systeme angewendet. Die resultierenden Taktstriche sind miteinander verbunden innerhalb einer Gruppe (`StaffGroup`) oder einem Klaviersystem (`PianoStaff` bzw. (`GrandStaff`).

```
<<
  \new StaffGroup <<
    \new Staff \relative {
      e'4 d
      \bar "||"
      f4 e
    }
    \new Staff \relative { \clef bass c'4 g e g }
  >>
  \new Staff \relative { \clef bass c'2 c2 }
>>
```



Ausgewählte Schnipsel

Der Befehl `\bar` *Taktart* ist eine Kurzform von: `\set Timing.whichBar =` *Taktart*. Immer, wenn `whichBar` auf einen Wert gesetzt wird, wird ein Taktstrich dieses Typs erzeugt.

Der automatisch erzeugte Taktstrich ist `"|"`. Das kann jederzeit durch den Befehl `\set Timing.measureBarType =` *Taktstrichart* geändert werden.

Siehe auch

Notationsreferenz: Abschnitt 27.1 [Zeilenumbrüche], Seite 536, Kapitel 4 [Wiederholungszeichen], Seite 144, Abschnitt 6.1.2 [Systeme gruppieren], Seite 185.

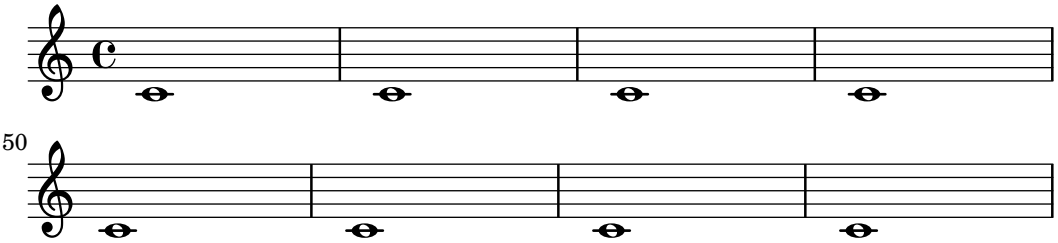
Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Referenz der Interna: Abschnitt “BarLine” in *Referenz der Interna* (erstellt auf Abschnitt “Staff” in *Referenz der Interna*-Ebene), Abschnitt “SpanBar” in *Referenz der Interna* (über Systeme), Abschnitt “Timing_translator” in *Referenz der Interna* (für Timing-Eigenschaften).

2.5.2 Taktzahlen

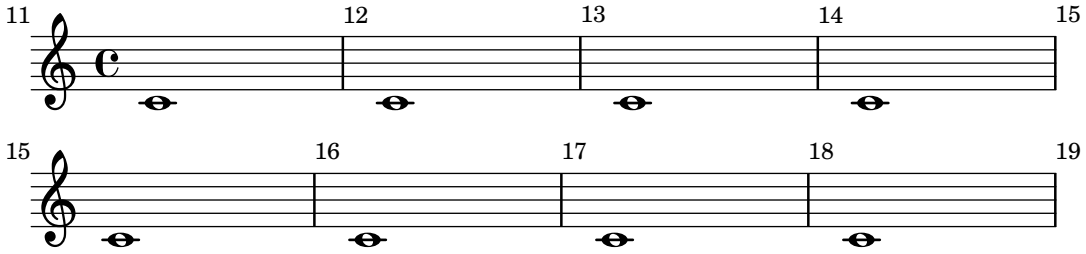
Taktzahlen werden standardmäßig zu Beginn eines jeden Systems ausgegeben, ausgenommen ist die erste Zeile. Die Zahl selber wird in der `currentBarNumber`-Eigenschaft gespeichert, die normalerweise für jeden Takt aktualisiert wird. Sie kann aber auch manuell gesetzt werden:

```
c1 c c c
\break
\set Score.currentBarNumber = #50
c1 c c c
```



Taktnummern können in regelmäßigem Abstand ausgegeben werden, anstatt dass sie nur am Beginn des Systems erscheinen. Um das zu erreichen, muss die Standardeinstellung verändert werden, um zu erlauben, dass Taktnummern an anderen Stellen als dem Beginn von Systemen ausgegeben werden. Das wird mit der Eigenschaft `break-visibility` von `BarNumber` vorgenommen. Sie braucht drei Werte, die auf `#t` (wahr) oder `#f` (falsch) gestellt werden können, womit angegeben wird, ob die Taktnummer an der entsprechenden Stelle sichtbar ist. Die Reihenfolge der Werte ist: *Ende der Zeile*, *Mitte der Zeile* und *Beginn der Zeile*. Im folgenden Beispiel werden die Taktlinien überall ausgegeben:

```
\override Score.BarNumber.break-visibility = ##(#t #t #t)
\set Score.currentBarNumber = #11
% Permit first bar number to be printed
\bar ""
c1 | c | c | c
\break
c1 | c | c | c
```

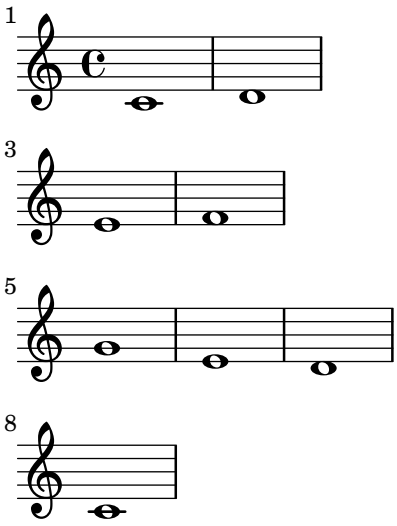


Setzen der Taktnummer für den ersten Takt

Standardmäßig wird die erste Taktzahl einer Partitur nicht gesetzt, wenn sie weniger oder gleich '1' ist. Indem man `barNumberVisibility` auf `all-bar-numbers-visible` setzt, kann eine beliebige Taktzahl für den ersten und die folgenden Takte gesetzt werden. Eine leere Taktlinie muss jedoch vor der ersten Note eingefügt werden, damit das funktioniert.

```
\paper {
  line-width = 50\mm
}

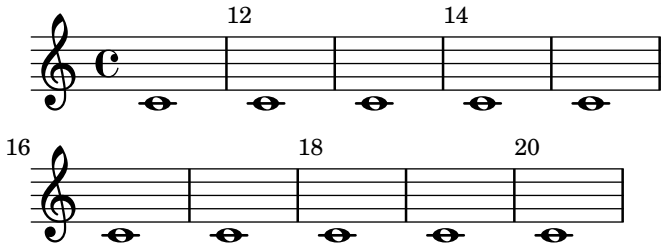
\relative c' {
  \set Score.barNumberVisibility = #all-bar-numbers-visible
  c1 | d | e | f \break
  g1 | e | d | c
}
```



Setzen der Taktnummern in regelmäßigen Intervallen

Taktnummern können in regelmäßigen Intervallen gesetzt werden, indem man die Eigenschaft `barNumberVisibility` definiert. In diesem Beispiel werden die Taktnummern jeden zweiten Takt gesetzt, außer am Ende einer Zeile.

```
\relative c' {
  \override Score.BarNumber.break-visibility = #end-of-line-invisible
  \set Score.currentBarNumber = 11
  % Print a bar number every second measure
  \set Score.barNumberVisibility = #(every-nth-bar-number-visible 2)
  c1 | c | c | c | c
  \break
  c1 | c | c | c | c
}
```



Alternative Taktnummerierung

Zwei alternative Methoden können eingestellt werden, die die Taktnummerierung beeinflussen, insbesondere bei Wiederholungen.

```
music = \relative c' {
  \repeat volta 3 {
    c4 d e f |
    \alternative {
      \volta 1 { c4 d e f | c2 d \break }
      \volta 2 { f4 g a b | f4 g a b | f2 a | \break }
      \volta 3 { c4 d e f | c2 d } } }
  c1 \bar "|."
}
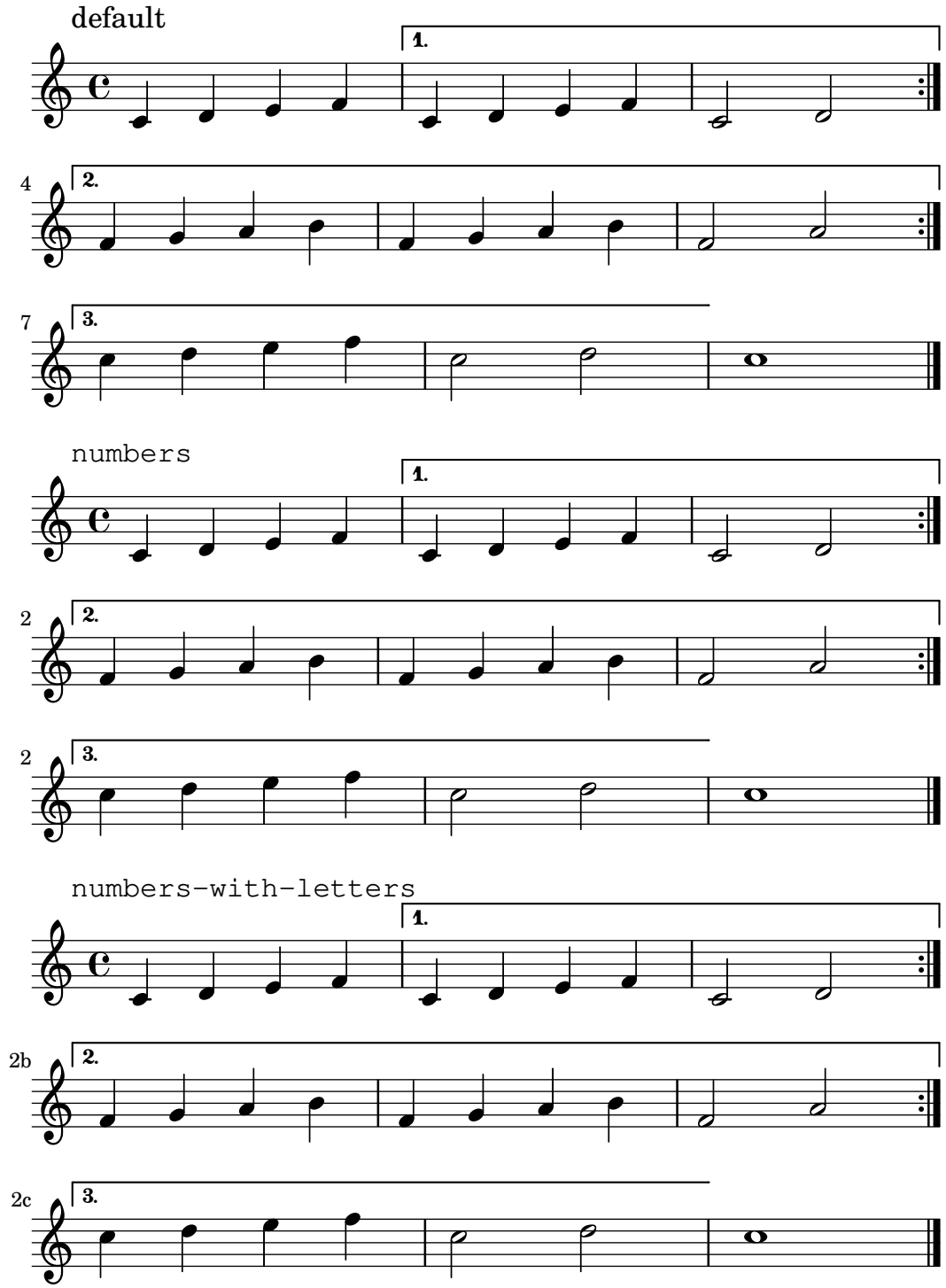
{
  \textMark \markup \large "default"
  \music
}

{
  \textMark \markup \large \typewriter "numbers"
  \set Score.alternativeNumberingStyle = #'numbers
  \music
}

{
  \textMark \markup \large \typewriter "numbers-with-letters"
  \set Score.alternativeNumberingStyle = #'numbers-with-letters
  \music
}

\layout {
  \context {
    \Score
    \override TextMark.Y-offset = #5
```

```
  }
}
```



Setzen von Taktnummern in Kästen oder Kreisen

Taktnummern können auch in Boxen oder Kreisen gesetzt werden.

```
\relative c' {
  % Center bar numbers except at the beginning of a staff.
  \override Score.BarNumber.self-alignment-X =
    #(break-alignment-list CENTER CENTER 0.3)

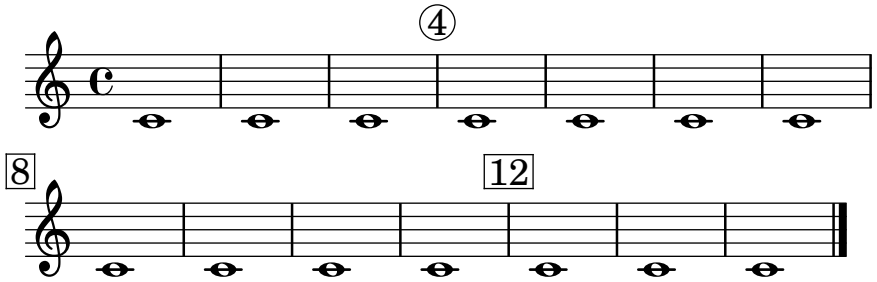
  % Prevent bar numbers at the end of a line and permit them elsewhere.
  \override Score.BarNumber.break-visibility = #end-of-line-invisible

  \set Score.barNumberVisibility = #(every-nth-bar-number-visible 4)

  % Increase the size of the bar number by 2.
  \override Score.BarNumber.font-size = 2

  % Draw a circle round the following bar number(s).
  \override Score.BarNumber.stencil
    = #(make-stencil-circler 0.1 0.25 ly:text-interface::print)
  \*7 c1 \break

  % Draw a box round the following bar number(s).
  \override Score.BarNumber.stencil
    = #(make-stencil-boxer 0.1 0.25 ly:text-interface::print)
  \*7 c1 \bar "|."
}
```



Taktnummern ausrichten

Taktnummern sind standardmäßig links an ihrem Ursprungsobjekt ausgerichtet. Das ist normalerweise die linke Ecke einer Linie oder, wenn die Nummern innerhalb einer Zeile gesetzt werden, auf der linken Seite eines Taktstrichs. Die Nummern können auch direkt über dem Taktstrich positioniert werden oder rechts vom Taktstrich gesetzt werden.

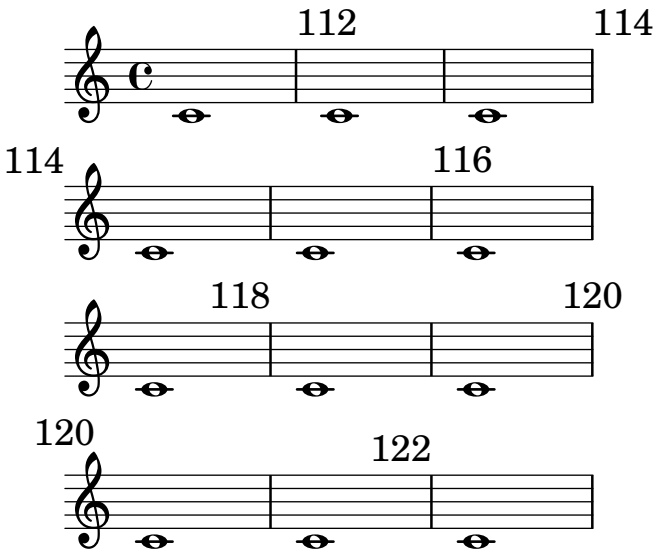
```
\relative c' {
  \set Score.currentBarNumber = 111
  \override Score.BarNumber.break-visibility = #all-visible
  % Increase the size of the bar number by 2
  \override Score.BarNumber.font-size = 2
  % Print a bar number every second measure
  \set Score.barNumberVisibility = #(every-nth-bar-number-visible 2)

  c1 | c1 | c1 | \break
  c1 | c1 | c1 | \break

  \override Score.BarNumber.self-alignment-X =
    #(break-alignment-list CENTER RIGHT CENTER)
  c1 | c1 | c1 | \break
```

```
  c1 | c1 | c1 |
}

\paper {
  line-width = 70\mm
}
```

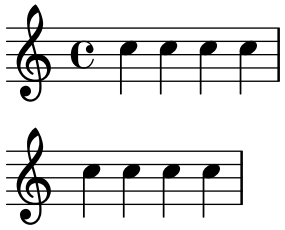


Entfernung von Taktnummern in einer Partitur

Taktnummern können vollkommen aus den Noten entfernt werden, indem man den `Bar_number_engraver` aus dem `Score`-Kontext entfernt.

```
\layout {
  \context {
    \Score
    \omit BarNumber
    % or:
    % \remove "Bar_number_engraver"
  }
}

\relative c'' {
  c4 c c c \break
  c4 c c c
}
```



Siehe auch

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Referenz der Interna: Abschnitt “BarNumber” in *Referenz der Interna*, Abschnitt “Bar_number_engraver” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Taktnummern können mit der oberen Ecke der Klammer zu Beginn des Systems zusammenstoßen. Um das zu verhindert, kann die padding-Eigenschaft von BarNumber verwendet werden, um die Zahl zu verschieben. Für mehr Information siehe Abschnitt "StaffGroup" in *Referenz der Interna* und Abschnitt "BarNumber" in *Referenz der Interna*.

2.5.3 Takt- und Taktzahlüberprüfung

Die Taktüberprüfung hilft, Fehler in den Notendauern zu entdecken. Eine Taktüberprüfung wird mit dem Taktstrichsymbol „ | " (Taste AltGr+<) eingegeben. Immer, wenn LilyPond bei der Ausgabe des Notendrucks auf dieses Zeichen stößt, sollte hier in den Noten auch ein Taktstrich erscheinen. Wenn das nicht der Fall ist, wird eine Warnung ausgegeben. Im nächsten Beispiel resultiert die zweite Taktüberprüfung in einer Fehlermeldung.

```
\time 3/4 c2 e4 | g2 |
```

Taktüberprüfungen können auch in Gesangstexten verwendet werden:
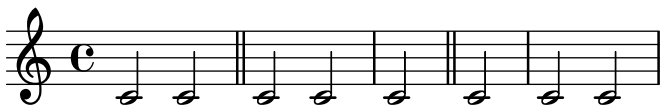
```
\lyricmode {
  \time 2/4
  Twin -- kle | Twin -- kle
}
```

Besonders in mehrstimmiger komplizierter Musik können falschen Notenwerte die ganze Partitur durcheinander bringen. Es lohnt sich also, die Fehlersuche damit zu beginnen, nicht bestandene Taktüberprüfungen zu kontrollieren.

Wenn aufeinander folgende Taktüberprüfungen mit dem gleichen Abstand Fehler produzieren, wird eventuell nur die erste Warnung ausgegeben. Damit wird die Warnung auf den Ursprung de Fehlers fokussiert.

Es ist auch möglich, die Bedeutung des Symbols | (Pipe) umzudefinieren, so dass hiermit eine andere Aktion als eine Taktüberprüfung erreicht wird. Das geschieht, indem man der Pipe ("|" einen musikalischen Ausdruck zuweist. Im nächsten Beispiel wird | dazu verwendet, eine doppelte Taktlinie auszugeben, woimmer man das Zeichen auch setzt. Gleichzeitig hört das Zeichen auf, als Taktüberprüfung zu funktionieren.

```
"|" = \bar "||"
{
  c'2 c' |
  c'2 c'
  c'2 | c'
  c'2 c'
}
```



Wenn man größere Musikstücke kopiert, kann es hilfreich sein, wenn LilyPond überprüft, ob die Taktnummer, in der Sie gerade kopieren, mit der des Originales übereinstimmt. Das kann mit dem Befehl \barNumberCheck folgenderweise überprüft werden:

```
\barNumberCheck #123
```

Eine Warnung wird ausgegeben, wenn der interne Zähler currentBarNumber von LilyPond nicht mit dem Wert 123 übereinstimmt.

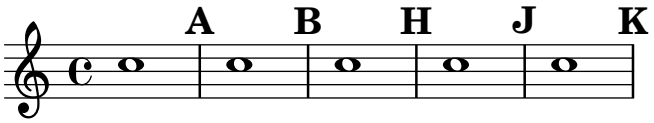
Siehe auch

Schnipsel: Abschnitt "Rhythms" in *Schnipsel*.

2.5.4 Übungszeichen

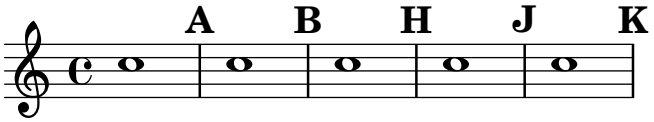
Übungszeichen können mit dem \mark-Befehl ausgegeben werden:

```
c1 \mark \default
c1 \mark \default
c1 \mark #8
c1 \mark \default
c1 \mark \default
```



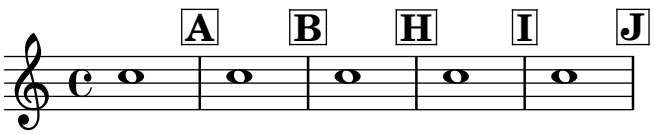
Das Zeichen wird automatisch um einen Wert heraufgesetzt, wenn man \mark \default benutzt, aber man kann auch eine Ganzzahl als Argument einsetzen, wenn man das Zeichen manuell setzen will. Der Wert, der eingesetzt werden soll, wird in der Eigenschaft rehearsalMark gespeichert.

```
\relative c'' {
  c1 \mark \default
  c1 \mark \default
  c1 \mark #8
  c1 \mark \default
  c1 \mark \default
}
```



Der Buchstabe „I“ wird ausgelassen, was den allgemeinen Notensatzregeln entspricht. Wenn Sie dennoch den Buchstaben „I“ benutzen, wollen, müssen Sie einen der folgenden Stile benutzen, je nachdem, was für einen Übungszeichenstil Sie wollen (Buchstaben, Buchstaben in einem Kasten, Buchstaben in einem Kreis).

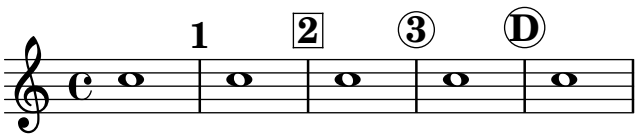
```
\set Score.rehearsalMarkFormatter = #format-mark-alphabet
\set Score.rehearsalMarkFormatter = #format-mark-box-alphabet
\set Score.rehearsalMarkFormatter = #format-mark-circle-alphabet
\relative c'' {
  \set Score.rehearsalMarkFormatter = #format-mark-box-alphabet
  c1 \mark \default
  c1 \mark \default
  c1 \mark #8
  c1 \mark \default
  c1 \mark \default
}
```



Der Stil der Übungszeichen wird von der Eigenschaft rehearsalMarkFormatter definiert. Das ist eine Funktion, die das aktuelle Zeichen und den aktuellen Kontext als Argument annimmt. Sie gibt dann ein Textbeschriftungsobjekt aus. Im folgenden Beispiel ist rehearsalMarkFormatter

so definiert, dass eine Zahl ausgegeben wird. Dann wird ein Übungszeichen in einem Kasten produziert.

```
\relative c'' {
  \set Score.rehearsalMarkFormatter = #format-mark-numbers
  c1 \mark \default
  c1 \mark \default
  \set Score.rehearsalMarkFormatter = #format-mark-box-numbers
  c1 \mark \default
  \set Score.rehearsalMarkFormatter = #format-mark-circle-numbers
  c1 \mark \default
  \set Score.rehearsalMarkFormatter = #format-mark-circle-letters
  c1
}
```



Die Datei scm/translation-functions.scm beinhaltet die Definitionen für format-mark-numbers (erstelle-Zeichen-Nummern), format-mark-box-numbers (erstelle-Zeichen-Kasten-Nummern), format-mark-letters (erstelle-Zeichen-Buchstaben) und format-mark-box-letters (erstelle-Zeichen-Kasten-Buchstaben). Sie können als Anleitung für eigene Formatierungsfunktionen dienen.

Die Funktionen format-mark-barnumbers, format-mark-box-barnumbersund format-mark-circle-barnumbers könenn eingesetzt werden, um Taktnummern anstelle der fortlaufenden Zahlen bzw. Buchstaben zu erhalten.

Andere Übungszeichenstile können auch manuell gesetzt werden:

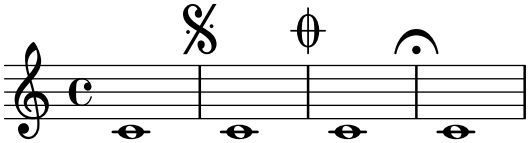
```
\mark "A1"
```

Score.rehearsalMarkFormatter hat keine Auswirkungen auf solcherart definierte Zeichen. Man kann aber auch mit \markup Textbeschriftungsobjekte zu dem selbstdefinierten Zeichen hinzufügen:

```
\mark \markup{ \box A1 }
```

Musikbuchstaben (wie etwa das Segno-Zeichen) können mit dem Befehl \musicglyph als ein \mark-Zeichen definierte werden:

```
\relative c' {
  c1 \mark \markup { \musicglyph "scripts.segno" }
  c1 \mark \markup { \musicglyph "scripts.coda" }
  c1 \mark \markup { \musicglyph "scripts.ufermata" }
  c1
}
```



Siehe Abschnitt A.8 [Die Emmentaler-Schriftart], Seite 658, wo alle Symbole gezeigt sind, die mit dem Befehl \musicglyph ausgegeben werden können.

Übliche Veränderungen der Positionierung von Übungszeichen finden sich in Abschnitt 8.2 [Text formatieren], Seite 233. Zu noch präzieserer Kontrolle siehe break-alignable-interface in Abschnitt 36.1 [Objekte ausrichten], Seite 627.

Die Datei `scm/translation-functions.scm` enthält die Definitionen von `format-mark-numbers` und `format-mark-letters`. Sie können als Anregung für andere Formatierungsfunktionen genommen werden.

Siehe auch

Notationsreferenz: Abschnitt A.8 [Die Emmentaler-Schriftart], Seite 658, Abschnitt 8.2 [Text formatieren], Seite 233, Abschnitt 36.1 [Objekte ausrichten], Seite 627.

Installierte Dateien: `scm/translation-functions.scm`.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.
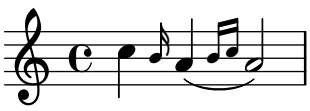
Referenz der Interna: Abschnitt “AdHocMarkEvent” in *Referenz der Interna*, Abschnitt “RehearsalMark” in *Referenz der Interna*, Abschnitt “RehearsalMarkEvent” in *Referenz der Interna*.

2.6 Besondere rhythmische Fragen

2.6.1 Verzierungen
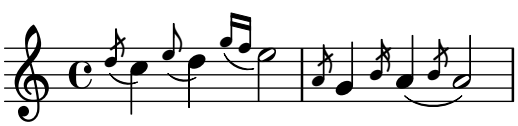
Verzierungen, mit dem Befehl `\grace` notiert, sind ausgeschriebene Ornamente. Sie werden in einer kleineren Schriftgröße gesetzt und nehmen keine logische Zeit im Takt ein.

```
\relative {
  c''4 \grace b16 a4(
  \grace { b16 c16 } a2)
}
```
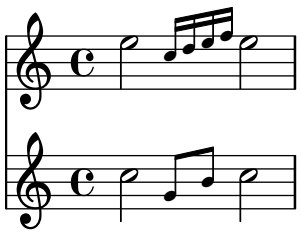


Es gibt drei Arten von Verzierungen: den Vorschlag (engl. acciaccatura), eine angebundene Verzierungsnote mit einem Schrägstrich durch den Hals, und den Vorhalt (engl. appoggiatura), welcher den Wert der Hauptnote um seinen eigenen Wert verkürzt und ohne Schrägstrich notiert wird. Man kann einen Vorschlag auch mit Schrägstrich durch den Hals, aber ohne Legatobogen notieren. Diese Verzierung wird mit dem Befehl `\slashedGrace` notiert und wird zwischen Noten notiert, die selber einen Legatobogen haben.

```
\relative {
  \acciaccatura d''8 c4
  \appoggiatura e8 d4
  \acciaccatura { g16 f } e2
  \slashedGrace a,8 g4
  \slashedGrace b16 a4(
  \slashedGrace b8 a2)
}
```
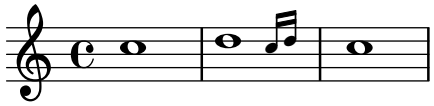


Die Position von Verzierungen ist zwischen Notensystemen synchronisiert. Im nächsten Beispiel stehen im einen System zwei 16-Noten für jede 8-Note des zweiten Systems:

```
<< \new Staff { e2 \grace { c16 d e f } e2 }
   \new Staff { c2 \grace { g8 b } c2 } >>
```
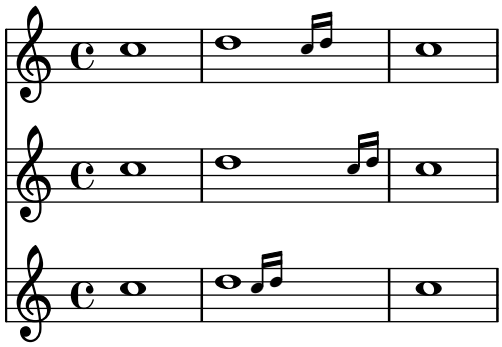


Wenn Sie eine Note mit einer Verzierung abschließen wollen, müssen Sie den `\afterGrace`-Befehl benutzen. Er benötigt zwei Argumente: die Hauptnote und die Verzierung, die nach der Hauptnote folgen soll:

```
\relative { c''1 \afterGrace d1 { c16[ d] } c1 }
```
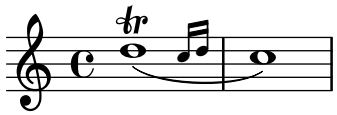


Damit wird die Verzierung mit einem Abstand von der Hauptnote gesetzt, der 3/4 der Dauer der Hauptnote entspricht. Dieser Standard kann durch Definition von `afterGraceFraction` verändert werden. Das nächste Beispiel zeigt, wie sich der Abstand verändert, wenn der Wert 3/4, 15/16 und 1/2 der Hauptnote beträgt.

```
<<
  \new Staff \relative {
    c''1 \afterGrace d1 { c16[ d] } c1
  }
  \new Staff \relative {
    #(define afterGraceFraction (cons 15 16))
    c''1 \afterGrace d1 { c16[ d] } c1
  }
  \new Staff \relative {
    #(define afterGraceFraction (cons 1 2))
    c''1 \afterGrace d1 { c16[ d] } c1
  }
>>
```
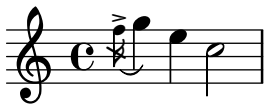


Der Abstand zwischen der Hauptnote und der Verzierung kann auch mit unsichtbaren Noten beeinflusst werden. Im nächsten Beispiel wird die Verzierung mit einem Abstand von 7/8 zur Hauptnote gesetzt.

```
\new Voice {
  << { d1^\trill_( }
     { s2 s4. \grace { c16 d } } >>
  c1)
}
```



Ein \grace-Notenabschnitt wird nach besonderen Satzregeln gesetzt, um z. B. kleinere Noten zu benutzen und die Richtung der Hälse einzustellen. Veränderungen am Layout müssen also innerhalb des Verzierungsausdrucks gesetzt werden, damit sie auch eine Auswirkung haben. Die Veränderungen müssen auch innerhalb des Verzierungsausdrucks rückgängig gemacht werden. In diesem Fall wird die Richtung der Hälse geändert und dann wieder der Standard eingestellt:

```
\new Voice \relative {
  \acciaccatura {
    \stemDown
    f''16->
    \stemNeutral
  }
  g4 e c2
}
```
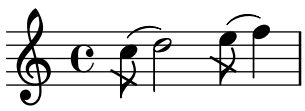


Ausgewählte Schnipsel

Using grace note slashes with normal heads

The slash through the stem found in acciaccaturas can be applied in other situations.
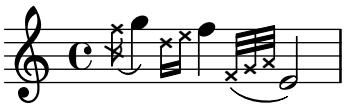
```
\relative c'' {
  \override Flag.stroke-style = "grace"
  c8( d2) e8( f4)
}
```



Veränderung des Layouts von Verzierungen innerhalb der Noten

Das Layout von Verzierungsausdrücken kann in der Musik verändert werden mit den Funktionen `add-grace-property` und `remove-grace-property`. Das folgende Beispiel definiert die Richtung von Hälsen (Stem) für diese Verzierung, sodass die Hälse nicht immer nach unten zeigen, und ändert den Standardnotenkopf in ein Kreuz.

```
\relative c'' {
  \new Staff {
    $(remove-grace-property 'Voice 'Stem 'direction)
    $(add-grace-property 'Voice 'NoteHead 'style 'cross)
    \new Voice {
       \acciaccatura { f16 } g4
       \grace { d16 e } f4
       \appoggiatura { f,32 g a } e2
    }
  }
}
```
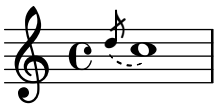


Globale Umdefinition von Verzierungsnoten

Die globalen Standardeinstellungen für Verzierungsnoten werden in den Variablen`startGraceMusic`, `stopGraceMusic`, `startAcciaccaturaMusic`, `stopAcciaccaturaMusic`, `startAppoggiaturaMusic` und `stopAppoggiaturaMusic` gespeichert, die in der Datei `ly/grace-init.ly` definiert sind. Wenn man sie umdefiniert, können andere Effekte erreicht werden.

```
startAcciaccaturaMusic = {
  <>(
  \override Flag.stroke-style = "grace"
  \slurDashed
}

stopAcciaccaturaMusic = {
  \revert Flag.stroke-style
  \slurSolid
  <>)
}

\relative c'' {
  \acciaccatura d8 c1
}
```



Positionierung von Verzierungen mit verschiebbarem Platz

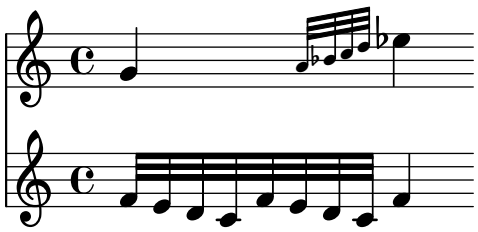
Wenn man die Eigenschaft `'strict-grace-spacing` aktiviert, werden die Verzierungsnoten "fließend" gemacht, d.h. sie sind von den normalen Noten los gekoppelt: Zuerst werden die normalen Noten platziert, dann erst die Verzierungen links von der Hauptnote gesetzt.

```
shiftedGrace =
#(define-music-function (offset music) (number? ly:music?)
   #{
     \override NoteHead.X-offset = #(- offset 0.85)
     \override Stem.X-offset = #offset
     \grace { $music }
```

```
    \revert NoteHead.X-offset
    \revert Stem.X-offset
  #})

\relative c'' <<
  { g4 \shiftedGrace #-1.3 a32 \shiftedGrace #-0.5 { bes c d } es4 }
  { f,32 e d c f e d c f4 }
>>

\layout {
  \context {
    \Score
    \override SpacingSpanner.strict-grace-spacing = ##t
  }
}
```



Siehe auch

Glossar: Abschnitt “grace notes” in *Glossar*, Abschnitt “acciaccatura” in *Glossar*, Abschnitt “appoggiatura” in *Glossar*.

Notationsreferenz: Abschnitt 2.1.3 [Tondauern skalieren], Seite 52, Abschnitt 2.4.3 [Manuelle Balken], Seite 94.

Installierte Dateien: `ly/grace-init.ly`.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.
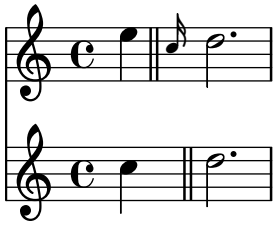
Referenz der Interna: Abschnitt “GraceMusic” in *Referenz der Interna*, Abschnitt “Grace_beam_engraver” in *Referenz der Interna*, Abschnitt “Grace_engraver” in *Referenz der Interna*, Abschnitt “Grace_spacing_engraver” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Ein Vorschlag (*acciaccatura*) mit mehreren Noten und Balken wird ohne den Schrägstrich gesetzt und sieht einem Vorhalt (*appoggiatura*) sehr ähnlich.
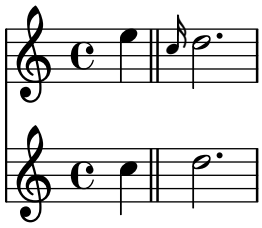
Die Synchronisation von Verzierungen kann auch zu Überraschungen führen. Auch andere Symbole der Systeme, wie Vorzeichen, Taktlinien usw., werden synchronisiert. Vorsicht ist geboten, wenn nur in bestimmten Systemen Verzierungen vorkommen:

```
<<
  \new Staff \relative { e''4 \section \grace c16 d2. }
  \new Staff \relative { c''4 \section d2. }
>>
```



Dem kann abgeholfen werden, indem unsichtbare Verzierungsnoten der selben Länge in die anderen Systeme gesetzt werden. Im obigen Beispiel müsste also

```
<<
  \new Staff { e4 \section \grace c16 d2. }
  \new Staff { c4 \section \grace s16 d2. }
>>
```
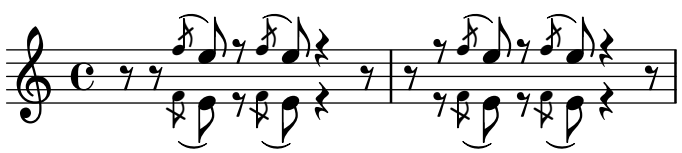


gesetzt werden.

Der Einsatz von Verzierungsnoten innerhalb von Stimmen-Kontexten kann den Notensatz der Stimme verwirren. Dieses Problem kann umgangen werden, indem man eine Note oder Pause zwischen dem `Voice`-Befehl und der Verzierungsnote setzt.

```
accMusic = {
  \acciaccatura { f8 } e8 r8 \acciaccatura { f8 } e8 r4
}

\new Staff {
  <<
    \new Voice {
      \relative c'' {
        r8 r8 \voiceOne \accMusic \oneVoice r8 |
        r8 \voiceOne r8 \accMusic \oneVoice r8 |
      }
    }
    \new Voice {
      \relative c' {
        s8 s8 \voiceTwo \accMusic \oneVoice s8 |
        s8 \voiceTwo r8 \accMusic \oneVoice s8 |
      }
    }
  >>
}
```



Verzierungsabschnitte sollten nur innerhalb von sequentiellen musikalischen Ausdrücken benützt werden. Wenn sie ineinandergeschachtelt werden, kann es zu Fehlermeldungen oder Abstürzen kommen.

Jede Verzierungsnote in der MIDI-Ausgabe hat ein Viertel der Dauer ihrer wirklichen Dauer. Wenn die addierte Dauer der Verzierungsnoten länger als die Dauer der vorhergehenden Note dauert, wird der Fehler „`Going back in MIDI time`“ ausgegeben. Man muss die Verzierungsnoten entweder kürzer machen, etwa:

```
\acciaccatura { c'8[ d' e' f' g'] }
```

wird zu:

```
\acciaccatura { c'16[ d' e' f' g'] }
```

oder die Dauern explizit ändern:

```
\acciaccatura { \scaleDurations 1/2 { c'8[ d' e' f' g'] } }
```

Siehe Abschnitt 2.1.3 [Tondauern skalieren], Seite 52.

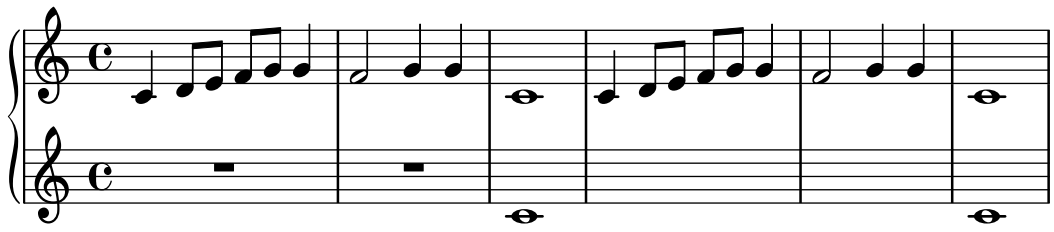
2.6.2 An Kadenzen ausrichten

In Orchesterpartituren stellen Kadenzen ein besonderes Problem dar: Wenn in der Partitur ein Instrument eine Kadenz spielt, die notiert wird, müssen die anderen Stimmen genau die entsprechende Anzahl Noten überspringen, damit sie nicht zu früh oder zu spät einsetzen.

Eine Lösung ist es, die Funktionen `mmrest-of-length` oder `skip-of-length` zu benutzen. Diese Scheme-Funktionen brauchen einen definierten Notenabschnitt (eine Variable) als Argument und produzieren entweder Ganztaktpausen oder leere Takte, die genauso lang sind wie der Notenabschnitt.

```
MyCadenza = \relative {
  c'4 d8 e f g g4
  f2 g4 g
}

\new GrandStaff <<
  \new Staff {
    \MyCadenza c'1
    \MyCadenza c'1
  }
  \new Staff {
    #(mmrest-of-length MyCadenza)
    c'1
    #(skip-of-length MyCadenza)
    c'1
  }
>>
```



Siehe auch

Glossar: Abschnitt “cadenza” in *Glossar*.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

2.6.3 Verwaltung der Zeiteinheiten

Die Zeit in einer Partitur wird vom `Timing_translator` verwaltet, der sich in den Standardeinstellungen im `Score`-Kontext befindet. Eine Parallelbezeichnung, `Timing`, wird dem Kontext hinzugefügt, in dem sich `Timing_translator` befindet. Um sicherzugehen, dass `Timing` erhältlich ist, muss man eventuell den enthaltenden Kontext manuell erstellen (also etwa einen `Voice`- oder `Staff`-Kontext.

Die folgenden Eigenschaften von `Timing` werden eingesetzt, um die Zeit in Partituren zu verwalten.

`currentBarNumber` (`aktuelle Taktnummer`)
: Die gerade aktuelle Taktzahl. Für ein Beispiel, das die Benutzung dieser Eigenschaft zeigt, siehe Abschnitt 2.5.2 [Taktzahlen], Seite 103.
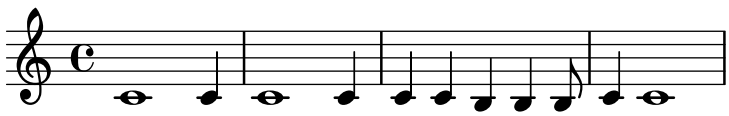
`measureLength` (`Taktlänge`)
: Die Länge der Takte mit der aktuellen Taktart. In einem 4/4-Takt ist sie 1, in einem 6/8-Takt 3/4. Dieser Wert bestimmt, wann eine Taktlinie gezogen wird und wie automatische Balken erstellt werden sollen.

`measurePosition` (`Taktposition`)
: Der Schlag im Takt zum aktuellen Moment. Dieser Wert wird zurückgesetzt, indem `measureLength` (die Taktlänge) abgezogen wird, wenn der Wert von `measureLength` erreicht oder überschritten wird. Wenn das passiert, wird der Zähler `currentBarNumber` (aktuelle Taktnummer) erhöht.

`timing` (`Zeitberechnung`)
: Wenn auf wahr gesetzt, werden die oben genannten Variablen zu jedem Zeitpunkt aktualisiert. Wenn auf falsch gesetzt, bleibt der Engraver unendlich lange im aktuellen Takt.

Zeitverwaltung kann geändert werden, indem man diese Variablen direkt beeinflusst. Im nächsten Beispiel wird die normale Taktart mit 4/4 angegeben, aber `measureLength` wird auf 5/4 gesetzt. An der Stelle 4/8 des dritten Taktes wird die Taktposition (`measurePosition`) um 1/8 auf 5/8 erhöht, so dass der Takt im Ergebnis 1/8 kürzer ist. Die nächste Taktlinie wird dann auch bei 9/8 gezogen und nicht bei 5/4.

```
\set Score.measureLength = #5/4
c1 c4
c1 c4
c4 c4
\set Score.measurePosition = \musicLength 8*5
b4 b4 b8
c4 c1
```



Wie das Beispiel zeigt, erstellt `ly:make-moment n m` die Dauer Zähler/Nenner einer ganzen Note. Zum Beispiel heißt `ly:make-moment 1 8` die Dauer einer Achtelnote, und `ly:make-moment 7 16` die Dauer von sieben Sechszehntelnoten.
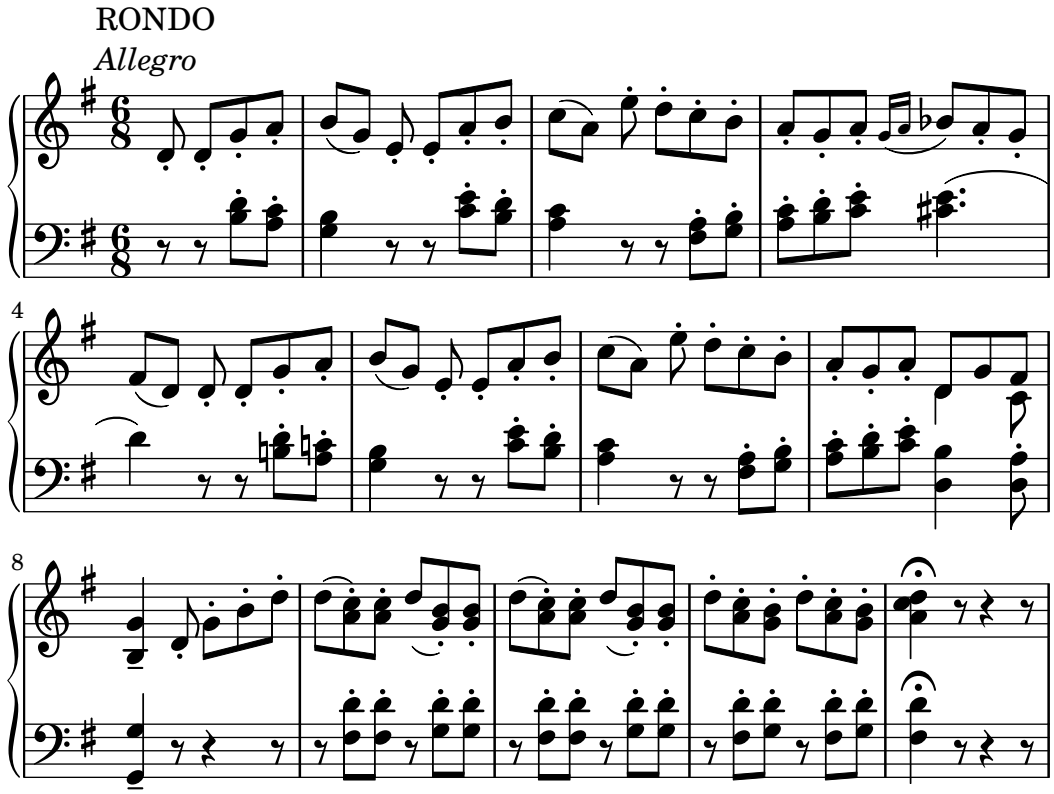
Siehe auch

Notationsreferenz: Abschnitt 2.5.2 [Taktzahlen], Seite 103, Abschnitt 2.3.4 [Musik ohne Metrum], Seite 73.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Referenz der Interna: Abschnitt “Timing_translator” in *Referenz der Interna*, Abschnitt “Score” in *Referenz der Interna*

3 Ausdrucksbezeichnungen



Dieser Abschnitt zeigt verschiedene Ausdrucksbezeichnungen, die zur Partitur hinzugefügt werden können.

3.1 Ausdrucksbezeichnungen an Noten angehängt

Dieser Abschnitt erklärt, wie man Ausdrucksbezeichnungen erstellt, die an Noten gebunden sind: Artikulationszeichen, Ornamente und Dynamikzeichen. Es werden auch Methoden gezeigt, eigene Ausdrucksbezeichnungen zu erstellen.
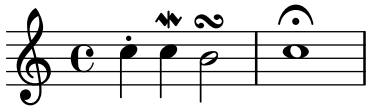
3.1.1 Artikulationszeichen und Verzierungen

Eine Vielfalt an Symbolen kann über und unter den Noten erscheinen, um zu markieren, auf welche Art die Note ausgeführt werden soll. Hierzu wird folgende Syntax benutzt:

```
Note\Bezeichnung
```

Die möglichen Werte für *Bezeichnung* sind aufgelistet in Abschnitt A.13 [Liste der Artikulationszeichen], Seite 760. Ein Beispiel:
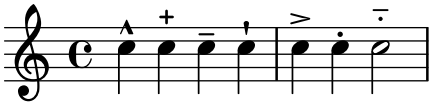
```
\relative {
  c''4\staccato c\mordent b2\turn
  c1\fermata
}
```



Einige dieser Artikulationszeichen haben eine Abkürzung, damit es einfacher ist, sie zu schreiben. Die Abkürzung wird an die Notenbezeichnung gehängt, wobei ihre Syntax aus einem Minuszeichen - besteht, gefolgt von dem Symbol, das dem Artikulationszeichen zugeordnet ist. Es

gibt diese Abkürzungen für *marcato*, *stopped* (gedämpft), *tenuto*, *staccatissimo*, *accent*, *staccato*, und *portato*. Die ihnen entsprechenden Symbole werden also folgendermaßen notiert:

```
\relative {
  c''4-^ c-+ c-- c-!
  c4-> c-. c2-_
}
```
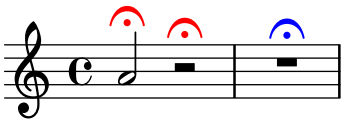


Die Regeln für die standardmäßige Platzierung von Artikulationszeichen werden in der Datei `scm/script.scm` definiert. Artikulationszeichen und Ornamente können manuell über oder unter dem System gesetzt werden, siehe Abschnitt 35.2 [Richtung und Platzierung], Seite 611.

Artikulationszeichen sind `Script`-Objekte. Ihre Eigenschaften werden ausführlich in Abschnitt “Script” in *Referenz der Interna* beschrieben.

Artikulationen können neben Noten auch an Pausen und Mehrtaktpausen gehängt werden. Beim Anhängen an eine Mehrtaktpause wird ein `MultiMeasureRestScript`-Objekt erstellt.

```
\override Script.color = #red
\override MultiMeasureRestScript.color = #blue
a'2\fermata r\fermata
R1\fermata
```



Zusätzlich zu den Artikulationszeichen können auch Text und Beschriftung an Noten angehängt werden. Siehe auch Abschnitt 8.1.1 [Textarten], Seite 226.

Zu weiterer Information über die Reihenfolge von Scripten und TextScripten, die an Noten angehängt werden, siehe Abschnitt “Positionierung von Objekten” in *Handbuch zum Lernen*.

Ausgewählte Schnipsel

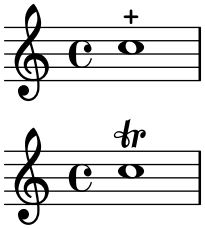
Die Standardwerte der Abkürzungen von Artikulationen verändern

Die Abkürzungen sind in der Datei `ly/script-init.ly` definiert, wo den Variablen `dashHat`, `dashPlus`, `dashDash`, `dashBar`, `dashLarger`, `dashDot` und `dashUnderscore` Standardwerte zugewiesen werden. Diese Standardwerte können verändert werden. Um zum Beispiel die Abkürzung `-+` (`dashPlus`) mit dem Triller anstatt mit dem +-Symbol zu assoziieren, muss der Wert `trill` der Variable `dashPlus` zugewiesen werden:

```
\relative c'' { c1-+ }

dashPlus = \trill

\relative c'' { c1-+ }
```



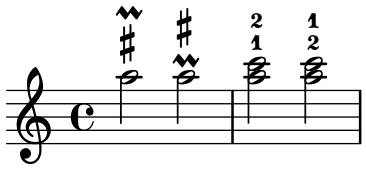
Die vertikale Anordnung von Beschriftungen kontrollieren

Die vertikale Anordnung von Beschriftungen wird mit der `'script-priority`-Eigenschaft kontrolliert. Um so kleiner die Zahl, umso näher wird die Beschriftung in Bezug auf die Note gesetzt. In diesem Beispiel hat das `TextScript`-Objekt (das Kreuz) zuerst die niedrigste Priorität, wird also auch am niedrigsten in dem ersten Beispiel gesetzt. Im zweiten Fall hat der Praller (das `Script`) die niedrigste Priorität, darum wird er am nächsten zum System gesetzt. Wenn zwei Objekte die gleiche Priorität haben, wird ihre Reihenfolge anhand ihres Auftretens in der Quelldatei entschieden.

```
\relative c''' {
  \once \override TextScript.script-priority = -100
  a2^\prall^\markup { \sharp }

  \once \override Script.script-priority = -100
  a2^\prall^\markup { \sharp }
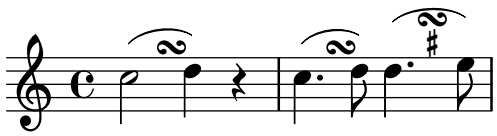
  \set fingeringOrientations = #'(up)
  <c-2 a-1>2
  <a-1 c\tweak script-priority -100 -2>2
}
```



Einen Doppelschlag mit Vorhalt erstellen

Einen Doppelschlag mit Vorhalt zu erstellen, wobei die untere Note das Vorzeichen benutzt, erfordert einige Einstellungsänderungen. Die `outside-staff-priority`-Eigenschaft muss auf falsch (`#f`) gesetzt werden, weil sie sonst über die Eigenschaft `avoid-slur` `property` dominieren würde. Der Wert von `halign` wird benutzt, um den Doppelschlag horizontal zu positionieren.

```
\relative c'' {
  \after 2*2/3 \turn c2( d4) r |
  \after 4 \turn c4.( d8)
  \after 4
  {
    \once \set suggestAccidentals = ##t
    \once \override AccidentalSuggestion.outside-staff-priority = ##f
    \once \override AccidentalSuggestion.avoid-slur = #'inside
    \once \override AccidentalSuggestion.font-size = -3
    \once \override AccidentalSuggestion.script-priority = -1
    \once \hideNotes
    cis8\turn \noBeam
  }
  d4.( e8)
}
```



Siehe auch

Glossar: Abschnitt “tenuto” in *Glossar*, Abschnitt “accent” in *Glossar*, Abschnitt “staccato” in *Glossar*, Abschnitt “portato” in *Glossar*.

Handbuch zum Lernen: Abschnitt “Positionierung von Objekten” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 8.1.1 [Textarten], Seite 226, Abschnitt 35.2 [Richtung und Platzierung], Seite 611, Abschnitt A.13 [Liste der Artikulationszeichen], Seite 760, Abschnitt 3.3.3 [Triller], Seite 141.

Installierte Dateien: `scm/script.scm`.

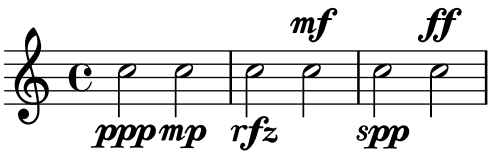
Schnipsel: Abschnitt “Expressive marks” in *Schnipsel*.

Referenz der Interna: Abschnitt “Script” in *Referenz der Interna*, Abschnitt “TextScript” in *Referenz der Interna*.

3.1.2 Dynamik

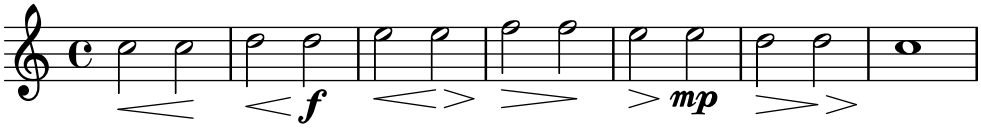
Absolute Dynamikbezeichnung wird mit Befehlen nach den Noten angezeigt, etwa `c4\ff`. Die vordefinierten Befehle lauten: `\ppppp`, `\pppp`, `\ppp`, `\pp`, `\p`, `\mp`, `\mf`, `\f`, `\ff`, `\fff`, `\ffff`, `fffff`, `\fp`, `\sf`, `\sff`, `\sp`, `\spp`, `\sfz`, and `\rfz`. Die Dynamikzeichen können manuell unter- oder oberhalb des Systems platziert werden, siehe Abschnitt 35.2 [Richtung und Platzierung], Seite 611.

```
\relative c'' {
  c2\ppp c\mp
  c2\rfz c^\mf
  c2_\spp c^\ff
}
```



Eine *Crescendo*-Klammer wird mit dem Befehl `\<` begonnen und mit `\!`, einem absoluten Dynamikbefehl oder einer weiteren Crescendo- oder Decrescendo-Klammer beendet. Ein *Decrescendo* beginnt mit `\>` und wird auch beendet mit `\!`, einem absoluten Dynamikbefehl oder einem weiteren Crescendo oder Decrescendo. `\cr` und `\decr` können anstelle von `\<` und `\>` benutzt werden. Die Befehle ergeben standardmäßig Crescendo-Klammern.

```
\relative c'' {
  c2\< c\!
  d2\< d\f
  e2\< e\>
  f2\> f\!
  e2\> e\mp
  d2\> d\>
  c1\!
}
```



Eine Crescendo-Klammer, die mit `\!` beendet wird, endet an der rechten Seite der Note, welcher `\!` zugeordnet ist. In dem Fall, dass es durch den Beginn eines anderen *crescendo*- oder

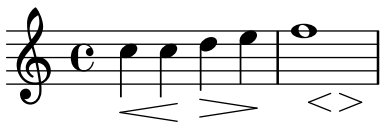
decrescendo-Zeichens beendet wird, endet es in der Mitte der Note, welche das nächste \< oder \> angehängt hat. Die nächste Klammer beginnt dann am rechten Rand der selben Note anstatt dem normalerweise linken Rand, wenn die vorherige Klammer mit \! beendet worden wäre.

```
\relative {
  c''1\< | c4 a c\< a | c4 a c\! a\< | c4 a c a\!
}
```
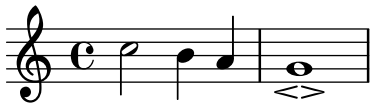


Leere Pausenzeichen werden benötigt, um mehrere Zeichen für eine Note zu notieren. Das ist insbesondere nützlich, wenn man *crescendo* und *decrescendo* zu der selben Note hinzufügen will:

```
\relative {
  c''4\< c\! d\> e\!
  << f1 { s4 s4\< s4\> s4\! } >>
}
```
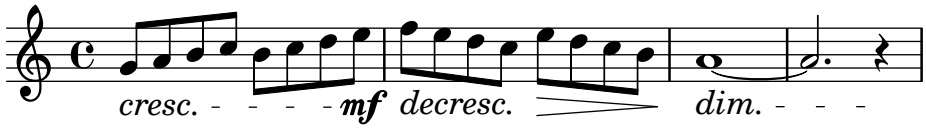


Der \espressivo-Befehl kann eingesetzt werden, um crescendo und decrescendo für die selbe Note anzuzeigen. Dieser Befehl ist jedoch als Artikulation, nicht als Dynamikzeichen implementiert.

```
\relative {
  c''2 b4 a
  g1\espressivo
}
```
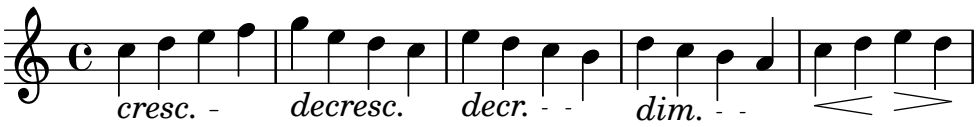


Mit Text gesetzte Crescendo-Bezeichnungen beginnen mit \cresc. Mit Text gesetzte Decrescendo-Bezeichnungen beginnen mit \decresc oder \dim. Fortsetzungslinien werden gesetzt, wenn sie benötigt werden.

```
\relative {
  g'8\cresc a b c b c d e\mf |
  f8\decresc e d c e\> d c b |
  a1\dim ~ |
  a2. r4\! |
}
```



Als Text gesetzte Dynamik-Bezeichnungen können auch die Crescendo-Klammern ersetzen:

```
\relative c'' {
  \crescTextCresc
  c4\< d e f\! |
  \dimTextDecresc
  g4\> e d c\! |
  \dimTextDecr
  e4\> d c b\! |
  \dimTextDim
  d4\> c b a\! |
  \crescHairpin
  \dimHairpin
  c4\< d\! e\> d\! |
}
```
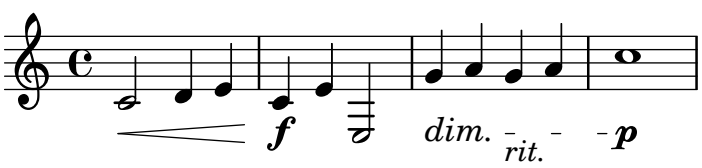


Um neue absolute Dynamikzeichen oder Text, der mit ihnen angeordnet wird, zu erstellen, siehe Abschnitt 3.1.3 [Neue Lautstärkezeichen], Seite 128.

Vertikale Position der Zeichen wird von der Funktion Abschnitt “DynamicLineSpanner” in *Referenz der Interna* verwaltet.

Es gibt einen besonderen `Dynamics`-Kontext, um Crescendi und Decrescendi auf einer eigenen Zeile zu notieren. Mit leeren Pausen (`s`) werden die Dauern gesetzt. (Auch Noten in einem `Dynamics`-Kontext nehmen eine Dauer ein, werden aber nicht gesetzt.) Der `Dynamics`-Kontext ist sehr nützlich, um andere Elemente wie Textbeschriftung, Text-Strecker und Klavierpedalbezeichnungen aufzunehmen.

```
<<
  \new Staff \relative {
    c'2 d4 e |
    c4 e e,2 |
    g'4 a g a |
    c1 |
  }
  \new Dynamics {
    s1\< |
    s1\f |
    s2\dim s2-"rit." |
    s1\p |
  }
>>
```



Vordefinierte Befehle

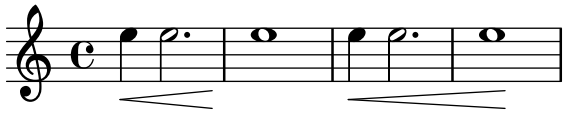
`\dynamicUp`, `\dynamicDown`, `\dynamicNeutral`, `\crescTextCresc`, `\dimTextDim`, `\dimTextDecr`, `\dimTextDecresc`, `\crescHairpin`, `\dimHairpin`.

Ausgewählte Schnipsel

Das Verhalten von Crescendo-Klammern an Taktlinien beeinflussen

Wenn die Note, an welcher eine Crescendo-Klammer endet, die erste Note eines Taktes ist, wird die Klammer an der vorhergehenden Taktlinie beendet. Dieses Verhalten kann auch mit der Eigenschaft `'to-barline` geändert werden:

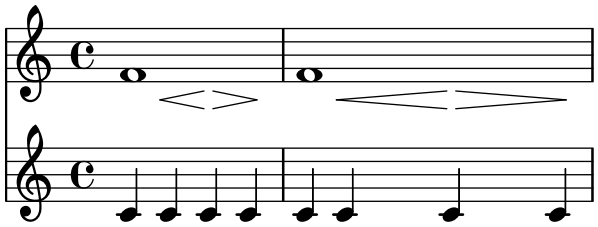
```
\relative c'' {
  e4\< e2.
  e1\!
  \override Hairpin.to-barline = ##f
  e4\< e2.
  e1\!
}
```



Die Mindestlänge von Crescendo-Klammern bestimmen

Wenn Crescendo-Klammern zu kurz sind, können sie verlängert werden, indem die `minimum-length`-Eigenschaft des `Hairpin`-Objektes verändert wird.
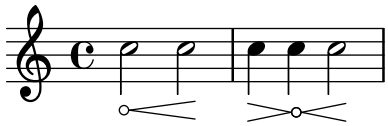
```
<<
  {
    \after 4 \< \after 2 \> \after 2. \! f'1
    \override Hairpin.minimum-length = 8
    \after 4 \< \after 2 \> \after 2. \! f'1
  }
  {
    \*8 c'4
  }
>>
```



Crescendo Klammern al niente schreiben

Crescendo-Klammern können mit einem kleinen Kreis vor der Spitze notiert werden (al niente = bis zum Nichts), indem die `circled-tip`-Eigenschaft des `Hairpin`-Objekts auf `#t` gesetzt wird.

```
\relative c'' {
  \override Hairpin.circled-tip = ##t
  c2\< c\!
  c4\> c\< c2\!
}
```



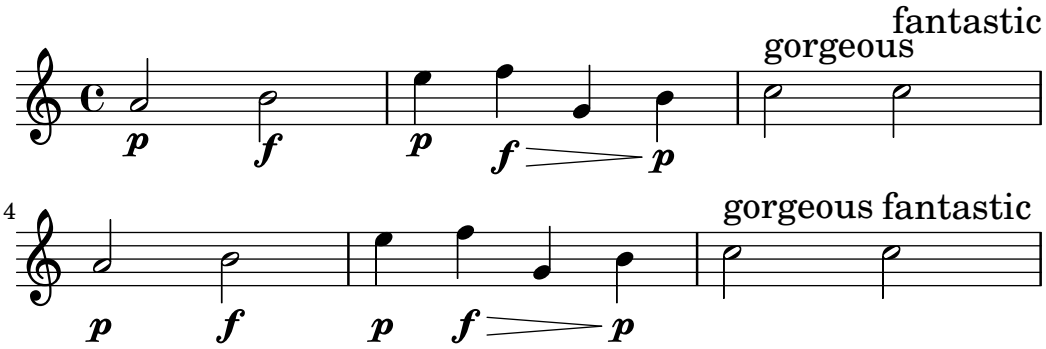
Vertikale Ausrichtung von Dynamik und Textbeschriftung beeinflussen

Indem man die `'Y-extent`-Eigenschaft auf einen passenden Wert setzt, können alle `DynamicLineSpanner`-Objekte (Crescendo-Klammern und Dynamik-Texte) (hairpins and dynamic texts) unabhängig von ihrer wirklichen Ausdehnung an einem gemeinsamen Referenzpunkt ausgerichtet werden. Auf diese Weise ist jedes Element vertikal ausgerichtet und der Notensatz sieht ansprechender aus.

Die gleiche Idee wird benutzt, um Textbeschriftungen an ihrer Grundlinie auszurichten.

```
music = \relative c' {
  a'2\p b\f
  e4\p f\f\> g, b\p
  c2^\markup { \huge gorgeous } c^\markup { \huge fantastic }
}

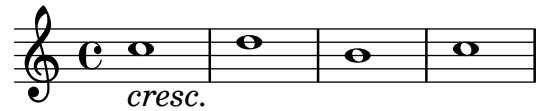
{
  \music
  \break
  \override DynamicLineSpanner.staff-padding = 3
  \textLengthOn
  \override TextScript.staff-padding = 1
  \music
}
```



Crescendo-Linien von Dynamik-Texten unterdrücken

Dynamik-Texte (wie cresc. und dim.) werden mit einer gestrichelten Linie gesetzt, die ihre Dauer anzeigt. Diese Linie kann auf folgende Weise unterdrückt werden:
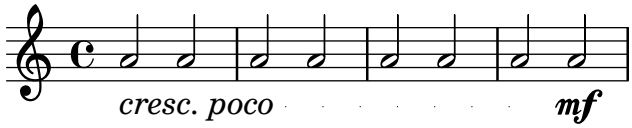
```
\relative c'' {
  \override DynamicTextSpanner.style = #'none
  \crescTextCresc
  c1\< | d | b | c\!
}
```



Text und Strecker-Stile für Dynamik-Texte ändern

Der Text, der für Crescendo und Decrescendo gesetzt wird, kann geändert werden, indem man die Eigenschaften `crescendoText` und `decrescendoText` verändert. Der Stil des Streckers kann auch geändert werden, indem die `'style`-Eigenschaft des `DynamicTextSpanner` beeinflusst wird. Der Standardwert ist `'hairpin`, andere Möglichkeiten sind `'line`, `'dashed-line` und `'dotted-line`.

```
\relative c'' {
  \set crescendoText = \markup { \italic { cresc. poco } }
  \set crescendoSpanner = #'text
  \override DynamicTextSpanner.style = #'dotted-line
  a2\< a
  a2 a
  a2 a
  a2 a\mf
}
```



Siehe auch

Glossar: Abschnitt “al niente” in *Glossar*, Abschnitt “crescendo” in *Glossar*, Abschnitt “decrescendo” in *Glossar*, Abschnitt “hairpin” in *Glossar*. Handbuch zum Lernen: Abschnitt “Artikulationszeichen und Lautstärke” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 35.2 [Richtung und Platzierung], Seite 611, Abschnitt 3.1.3 [Neue Lautstärkezeichen], Seite 128, Abschnitt 23.3 [Was geht in die MIDI-Ausgabe], Seite 509, Abschnitt 23.5 [MIDI-Lautstärke kontrollieren], Seite 510.

Schnipsel: Abschnitt “Expressive marks” in *Schnipsel*.
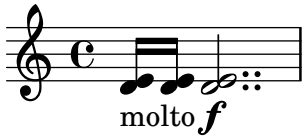
Referenz der Interna: Abschnitt “DynamicText” in *Referenz der Interna*, Abschnitt “Hairpin” in *Referenz der Interna*, Abschnitt “DynamicLineSpanner” in *Referenz der Interna*, Abschnitt “Dynamics” in *Referenz der Interna*.

3.1.3 Neue Lautstärkezeichen

Die einfachste Art, eigene Dynamikbezeichnungen zu erstellen, ist die Benutzung von \markup-(Textbeschriftungs)-Objekten.

```
moltoF = \markup { molto \dynamic f }

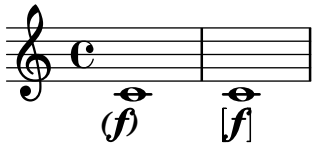
\relative {
  <d' e>16_\moltoF <d e>
  <d e>2..
}
```



Mit einer Textbeschriftung können editorische Dynamikzeichen (in runden oder eckigen Klammern) erstellt werden. Die Syntax für den Textbeschriftungsmodus wird erklärt in Abschnitt 8.2 [Text formatieren], Seite 233.
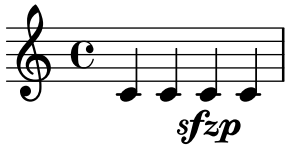
```
roundF = \markup {
    \center-align \concat { \bold { \italic ( }
          \dynamic f \bold { \italic ) } } }
boxF = \markup { \bracket { \dynamic f } }
\relative {
  c'1_\roundF
  c1_\boxF
```

```
}
```



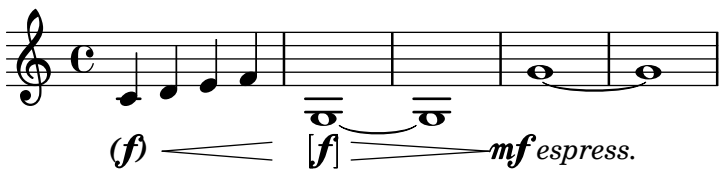
Einfache, mittig gesetzte Dynamikzeichen können schnell mit der `make-dynamic-script`-Funktion erstellt werden.

```
sfzp = #(make-dynamic-script "sfzp")
\relative {
  c'4 c c\sfzp c
}
```



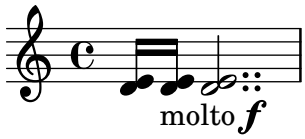
Allgemein gesagt kann `make-dynamic-script` jegliches Textbeschriftungsobjekt als Argument haben. Die Schriftart für Dynamikzeichen enthält nur die Buchstaben `f,m,p,r,s` sowie z; ein Dynamikzeichen, das anderen Text oder Satzzeichen enthalten soll, benötigt Textbeschriftungsbefehle, die die normale Schriftart einschalten, etwa `\normal-text`. Die Funktion `make-dynamic-script` sollte anstelle einer normalen Textbeschriftung vor allem deshalb benutzt werden, weil auf diese Weise die vertikale Ausrichtung von den Textbeschriftungen (engl. markup) und den spitzen Klammern an der selben Linie gewährleistet wird.

```
roundF = \markup { \center-align \concat {
           \normal-text { \bold { \italic ( } }
           \dynamic f
           \normal-text { \bold { \italic ) } } } }
boxF = \markup { \bracket { \dynamic f } }
mfEspress = \markup { \center-align \line {
              \hspace #3.7 mf \normal-text \italic espress. } }
roundFdynamic = #(make-dynamic-script roundF)
boxFdynamic = #(make-dynamic-script boxF)
mfEspressDynamic = #(make-dynamic-script mfEspress)
\relative {
  c'4_\roundFdynamic\< d e f
  g,1~_\boxFdynamic\>
  g1
  g'1~\mfEspressDynamic
  g1
}
```



Anstelle dessen kann auch die Scheme-Form des Beschriftungs-Modus verwendet werden. Seine Syntax ist erklärt in Abschnitt “Beschriftungskonstruktionen in Scheme” in *Extending*.

```
moltoF = #(make-dynamic-script
            (markup #:normal-text "molto"
                    #:dynamic "f"))
\relative {
  <d' e>16 <d e>
  <d e>2..\moltoF
}
```



Die Auswahl von Schriftarten in Textbeschriftungen ist erklärt in Abschnitt 8.2.2 [Überblick über die wichtigsten Textbeschriftungsbefehle], Seite 234.

Siehe auch

Notationsreferenz: Abschnitt 8.2 [Text formatieren], Seite 233, Abschnitt 8.2.2 [Überblick über die wichtigsten Textbeschriftungsbefehle], Seite 234, Abschnitt 23.3 [Was geht in die MIDI-Ausgabe], Seite 509, Abschnitt 23.5 [MIDI-Lautstärke kontrollieren], Seite 510.

Erweitern: Abschnitt “Beschriftungskonstruktionen in Scheme” in *Extending*.

Schnipsel: Abschnitt “Expressive marks” in *Schnipsel*.

3.2 Ausdrucksbezeichnungen als Bögen
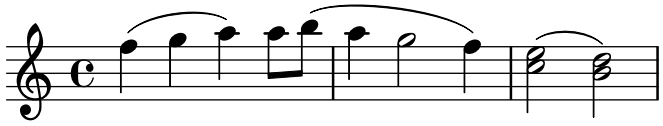
Dieser Abschnitt erklärt, wie man verschiedene gebogene Ausdrucksbezeichnungen erstellt: Legato- und Phrasierungsbögen, Atemzeichen und Glissandos zu unbestimmten Tonhöhen.

3.2.1 Legatobögen

Ein Legatobogen (engl. slur) zeigt an, dass die Noten *legato* gespielt werden sollen. Er wird mit Klammern hinter den Notenwerten notiert.

> **Achtung:** In polyphoner Musik muss ein Legatobogen in der gleichen Stimme beendet werden, in der er begonnen wurde.

```
\relative {
  f''4( g a) a8 b(
  a4 g2 f4)
  <c e>2( <b d>2)
}
```
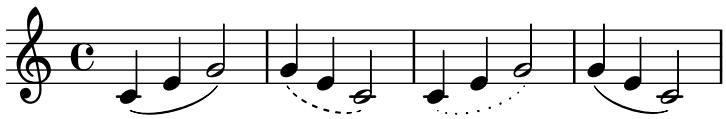


Legatobögen können manuell ober- oder unterhalb des Notensystems besetzt werden, siehe Abschnitt 35.2 [Richtung und Platzierung], Seite 611.

Gleichzeitige, überlappende Legatobögen sind nicht erlaubt, aber ein Phrasierungsbogen kann einen Legatobogen überlappen. Damit können zwei Bögen gleichzeitig ausgegeben werden. Siehe auch Abschnitt 3.2.2 [Phrasierungsbögen], Seite 133.
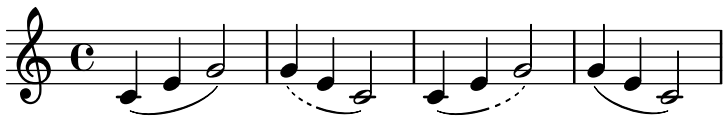
Legatobögen können durchgehend, gepunktet oder gestrichelt dargestellt werden. Standard ist der durchgehende Bogen:

```
\relative {
  c'4( e g2)
  \slurDashed
  g4( e c2)
  \slurDotted
  c4( e g2)
  \slurSolid
  g4( e c2)
}
```
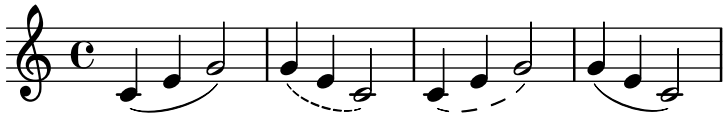


Bögen können auch halb gestrichelt (die erste Hälfte gestrichelt, die zweite Hälfte durchgehend) erstellt werden, oder als halb durchgehend (die erste Hälfte durchgehend, die zweite Hälfte gestrichelt:

```
\relative {
  c'4( e g2)
  \slurHalfDashed
  g4( e c2)
  \slurHalfSolid
  c4( e g2)
  \slurSolid
  g4( e c2)
}
```



Eigene Muster für die Strichelung können definiert werden:

```
\relative {
  c'4( e g2)
  \slurDashPattern #0.7 #0.75
  g4( e c2)
  \slurDashPattern #0.5 #2.0
  c4( e g2)
  \slurSolid
  g4( e c2)
}
```



Vordefinierte Befehle

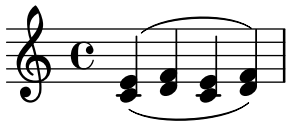
`\slurUp`, `\slurDown`, `\slurNeutral`, `\slurDashed`, `\slurDotted`, `\slurHalfDashed`, `\slurHalfSolid`, `\slurDashPattern`, `\slurSolid`.

Ausgewählte Schnipsel

Doppelte Bögen für Legato-Akkorde benutzen

Einige Komponisten schreiben doppelte Bögen, wenn Legato-Akkorde notiert werden. Das kann mit der Eigenschaft `doubleSlurs` erreicht werden.
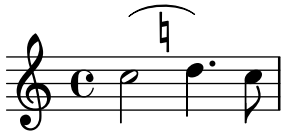
```
\relative c' {
  \set doubleSlurs = ##t
  <c e>4( <d f> <c e> <d f>)
}
```



Textbeschriftung innerhalb von Bögen positionieren

Textbeschriftung kann innerhalb von Bögen gesetzt werden, wenn die `outside-staff-priority`-Eigenschaft auf falsch gesetzt wird.

```
\relative c'' {
  \override TextScript.avoid-slur = #'inside
  \override TextScript.outside-staff-priority = ##f
  c2(^\markup { \halign #-10 \natural } d4.) c8
}
```
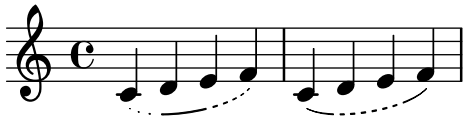


Legatobögen mit kompliziertem Strichelmuster definieren

Legatobögen können mit einem komplizierten Strichelmuster gesetzt werden, indem die `dash-definition`-Eigenschaft definiert wird. `dash-definition` ist eine Liste bestehend aus `dash-elements`-Elementen. Ein `dash-element` ist eine Liste an Parametern, die das Strichverhalten für einen Abschnitt des Legatobogens definieren.

Der Bogen wird nach dem Bezierparameter t definiert, welcher von 0 am linken Ende des Bogens zu 1 am rechten Ende des Bogens reicht. `dash-element` ist eine Liste (`start-t stop-t dash-Unterbrechung dash-Abschnitt`). Die Region des Bogens von `start-t` bis `stop-t` hat eine Unterbrechung von `dash-Unterbrechung` von jedem `dash-Abschnitt`-Schwarzabschnitt. `dash-Abschnitt` ist in Notenlinienzwischenräumen definiert. `dash-Abschnitt` ist auf 1 für einen durchgehenden Bogen gesetzt.

```
\relative c' {
  \once \override
    Slur.dash-definition = #'((  0  0.3  0.1 0.75)
                              (0.3  0.6  1   1   )
                              (0.65 1.0  0.4 0.75))
  c4( d e f)
  \once \override
    Slur.dash-definition = #'((0    0.25  1   1   )
                              (0.3  0.7   0.4 0.75)
                              (0.75 1.0   1   1   ))
  c4( d e f)
}
```



Siehe auch

Glossar: Abschnitt “slur” in *Glossar*.

Handbuch zum Lernen: Abschnitt “Über die Nicht-Schachtelung von Klammern und Bindebögen” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 35.2 [Richtung und Platzierung], Seite 611, Abschnitt 3.2.2 [Phrasierungsbögen], Seite 133.
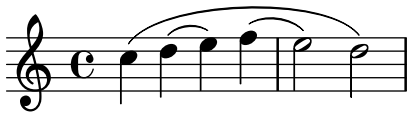
Schnipsel: Abschnitt “Expressive marks” in *Schnipsel*.

Referenz der Interna: Abschnitt “Slur” in *Referenz der Interna*.

3.2.2 Phrasierungsbögen

Ein Phrasierungsbogen verbindet Noten und wird verwendet, um einen musikalischen Ausdruck anzuzeigen. Er wird mit den Befehlen `\(` und `\)` eingegeben.

```
\relative {
  c''4\( d( e) f(
  e2) d\)
}
```
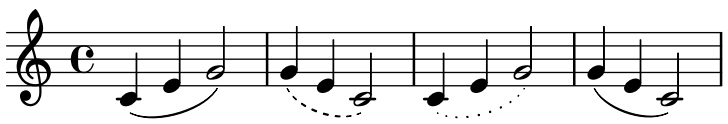


Im typographischen Sinne verhalten sich Phrasierungsbögen genauso wie Legatobögen. Sie werden aber als eigene Objekte behandelt. Ein `\slurUp` hat also keine Auswirkung auf die Phrasierungsbögen. Phrasierungsbögen können manuell oberhalb oder unterhalb des Notensystems gesetzt werden, siehe Abschnitt 35.2 [Richtung und Platzierung], Seite 611.

Simultane oder überlappende Phrasierungsbögen sind nicht erlaubt.

Phrasierungsbögen können durchgehend, gepunktet oder gestrichelt dargestellt werden. Standard ist der durchgehende Bogen:
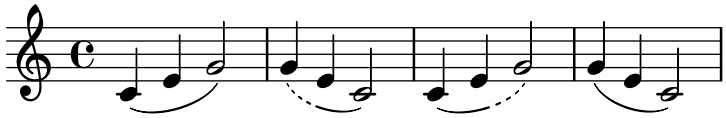
```
\relative {
  c'4\( e g2\)
  \phrasingSlurDashed
  g4\( e c2\)
  \phrasingSlurDotted
  c4\( e g2\)
  \phrasingSlurSolid
  g4\( e c2\)
}
```



funindex phrasingSlurHalfDashed
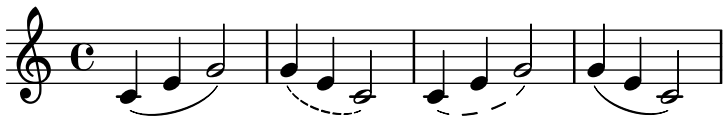
Phrasierungsbögen können auch als halbgestrichelt dargestellt werden (die erste Hälfte gestrichelt, die zweite Hälfte durchgehend, oder halb durchgehend (die erste Hälfte durchgehend, die zweite gestrichelt):

```
\relative {
  c'4\( e g2\)
  \phrasingSlurHalfDashed
  g4\( e c2\)
  \phrasingSlurHalfSolid
  c4\( e g2\)
  \phrasingSlurSolid
  g4\( e c2\)
}
```



Eigene Strichelmuster für Phrasierungsbögen können definiert werden:

```
\relative {
  c'4\( e g2\)
  \phrasingSlurDashPattern #0.7 #0.75
  g4\( e c2\)
  \phrasingSlurDashPattern #0.5 #2.0
  c4\( e g2\)
  \phrasingSlurSolid
  g4\( e c2\)
}
```



Strichelmusterdefinitionen für Phrasierungsbögen haben die gleiche Struktur wie die Definitionen für Legatobögen. Zu mehr Information über komplizierte Strichelmuster, siehe die Schnipsel im Abschnitt Abschnitt 3.2.1 [Legatobögen], Seite 130.

Vordefinierte Befehle

`\phrasingSlurUp`, `\phrasingSlurDown`, `\phrasingSlurNeutral`, `\phrasingSlurDashed`, `\phrasingSlurDotted`, `\phrasingSlurHalfDashed`, `\phrasingSlurHalfSolid`, `\phrasingSlurDashPattern`, `\phrasingSlurSolid`.

Siehe auch

Handbuch zum Lernen: Abschnitt “Über die Nicht-Schachtelung von Klammern und Bindebögen” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 35.2 [Richtung und Platzierung], Seite 611.
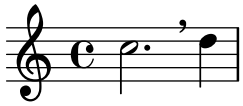
Schnipsel: Abschnitt “Expressive marks” in *Schnipsel*.

Referenz der Interna: Abschnitt “PhrasingSlur” in *Referenz der Interna*.

3.2.3 Atemzeichen
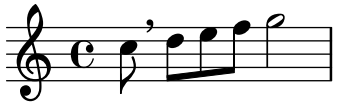
Atemzeichen werden mit dem Befehl `\breathe` eingegeben.

```
{ c''2. \breathe d''4 }
```



Ein Atemzeichen bezeichnet gleichzeitig das Ende eines automatischen Balkens. Um das Verhalten zu verändern siehe Abschnitt 2.4.3 [Manuelle Balken], Seite 94.

```
\relative { c''8 \breathe d e f g2 }
```



Musikalische Zeichen für Atemzeichen in Alter Notation, auch Divisiones genannt, sind unterstützt. Für Einzelheiten siehe Abschnitt 17.4.4 [Divisiones], Seite 433.

Ausgewählte Schnipsel

Das Atemzeichen-Symbol verändern

Das Schriftzeichen für das Atemzeichen kann verändert werden, indem die Text-Eigenschaft des `BreathingSign`-Layoutobjekts mit einer beliebigen Textbeschriftung definiert wird.

```
\relative c'' {
  c2
  \override BreathingSign.text =
    \markup { \musicglyph "scripts.rvarcomma" }
  \breathe
  d2
}
```
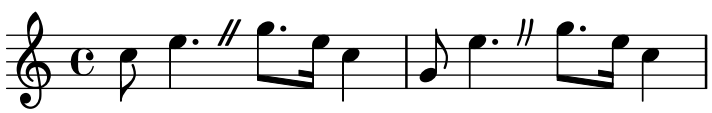


Eine Zäsur einfügen

Zäsurzeichen können erstellt werden, indem die `'text`-Eigenschaft des `BreathingSign`-Objektes verändert wird. Ein gekrümmtes Zäsurzeichen ist auch möglich.

```
\relative c'' {
  \override BreathingSign.text = \markup {
    \musicglyph "scripts.caesura.straight"
  }
  c8 e4. \breathe g8. e16 c4

  \override BreathingSign.text = \markup {
    \musicglyph "scripts.caesura.curved"
  }
  g8 e'4. \breathe g8. e16 c4
}
```



Siehe auch

Glossar: Abschnitt “caesura” in *Glossar*.

Notationsreferenz: Abschnitt 17.4.4 [Divisiones], Seite 433.

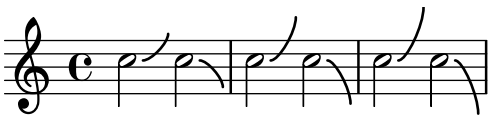
Schnipsel: Abschnitt “Expressive marks” in *Schnipsel*.

Referenz der Interna: Abschnitt “BreathingEvent” in *Referenz der Interna*, Abschnitt “BreathingSign” in *Referenz der Interna*, Abschnitt “Breathing_sign_engraver” in *Referenz der Interna*.

3.2.4 Glissando zu unbestimmter Tonhöhe

Gleiten nach oben und unten kann mit dem Befehl `\bendAfter` notiert werden. Die Richtung des Glissandos wird mit einem Plus oder Minus (nach oben bzw. nach unten) angezeigt. Die Zahl zeigt die Intervallgröße an, über die sich das Glissando *nach* der Note erstreckt.

```
\relative c'' {
  c2\bendAfter #+4
  c2\bendAfter #-4
  c2\bendAfter #+6.5
  c2\bendAfter #-6.5
  c2\bendAfter #+8
  c2\bendAfter #-8
}
```



Ausgewählte Schnipsel

Das Aussehen von unbestimmten Glissandi anpassen

Die `shortest-duration-space`-Eigenschaft kann verändert werden, um das Aussehen von unbestimmten Glissandi anzupassen.

```
\relative c'' {
  \override Score.SpacingSpanner.shortest-duration-space = 4.0
  c2-\bendAfter 5
  c2-\bendAfter -4.75
  c2-\bendAfter 8.5
  c2-\bendAfter -6
}
```



Siehe auch

Glossar: Abschnitt “fall” in *Glossar*, Abschnitt “doit” in *Glossar*.

Schnipsel: Abschnitt “Expressive marks” in *Schnipsel*.
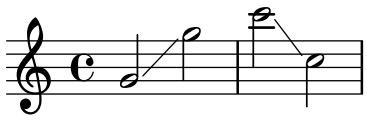
3.3 Ausdrucksbezeichnungen als Linien

Dieser Abschnitt zeigt, wie man verschiedene Ausdrucksbezeichnungen erstellt, die sich linear erstrecken: Glissando, Arpeggio und Triller.

3.3.1 Glissando

Ein *Glissando* wird mit dem Befehl `\glissando` auf eine Note folgend notiert:

```
g2\glissando g'
c2\glissando c,
```
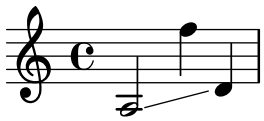


Verschiedene Glissando-Stile sind möglich. Für Einzelheiten siehe Abschnitt 35.7 [Linienstile], Seite 624.

Ausgewählte Schnipsel

Glissando kann Grobs überspringen

`NoteColumn`-Grobs können bei Glissandos übersprungen werden.
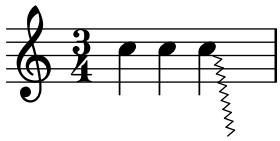
```
\relative c' {
  a2 \glissando
  \once \override NoteColumn.glissando-skip = ##t
  f''4 d,
}
```



Moderne Glissandi

Ein modernes Glissando ohne eine Endnote kann gesetzt werden, indem eine Kadenz eingesetzt wird und die Endnote unsichtbar gemacht wird.

```
\relative c'' {
  \time 3/4
  \override Glissando.style = #'zigzag
  c4 c
  \cadenzaOn
  c4\glissando
  \hideNotes
  c,,4
  \unHideNotes
  \cadenzaOff
  \bar "|"
}
```



Siehe auch

Glossar: Abschnitt “glissando” in *Glossar*.

Notationsreferenz: Abschnitt 35.7 [Linienstile], Seite 624.

Schnipsel: Abschnitt “Expressive marks” in *Schnipsel*.

Referenz der Interna: Abschnitt “Glissando” in *Referenz der Interna*.
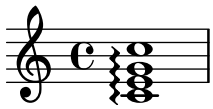
Bekannte Probleme und Warnungen

Printing text over the line (such as *gliss.*) is not supported.

3.3.2 Arpeggio

Ein *Arpeggio* als Zeichen, dass ein Akkord gebrochen gespielt werden soll, kann mit dem Befehl `\arpeggio` hinter der Akkord-Konstruktion erzeugt werden.

```
\relative { <c' e g c>1\arpeggio }
```



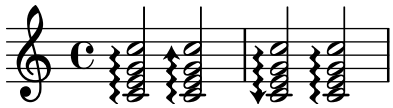
Unterschiedliche Arpeggio-Typen können benutzt werden. `\arpeggioNormal` stellt wieder das normale Verhalten her:

```
\relative {
  <c' e g c>2\arpeggio

  \arpeggioArrowUp
  <c e g c>2\arpeggio

  \arpeggioArrowDown
  <c e g c>2\arpeggio

  \arpeggioNormal
  <c e g c>2\arpeggio
}
```



Besondere Arpeggios mit Klammern können erstellt werden:

```
\relative {
  <c' e g c>2

  \arpeggioBracket
  <c e g c>2\arpeggio

  \arpeggioParenthesis
  <c e g c>2\arpeggio

  \arpeggioParenthesisDashed
  <c e g c>2\arpeggio

  \arpeggioNormal
  <c e g c>2\arpeggio
}
```



Die dash-Eigenschaft der Arpeggioklammern werden von der 'dash-definition-Eigenschaft kontrolliert, die beschrieben ist in Abschnitt 3.2.1 [Legatobögen], Seite 130.

Ein Arpeggio kann auch explizit ausgeschrieben werden, indem Überbindungsbögen benutzt werden. Für mehr Information siehe Abschnitt 2.1.4 [Bindebögen], Seite 53.

Vordefinierte Befehle

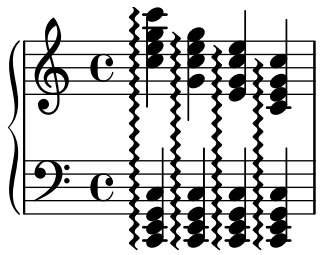
\arpeggio, \arpeggioArrowUp, \arpeggioArrowDown, \arpeggioNormal, \arpeggioBracket, \arpeggioParenthesis, \arpeggioParenthesisDashed.

Ausgewählte Schnipsel

Arpeggio über mehrere Systeme in anderen Kontexten

Arpeggio über mehrere Systeme können in anderen Kontexten als dem PianoStaff erstellt werden, wenn der Span_arpeggio_engraver in den Score-Kontext eingefügt wird.

```
\new PianoStaff \relative c'' <<
  \set PianoStaff.connectArpeggios = ##t
  \new Staff {
    <c e g c>4\arpeggio
    <g c e g>4\arpeggio
    <e g c e>4\arpeggio
    <c e g c>4\arpeggio
  }
  \new Staff {
    \clef bass
    \*4 <c,, e g c>4\arpeggio
  }
>>
```



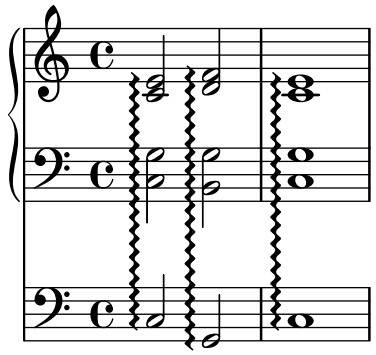
Arpeggio zwischen Systemen in einem Klaviersystem erstellen

In einem Klaviersystem (PianoStaff) ist es möglich, ein Arpeggio zwischen beiden Systemen zu verbinden, indem die PianoStaff.connectArpeggios-Eigenschaft gesetzt wird.

```
<<
  \new PianoStaff <<
    \new Voice \relative c' {
      <c e>2\arpeggio <d f>2\arpeggio
      <c e>1\arpeggio
    }
    \new Voice \relative c {
      \clef bass
      <c g'>2\arpeggio <b g'>2\arpeggio
      <c g'>1\arpeggio
    }
  >>
```

```
  \new Staff \relative c {
    \set Score.connectArpeggios = ##t
    \clef bass
    c2\arpeggio g\arpeggio
    c1\arpeggio
  }
>>

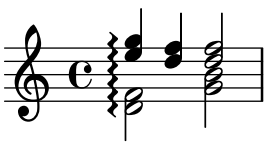
\layout {
  \context {
    \Score
    \consists "Span_arpeggio_engraver"
  }
}
```



Arpeggios zwischen unterschiedlichen Stimmen erzeugen

Ein Arpeggio kann zwischen Noten aus unterschiedlichen Stimmen auf demselben System gezogen werden, wenn der `Span_arpeggio_engraver` in den `Staff`-Kontext verschoben wird:

```
\new Staff \with {
  \consists "Span_arpeggio_engraver"
}
\relative c' {
  \set Staff.connectArpeggios = ##t
  <<
    { <e' g>4\arpeggio <d f> <d f>2 }
    \\
    { <d, f>2\arpeggio <g b>2 }
  >>
}
```



Siehe auch

Glossar: Abschnitt “arpeggio” in *Glossar*.

Notationsreferenz: Abschnitt 3.2.1 [Legatobögen], Seite 130, Abschnitt 2.1.4 [Bindebögen], Seite 53.

Schnipsel: Abschnitt “Expressive marks” in *Schnipsel*.

Referenz der Interna: Abschnitt “Arpeggio” in *Referenz der Interna*, Abschnitt “Slur” in *Referenz der Interna*, Abschnitt “PianoStaff” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Es ist nicht möglich, Arpeggios zwischen Systemen und solche, die sich nur auf ein System erstrecken, zum gleichen Zeitpunkt in einem Klaviersystem (`PianoStaff`) zu benutzen.

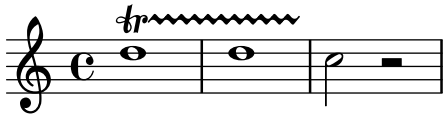
Die Arpeggios im Klammer-Stil funktionieren nicht über mehrere Notensysteme.

3.3.3 Triller

Kurze Triller ohne eine Dauer werden mit dem Befehl `\trill` notiert, siehe auch Abschnitt 3.1.1 [Artikulationszeichen und Verzierungen], Seite 120.
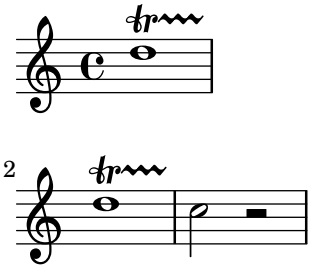
Längere Triller mit einer Dauer werden mit den Befehlen `\startTrillSpan` zu Beginn und `\stopTrillSpan` am Ende erstellt.

```
\relative {
  d''1\startTrillSpan
  d1
  c2\stopTrillSpan
  r2
}
```



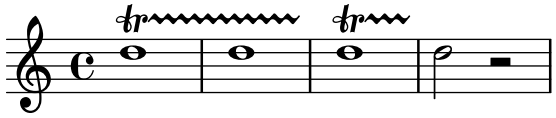
Ein Triller-Strekcer, der über einen Zeilenumbruch geht, beginnt genau über der ersten Note auf der neue Zeile erneut.

```
\relative {
  d''1\startTrillSpan
  \break
  d1
  c2\stopTrillSpan
  r2
}
```
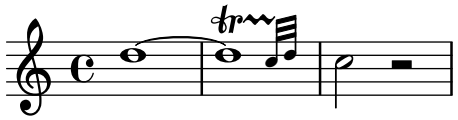


Aufeinanderfolgende Trillerstrecker funktieren ohne einen `\stopTrillSpan`-Befehl, weil ein folgender Strecker automatisch die rechte Begrenzung des vorhergehenden beendet.

```
d1\startTrillSpan
d1
d1\startTrillSpan
d2\stopTrillSpan
r2
```



Triller können auch mit Vorschlägen kombiniert werden. Die Syntax für diese Konstruktion und die Methode, um die Position der Vorschläge präzise zu positionieren, wird gezeigt in Abschnitt 2.6.1 [Verzierungen], Seite 112.

```
\relative {
  d''1~\afterGrace
  d1\startTrillSpan { c32[ d]\stopTrillSpan }
  c2 r2
}
```



Triller, die auf einer bestimmten Note ausgeführt werden sollen, können mit dem Befehl `pitchedTrill` notiert werden. Das erste Argument ist die Hauptnote, das zweite die Note, auf der getrillert wird. Sie wird als Note ohne Hals in Klammern ausgegeben.
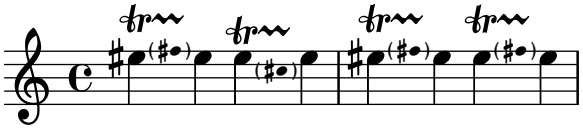
```
\relative {
  \pitchedTrill
  d''2\startTrillSpan fis
  d2
  c2\stopTrillSpan
  r2
}
```



Aufeinanderfolgende Versetzungszeichen der selben Note im selben Takt müssen selbst hinzugefügt werden. Nur das Versetzungszeichen des ersten Trillers mit Tonhöhe innerhalb eines Taktes wird ausgegeben.

```
\relative {
  \pitchedTrill
  eis''4\startTrillSpan fis
  eis4\stopTrillSpan
  \pitchedTrill
  eis4\startTrillSpan cis
  eis4\stopTrillSpan
  \pitchedTrill
  eis4\startTrillSpan fis
  eis4\stopTrillSpan
  \pitchedTrill
```

```
  eis4\startTrillSpan fis!
  eis4\stopTrillSpan
}
```



Vordefinierte Befehle

`\startTrillSpan`, `\stopTrillSpan`.

Siehe auch
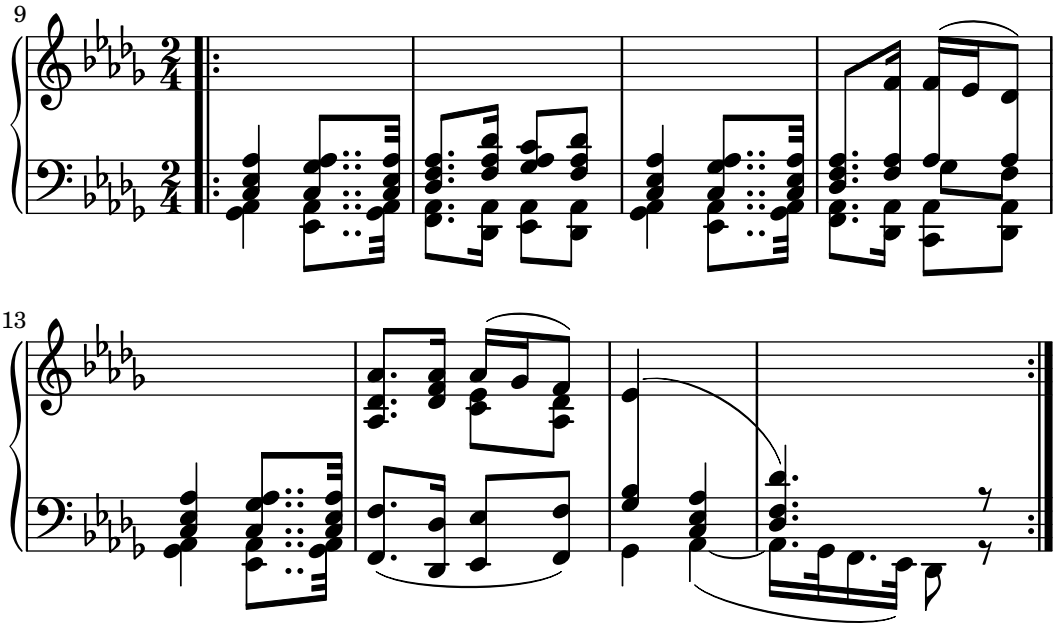
Glossar: Abschnitt “trill” in *Glossar*.

Notationsreferenz: Abschnitt 3.1.1 [Artikulationszeichen und Verzierungen], Seite 120, Abschnitt 2.6.1 [Verzierungen], Seite 112.

Schnipsel: Abschnitt “Expressive marks” in *Schnipsel*.

Referenz der Interna: Abschnitt “TrillSpanner” in *Referenz der Interna*.

4 Wiederholungszeichen



Wiederholung ist ein zentrales Konzept in der Musik, und es gibt eine ganze Vielzahl von Notationsmöglichkeiten für Wiederholungen. LilyPond unterstützt folgende Arten von Wiederholungen:

`volta (Wiederholungsklammer)`
: Die wiederholte Musik wird nicht geschrieben, sondern zwischen zwei Wiederholungstaktstrichen eingeschlossen. Wenn die Wiederholung am Anfang eines Stückes beginnt, wird nur am Ende der Wiederholung eine Wiederholungstaktlinie gesetzt. Alternative Schlüsse (Volta) werden von links nach rechts mit Klammern gesetzt. Das ist die Standardnotationspraxis für Wiederholungen mit alternativen Schlüssen.

`unfold (aufklappen)`
: Die wiederholte Musik wird ausgeschrieben, so oft, wie es durch *`Wiederholungszähler`* definiert wird. Das erspart Arbeit, wenn repetitive Musik notiert wird.

`percent (Prozent-Wiederholung)`
: Das sind Noten- oder Taktwiederholungen, sie sehen aus wie ein Schrägstrich bzw. wie ein Prozentzeichen.

`tremolo`
: Das wird benutzt, um Tremolo-Wiederholungen am Notenhals zu notieren.

4.1 Lange Wiederholungen

4.1.1 Normale Wiederholungen
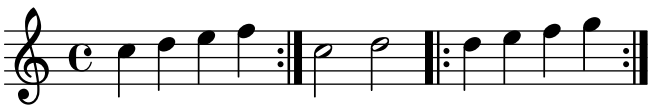
Die Syntax für normale Wiederholungen ist

```
\repeat Typ Wiederholungszähler musikAusdr
```

wobei *`musikAusdr`* ein musikalischer Ausdruck ist.

Wiederholung ohne alternativen Schluss:

```
\relative {
  \repeat volta 2 { c''4 d e f }
  c2 d
  \repeat volta 2 { d4 e f g }
}
```



Alternative Schlüsse können mit `\alternative` gesetzt werden. Damit die alternativen Schlüsse von den wiederholten Noten abgegrenzt werden, müssen sie in geschweiften Klammern zusammengefasst werden.
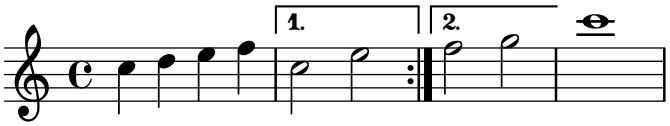
```
\repeat volta Wiederholungszähler musikAusdr
\alternative {
  { musikAusdr }
}
```

wobei *musikAusdr* ein musikalischer Ausdruck ist.

Wenn es mehr Wiederholungen gibt, als Alternativen angegeben sind, erhalten die ersten Wiederholungen den ersten Schluss.
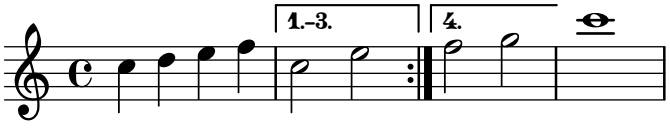
Eine einfache Wiederholung mit einer Alternative:

```
\relative {
  \repeat volta 2 { c''4 d e f | }
  \alternative {
    { c2 e | }
    { f2 g | }
  }
  c1
}
```



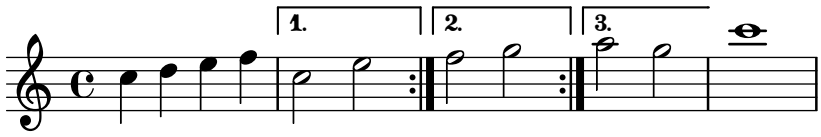
Eine einfache Wiederholung mit mehr als einer Alternative:

```
\relative {
  \repeat volta 4 { c''4 d e f | }
  \alternative {
    { c2 e | }
    { f2 g | }
  }
  c1
}
```



Mehrfache Wiederholungen mit mehr als einer Alternative:

```
\relative {
  \repeat volta 3 { c''4 d e f | }
  \alternative {
    { c2 e | }
    { f2 g | }
    { a2 g | }
  }
  c1
}
```
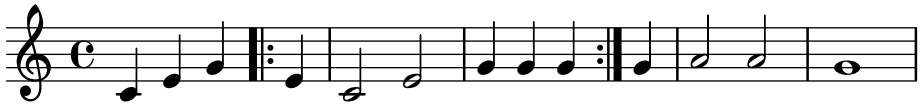


Achtung: Wenn es zwei oder mehr Alternativen gibt, darf nichts zwischen der schließenden Klammer der einen und der öffnenden Klammer der anderen Wiederholung stehen, weil sonst nicht die erwartete Anzahl von Endungen produziert wird.

Achtung: Wenn man `\relative` innerhalb von `\repeat` notiert, ohne den `Voice`-Kontext explizit zu beginnen, erscheinen zusätzliche (ungewollte) Systeme. Siehe auch Abschnitt "Ein zusätzliches System erscheint" in *Anwendungsbenutzung*.
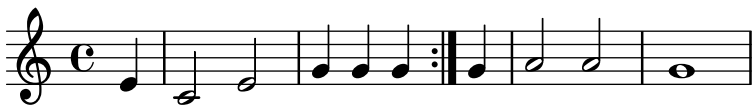
Wenn eine Wiederholung mitten in einem Takt beginnt und keine Alternativen hat, fällt normalerweise auch das Ende der Wiederholung mitten in einen Takt, sodass beide unvollständigen Takt einen vollständigen Takt ergeben. In diesem Fall bezeichnen die Wiederholungsstriche keine richtigen Taktstriche. Benutzen Sie nicht `\partial`-Befehle oder Taktüberprüfung, wo die Wiederholungslinien gesetzt werden:

```
\relative { % no \partial here
  c'4 e g  % no bar check here
  % no \partial here
  \repeat volta 4 {
    e4 |
    c2 e |
    % no \partial here
    g4 g g  % no bar check here
  }
  % no \partial here
  g4 |
  a2 a |
  g1 |
}
```



Ähnlich ist es, wenn eine Wiederholung mit einem Auftakt beginnt und keine Alternativen hat. In diesem Fall muss man aber den `\partial`-Befehl zu Beginn der Partitur setzen:
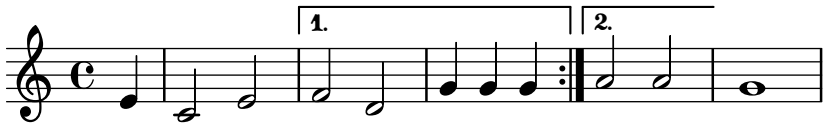
```
\relative {
  \partial 4  % required
  \repeat volta 4 {
    e'4 |
    c2 e |
    % no \partial here
    g4 g g  % no bar check here
  }
  % no \partial here
  g4 |
  a2 a |
  g1 |
}
```



Wenn alternative Endungen zu einer Wiederholung hinzugefügt werden, die mit einem Auftakt beginnt, muss die `Timing.measureLength`-Eigenschaft manuell gesetzt werden, und an folgenden Stellen:

- am Beginn eines unvollständigen Taktes innerhalb der `\alternative`-Umgebung, die normalerweise am Ende jeder Alternative auftreten, außer (in den meisten Fällen) in der letzten.
- zu Beginn jeder Alternative außer der ersten.

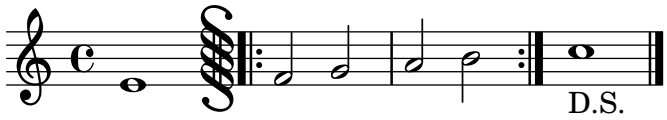
```
\partial 4
\repeat volta 2 { e4 | c2 e | }
\alternative {
  {
    f2 d |
    \set Timing.measureLength = #3/4
    g4 g g  % optional bar check is allowed here
  }
  {
    \set Timing.measureLength = #4/4
    a2 a |
  }
}
g1 |
```



Die `measureLength`-Eigenschaft ist beschrieben in Abschnitt 2.6.3 [Verwaltung der Zeiteinheiten], Seite 118.

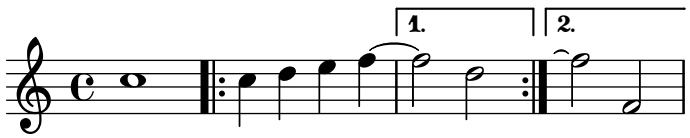
Der `\inStaffSegno`-Befehl kann angewandt werden, um das Segno-Zeichen in die Notenzeile einzubinden, auch in Kooperation mit dem `\repeat volta`-Befehl. Die benutzten Taktstrichsymbole können durch Überschreiben der Eigenschaften `segnoBarType`, `startRepeatSegnoBarType`, `endRepeatSegnoBarType` bzw. `doubleRepeatSegnoBarType` geändert werden.

```
e1
\repeat volta 2 {
  \inStaffSegno
  f2 g a b
}
c1_"D.S." \bar "|."
```



Bindebögen können auch an eine zweite Klammer angefügt werden:

```
\relative {
  c''1
  \repeat volta 2 { c4 d e f~ }
  \alternative {
    { f2 d }
    { f2\repeatTie f, }
  }
}
```
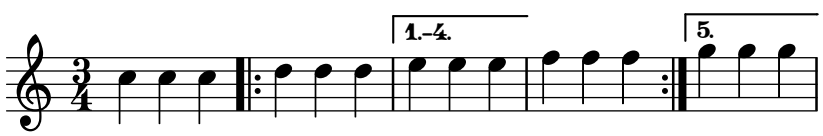


Ausgewählte Schnipsel

Volta-Klammern verkürzen

Volta-Klammern werden normalerweise über alle Noten der Klammer gezogen, aber es ist möglich sie zu verkürzen. Hierzu muss `voltaSpannerDuration` definiert werden, in dem Beispiel etwa als 3/4, sodass die Klammer nur einen Takt dauert.

```
\fixed c'' {
  \time 3/4
  c4 c c
  \repeat volta 5 {
    d4 d d
    \alternative {
      \volta 1,2,3,4 {
        \once \override Score.VoltaBracket.musical-length =
          \musicLength 2.
        e4 e e
        f4 f f
      }
      \volta 5 {
        g4 g g } } }
}
```



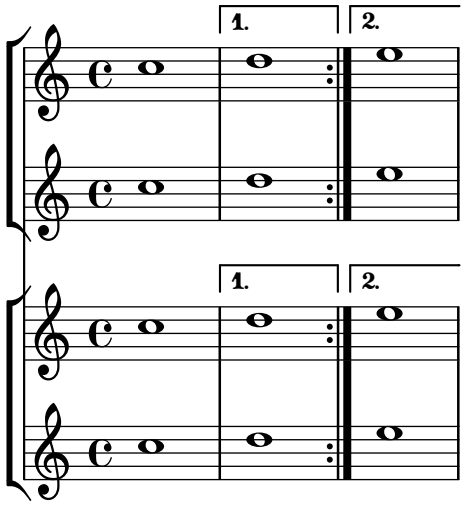
Volta brackets in multiple staves

By adding the `Volta_engraver` to the relevant staff, volte can be put over staves other than the topmost one in a score.

`\repeat` and related commands should be present in all staves.

```
voltaMusic = \relative c'' {
  \repeat volta 2 {
    c1
    \alternative {
      \volta 1 { d1 }
      \volta 2 { e1 }
    }
  }
}

<<
  \new StaffGroup <<
    \new Staff \voltaMusic
    \new Staff \voltaMusic
  >>
  \new StaffGroup <<
    \new Staff \with { \consists "Volta_engraver" }
      \voltaMusic
    \new Staff \voltaMusic
  >>
>>
```

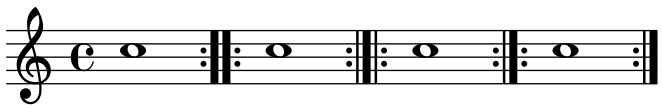


Setting the double repeat default for volte

There are different double repeat styles for volte that can be selected using the context property `doubleRepeatBarType`.

```
\relative c'' {
  \repeat volta 2 { c1 }
  \set Score.doubleRepeatBarType = ":..:"
  \repeat volta 2 { c1 }
  \set Score.doubleRepeatBarType = ":|.|:"
  \repeat volta 2 { c1 }
  \set Score.doubleRepeatBarType = ":|.:"
```

```
  \repeat volta 2 { c1 }
}
```



Alternative Taktnummerierung

Zwei alternative Methoden können eingestellt werden, die die Taktnummerierung beeinflussen, insbesondere bei Wiederholungen.

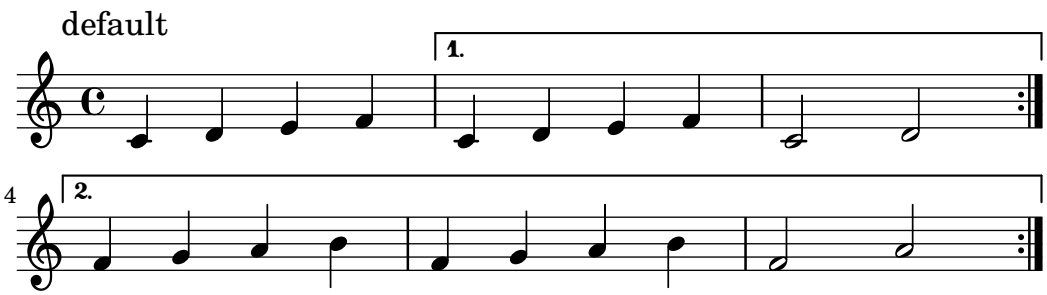
```
music = \relative c' {
  \repeat volta 3 {
    c4 d e f |
    \alternative {
      \volta 1 { c4 d e f | c2 d \break }
      \volta 2 { f4 g a b | f4 g a b | f2 a | \break }
      \volta 3 { c4 d e f | c2 d } } }
  c1 \bar "|."
}
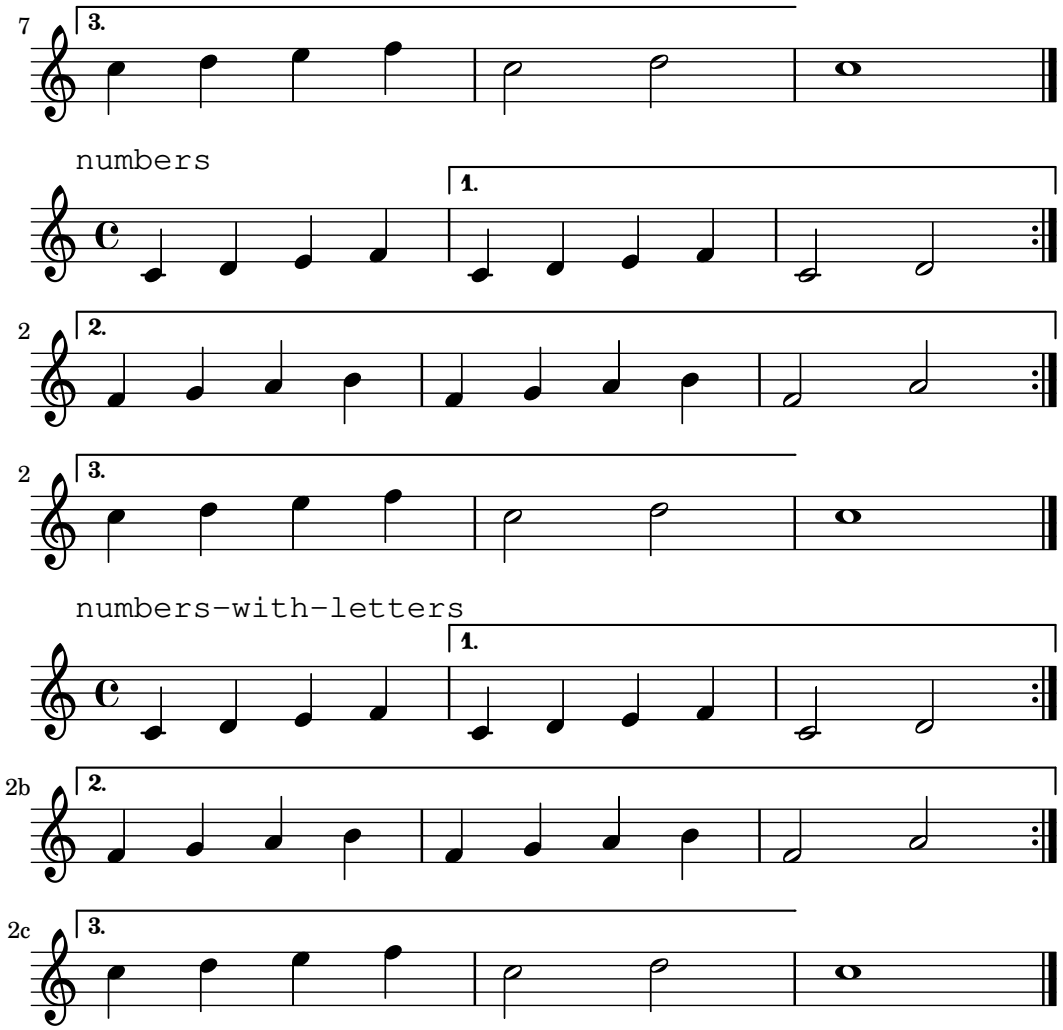
{
  \textMark \markup \large "default"
  \music
}

{
  \textMark \markup \large \typewriter "numbers"
  \set Score.alternativeNumberingStyle = #'numbers
  \music
}

{
  \textMark \markup \large \typewriter "numbers-with-letters"
  \set Score.alternativeNumberingStyle = #'numbers-with-letters
  \music
}

\layout {
  \context {
    \Score
    \override TextMark.Y-offset = #5
  }
}
```





Siehe auch

Glossar: Abschnitt “repeat” in *Glossar*, Abschnitt “volta” in *Glossar*.

Notationsreferenz: Abschnitt 2.5.1 [Taktstriche], Seite 97, Abschnitt 32.4 [Umgebungs-Plugins verändern], Seite 586, Abschnitt 2.6.3 [Verwaltung der Zeiteinheiten], Seite 118.

Schnipsel: Abschnitt “Repeats” in *Schnipsel*.

Referenz der Interna: Abschnitt “VoltaBracket” in *Referenz der Interna*, Abschnitt “VoltaRepeatedMusic” in *Referenz der Interna*, Abschnitt “UnfoldedRepeatedMusic” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Bindebögen, die von einer `\repeat`-Umgebung in eine `\alternative`-Umgebung ragen, funktionieren nur in der ersten Klammer. Bindebögen können auch nicht von der Ende einer Wiederholungsklammer auf den Anfang der Wiederholung verweisen.

Wenn eine Wiederholung innerhalt eines unvollständigen Taktes beginnt und eine `\alternative`-Umgebung mit einer Veränderung von `measureLength` enghält, führt die Verwendung von `\unfoldRepeats` zu falsch gesetzten Taktstrichen und Taktüberprüfungswarnungen.

Eine ineinandergeschachtelte Wiederholung wie

```
\repeat ...
\repeat ...
```

`\alternative`

ist mehrdeutig, weil nicht klar ist, zu welchem `\repeat`-Abschnitt die `\alternative`-Endung gehört. Diese Mehrdeutigkeit wird von LilyPond aufgelöst, indem die alternative Endung immer zu der innersten Wiederholung gehört. Um Klarheit zu schaffen, bietet es sich an, in solchen Situationen Klammern zu benutzen.

4.1.2 Manuelle Wiederholungszeichen
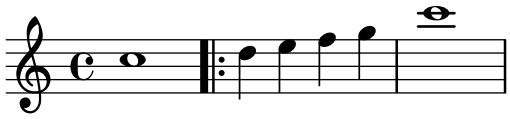
> **Achtung:** Diese Methoden werden nur verwendet, um ungewöhnliche Wiederholungskonstruktionen darzustellen und können sich unerwünscht verhalten. In den meisten Fällen sollten Wiederholungen mit dem Befehl `\repeat` erstellt werden oder indem die entsprechenden Taktstriche eingegeben werden. Mehr Information in Abschnitt 2.5.1 [Taktstriche], Seite 97.

Die Eigenschaft `repeatCommands` kann verwendet werden, um das Aussehen der Wiederholungen zu beeinflussen. Ihr Argument ist eine Scheme-Liste an Wiederholungsbefehlen.

`start-repeat`

Setzt eine `|:` Taktlinie.

```
\relative {
  c''1
  \set Score.repeatCommands = #'(start-repeat)
  d4 e f g
  c1
}
```
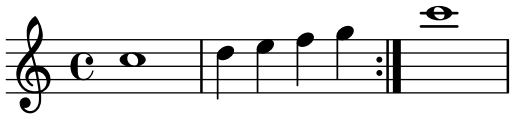


Der Notensatzpraxis folgend werden Wiederholungstaktstriche nicht zu Beginn eines Stückes gesetzt.

`end-repeat`

Setzt eine `:|` Taktlinie.
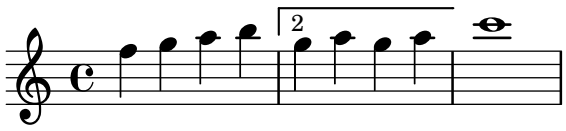
```
\relative {
  c''1
  d4 e f g
  \set Score.repeatCommands = #'(end-repeat)
  c1
}
```



`(volta` *Zahl*`) ... (volta #f)`

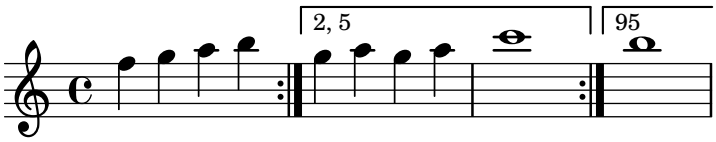
Setzt eine Volta-Klammer mit der Beschriftung *Nummer*. Die Volta-Klammer muss explizit beendet werden, sonst wird sie nicht ausgegeben.

```
\relative {
  f''4 g a b
  \set Score.repeatCommands = #'((volta "2"))
  g4 a g a
  \set Score.repeatCommands = #'((volta #f))
  c1
}
```
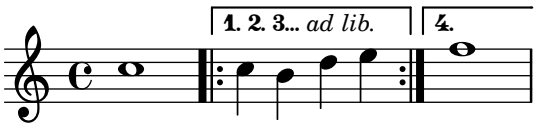


Mehrfache Wiederholungszeichen können an der selben Stelle vorkommen:

```
\relative {
  f''4 g a b
  \set Score.repeatCommands = #'((volta "2, 5") end-repeat)
  g4 a g a
  c1
  \set Score.repeatCommands = #'((volta #f) (volta "95") end-repeat)
  b1
  \set Score.repeatCommands = #'((volta #f))
}
```



Text kann auch in der Volta-Klammer gesetzt werden. Der Text kann aus Zahlen oder einer Zahl oder einer Textbeschriftung bestehen, siehe Abschnitt 8.2 [Text formatieren], Seite 233. Die einfachste Art Text zu benutzen ist, die Beschriftung zuerst zu definieren und dann die Beschriftung in einer Scheme-Liste einzufügen.

```
voltaAdLib = \markup { \volta-number { 1. 2. 3... } \italic { ad lib. } }
\relative {
  c''1
  \set Score.repeatCommands = #`((volta ,voltaAdLib) start-repeat)
  c4 b d e
  \set Score.repeatCommands = #`((volta #f)
                                 (volta ,#{ \markup \volta-number "4." #})
                                 end-repeat)
  f1
  \set Score.repeatCommands = #'((volta #f))
}
```



Siehe auch

Notationsreferenz: Abschnitt 2.5.1 [Taktstriche], Seite 97, Abschnitt 8.2 [Text formatieren], Seite 233.

Schnipsel: Abschnitt “Repeats” in *Schnipsel*.

Referenz der Interna: Abschnitt “VoltaBracket” in *Referenz der Interna*, Abschnitt “VoltaRepeatedMusic” in *Referenz der Interna*.
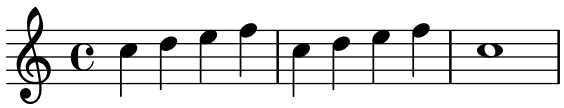
4.1.3 Ausgeschriebene Wiederholungen

Mit dem `unfold`-Befehl können Wiederholungen eingesetzt werden, um repetitive Musik zu notieren. Die Syntax ist

```
\repeat unfold Wiederholungszähler musikAusdr
```

wobei *musikAusdr* ein musikalischer Ausdruck ist und *Wiederholungszähler* die Anzahl bezeichnet, mit der *musikAusdr* wiederholt wird.

```
\relative {
  \repeat unfold 2 { c''4 d e f }
  c1
}
```



In einigen Fällen, insbesondere in einer `\relative`-Umgebung, bedeutet die Funktion `\repeat unfold` nicht das gleiche wie die ausgeschriebenen Noten mehrere Male. Beispielsweise ist
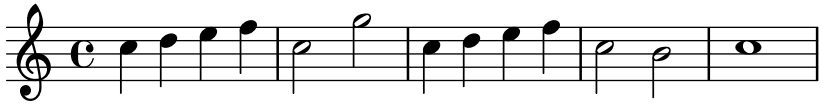
```
\repeat unfold 2 { a'4 b c }
```

nicht das Selbe wie

```
a'4 b c | a'4 b c
```
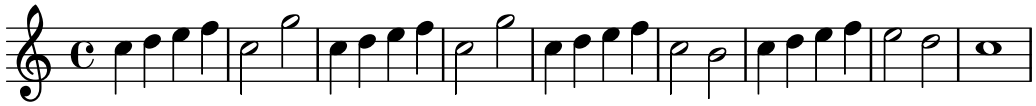
Repititive Wiederholungen können auch mit mehreren Alternativeklammern notiert werden:

```
\relative {
  \repeat unfold 2 { c''4 d e f }
  \alternative {
    { c2 g' }
    { c,2 b }
  }
  c1
}
```
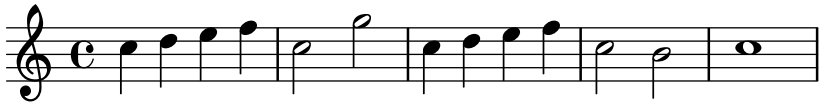


Wenn es mehr Wiederholungen als Alternativen gibt, wird die erste Alternative so oft eingesetzt, bis sich zusammen mit den restlichen Alternativen die Gesamtanzahl der Wiederholungen ergeben.

```
\relative {
  \repeat unfold 4 { c''4 d e f }
  \alternative {
    { c2 g' }
    { c,2 b }
    { e2 d }
   }
  c1
}
```
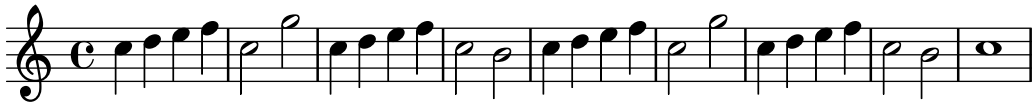


Wenn es mehr Alternativen als Wiederholungen gibt, wird nur die ersten Alternativen ausgegeben und die restlichen Alternativen ignoriert und nicht gesetzt.

```
\relative {
  \repeat unfold 2 { c''4 d e f }
  \alternative {
    { c2 g' }
    { c,2 b }
    { e2 d }
  }
  c1
}
```



Es ist auch möglich, mehrere `unfold`-Wiederholungen (mit oder ohne Alternativen) ineinander zu verschachteln:

```
\relative {
  \repeat unfold 2 {
    \repeat unfold 2 { c''4 d e f }
    \alternative {
      { c2 g' }
      { c,2 b }
    }
  }
  c1
}
```



Akkordkonstruktionen können mit dem Akkordwiederholungssymbol `q` wiederholt werden. Siehe Abschnitt 5.1.2 [Akkord-Wiederholungen], Seite 162.

> **Achtung:** Wenn man `\relative` innerhalb von `\repeat` notiert, ohne den `Voice`-Kontext explizit zu beginnen, erscheinen zusätzliche (ungewollte) Systeme. Siehe auch Abschnitt “Ein zusätzliches System erscheint” in *Anwendungsbenutzung*.

Siehe auch

Notationsreferenz: Abschnitt 5.1.2 [Akkord-Wiederholungen], Seite 162.

Handbuch zur Benutzung: Abschnitt “Ein zusätzliches System erscheint” in *Anwendungsbenutzung*.

Schnipsel: Abschnitt “Repeats” in *Schnipsel*.

Referenz der Interna: Abschnitt “UnfoldedRepeatedMusic” in *Referenz der Interna*.

4.2 Kurze Wiederholungen

Dieser Abschnitt zeigt, wie man kurze Wiederholungen notiert. Kurze Wiederholungen haben zwei Formen: Wiederholungen von einer Note bis zu zwei Takten, die mit Schrägstrichen oder Prozentzeichen dargestellt werden, und Tremolos.

4.2.1 Prozent-Wiederholungen

Kurze wiederholte Muster werden einmal gesetzt und das wiederholte Muster wird durch ein besonderes Zeichen ersetzt.
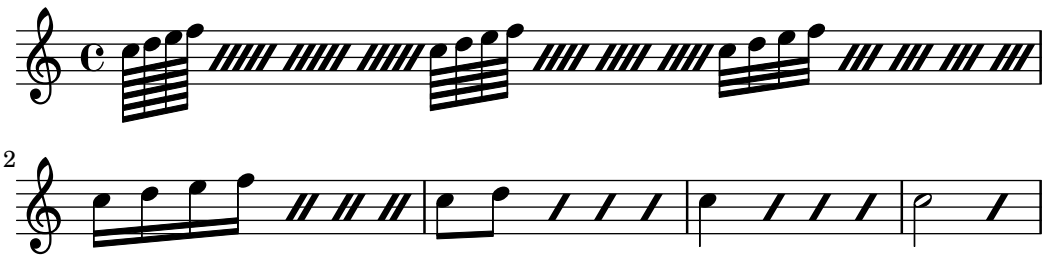
Die Syntax lautet:

```
\repeat percent Wiederholungszahl musikAusdr
```

wobei *musikAusdr* ein musikalischer Ausdruck ist.
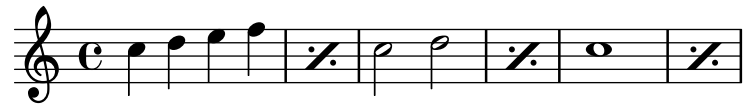
Muster, die kürzer als ein Takt sind, werden mit Schrägstrichen ersetzt:

```
\relative c'' {
  \repeat percent 4 { c128 d e f }
  \repeat percent 4 { c64 d e f }
  \repeat percent 5 { c32 d e f }
  \repeat percent 4 { c16 d e f }
  \repeat percent 4 { c8 d }
  \repeat percent 4 { c4 }
  \repeat percent 2 { c2 }
}
```
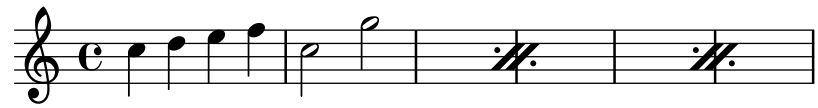


Muster von einem oder zwei Takten Dauer werden mit prozentartigen Symbolen ersetzt:

```
\relative c'' {
  \repeat percent 2 { c4 d e f }
  \repeat percent 2 { c2 d }
  \repeat percent 2 { c1 }
}
```

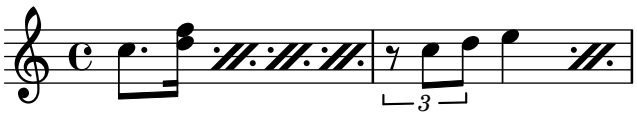


```
\relative {
  \repeat percent 3 { c''4 d e f | c2 g' }
}
```



Muster, die kürzer als ein Takt sind, aber unterschiedliche Dauern beinhalten, benützten ein doppeltes Prozent-Symbol.

```
\relative {
  \repeat percent 4 { c''8. <d f>16 }
  \repeat percent 2 { \tuplet 3/2 { r8 c d } e4 }
}
```

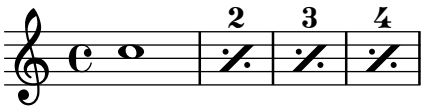


Ausgewählte Schnipsel

Prozent-Wiederholungen zählen

Ganztaktwiederholungen mit mehr als zwei Wiederholungen erhalten einen Zähler, wenn man die entsprechende Eigenschaft einsetzt:

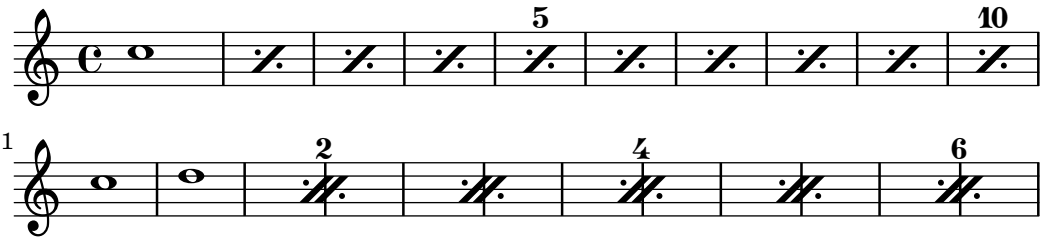
```
\relative c'' {
  \set countPercentRepeats = ##t
  \%4 c1
}
```



Sichtbarkeit von Prozent-Wiederholungen

Prozentwiederholungszähler können in regelmäßigen Intervallen angezeigt werden, indem man die Eigenschaft `repeatCountVisibility` beeinflusst.

```
\relative c'' {
  \set countPercentRepeats = ##t
  \set repeatCountVisibility = #(every-nth-repeat-count-visible 5)
  \%10 c1 \break
  \set repeatCountVisibility = #(every-nth-repeat-count-visible 2)
  \%6 { c1 d1 }
}
```



Isolierte Prozentwiederholungen

Isolierte Prozentwiederholungen können auch ausgegeben werden. Das wird erreicht, indem man eine Ganztaktpause notiert und ihre Ausgabeform ändert:

```
makePercent =
#(define-music-function (note) (ly:music?)
   "Make a percent repeat the same length as NOTE."
   (make-music 'PercentEvent
               'length (ly:music-length note)))

\relative c'' {
```

```
  \makePercent s1
}
```



Siehe auch

Glossar: Abschnitt “percent repeat” in *Glossar*, Abschnitt “simile” in *Glossar*.

Schnipsel: Abschnitt “Repeats” in *Schnipsel*.
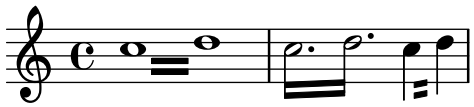
Referenz der Interna: Abschnitt “RepeatSlash” in *Referenz der Interna*, Abschnitt “RepeatSlashEvent” in *Referenz der Interna*, Abschnitt “DoubleRepeatSlash” in *Referenz der Interna*, Abschnitt “PercentRepeat” in *Referenz der Interna*, Abschnitt “PercentRepeatCounter” in *Referenz der Interna*, Abschnitt “PercentRepeatedMusic” in *Referenz der Interna*, Abschnitt “Percent_repeat_engraver” in *Referenz der Interna*, Abschnitt “DoublePercentEvent” in *Referenz der Interna*, Abschnitt “DoublePercentRepeat” in *Referenz der Interna*, Abschnitt “DoublePercentRepeatCounter” in *Referenz der Interna*, Abschnitt “Double_percent_repeat_engraver” in *Referenz der Interna*, Abschnitt “Slash_repeat_engraver” in *Referenz der Interna*.

4.2.2 Tremolo-Wiederholung

Tremolos können in zwei Arten notiert werden: als Wechsel zwischen zwei Noten oder Akkorden oder als schnelle Wiederholung einer einzigen Note. Tremolos, die als Wechsel realisiert werden, werden dargestellt, indem Balken zwischen die Noten gesetzt werden, Tremolos, die eine schnelle Wiederholung darstellen, haben Balken oder Schrägstriche am Hals einer einzigen Note.

Um Tremolobalken zwischen Noten zu setzen, kann der `\repeat`-Befehl mit dem Tremolo-Stil benutzt werden:

```
\relative c'' {
  \repeat tremolo 8 { c16 d }
  \repeat tremolo 6 { c16 d }
  \repeat tremolo 2 { c16 d }
}
```



Die `\repeat tremolo`-Syntax braucht genau zwei Noten innerhalb der geschweiften Klammern, und die Anzahl der Wiederholungen muss einem Wert entsprechen, der mit einfachen oder punktierten Noten ausgedrückt werden kann. `\repeat tremolo 7` funktioniert und setzt Tremolo für die Dauer einer Doppeltpunktierten, aber `\repeat tremolo 9` funktioniert nicht.

Die Dauer des Tremolos entspricht der Dauer der Wertes in Klammern, multipliziert mit der Zahl der Wiederholungen: `\repeat tremolo 8 { c16 d16 }` ergibt ein Tremolo für eine Ganze, notiert als zwei Ganze, die zwei Tremolobalken zwischen sich haben.
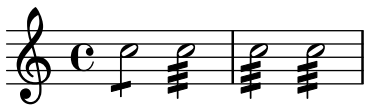
Es gibt zwei Möglichkeiten, ein Tremolozeichen zu einer einzelnen Noten hinzuzufügen. Die `\repeat tremolo`-Syntax kann hier auch benutzt werden; in diesem Fall wird die Note allerdings nicht eingeklammert:

```
\repeat tremolo 4 c'16
```



Die gleiche Darstellung wird erreicht, indem nach der Note `:Zahl` geschrieben wird. Die Zahl zeigt die Dauer der Unterteilung an, und sie muss mindestens den Wert 8 haben. Ein Wert von 8 ergibt einen Balken durch den Notenhals. Wenn die Zahl ausgelassen wird, wird der letzte benutzte Wert eingesetzt (gespeichert in `tremoloFlags`):

```
\relative {
  c''2:8 c:32
  c: c:
}
```
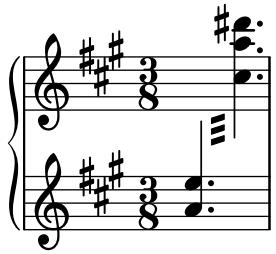


Ausgewählte Schnipsel

Cross-staff tremolos

Since `\repeat tremolo` expects exactly two musical arguments for chord tremolos, the note or chord which changes staff within a cross-staff tremolo should be placed inside curly braces together with its `\change Staff` command.

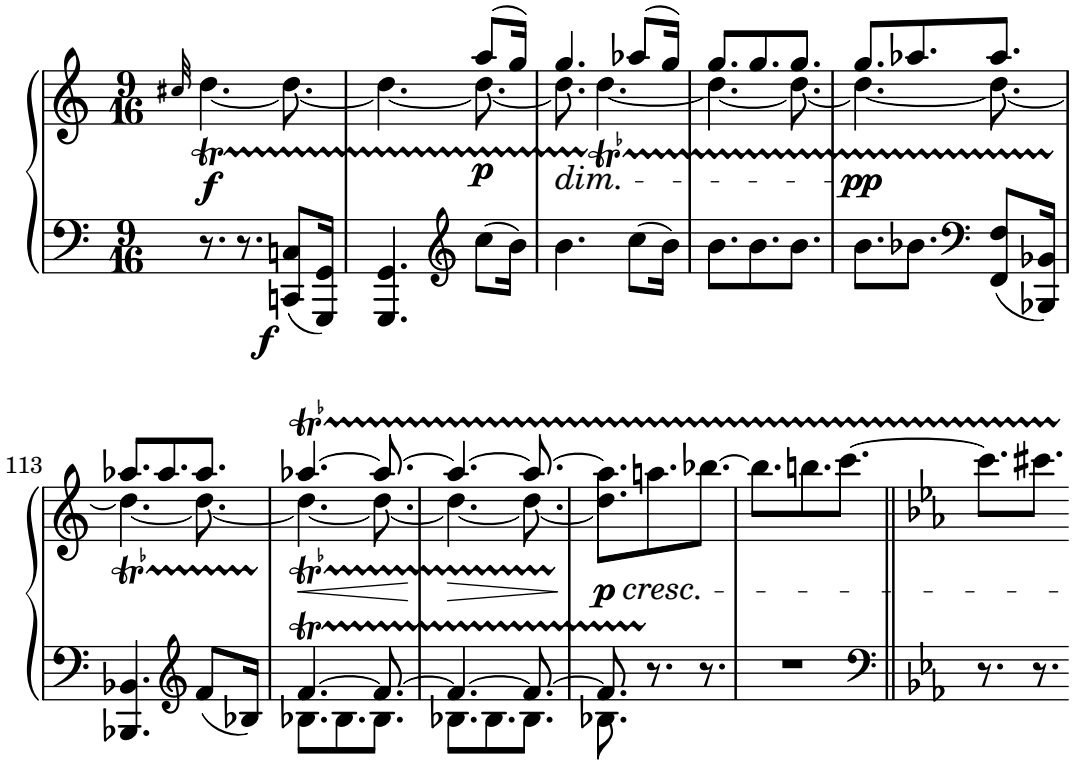
```
\new PianoStaff <<
  \new Staff = "up" \relative c'' {
    \key a \major
    \time 3/8
    s4.
  }
  \new Staff = "down" \relative c'' {
    \key a \major
    \time 3/8
    \voiceOne
    \repeat tremolo 6 {
      <a e'>32
      {
        \change Staff = "up"
        \voiceTwo
        <cis a' dis>32
      }
    }
  }
>>
```



Siehe auch

Schnipsel: Abschnitt “Repeats” in *Schnipsel*.

5 Gleichzeitig erscheinende Noten



Polyphonie bedeutet in der musikalischen Terminologie das Vorhandensein von mehr als einer (eigenständigen) Stimme in einem Stück. Für LilyPond bedeutet es aber das Vorhandensein von mehr als einer Stimme pro System.

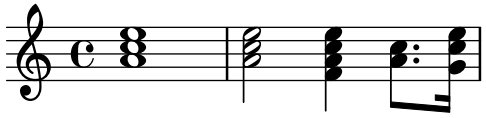
5.1 Eine einzelne Stimme

Dieser Abschnitt behandelt gleichzeitige Noten innerhalb derselben Stimme.

5.1.1 Noten mit Akkorden

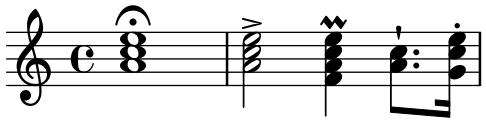
Ein Akkord wird notiert, indem die zu ihm gehörenden Tonhöhen zwischen spitze Klammern (< und >) gesetzt werden. Auf einen Akkord kann eine Dauer-Angabe folgen, genauso wie bei einfachen Noten.

```
\relative {
  <a' c e>1 <a c e>2 <f a c e>4 <a c>8. <g c e>16
}
```



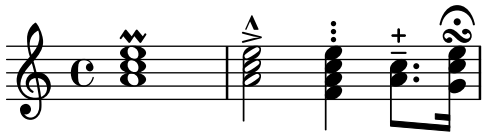
Akkorde können auch von Artikulationen gefolgt werden, genau wie auch einfache Noten.

```
\relative {
  <a' c e>1\fermata <a c e>2-> <f a c e>4\prall <a c>8.^! <g c e>16-.
}
```



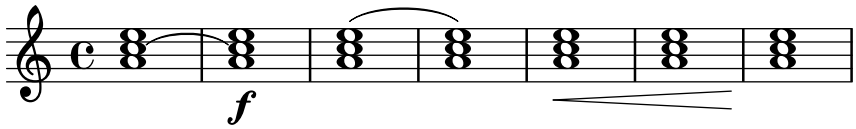
Die Noten innerhalb der Akkorde konnen auch von Artikulationen oder Ornamenten gefolgt werden.

```
\relative {
  <a' c\prall e>1 <a-> c-^ e>2 <f-. a c-. e-.>4
  <a-+ c-->8. <g\fermata c e\turn>16
}
```
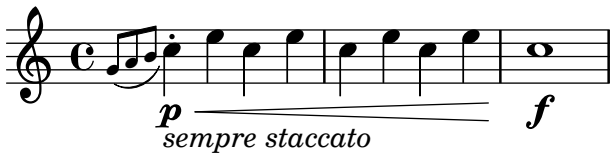


Manche Notationselemente, wie etwa Dynamik, Crescendo-Klammern und Legatobögen müssen an den gesamten Akkord gehängt werden und nicht an einzelne Noten, damit sie ausgegeben werden.

```
\relative {
  <a'\f c( e>1 <a c) e>\f <a\< c e>( <a\! c e>)
  <a c e>\< <a c e> <a c e>\!
}
```



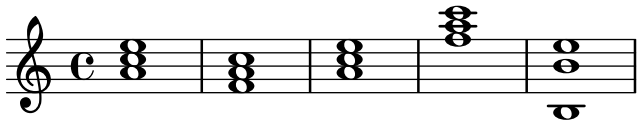
Ein Akkord ist sozusagen ein Container für die Noten, ihre Artikulationen und andere angehänge Elemente. Demzufolge hat also auch ein Akkord ohne wirkliche Noten innerhalb der Klammern keine Dauer. Alle angehängten Artikulationen geschehen zur selben musikalischen Zeit wie die folgende Note oder der folgende Akkord und werden damit kombiniert (für komplexere Möglichkeiten, derartige Elemente zu kombinieren, siehe Abschnitt 5.1.3 [Gleichzeitige Ausdrücke], Seite 164.

```
\relative {
  \grace { g'8( a b }
  <> ) \p \< -. -\markup \italic "sempre staccato"
  \repeat unfold 4 { c4 e }  c1\f
}
```



Der relative Modus kann auch für Tonhöhen in Akkorden eingesetzt werden. Die erste Note eines Akkordes ist immer relativ zur ersten Note des vorherigen Akkordes, oder mit der Tonhöhe der letzten Note vor dem Akkord (wenn kein Akkord vorhergeht). Alle anderen Noten innerhalb des Akkordes sind relativ zu der Note vorher innerhalb des selben Akkordes.

```
\relative {
  <a' c e>1 <f a c> <a c e> <f' a c> <b, e b,>
}
```



Mehr Information über Akkorden findet sich in Kapitel 15 [Notation von Akkorden], Seite 397.

Siehe auch

Musikglossar: Abschnitt “chord” in *Glossar*.

Handbuch zum Lernen: Abschnitt “Noten zu Akkorden verbinden” in *Handbuch zum Lernen*.

Notationsreferenz: Kapitel 15 [Notation von Akkorden], Seite 397, Abschnitt 3.1.1 [Artikulationszeichen und Verzierungen], Seite 120, Abschnitt 1.1.2 [Relative Oktavenbezeichnung], Seite 4, Abschnitt 5.2 [Mehrere Stimmen], Seite 166.

Schnipsel: Abschnitt “Simultaneous notes” in *Schnipsel*.

Bekannte Probleme und Warnungen

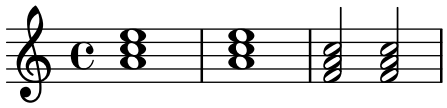
Akkorde, die mehr als zwei Tonhöhen für einen Notenlinienzwischenraum enthalten (wie etwa ‘`<e f! fis!>`’) produzieren überlappende Notenköpfe. Abhängig von der Situation kann eines der folgenden Dinge helfen, die Darstellung zu verbessern:

- Kurzzeitig mehrere Stimmen benutzen, siehe Abschnitt 5.2 [Mehrere Stimmen], Seite 166: ‘`<y f! \\ <e fis!> >>`’,
- enharmonische Transkription für einen oder mehrere Tonhöhen vornehmen: ‘`<e f ges>`’ oder
- Cluster, siehe Abschnitt 5.1.4 [Cluster], Seite 165.

5.1.2 Akkord-Wiederholungen

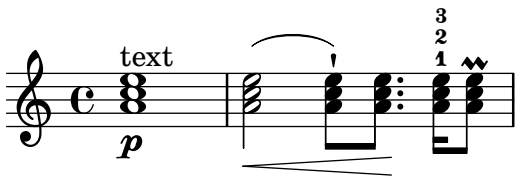
Um Schreibarbeit zu ersparen, kann ein Zeichen benutzt werden, um den vorhergehenden Akkord zu wiederholen. Das Symbol hierzu ist `q`:

```
\relative {
  <a' c e>1 q <f a c>2 q
}
```



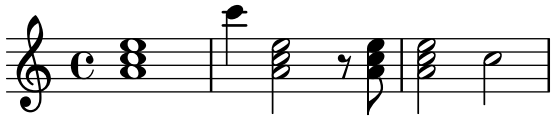
Genauso wie normale Akkorde kann auch das Akkord-Wiederholungssymbol in Verbindung mit Tondauern, Artikulationen, Beschriftungen, Legatobögen, Balken usw. benutzt werden, weil nur die Tonhöhen des vorangehenden Akkordes wiederholgt werden.

```
\relative {
  <a' c e>1\p^"text" q2\<( q8)[-! q8.]\! q16-1-2-3 q8\prall
}
```
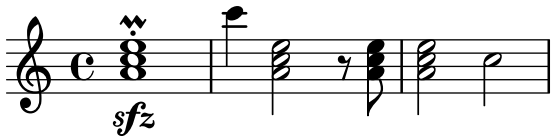


Das Akkordwiederholungssymbol erinnert sich an das letzte Vorkommen eines Akkordes, sodass man den letzten Akkord wiederholen kann, auch wenn in der Zwischenzeit nicht-Akkord-Noten oder -Pause auftetreten sind.

```
\relative {
  <a' c e>1 c'4 q2 r8 q8 |
  q2 c, |
}
```



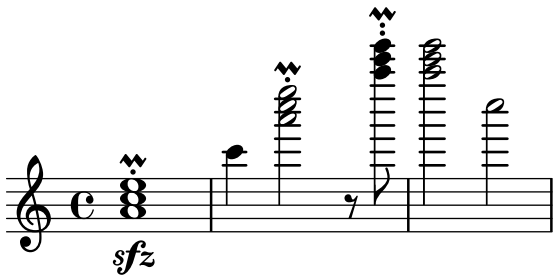
Das Akkord-Wiederholungssymbol behält keine Dynamikzeichen, Artikulationen oder Ornamente, die in oder an den vorhergehenden Akkord gehängt waren.

```
<a-. c\prall e>1\sfz c'4 q2 r8 q8 |
q2 c, |
```



Damit auch diese Zeichen erhalten bleiben, kann die Funktion `\chordRepeats` explizit mit einem zusätzlichen Argument aufgerufen werden, um eine Liste an Ereignistypen (engl. event) zu spezifizieren, die mit wiederholt werden, es sei denn, der gleiche Ereignistyp wird selber mit dem `q` verwendet:
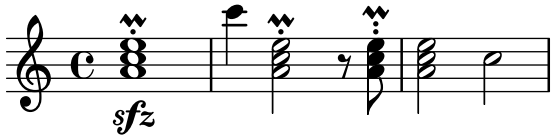
```
\relative {
  \chordRepeats #'(articulation-event)
  { <a'-. c\prall e>1\sfz c'4 q2 r8 q8-. } |
  q2 c, |
}
```



Die Benutzung von `\chordRepeats` innerhalb einer `\relative`-Konstruktion führt zu unerwarteten Ergebnissen: Nachdem die Akkordereignisse einmal erweitert sind, können sie nicht mehr von normal eingegebenen Akkorden unterschieden werden. Dadurch fügt `\relative` einen Oktavsprung entsprechend des aktuellen Kontexts ein.

Weil geschachtelte `\relative`-Umgebungen sich nicht gegenseitig beeinflussen, kann man eine zusätzliche `\relative`-Umgebung innerhalb von `\chordRepeats` benutzen, um die Oktavbeziehungen darzustellen, bevor die wiederholten Akkorde eingesetzt werden. In diesem Fall beeinflusst der gesamte Inhalt der inneren `\relative`-Umgebung nicht die äußere. Daraus ergibt sich die unterschiedliche Oktave der letzten Note in diesem Beispiel:

```
\new Voice
\relative c'' {
  \chordRepeats #'(articulation-event)
  \relative
  { <a'-. c\prall e>1\sfz c'4 q2 r8 q8-. } |
  q2 c |
}
```



Derartige Probleme mit `\relative` treten nur auf, wenn `\chordRepeats` explizit aufgerufen wird: die Verarbeitung von einfachem `q` wird erst vorgenommen, wenn alle `\relative`-Umgebungen schon verarbeitet sind.

Siehe auch

Notationsreferenz: Kapitel 15 [Notation von Akkorden], Seite 397, Abschnitt 3.1.1 [Artikulationszeichen und Verzierungen], Seite 120.
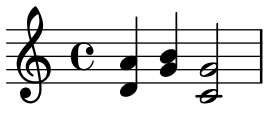
Installierte Dateien: `ly/chord-repetition-init.ly`.

5.1.3 Gleichzeitige Ausdrücke
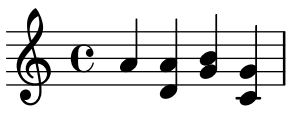
Eine oder mehrere musikalische Ausdrücke, die in doppelte spitze Klammern eingeschlossen werden, werden gleichzeitig gesetzt. Wenn der erste Ausdruck mit einer einzelnen Note beginnt oder die gesamte Konstruktion explizit in einer einzelnen Stimme erstellt wird, wird auch nur ein Notensystem erstellt. In anderem Falle werden die Elemente der simultanen Konstruktion auf unterschiedlichen Systemen gesetzt.

Das nächste Beispiel zeigt simultane Konstruktionen auf einem System:

```
\new Voice {  % explicit single voice
  << \relative { a'4 b g2 }
     \relative { d'4 g c,2 } >>
}
```
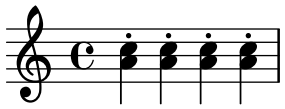


```
\relative {
  % single first note
  a' << \relative { a'4 b g }
        \relative { d'4 g c, } >>
}
```



Dass kann benutzt werden, wenn die simultanen Abschnitte einen identischen Rhythmus haben, aber wenn versucht wird, Noten mit unterschiedlicher Dauer an denselben Hals zu setzen, gibt es Fehlermeldungen. Noten, Artikulationen und Eigenschaftsänderungen in einer *einzelnen* ‘`Voice`’ werden gesammelt und in musikalischer Reihenfolge gesetzt:
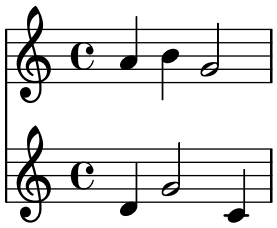
```
\relative {
  <a' c>4-.  <>-. << c a >>  << { c-. <c a> } { a s-. } >>
}
```



Mehrfache Hälse oder Balken oder unterschiedliche Notendauern oder Eigenschaften zur selben musikalischen Zeit erfordern den Einsatz von mehreren Stimmen.

Das nächste Beispiel zeigt, wie ein simultaner Ausdruck implizit mehrere Systeme erstellt:

```
% no single first note
<< \relative { a'4 b g2 }
   \relative { d'4 g2 c,4 } >>
```



In diesem Fall stellt der unterschiedliche Rhythmus kein Problem dar, weil sie in unterschiedlichen Stimmen interpretiert werden.

Bekannte Probleme und Warnungen

Wenn Noten zweier oder mehrerer Stimmen mit Hälsen in die gleiche Richtung an der selben Position auf dem System gesetzt werden und keinen Versatz durch `shift` aufweisen (oder den gleichen Versatz besitzen), erscheint die Nachricht

```
Warnung: zu viele kollidierende Notenspalten werden ignoriert
```

während der Kompilation. Diese Nachricht kann unterdrückt werden durch

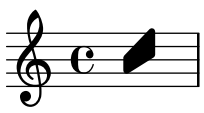
```
\override NoteColumn.ignore-collision = ##t
```

Das unterdrückt jedoch nicht nur die Warnungen, sondern schaltet auch die Auflösung von Zusammenstößen ab und kann also zu unbeabsichtigten Resultaten führen. (Siehe auch Abschnitt 5.2.3 [Auflösung von Zusammenstößen], Seite 170.)

5.1.4 Cluster

Ein Cluster zeigt an, dass alle Tonhöhen in einem Bereich gleichzeitig gespielt werden sollen. Cluster können gedeutet werden als eine Zusammenfassung einer ganzen Anzahl von Noten. Sie werden notiert, indem die Funktion `\makeClusters` auf eine Reihe von Akkorden angewendet wird:

```
\relative \makeClusters { <g' b>2 <c g'> }
```



Normale Noten und Cluster können zusammen im selben System notiert werden, sogar gleichzeitig. In solchen Fällen wird nicht versucht, automatisch Zusammenstöße zwischen normalen Noten und Clustern aufzulösen.

Siehe auch

Musikglossar: Abschnitt “cluster” in *Glossar*.

Schnipsel: Abschnitt “Simultaneous notes” in *Schnipsel*.

Referenz der Interna: Abschnitt “ClusterSpanner” in *Referenz der Interna*, Abschnitt “ClusterSpannerBeacon” in *Referenz der Interna*, Abschnitt “Cluster_spanner_engraver” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Cluster sehen nur gut aus, wenn sie wenigstens über zwei Akkorde reichen – andernfalls sind sie zu schmal.

Cluster haben keine Hälse und können auch selber keine Dauern darstellen, aber die Länge des gesetzten Clusters wird erschlossen anhand der Dauern der definierten Akkorde. Voneinander getrennte Cluster brauchen eine unsichtbare Pause zwischen sich.

Cluster produzieren kein MIDI.

5.2 Mehrere Stimmen

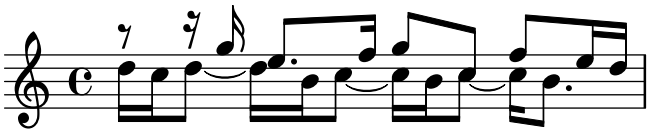
Dieser Abschnitt behandelt gleichzeitige Noten in mehreren Stimmen oder mehreren Systemen.

5.2.1 Mehrstimmigkeit in einem System

Stimmen explicit beginnen

Die grundlegende Struktur, die man benötigt, um mehrere unabhängige Stimmen in einem Notensystem zu setzen, ist im Beispiel unten dargestellt:

```
\new Staff <<
  \new Voice = "first"
    \relative { \voiceOne r8 r16 g'' e8. f16 g8[ c,] f e16 d }
  \new Voice= "second"
    \relative { \voiceTwo d''16 c d8~ 16 b c8~ 16 b c8~ 16 b8. }
>>
```



Stimmen werden hier explizit erstellt und erhalten Bezeichnungen zugewiesen. Die `\voiceOne ... \voiceFour`-Befehle stellen die Stimmen so ein, dass für die erste und dritte Stimme die Hälse nach oben zeigen, für die zweite und vierte Stimme hingegen nach unten. Die Noten der dritten und vierten Stimme werden horizontal verschoben, und Pausen in den entsprechenden Stimmen werden automatisch verschoben, um Zusammenstöße zu vermeiden. Der `\oneVoice`-Befehl stellt das Standardverhalten mit neutralen Halsrichtungen wieder her.
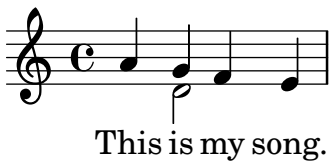
Vorübergehende polyphone Passagen

Ein vorübergehender polyphoner Abschnitt kann mit folgender Konstruktion erstellt werden:

```
<< { \voiceOne ... }
  \new Voice { \voiceTwo ... }
>> \oneVoice
```

Der erste Ausdruck innerhalb des polyphonen Abschnitts wird in den `Voice`-Kontext gestellt, der unmittelbar vor dem polyphonen Abschnitt aktiv war, und der gleiche `Voice`-Kontext setzt sich nach dem Abschnitt fort. Andere Ausdrücke innerhalb der eckigen Klammern werden anderen Stimmennummern zugewiesen. Damit lassen sich auch Gesangstexte einer durchgehenden Stimme vor, während und nach dem polyphonen Abschnitt zuweisen:

```
\relative <<
  \new Voice = "melody" {
    a'4
    <<
      {
        \voiceOne
        g f
      }
      \new Voice {
        \voiceTwo
        d2
      }
    >>
    \oneVoice
    e4
  }
  \new Lyrics \lyricsto "melody" {
  This is my song.
  }
>>
```
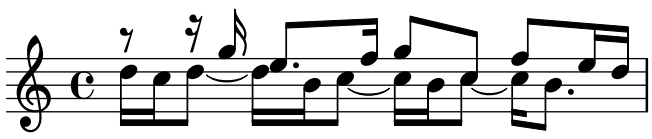


Hierbei sind die Befehle `\voiceOne` und `\voiceTwo` notwendig, um die Einstellungen für jede Stimme zu initialisieren.

Die Konstruktion mit doppeltem Backslash

Die `<< {...} \\ {...} >>`-Konstruktion, in welcher die beiden (oder mehreren) Ausdrücke durch doppelte Backslash-Zeichen (Taste AltGr+ß) getrennt werden, verhält sich anderes als die ähnliche Konstruktion ohne die doppelten Schrägstriche: *alle* Ausdrücke innerhalb der eckigen Klammern werden in diesem Fall jeweils neuen `Voice`-Kontexten zugeordnet. Diese neuen `Voice`-Kontexte werden implizit erstellt und haben die festen Bezeichnungen `"1"`, `"2"` usw.

Das erste Beispiel könnte also auch wie folgt notiert werden:

```
<<
  \relative { r8 r16 g'' e8. f16 g8[ c,] f e16 d }
  \\
  \relative { d''16 c d8~ 16 b c8~ 16 b c8~ 16 b8. }
>>
```
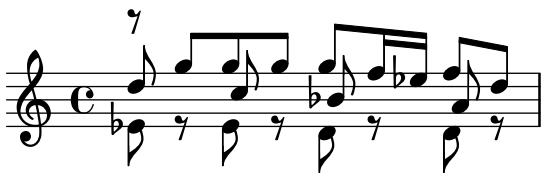


Diese Syntax kann benutzt werden, wenn es keine Rolle spielt, ob vorübergehend Stimmen erstellt werden und dann wieder verworfen werden. Diese implizit erstellten Stimmen erhalten die Einstellungen, die in den Befehlen `\voiceOne` ... `\voiceFour` enthalten sind, in der Reihenfolge, in der sie im Quelltext auftauchen.

Im nächsten Beispiel zeigen die Hälse der zeitweiligen Stimme nach oben, sie wird deshalb erst als dritte in der Konstruktion notiert, damit sie die Eigenschaften von `voiceThree` zugewie-

sen bekommt. Unsichtbare Pause werden eingesetzt, damit keine doppelten Pausen ausgegeben werden.

```
<<
  \relative { r8 g'' g  g g f16 ees f8 d }
  \\
  \relative { ees'8 r ees r d r d r }
  \\
  \relative { d''8 s c s bes s a s }
>>
```



Es wird sehr empfohlen, in allen außer den allereinfachsten Stücken explizite Stimmenkontexte zu erstellen, wie erklärt inAbschnitt “Kontexte und Engraver” in *Handbuch zum Lernen* und Abschnitt “Stimmen explizit beginnen” in *Handbuch zum Lernen*.
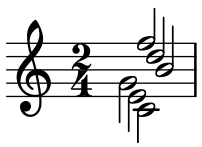
Stimmen-Anordnung

Wenn mehrere Stimmen notiert werden, sollte folgende Anordnung eingehalten werden:

```
Stimme 1: höchste
Stimme 2: tiefste
Stimme 3: zweithöchste
Stimme 4: zweittiefste
Stimme 5: dritthöchste
Stimme 6: dritttiefste
usw.
```

Auch wenn das erst nicht einleuchtend erscheint, erleichtert es den automatischen Layoutprozess doch sehr. Die ungeraden Stimmen erhalten Hälse nach oben, die graden Stimmen Hälse nach unten:

```
\new Staff <<
  \time 2/4
  { f''2 }  % 1: highest
  \\
  { c'2  }  % 2: lowest
  \\
  { d''2 }  % 3: second-highest
  \\
  { e'2  }  % 4: second-lowest
  \\
  { b'2  }  % 5: third-highest
  \\
  { g'2  }  % 6: third-lowest
>>
```
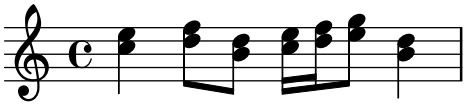


Achtung: Gesangstext und Strecker (etwa Bögen, Bindebögen, Crescendoklammern usw.) können nicht zwischen zwei Stimmen erstellt werden.

Identische Rhythmen

Wenn parallele Abschnitte gesetzt werden sollen, die identischen Rhythmus haben, kann man die Ausdrücke in einen einzigen `Voice`-Kontext parallel kombinieren, sodass sich Akkorde ergeben. Um das zu erreichen, müssen sie einfach von spitzen Klammern innerhalb einer expliziten Stimme umgeben werden:

```
\new Voice <<
  \relative { e''4 f8 d e16 f g8 d4 }
  \relative { c''4 d8 b c16 d e8 b4 }
>>
```



Mit dieser Methode können sich seltsame Balken und Warnungen ergeben, wenn die Musikausdrücke nicht den gleichen Rhythmus haben.

Vordefinierte Befehle

`\voiceOne`, `\voiceTwo`, `\voiceThree`, `\voiceFour`, `\oneVoice`.

Siehe auch

Handbuch zum Lernen: Abschnitt “Voice enthält Noten” in *Handbuch zum Lernen*, Abschnitt “Stimmen explizit beginnen” in *Handbuch zum Lernen*.

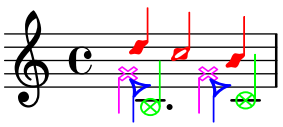
Notationsreferenz: Abschnitt 13.1.5 [Schlagzeugsysteme], Seite 377, Abschnitt 2.2.2 [Unsichtbare Pausen], Seite 59, Abschnitt 7.1.6 [Hälse], Seite 220.

Schnipsel: Abschnitt “Simultaneous notes” in *Schnipsel*.

5.2.2 Stimmenstile

Stimmen können unterschiedliche Farben erhalten, um einfach erkennbar zu sein:

```
<<
  \relative { \voiceOneStyle d''4 c2 b4 }
  \\
  \relative { \voiceTwoStyle e'2 e }
  \\
  \relative { \voiceThreeStyle b2. c4 }
  \\
  \relative { \voiceFourStyle g'2 g }
>>
```



Der `\voiceNeutralStyle`-Befehl wird benutzt, um wieder die Standardausgabe einzuschalten.

Vordefinierte Befehle

`\voiceOneStyle`, `\voiceTwoStyle`, `\voiceThreeStyle`, `\voiceFourStyle`, `\voiceNeutralStyle`.

Siehe auch
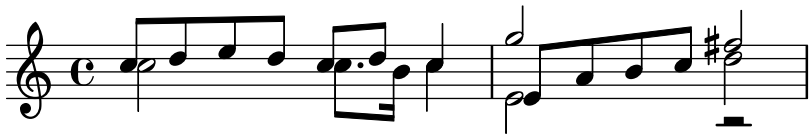
Handbuch zum Lernen: Abschnitt “Ich höre Stimmen” in *Handbuch zum Lernen*, Abschnitt “Mehr Information” in *Handbuch zum Lernen*.

Schnipsel: Abschnitt “Simultaneous notes” in *Schnipsel*.

5.2.3 Auflösung von Zusammenstößen

Die Notenköpfe von Noten in unterschiedlichen Stimmen mit derselben Tonhöhe, demselben Notenkopf und den Hälsen in entgegengesetzte Richtungen werden automatisch verschmolzen, aber Noten mit unterschiedlichen Köpfen oder den Hälsen in die selbe Richtung werden nicht verschmolzen. Pausen, die einem Hals in einer anderen Stimme gegenüberstehen, werden vertikal verschoben. Das folgende Beispiel zeigt drei unterschiedliche Situationen, auf Taktposition 1 und 3 in Takt 1 und Taktposition 1 in Takt 2, wo das automatische Verschmelzen nicht funktioniert.
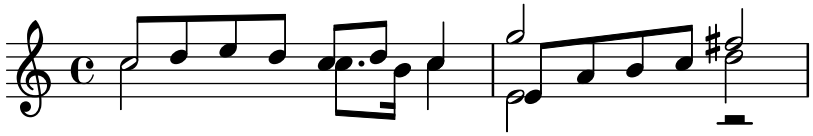
```
<<
  \relative {
    c''8 d e d c d c4
    g'2 fis
  } \\
  \relative {
    c''2 c8. b16 c4
    e,2 r
  } \\
  \relative {
    \oneVoice
    s1
    e'8 a b c d2
  }
>>
```



Noten mit unterschiedlichen Notenköpfen können verschmolzen werden, mit der Ausnahme von Halben- und Viertelnotenköpfen, wie im Beispiel unten gezeigt. Hier werden die Notenköpfe auf Taktposition 1 im ersten Takt verschmolzen:
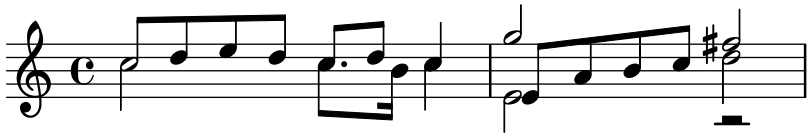
```
<<
  \relative {
    \mergeDifferentlyHeadedOn
    c''8 d e d c d c4
    g'2 fis
  } \\
  \relative {
    c''2 c8. b16 c4
    e,2 r
  } \\
  \relative {
```

```
    \oneVoice
    s1
    e'8 a b c d2
  }
>>
```



Auch Köpfe mit unterschiedlichen Punktierungen wie auf Taktposition 3 im ersten Takt können verschmolzen werden:
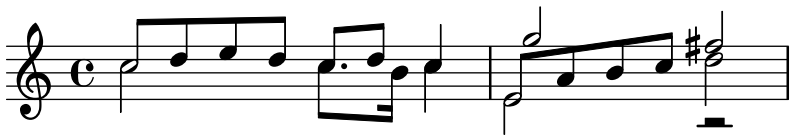
```
<<
  \relative {
    \mergeDifferentlyHeadedOn
    \mergeDifferentlyDottedOn
    c''8 d e d c d c4
    g'2 fis
  } \\
  \relative {
    c''2 c8. b16 c4
    e,2 r
  } \\
  \relative {
    \oneVoice
    s1
    e'8 a b c d2
  }
>>
```



Die Halbe und die Achtel am Anfang des zweiten Taktes werden fehlerhaft verschmolzen, weil die automatische Verschmelzung nicht richtig arbeiten kann, wenn drei oder mehr Noten zur gleichen Zeit auftreten – und in diesem Fall ist der verschmolzene Notenkopf nicht richtig. Um das Verschmelzen zuzulassen, muss ein `\shift` (Verschiebung) auf die Note angewendet werden, die nicht verschmolzen werden soll. In diesem Fall wurde `\shiftOn` gesetzt, um das oberste *g* aus der Kolumne zu entfernen. Jetzt funktioniert `\mergeDifferentlyHeadedOn` (verschmelze Noten mit unterschiedlichen Köpfen) so wie es soll.

```
<<
  \relative {
    \mergeDifferentlyHeadedOn
    \mergeDifferentlyDottedOn
    c''8 d e d c d c4
    \shiftOn
    g'2 fis
  } \\
  \relative {
    c''2 c8. b16 c4
```

```
    e,2 r
  } \\
  \relative {
    \oneVoice
    s1
    e'8 a b c d2
  }
>>
```
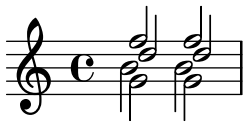


Der `shiftOn`-Befehl ermöglicht die Noten einer Stimme zu verschieben, erzwingt dieses Verhalten aber nicht. Wenn `shiftOn` auf eine Stimme angewendet wird, eine Note oder ein Akkord in der Stimme wird nur verschoben, wenn sein Hals mit dem Hals der Note einer anderen Stimme kollidieren würde, und nur, wenn der Hals der Kollisionsnote in die gleiche Richtung zeigt. Der `shiftOff`-Befehl verhindert, dass eine derartige Verschiebung stattfinden kann.

Die äußeren Stimmen (also normalerweise Stimmen 1 und 2) haben als Standard `shiftOff` eingestellt, während die inneren Stimmen (3 und mehr) `shiftOn` definiert haben. Wenn eine Verschiebung stattfindet, werden Stimmen mit den Hälsen nach oben (also ungerade Stimmen) nach rechts verschoben, während Stimmen mit den Hälsen nach unten (also gerade Stimmen) nach links verschoben werden.

Hier ein Beispiel, das verstehen hilft, wie ein verkürzter polyphonischer Abschnitt intern ausgeweitet wird.

> **Achtung:** Wenn Sie drei oder mehr Stimmen haben, sollte die vertikale Anordnung der Stimmen in der Eingabedatei nicht die gleiche sein wie die vertikale Anordnung der Stimmen im Notensystem!

```
\new Staff \relative {
  %% abbreviated entry
  <<
    { f''2  }  % 1: highest
    \\
    { g,2 }  % 2: lowest
    \\
    { d'2 }  % 3: upper middle
    \\
    { b2  }  % 4: lower middle
  >>
  %% internal expansion of the above
  <<
    \new Voice = "1" { \voiceOne   \shiftOff f'2 }
    \new Voice = "2" { \voiceTwo   \shiftOff g,2 }
    \new Voice = "3" { \voiceThree \shiftOn  d'2 } % shifts right
    \new Voice = "4" { \voiceFour  \shiftOn  b2  } % shifts left
  >>
}
```



Zwei zusätzliche Befehle, `shiftOnn` und `shiftOnnn` stellen weitere Verschiebungsebenen zu verfügung, die vorübergehend eingesetzt werden können um Zusammenstöße in komplizierten Situationen aufzulösen. Siehe auch Abschnitt "Beispiel aus dem Leben" in *Handbuch zum Lernen*.

Noten werden nur verschmolzen, wenn ihre Hälse in entgegengesetzte Richtungen zeigen (also etwa wie `Voice` 1 und 2 in den Standardeinstellungen oder wenn die Hälse explizit in unterschiedliche Richtungen gedreht sind).

Vordefinierte Befehle

`\mergeDifferentlyDottedOn`, `\mergeDifferentlyDottedOff`, `\mergeDifferentlyHeadedOn`, `\mergeDifferentlyHeadedOff`, `\shiftOn`, `\shiftOnn`, `\shiftOnnn`, `\shiftOff`.

Ausgewählte Schnipsel
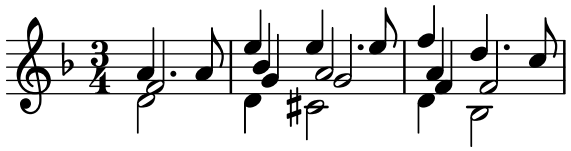
Zusätzliche Stimmen, um Zusammenstöße zu vermeiden

Ein einigen Fällen von sehr komplexer polyphoner Musik sind zusätzliche Stimmen notwendig, um Zusammenstöße zwischen den Noten zu vermeiden. Wenn mehr als vier parallele Stimmen benötigt werden, können zusätzliche Stimmen definiert werden, indem eine Variable mit der Funktion `context-spec-music` definiert wird.

```
voiceFive = #(context-spec-music (make-voice-props-set 4) 'Voice)

\relative c'' {
  \time 3/4
  \key d \minor
  \partial 2
  <<
    \new Voice  {
      \voiceOne
      a4. a8
      e'4 e4. e8
      f4 d4. c8
    }
    \new Voice {
      \voiceTwo
      d,2
      d4 cis2
      d4 bes2
    }
    \new Voice {
      \voiceThree
      f'2
      bes4 a2
      a4 s2
    }
    \new Voice {
      \voiceFive
      s2
```
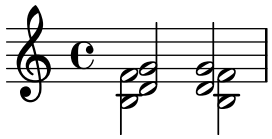
```
      g4 g2
      f4 f2
    }
  >>
}
```



Horizontale Verschiebung von Noten erzwingen

Wenn es zu Zusammenstößen kommt, kann mit folgender Lösung eine andere Position manuell eingestellt werden. Die Einheiten hier sind Notenlinienzwischenräume.

```
\relative c' <<
  {
    <d g>2 <d g>
  }
  \\
  {
    <b f'>2
    \once \override NoteColumn.force-hshift = 1.7
    <b f'>2
  }
>>
```



Siehe auch

Musikglossar: Abschnitt “polyphony” in *Glossar*.

Handbuch zum Lernen: Abschnitt “Mehrere Noten auf einmal” in *Handbuch zum Lernen*, Abschnitt “Voice enthält Noten” in *Handbuch zum Lernen*, Abschnitt “Beispiel aus dem Leben” in *Handbuch zum Lernen*.

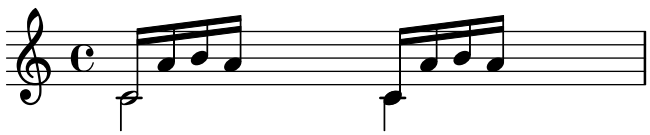
Schnipsel: Abschnitt “Simultaneous notes” in *Schnipsel*.

Referenz der Interna: Abschnitt “NoteColumn” in *Referenz der Interna*, Abschnitt “NoteCollision” in *Referenz der Interna*, Abschnitt “RestCollision” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Die Benutzung von `\override NoteColumn.ignore-collision = ##t` führt dazu, dass Noten mit unterschiedlichen Köpfen in unterschiedlichen Stimmen falsch verschmolzen werden.

```
\mergeDifferentlyHeadedOn
<< \relative { c'16 a' b a } \\ \relative { c'2 } >>
\override NoteColumn.ignore-collision = ##t
<< \relative { c'16 a' b a } \\ \relative { c'2 } >>
```



5.2.4 Automatische Kombination von Stimmen

Automatische Kombination von Stimmen wird verwendet, um zwei selbständige Stimmen auf einem Notensystem zu setzen. Es wird vor allem in Orchesterpartituren eingesetzt. Wenn beide Stimmen die gleichen Noten haben, wird nur eine Stimme gesetzt, wenn sie sich unterscheiden, werden sie als unterschiedliche Stimmen (`Voice`) gesetzt, und die Richtung der Hälse wird automatisch bestimmt. Zusätzlich werden *solo* und a *due*-Stellen erkannt und bezeichnet.

Die Syntax zur automatischen Stimmenkombination lautet:
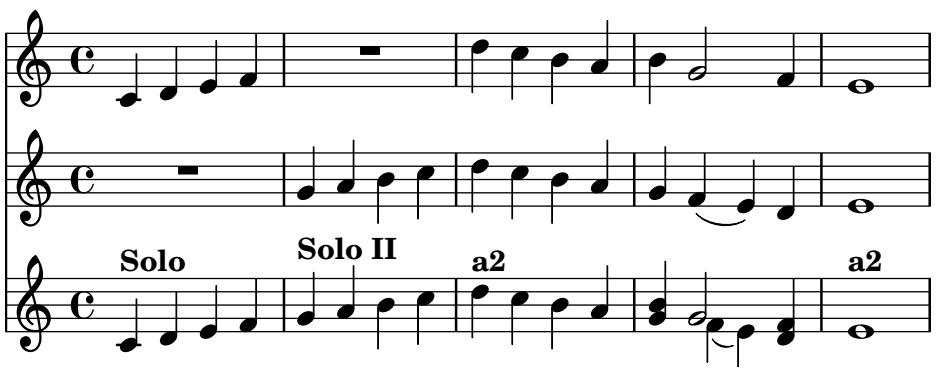
`\partCombine` *musikAusdr1 musikAusdr2*

Das nächste Beispiel zeigt, wie die Kombination funktioniert. Hier werden die Stimmen erst auf einem jeweils eigenen System und dann kombiniert gesetzt. Die gleichen Variablen werden für die Stimmen und das kombinierte System benutzt.

```
instrumentOne = \relative {
  c'4 d e f |
  R1 |
  d'4 c b a |
  b4 g2 f4 |
  e1 |
}

instrumentTwo = \relative {
  R1 |
  g'4 a b c |
  d4 c b a |
  g4 f( e) d |
  e1 |
}

<<
  \new Staff \instrumentOne
  \new Staff \instrumentTwo
  \new Staff \partCombine \instrumentOne \instrumentTwo
>>
```



Beide Stimmen haben die gleichen Noten im dritten Takt, sodass sie nur als eine Stimme gesetzt werden. Die Richtung von Hälsen und Bögen werden automatisch gewählt, abhängig davon ob es eine Solo-Stelle oder Unisono ist. In polyphonen Situationen erhält die erste Stimme (mit dem Kontext `one`) immer Hälse nach oben, die zweite Stimme (mit dem Kontext `two`) Hälse nach unten. An Solo-Stellen werden die Stimmen mit „Solo“ bzw. „Solo II“ bezeichnet. Die Unisono-Stellen (*a due*) werden mit dem Text „a2“ gekennzeichnet.

Beide Argumente von \partCombine werden als eigenständige Voice-Kontexte interpretiert. Wenn relative Oktaven benutzt werden, muss \relative für beide Stimmen benutzt werden, also:

```
\partCombine
  \relative ... musikAusdr1
  \relative ... musikAusdr2
```

Ein \relative-Abschnitt, der \partCombine umschließt, hat keinen Einfluss auf die Tonhöhen von *musikAusdr1* oder *musikAusdr2*.

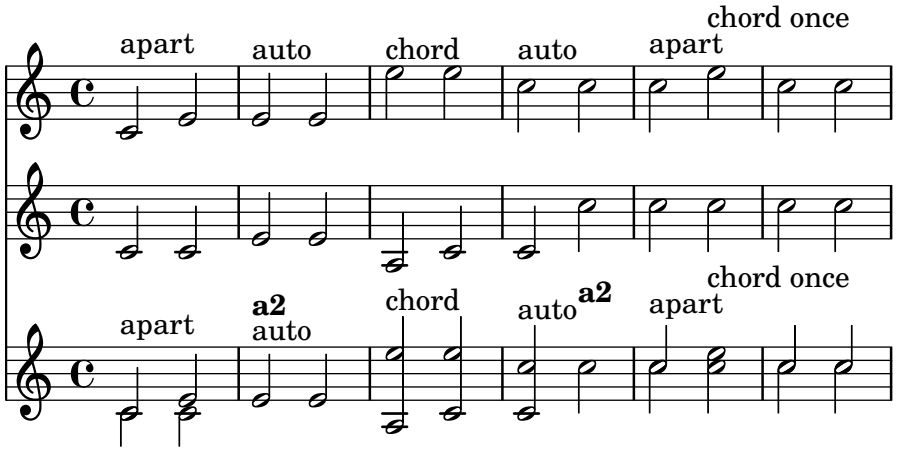
In professionellen Partituren werden Stimmen oft für lange Strecken auseinander gehalten, auch wenn eine oder mehrere Noten tatsälich aufeinander treffen und einfach als Unisono gesetzt werden könnten. Noten zu Akkorden zusammenzufassen oder eine Stimme als Solo anzuzeigen ist darum nicht ideal, weil die \partCombine-Funktion jede Note separat betrachtet. Aus diesem Grund kann die \partCombine-Funktion mit folgenden Befehlen verändert werden:

Befehle, die auf ... Once enden, gelten nur für die nächste Note eines musikalischen Ausdrucks.

- \partCombineApart und \once \partCombineApart erhalten die Noten als zwei unterschiedliche Stimmen, auch wenn sie als Akkord oder Unisono kombiniert werden könnten.
- \partCombineChords und \once \partCombineChords kombinieren die Noten als Akkord.
- \partCombineUnisono und \once \partCombineUnisono kombinieren beide Stimmen als Unisono.
- \partCombineSoloI und \once \partCombineSoloI setzen nur Stimme eins und markieren sie als „Solo“.
- \partCombineSoloII und \once \partCombineSoloII setzen nur Stimme zwei und markieren sie als „Solo“.
- \partCombineAutomatic und \once \partCombineAutomatic beenden die Wirkung der Befehle oben und stellt das normale Verhalten des Kombinationsmechanismus wieder her.

```
instrumentOne = \relative c' {
  \partCombineApart c2^"apart" e |
  \partCombineAutomatic e2^"auto" e |
  \partCombineChords e'2^"chord" e |
  \partCombineAutomatic c2^"auto" c |
  \partCombineApart c2^"apart" \once \partCombineChords e^"chord once" |
  c2 c |
}
instrumentTwo = \relative {
  c'2 c |
  e2 e |
  a,2 c |
  c2 c' |
  c2 c |
  c2 c |
}

<<
  \new Staff { \instrumentOne }
  \new Staff { \instrumentTwo }
  \new Staff { \partCombine \instrumentOne \instrumentTwo }
>>
```



Ausgewählte Schnipsel

Zwei Stimmen auf einem System kombinieren

Die Funktion, die Stimmen kombiniert (also der `\partCombine`-Befehl) ermöglicht die Kombination unterschiedlicher Stimmen auf einem System. Textanweisungen wie "solo" oder "a2" werden automatisch hinzugefügt. Wenn man sie entfernen will, muss man die Eigenschaft `printPartCombineTexts` auf falsch setzen. Für Klavierauszüge muss natürlich kein "solo"/"a2" usw. hinzugefügt werden, man sollte sie also ausschalten. Wenn aber Solo-Stellen in einem Klavierauszug oder einer Chorpartitur angezeigt werden, ist es besser, normale Polyphonie zu verwenden, weil so die Solostellen angezeigt werden, auch wenn der Text des Stimmenkombinierers ausgeschaltet ist.

Der Schnipsel zeigt drei Möglichkeiten, Stimmen auf einem System zu kombinieren: Standardpolyphonie, `\partCombine` ohne Text und `\partCombine` mit Text.

```
musicUp = \relative c'' {
  \time 4/4
  a4 c4.( g8) a4 |
  g4 e' g,( a8 b) |
  c b a2.
}

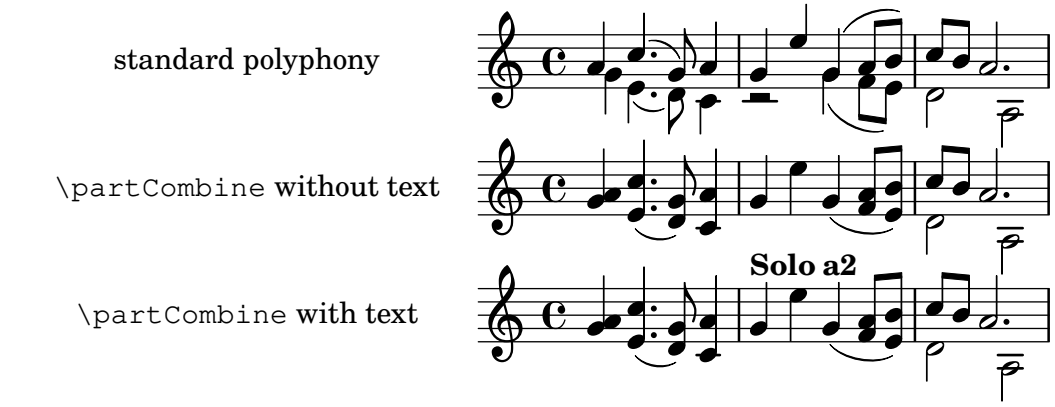
musicDown = \relative c'' {
  g4 e4.( d8) c4 |
  r2 g'4( f8 e) |
  d2 \stemDown a
}

\score {
  <<
    \new Staff \with {
      instrumentName = "standard polyphony"
    } << \musicUp \\ \musicDown >>

    \new Staff \with {
      instrumentName =
        \markup { \typewriter "\\partCombine" without text}
      printPartCombineTexts = ##f
    } \partCombine \musicUp \musicDown
```

```
    \new Staff \with {
      instrumentName =
        \markup { \typewriter "\\partCombine" with text}
    } \partCombine \musicUp \musicDown
  >>
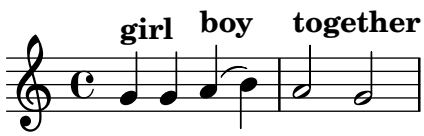
  \layout {
    indent = 6.0\cm
    \context {
      \Score
      % Setting this to a large value avoids a bar line at the
      % beginning that would connect the three staves otherwise.
      \override SystemStartBar.collapse-height = 30
    }
  }
}
```



Changing \partCombine texts

When using the automatic part combining feature, the printed text for the solo and unison sections may be changed.

```
\new Staff <<
  \set Staff.soloText = "girl"
  \set Staff.soloIIText = "boy"
  \set Staff.aDueText = "together"
  \partCombine
    \relative c'' {
      g4 g r r
      a2 g
    }
    \relative c'' {
      r4 r a( b)
      a2 g
    }
>>
```



Siehe auch

Musikglossar: Abschnitt “a due” in *Glossar*, Abschnitt “part” in *Glossar*.

Notationsreferenz: Abschnitt 6.3 [Orchesterstimmen erstellen], Seite 201.

Schnipsel: Abschnitt “Simultaneous notes” in *Schnipsel*.

Referenz der Interna: Abschnitt “PartCombineMusic” in *Referenz der Interna*, Abschnitt “Voice” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Alle `\partCombine...`-Funktionen können nur zwei Stimmen aufnehmen und sind nicht für die Behandlung von Gesangstext geeignet. Das heißt, dass die Funktion nicht funktioniert, wenn einer der Kontexte explizit eine Bezeichnung erhält, um ihm etwa Gesangstext zuweisen zu können.

`\partCombine...`-Funktionen können nicht innerhalb von `\times`- oder `\relative`-Umgebungen geschrieben werden.

Wenn `printPartCombineTexts` definiert ist und die zwei Stimmen die gleichen Noten abwechselnd spielen, kann es sein, dass die Kombinationsfunktion den Text „a2“ mehr als einmal im gleichen Takt setzt.

`\partCombine` merkt nur, wenn eine Note in einer `Voice`-Umgebung beginnt; Information, dass eine Note in einer Stimme etwa schon begonnen hat, wenn die andere Stimme die gleiche Note spielt, fehlt. Das kann zu einigen unerwarteten Problemen führen, wie etwa dass „Solo“ bzw. „Unison“ falsch gesetzt werden.

`\partCombine` hält alle Strecker (Haltebögen, Legatobögen, Klammern usw.) in der gleichen `Voice`-Umgebung, sodass es sein kann, dass diese Strecker nicht korrekt erscheinen, wenn sie in einer anderen Stimme enden.

Wenn die `\partCombine`-Funktion beide Noten nicht kombinieren kann (beispielsweise weil beide Stimmen unterschiedliche Dauern haben), werden die Stimmen intern in `one` und `two` benannt. Das heißt, dass jeglicher Kontextwechsel zu einem `Voice`-Kontext mit anderer Bezeichnung ignoriert wird.

Lesen Sie auch den Abschnitt *Bekannte Probleme und Warnungen*, wenn Sie `\partCombine` in Tabulaturen benutzen (siehe Abschnitt 12.1.3 [Standardtabulaturen], Seite 330).

5.2.5 Musik parallel notieren

Noten für mehrere Stimmen können verschachtelt notiert werden. Die Funktion `\parallelMusic` akzeptiert eine Liste mit den Bezeichnungen einer Reihe von Variablen und einen musikalischen Ausdruck. Der Inhalt der verschiedenen Takte in dem musikalischen Ausdruck bekommt die Bezeichnung der Variablen zugewiesen, sodass sie benutzt werden können, um die Musik dann zu setzen. Dabei entspricht jede Zeile einer Stimme.

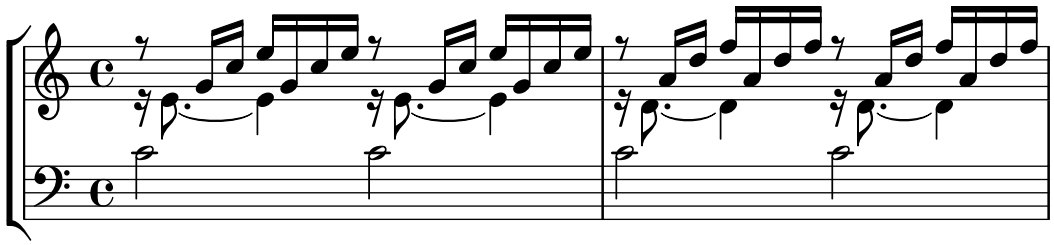
> **Achtung:** Taktüberprüfungen `|` müssen benutzt werden, und die Takte müssen die gleiche Länge haben.

```
\parallelMusic voiceA,voiceB,voiceC {
  % Bar 1
  r8 g'16 c'' e'' g' c'' e'' r8 g'16 c'' e'' g' c'' e'' |
  r16 e'8.~   4             r16 e'8.~   4               |
  c'2                       c'2                         |

  % Bar 2
  r8 a'16 d'' f'' a' d'' f'' r8 a'16 d'' f'' a' d'' f'' |
```

```
  r16 d'8.~   4              r16 d'8.~   4              |
  c'2                        c'2                        |

}
\new StaffGroup <<
  \new Staff << \voiceA \\ \voiceB >>
  \new Staff { \clef bass \voiceC }
>>
```

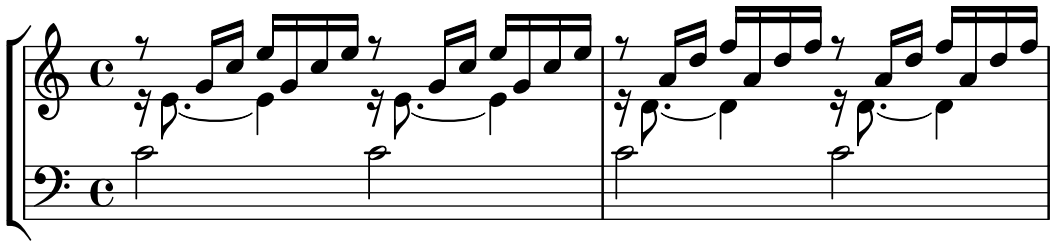


Der relative Modus kann auch benutzt werden. Beachten Sie, dass der `\relative`-Befehl nicht innerhalb von `\parallelMusic` benutzt wird. Die Noten sind parallel zu der vorherigen Note der gleichen Stimme , nicht zu der vorherigen Note in der Quelldatei. Anders gesagt ignorieren relative Noten von `voiceA` die Noten von `voiceB`.

```
\parallelMusic voiceA,voiceB,voiceC {
  % Bar 1
  r8 g16 c e g, c e r8 g,16 c e g, c e  |
  r16 e8.~ 4        r16 e8.~  4         |
  c2                c                   |

  % Bar 2
  r8 a,16 d f a, d f r8 a,16 d f a, d f |
  r16 d8.~  4        r16 d8.~  4        |
  c2                 c                  |

 }
\new StaffGroup <<
  \new Staff << \relative c'' \voiceA \\ \relative c' \voiceB >>
  \new Staff \relative c' { \clef bass \voiceC }
>>
```



Das funktioniert ziemlich gut für Klaviernoten. Dieses Beispiel speichert vier konsekutive Takte in vier Variablen:
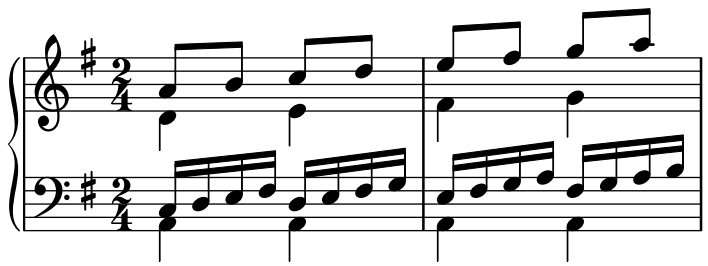
```
global = {
  \key g \major
  \time 2/4
}

\parallelMusic voiceA,voiceB,voiceC,voiceD {
  % Bar 1
  a8    b     c   d     |
  d4          e         |
  c16 d e fis d e fis g  |
  a4          a         |

  % Bar 2
  e8      fis  g     a   |
  fis4         g         |
  e16 fis g  a fis g a b  |
  a4           a         |

  % Bar 3 ...
}

\score {
  \new PianoStaff <<
     \new Staff {
       \global
       <<
         \relative c'' \voiceA
         \\
         \relative c'  \voiceB
       >>
     }
     \new Staff {
       \global \clef bass
       <<
         \relative c \voiceC
         \\
         \relative c \voiceD
       >>
     }
  >>
}
```

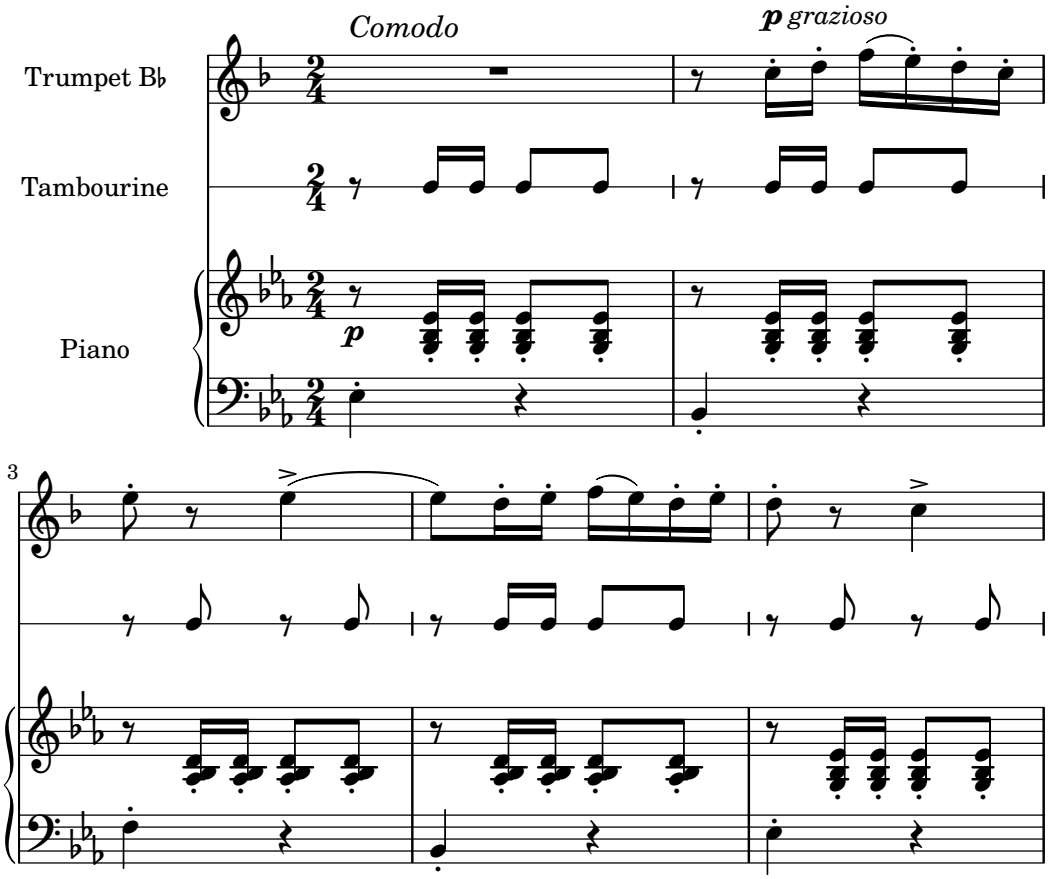


Siehe auch

Handbuch zum Lernen: Abschnitt “Stücke durch Variablen organisieren” in *Handbuch zum Lernen*.

Schnipsel: Abschnitt “Simultaneous notes” in *Schnipsel*.

6 Notation auf Systemen



Dieser Abschnitt zeigt, wie die Erscheinung von Systemen beeinflusst wird, wie Partituren mit mehr als einem System gesetzt werden und wie man Aufführungsanweisungen und Stichnoten zu einzelnen Systemen hinzufügt.

6.1 Systeme anzeigen lassen
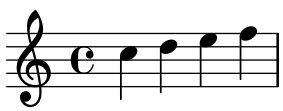
Dieser Abschnitt zeigt unterschiedliche Methoden, Notensysteme und Gruppen von Systemen zu erstellen.

6.1.1 Neue Notensysteme erstellen

Notensysteme (engl. *staff*, Pl. *staves*) werden mit dem `\new` oder `\context`-Befehl erstellt. Zu Einzelheiten siehe Abschnitt 32.2 [Kontexte erstellen und referenzieren], Seite 582.

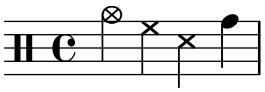
Der einfachste Notensystem-Kontext ist `Staff`:

```
\new Staff \relative { c''4 d e f }
```



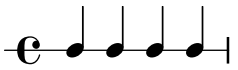
`DrumStaff` (Perkussionsnotensystem) erstellt ein Notensystem mit fünf Linien, das für ein typisches Schlagzeug eingerichtet ist. Für jedes Instrument werden unterschiedliche Symbole dargestellt. Die Instrumente werden innerhalb der `drummode`-Umgebung gesetzt, wo jedes Instrument seine eigene Bezeichnung hat. Zu Einzelheiten siehe Abschnitt 13.1.5 [Schlagzeugsysteme], Seite 377.

```
\new DrumStaff {
  \drummode { cymc hh ss tomh }
}
```



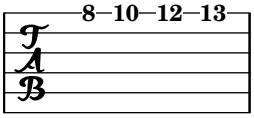
RhythmicStaff (Rhythmus-System) erstellt ein Notensystem mit nur einer Notenlinie, auf welcher nur die rhythmischen Werte der eingegebenen Noten dargestellt werden. Die wirklichen Längen bleiben erhalten. Zu Einzelheiten, siehe Abschnitt 2.3.7 [Melodierhythmus anzeigen], Seite 79.

```
\new RhythmicStaff { c4 d e f }
```



TabStaff (Tabulatursystem) erstellt eine Tabulatur mit sechs Saiten in der üblichen Gitarrenstimmung. Zu Einzelheiten siehe Abschnitt 12.1.3 [Standardtabulaturen], Seite 330.

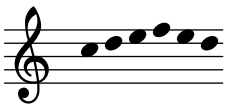
```
\new TabStaff \relative { c''4 d e f }
```



Es gibt zwei Notensysteme, die zur Notation von Alter Musik eingesetzt werden: MensuralStaff and VaticanaStaff. Sie sind erklärt in Abschnitt 17.2.1 [Vordefinierte Umgebungen], Seite 422.

Das GregorianTranscriptionStaff (System zur Transkription des Gregorianischen Chorals) erstellt ein Notensystem, um modernen Gregorianischen Choral zu notieren. Es hat keine Taktstriche.

```
\new GregorianTranscriptionStaff \relative { c''4 d e f e d }
```



Neue Notensystem-Kontexte können selber definiert werden. Zu Einzelheiten, siehe Abschnitt 32.6 [Neue Kontexte definieren], Seite 593.

Siehe auch

Glossar: Abschnitt “staff” in *Glossar*, Abschnitt “staves” in *Glossar*.

Notationsreferenz: Abschnitt 32.2 [Kontexte erstellen und referenzieren], Seite 582, Abschnitt 13.1.5 [Schlagzeugsysteme], Seite 377, Abschnitt 2.3.7 [Melodierhythmus anzeigen], Seite 79, Abschnitt 12.1.3 [Standardtabulaturen], Seite 330, Abschnitt 17.2.1 [Vordefinierte Umgebungen], Seite 422, Abschnitt 6.2.1 [Das Notensystem], Seite 191, Abschnitt 17.4.1 [Gregorianische Gesangs-Kontexte], Seite 431, Abschnitt 17.3.1 [Mensural-Kontexte], Seite 424, Abschnitt 32.6 [Neue Kontexte definieren], Seite 593.

Schnipsel: Abschnitt “Staff notation” in *Schnipsel*.

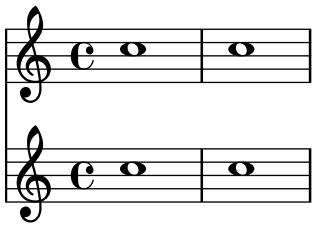
Referenz der Interna: Abschnitt “Staff” in *Referenz der Interna*, Abschnitt “DrumStaff” in *Referenz der Interna*, Abschnitt “GregorianTranscriptionStaff” in *Referenz der Interna*, Abschnitt “RhythmicStaff” in *Referenz der Interna*, Abschnitt “TabStaff” in *Referenz der Interna*, Abschnitt “MensuralStaff” in *Referenz der Interna*, Abschnitt “VaticanaStaff” in *Referenz der Interna*, Abschnitt “StaffSymbol” in *Referenz der Interna*.

6.1.2 Systeme gruppieren

Es gibt verschiedene Kontexte, um einzelne Notensysteme zu gruppieren und einer Partitur zu verbinden. Jeder Gruppenstil beeinflusst das Aussehen des Systemanfangs und das Verhalten der Taktlinien.

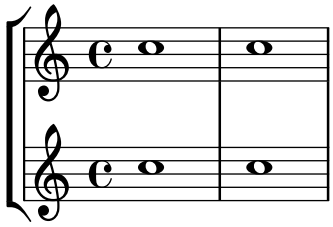
Wenn kein Kontext angegeben ist, wird die Standardeinstellung eingesetzt: die Gruppe beginnt mit einer vertikalen Linie und die Taktlinien sind nicht verbunden.

```
<<
  \new Staff \relative { c''1 c }
  \new Staff \relative { c''1 c }
>>
```



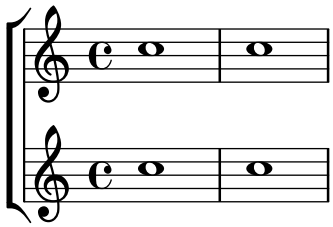
Im `StaffGroup`-Kontext die Gruppe mit einer eckigen Klammer begonnen und die Taktlinien durch alle Systeme gezogen.

```
\new StaffGroup <<
  \new Staff \relative { c''1 c }
  \new Staff \relative { c''1 c }
>>
```



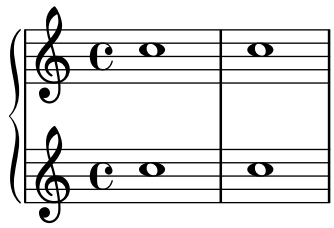
In einem `ChoirStaff` (Chorsystem) beginnt die Gruppe mit einer eckigen Klammer, aber die Taktlinien sind nicht verbunden.

```
\new ChoirStaff <<
  \new Staff \relative { c''1 c }
  \new Staff \relative { c''1 c }
>>
```



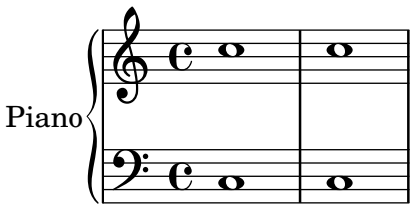
In einem `GrandStaff` (Akkolade) beginnt die Gruppe mit einer geschweiften Klammer und die Taktlinien sind durchgezogen.

```
\new GrandStaff <<
  \new Staff \relative { c''1 c }
  \new Staff \relative { c''1 c }
>>
```



Der PianoStaff-(Klaviersystem)-Kontext ist identisch mit dem GrandStaff-Kontext, aber es ermöglicht zusätzlich direkt die Angabe einer Instrumentbezeichnung. Zu Einzelheiten siehe Abschnitt 6.3.1 [Instrumentenbezeichnungen], Seite 201.

```
\new PianoStaff <<
  \set PianoStaff.instrumentName = "Piano"
  \new Staff \relative { c''1 c }
  \new Staff \relative { \clef bass c1 c }
>>
```



Jede Systemgruppe stellt die Eigenschaft systemStartDelimiter (SystemBeginnBegrenzer) auf einen der folgenden Werte: SystemStartBar, SystemStartBrace oder SystemStartBracket. Ein vierter Begrenzer, SystemStartSquare, ist auch erreichbar, aber man muss ihr explizit einstellen.

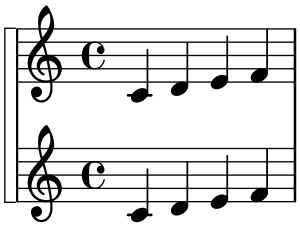
Neue Systemgruppen können definiert werden. Zu Einzelheiten siehe Abschnitt 32.6 [Neue Kontexte definieren], Seite 593.

Ausgewählte Schnipsel

Eine eckige Klammer zu Beginn von Systemgruppen benutzen

Die Klammer zu Beginn von Systemgruppen kann auch in eine eckige Klammer (SystemStartSquare) umgewandelt werden, wenn man sie explizit im StaffGroup- oder ChoirStaffGroup-Kontext setzt.

```
\score {
  \new StaffGroup { <<
  \set StaffGroup.systemStartDelimiter = #'SystemStartSquare
    \new Staff { c'4 d' e' f' }
    \new Staff { c'4 d' e' f' }
  >> }
}
```

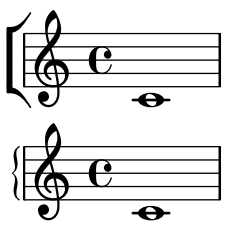


Klammer anzeigen, wenn nur ein System gesetzt wird

Wenn nur ein System einer Systemgruppe vom Typ `ChoirStaff` oder `StaffGroup` angezeigt wird, wird die Klammer zu Beginn normalerweise nicht gesetzt. Das kann verändert werden, indem man die entsprechende Eigenschaft verändert.

Bei Systemen wie `PianoStaff` und `GrandStaff`, die mit einer geschweiften Klammer beginne, muss eine andere Eigenschaft verändert werden, wie das zweite Beispiel zeigt.

```
\score {
  \new StaffGroup <<
    % Must be lower than the actual number of staff lines
    \override StaffGroup.SystemStartBracket.collapse-height = 4
    \override Score.SystemStartBar.collapse-height = 4
    \new Staff {
      c'1
    }
  >>
}
\score {
  \new PianoStaff <<
    \override PianoStaff.SystemStartBrace.collapse-height = 4
    \override Score.SystemStartBar.collapse-height = 4
    \new Staff {
      c'1
    }
  >>
}
```



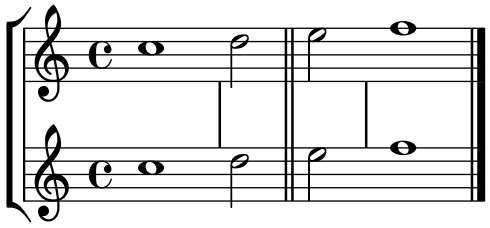
Mensurstriche-Layout (Taktstriche zwischen den Systemen

Das Mensurstiche-Layout, in welchem die Taktlinien nicht auf den Systemen, sondern zwischen den Systemen gesetzt werden, kann mit einer `StaffGroup` anstelle von `ChoirStaff` erreicht werden. Die Taktlinien auf den Systemen werden mit der `transparent`-Eigenschaft ausgelöscht.

```
\layout {
  \context {
    \Staff
    measureBarType = "-span|"
  }
}

music = \fixed c'' {
  c1
  d2 \section e2
  f1 \fine
}

\new StaffGroup <<
  \new Staff \music
  \new Staff \music
>>
```



Siehe auch

Glossar: Abschnitt “brace” in *Glossar*, Abschnitt “bracket” in *Glossar*, Abschnitt “grand staff” in *Glossar*.

Notationsreferenz: Abschnitt 6.3.1 [Instrumentenbezeichnungen], Seite 201, Abschnitt 32.6 [Neue Kontexte definieren], Seite 593.

Schnipsel: Abschnitt “Staff notation” in *Schnipsel*.

Referenz der Interna: Abschnitt “Staff” in *Referenz der Interna*, Abschnitt “StaffGroup” in *Referenz der Interna*, Abschnitt “ChoirStaff” in *Referenz der Interna*, Abschnitt “GrandStaff” in *Referenz der Interna*, Abschnitt “PianoStaff” in *Referenz der Interna*, Abschnitt “SystemStartBar” in *Referenz der Interna*, Abschnitt “SystemStartBrace” in *Referenz der Interna*, Abschnitt “SystemStartBracket” in *Referenz der Interna*, Abschnitt “SystemStartSquare” in *Referenz der Interna*.
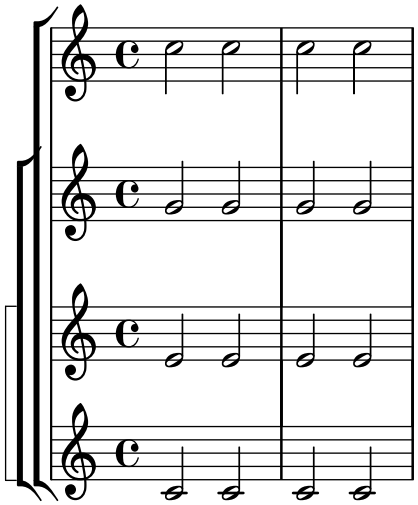
Bekannte Probleme und Warnungen

`PianoStaff` nimmt standardmäßig keine `ChordNames` (Akkordbezeichnungen) auf.

6.1.3 Verschachtelte Notensysteme

System-Gruppen können in beliebiger Tiefe geschachtelt werden. In diesem Fall erstellt jeder neue, innen liegende Kontext eine neue Klammer außerhalb der Klammer der Systemgruppe, in der er sich befindet.

```
\new StaffGroup <<
  \new Staff \relative { c''2 c | c2 c }
  \new StaffGroup <<
    \new Staff \relative { g'2 g | g2 g }
    \new StaffGroup \with {
      systemStartDelimiter = #'SystemStartSquare
    }
    <<
      \new Staff \relative { e'2 e | e2 e }
      \new Staff \relative { c'2 c | c2 c }
    >>
  >>
>>
```



Neue geschachtelte Systemgruppen können definiert werden. Zu Einzelheiten siehe Abschnitt 32.6 [Neue Kontexte definieren], Seite 593.

Ausgewählte Schnipsel

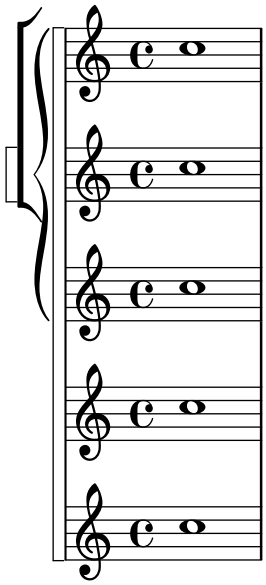
Systeme schachteln

Die Eigenschaft systemStartDelimiterHierarchy kann eingesetzt werden, um komplizierte geschachtelte Systemklammern zu erstellen. Der Befehl \set StaffGroup.systemStartDelimiterHierarchy nimmt eine Liste mit der Anzahl der Systeme, die ausgegeben werden, auf. Vor jedem System kann eine Systemanfangsklammer angegeben werden. Sie muss in Klammern eingefügt werden und umfasst so viele Systeme, wie die Klammer einschließt. Elemente in der Liste können ausgelassen werden, aber die erste Klammer umfasst immer die gesamte Gruppe. Die Möglichkeiten der Anfangsklammer sind: SystemStartBar, SystemStartBracket, SystemStartBrace und SystemStartSquare.

```
\new StaffGroup
\relative c'' <<
  \override StaffGroup.SystemStartSquare.collapse-height = 4
  \set StaffGroup.systemStartDelimiterHierarchy
    = #'(SystemStartSquare
          (SystemStartBrace
            (SystemStartBracket a
              (SystemStartSquare b))
            c)
```

```
        d)

  \new Staff { c1 }
  \new Staff { c1 }
  \new Staff { c1 }
  \new Staff { c1 }
  \new Staff { c1 }
>>
```



Siehe auch

Notationsreferenz: Abschnitt 6.1.2 [Systeme gruppieren], Seite 185, Abschnitt 6.3.1 [Instrumentenbezeichnungen], Seite 201, Abschnitt 32.6 [Neue Kontexte definieren], Seite 593.

Schnipsel: Abschnitt “Staff notation” in *Schnipsel*.

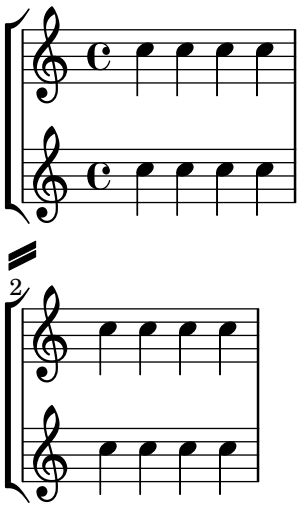
Referenz der Interna: Abschnitt “StaffGroup” in *Referenz der Interna*, Abschnitt “ChoirStaff” in *Referenz der Interna*, Abschnitt “SystemStartBar” in *Referenz der Interna*, Abschnitt “SystemStartBrace” in *Referenz der Interna*, Abschnitt “SystemStartBracket” in *Referenz der Interna*, Abschnitt “SystemStartSquare” in *Referenz der Interna*.

6.1.4 Systeme trennen

Wenn die Anzahl der Systeme sich von Seite zu Seite ändert, wird normalerweise ein Trennzeichen hinzugefügt, dass die Systeme voneinander trennt. Die Standardeinstellung ist, dass der Trenner nicht gesetzt wird, aber man kann ihn mit einer Option in der \paper-Umgebung angeschalten.

```
\book {
  \score {
    \new StaffGroup <<
      \new Staff {
        \relative {
          c''4 c c c
          \break
          c4 c c c
        }
      }
      \new Staff {
```

```
        \relative {
          c''4 c c c
          \break
          c4 c c c
        }
      }
    >>
  }
  \paper {
    system-separator-markup = \slashSeparator
    % following commands are needed only to format this documentation
    paper-width = 100\mm
    paper-height = 100\mm
    tagline = ##f
  }
}
```



Siehe auch

Notationsreferenz: Kapitel 25 [Seitenlayout], Seite 521.

Schnipsel: Abschnitt “Staff notation” in *Schnipsel*.
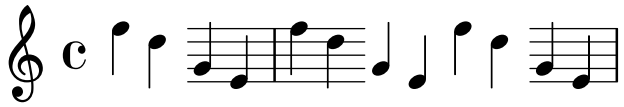
6.2 Einzelne Systeme verändern

Dieser Abschnitt zeigt, wie man bestimmte Eigenschaften eines Systems ändert – etwa die Anzahl der Notenlinien oder die Größe des Systems. Es werden auch Methoden dargestellt, ein System zu beginnen und zu beenden sowie eine Methode, Ossia-Systeme zu erstellen.

6.2.1 Das Notensystem

Die Befehle `\stopStaff` und `\startStaff` können benutzt werden, um ein Notensystem zu stoppen oder (wieder) zu beginnen.

```
\relative {
  \stopStaff f''4 d \startStaff g, e
  f'4 d \stopStaff g, e
  f'4 d \startStaff g, e
}
```



Vordefinierte Befehle
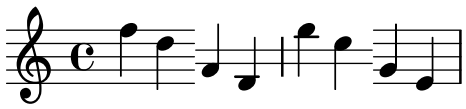
`\startStaff`, `\stopStaff`.

Die Linien eines Notensystems gehören zu dem `StaffSymbol`-(NotensystemSymbol)-Grob (hierzu gehören auch Hilfslinien). `StaffSymbol`-Eigenschaften können verändert werden, um die Erscheinung des Notensystems zu beeinflussen, aber sie müssen gesetzt werden, bevor das System erstellt wird.

Die Anzahl der Notenlinien kann verändert werden:
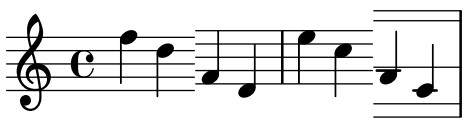
```
\relative {
  f''4 d \stopStaff
  \override Staff.StaffSymbol.line-count = #2
  \startStaff g, e |

  f'4 d \stopStaff
  \revert Staff.StaffSymbol.line-count
  \startStaff g, e |
}
```



Auch die Position der Notenlinien kann geändert werden. Die Werte werden in *halben* Notensystemabständen eingegeben und die neue Position ist relativ zur ursprünglichen Mittellinie. Eine einzelne Linie wird für jeden Wert ausgegeben, sodass die Anzahl der Linien sowie ihre Position im Notensystem mit einem Befehl geändert werden können.
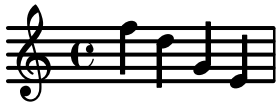
```
\relative {
  f''4 d \stopStaff
  \override Staff.StaffSymbol.line-positions = #'(1 3 5 -1 -3)
  \startStaff g, e |
  f'4 d \stopStaff
  \override Staff.StaffSymbol.line-positions = #'(8 6.5 -6 -8 -0.5)
  \startStaff g, e |
}
```



die Position des Notenschlüssels und die Position von c' können geändert werden, um dem neuen System zu entsprechen. Siehe auch Abschnitt 1.3.1 [Notenschlüssel], Seite 19.
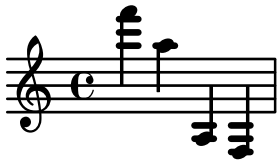
Die Liniendicke der Notenlinien kann verändert werden. Die Dicke der Hilfslinien und Notenhälse wird auch beeinflusst, weil sie von der Notenliniendicke abhängen.

```
\new Staff \with {
  \override StaffSymbol.thickness = #3
} \relative {
  f''4 d g, e
}
```



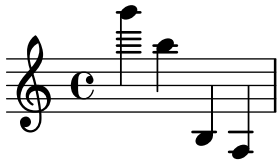
Die Dicke der Hilfslinien (engl. ledger lines) kann allerdings auch unabhängig von der Notenliniendicke verändert werden. Die zwei Zahlen, die nötig sind, sind Faktoren, mit denen die Notenlinien-Dicke und der Notenlinienabstand multipliziert werden. Die Addition beider Werte ergibt die Dicke der Hilfslinien.

```
\new Staff \with {
  \override StaffSymbol.thickness = #2
  \override StaffSymbol.ledger-line-thickness = #'(0.5 . 0.4)
} \relative {
  f'''4 a, a,, f
}
```
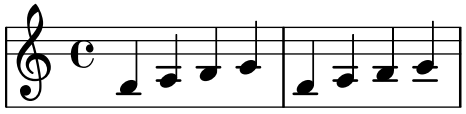


Die vertikale Position der Hilfslinien kann verändert werden:

```
\new Staff \with {
  \override StaffSymbol.ledger-positions = #'(-3 -2 -1 2 5 6)
} \relative {
  f'''4 a, a,, f
}
```

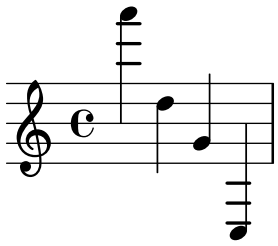


Zusätzliche Hilfslinien können auch innerhalb des Systems gesetzt werden, wenn sie von einem selbsdefinierten Notensystem genötigt werden. Das Beispiel zeigt die Standardposition der Hilfslinien wenn die explizite Position mit `legder-position` nicht definiert ist. Der Befehl `stopStaff` wird benötigt, damit der Befehl sich auf das gesamte System (`StaffSymbol`) auswirkt.



Der Abstand zwischen Notenlinien kann verändert werden. Diese Einstellung wirkt sich auch auf den Abstand der Hilfslinien aus.

```
\new Staff \with {
  \override StaffSymbol.staff-space = #1.5
} \relative {
  f'''4 d, g, e,
}
```
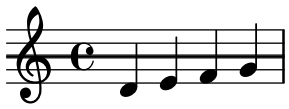


Ausgewählte Schnipsel

Eine Linie des Notensystems dicker als die anderen machen

Für den pädagogischen Einsatz kann eine Linie des Notensystems dicker gezeichnet werden (z. B. die Mittellinie, oder um den Schlüssel hervorzuheben). Das ist möglich, indem man zusätzliche Linien sehr nahe an der Linie, die dicker erscheinen soll, einfügt. Dazu wird die `line-positions`-Eigenschaft herangezogen.

```
{
  \override Staff.StaffSymbol.line-positions =
    #'(-4 -2 -0.2 0 0.2 2 4)
  d'4 e' f' g'
}
```



Siehe auch

Glossar: Abschnitt “line” in *Glossar*, Abschnitt “ledger line” in *Glossar*, Abschnitt “staff” in *Glossar*.

Notationsreferenz: Abschnitt 1.3.1 [Notenschlüssel], Seite 19.
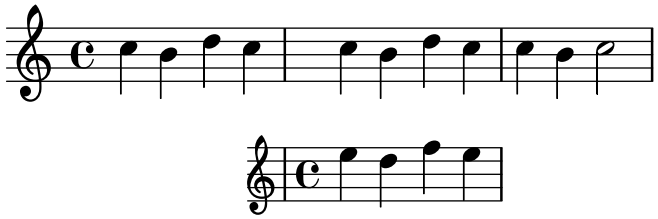
Schnipsel: Abschnitt “Staff notation” in *Schnipsel*.

Referenz der Interna: Abschnitt “StaffSymbol” in *Referenz der Interna*, Abschnitt “staff-symbol-interface” in *Referenz der Interna*.

6.2.2 Ossia-Systeme

Ossia-Systeme können gesetzt werden, indem zwei gleichzeitige Notensysteme an der entsprechenden Position erstellt werden:

```
\new Staff \relative {
  c''4 b d c
  <<
    { c4 b d c }
    \new Staff { e4 d f e }
  >>
  c4 b c2
}
```

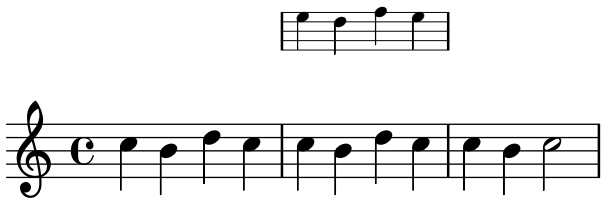


Dieses Beispiel ist aber normalerweise nicht erwünscht. Um Ossia-Systeme zu setzen, die sich über dem eigentlichen System befinden, keine Takt- und Schlüsselangaben haben und kleiner gesetzt sind, müssen einige Optimierungen angewendet werden. Im Handbuch zum Lernen wird eine Technik vorgestellt, mit der das gewünschte Ergebnis erreicht werden kann, beginnend in Abschnitt "Musikalische Ausdrücke ineinander verschachteln" in *Handbuch zum Lernen*.

Das Beispiel unten setzt die `alignAboveContext`-(oberhalbAusrichtenKontext)-Eigenschaft ein, um den Ossia-Abschnitt auszurichten. Diese Methode bietet sich an, wenn nur einige Ossia-Systeme benötigt werden.

```
\new Staff = main \relative {
  c''4 b d c
  <<
    { c4 b d c }

    \new Staff \with {
      \remove Time_signature_engraver
      alignAboveContext = "main"
      fontSize = #-3
      \override StaffSymbol.staff-space = #(magstep -3)
      \override StaffSymbol.thickness = #(magstep -3)
      firstClef = ##f
    }
    { e4 d f e }
  >>
  c4 b c2
}
```
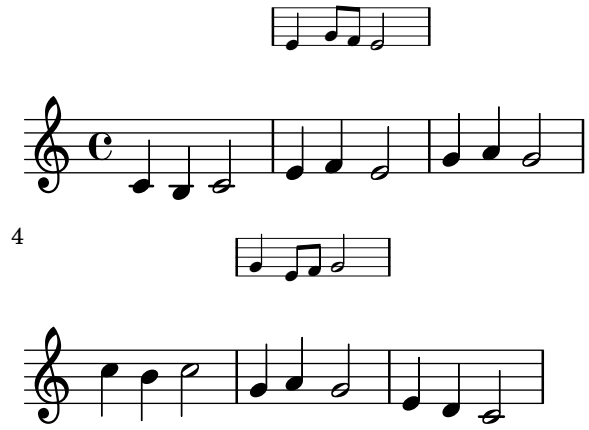


Wenn mehrere isolierte Ossia-Systeme gebraucht werden, kann es günstiger sein, einen leeren `Staff`-Kontext mit einer spezifischen *Kontextidentifikation* zu erstellen. Die Ossia-Abschnitte werden dann erstellt, indem dieser Kontext *aufgerufen* wird und mit `\startStaff` und `\stopStaff` an den richtigen Stellen sichtbar gemacht wird. Der Vorteil dieser Methode zeigt sich, wenn man längere Stücke setzt.

```
<<
  \new Staff = ossia \with {
    \remove Time_signature_engraver
    \hide Clef
    fontSize = #-3
    \override StaffSymbol.staff-space = #(magstep -3)
    \override StaffSymbol.thickness = #(magstep -3)
```

```
  }
  { \stopStaff s1*6 }

  \new Staff \relative {
    c'4 b c2
    <<
      { e4 f e2 }
      \context Staff = ossia {
        \startStaff e4 g8 f e2 \stopStaff
      }
    >>
    g4 a g2 \break
    c4 b c2
    <<
      { g4 a g2 }
      \context Staff = ossia {
        \startStaff g4 e8 f g2 \stopStaff
      }
    >>
    e4 d c2
  }
>>
```

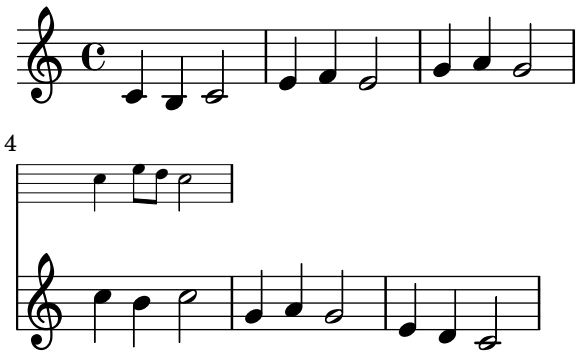


Man kann auch den \Staff \RemoveEmptyStaves-Befehl einsetzen, um Ossia-Systeme zu erstellen. Diese Methode eignet sich am besten, wenn nach dem Ossia sofort ein Zeilenumbruch erfolgt. Mehr Information zu \Staff \RemoveEmptyStaves findet sich in Abschnitt 6.2.3 [Systeme verstecken], Seite 198.

```
<<
  \new Staff = ossia \with {
    \remove Time_signature_engraver
    \hide Clef
    fontSize = #-3
    \override StaffSymbol.staff-space = #(magstep -3)
    \override StaffSymbol.thickness = #(magstep -3)
  } \relative {
    R1*3
    c''4 e8 d c2
  }
  \new Staff \relative {
```

```
    c'4 b c2
    e4 f e2
    g4 a g2 \break
    c4 b c2
    g4 a g2
    e4 d c2
  }
>>

\layout {
  \context {
    \Staff \RemoveEmptyStaves
    \override VerticalAxisGroup.remove-first = ##t
  }
}
```



Ausgewählte Schnipsel

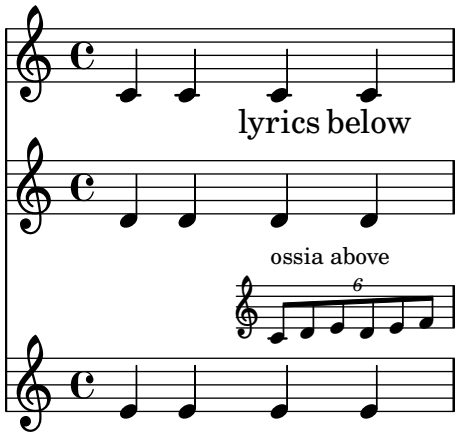
Gesangstext und Ossia vertikal ausrichten

Dieser Schnipsel zeigt, wie man die Kontexteigenschaften `alignBelowContext` und `alignAboveContext` benutzen kann, um die Positionierung von Gesangstext und Ossia-Abschnitten zu kontrollieren.

```
\relative c' <<
  \new Staff = "1" { c4 c c c }
  \new Staff = "2" { d4 d d d }
  \new Staff = "3" { e4 e e e }

  { \skip 2
    <<
      \lyrics {
        \set alignBelowContext = "1"
        lyrics4 below
      }
      \new Staff \with {
        alignAboveContext = "3"
        fontSize = -2
        \override StaffSymbol.staff-space = #(magstep -2)
        \remove "Time_signature_engraver"
        \override VerticalAxisGroup.staff-staff-spacing =
          #'((minimum-distance . 0)
             (basic-distance . 0)
```

```
                (padding . 1))
        } {
          \tuplet 6/4 {
            \override TextScript.padding = 2
            c8[^"ossia above" d e d e f]
          }
        }
      >>
    }
>>
```



Siehe auch

Glossar: Abschnitt “ossia” in *Glossar*, Abschnitt “staff” in *Glossar*, Abschnitt “Frenched staff” in *Glossar*.

Handbuch zum Lernen: Abschnitt “Musikalische Ausdrücke ineinander verschachteln” in *Handbuch zum Lernen*, Abschnitt “Größe von Objekten” in *Handbuch zum Lernen*, Abschnitt “Länge und Dicke von Objekten” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 6.2.3 [Systeme verstecken], Seite 198.

Schnipsel: Abschnitt “Staff notation” in *Schnipsel*.

Referenz der Interna: Abschnitt “StaffSymbol” in *Referenz der Interna*.

6.2.3 Systeme verstecken

Die Notenlinien können entfernt werden, indem der `Staff_symbol_engraver` aus dem `Staff`-Kontext entfernt wird. Alternativ kann auch `\stopStaff` eingesetzt werden.

```
\new Staff \with {
  \remove Staff_symbol_engraver
}
\relative { a''8 f e16 d c b a2 }
```

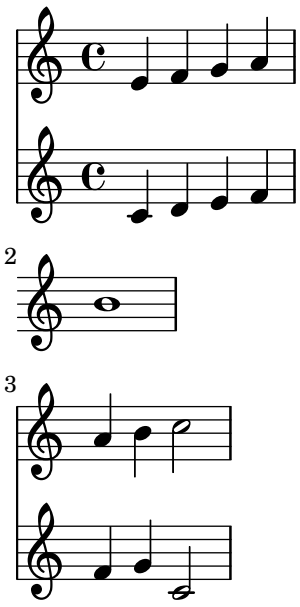


Leere Systeme können versteckt werden, wenn der `\Staff \RemoveEmptyStaves`-Befehl im `\layout`-Abschnitt benutzt wird. In großen Orchesterpartituren wird dies oft verwendet, um die leeren Systeme von gerade nicht spielenden Instrumenten zu verstecken. In der Standardeinstellung werden alle leeren Notenzeilen außer die des ersten Systems entfernt.

Achtung: Eine Notenzeile gilt als leer, wenn sie nur Ganztaktpausen, Pausen, unsichtbare Noten, `\skip`-Befehle oder eine Kombination der drei enthält.

```
\layout {
  \context {
    \Staff \RemoveEmptyStaves
  }
}

\relative <<
  \new Staff {
    e'4 f g a \break
    b1 \break
    a4 b c2
  }
  \new Staff {
    c,4 d e f \break
    R1 \break
    f4 g c,2
  }
>>
```



`\Staff \RemoveEmptyStaves` kann auch eingesetzt werden, um Ossiaabschnitte zu erstellen. Zu Einzelheiten, siehe Abschnitt 6.2.2 [Ossia-Systeme], Seite 194.

Der `\VaticanaStaff \RemoveEmptyStaves`-Befehl kann benutzt werden, um leere Takte in Notation der Alten Musik zu entfernen. Gleichermaßen kann `\RhythmicStaff \RemoveEmptyStaves` eingesetzt werden, um leere Takte in einem `RhythmicStaff`-Kontext zu entfernen.

Vordefinierte Befehle

`\Staff \RemoveEmptyStaves`, `\VaticanaStaff \RemoveEmptyStaves`, `\RhythmicStaff \RemoveEmptyStaves`.

Ausgewählte Schnipsel

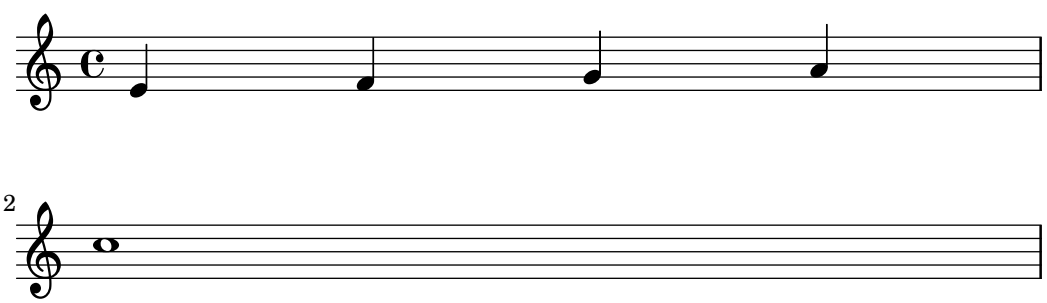
Die erste leere Notenzeile auch entfernen

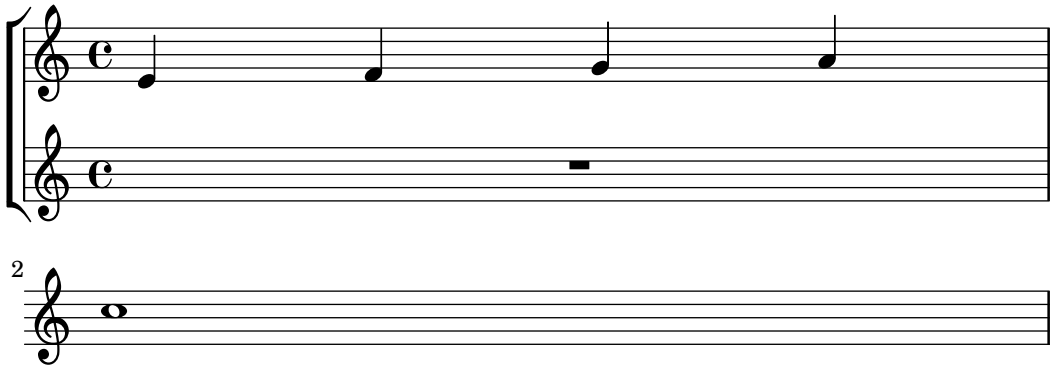
Ein leeres Notensystem kann auch aus der ersten Zeile einer Partitur entfernt werden, indem die Eigenschaft `remove-first` der `VerticalAxisGroup`-Eigenschaft eingesetzt wird. Das kann man global in einer `\layout`-Umgebung oder lokal in dem bestimmten Notensystem machen, das entfernt werden soll. In letzterem Fall muss man den Kontext angeben.

Das untere Notensystem der zweiten Systemgruppe wird nicht entfernt, weil in die Einstellungen in dem Schnipsel nur für das eine Notensystem gültig sind.

```
\layout {
  \context {
    \Staff \RemoveEmptyStaves
    % To use the setting globally, uncomment the following line:
    % \override VerticalAxisGroup.remove-first = ##t
  }
}
\new StaffGroup <<
  \new Staff \relative c' {
    e4 f g a \break
    c1
  }
  \new Staff {
    % To use the setting globally, comment this line,
    % uncomment the line in the \layout block above
    \override Staff.VerticalAxisGroup.remove-first = ##t
    R1 \break
    R
  }
>>

\new StaffGroup <<
  \new Staff \relative c' {
    e4 f g a \break
    c1
  }
  \new Staff {
    R1 \break
    R
  }
>>
```





Siehe auch

Glossar: Abschnitt “Frenched staff” in *Glossar*.

Handbuch zum Lernen: Abschnitt “Sichtbarkeit und Farbe von Objekten” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 32.5 [Die Standardeinstellungen von Kontexten ändern], Seite 588, Abschnitt 6.2.1 [Das Notensystem], Seite 191, Abschnitt 6.2.2 [Ossia-Systeme], Seite 194, Abschnitt 7.1.3 [Unsichtbare Noten], Seite 217, Abschnitt 35.6 [Sichtbarkeit von Objekten], Seite 619.

Schnipsel: Abschnitt “Staff notation” in *Schnipsel*.

Referenz der Interna: Abschnitt “ChordNames” in *Referenz der Interna*, Abschnitt “FiguredBass” in *Referenz der Interna*, Abschnitt “Lyrics” in *Referenz der Interna*, Abschnitt “Staff” in *Referenz der Interna*, Abschnitt “VerticalAxisGroup” in *Referenz der Interna*, Abschnitt “Staff_symbol_engraver” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Wenn man den `Staff_symbol_engraver` entfernt, werden auch die Taktlinien entfernt. Wenn eine sichtbare Taktlinie angefordert wird, kann es zu Formatierungsfehlern kommen. In diesem Fall sollten folgende Befehle eingesetzt werden, anstatt den Engraver zu entfernen:

```
\omit StaffSymbol
\override NoteHead.no-ledgers = ##t
```

Zu den bekannten Fehlern und Warnungen, die mit `\Staff \RemoveEmptyStaves` zusammenhängen, siehe Abschnitt 32.5 [Die Standardeinstellungen von Kontexten ändern], Seite 588.

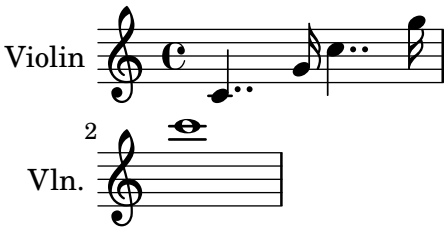
6.3 Orchesterstimmen erstellen

Dieser Abschnitt zeigt, wie man Tempo-Anweisungen und Instrumentenbezeichnungen einfügt. Es werden auch Möglichkeiten vorgestellt, andere Stimmen zu zitieren und Stichnoten zu formatieren.

6.3.1 Instrumentenbezeichnungen
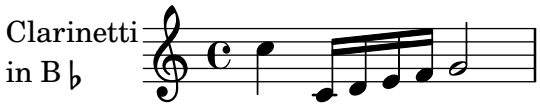
Instrumentbezeichnungen können an der linken Seite von Notensystemen im `Staff`-, `PianoStaff`-, `StaffGroup`, `GrandStaff` und `ChoirStaff`-Kontext gesetzt werden. Der Wert von `instrumentName` wird für das erste System eingesetzt, der Wert von `shortInstrumentName` für alle weiteren Systeme.

```
\new Staff \with {
  instrumentName = "Violin "
  shortInstrumentName = "Vln. "
} \relative {
  c'4.. g'16 c4.. g'16 \break | c1 |
}
```



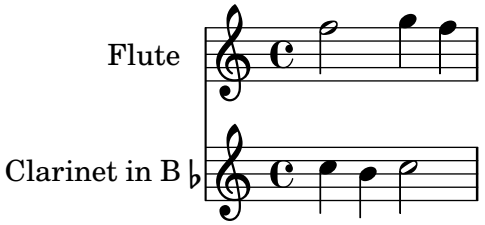
Mit dem Textbeschriftungsmodus (\markup) können auch komplizierte Instrumentenbezeichnungen erstellt werden:

```
\new Staff \with {
  instrumentName = \markup {
    \column { "Clarinetti"
      \line { "in B" \smaller \flat }
    }
  }
} \relative {
  c''4 c,16 d e f g2
}
```



Wenn zwei oder mehr Systeme gruppiert werden, werden die Instrumentenbezeichnungen automatisch zentriert. Um auch mehrzeilige Instrumentenbezeichnungen zentriert zu setzen, muss \center-column benutzt werden:

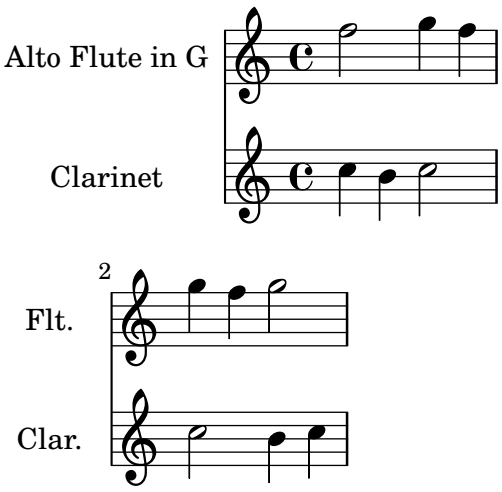
```
<<
  \new Staff \with {
    instrumentName = "Flute"
  }
  { f2 g4 f }
  \new Staff \with {
    instrumentName = \markup {
      \center-column { "Clarinet" }
      \line { "in B" \smaller \flat }
    }
  }
  { c4 b c2 }
>>
```



Wenn die Instrumentenbezeichnung zu lang ist, kann es vorkommen, dass die Bezeichnungen in einer Gruppe nicht zentriert werden. Um dennoch eine Zentrierung zu erhalten, müssen die Werte des Einzugs (`indent` und `short-indent`) vergrößert werden. Zu Einzelheiten siehe Abschnitt 25.5.3 [\paper-Variablen für Verschiebungen und Einrückungen], Seite 528.

```
<<
  \new Staff \with {
    instrumentName = "Alto Flute in G"
    shortInstrumentName = "Flt."
  } \relative {
    f''2 g4 f \break
    g4 f g2
  }
  \new Staff \with {
    instrumentName = "Clarinet"
    shortInstrumentName = "Clar."
  } \relative {
    c''4 b c2 \break
    c2 b4 c
  }
>>

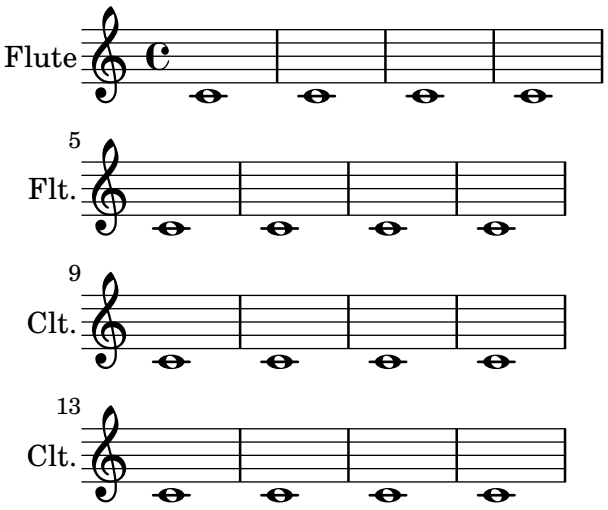
\layout {
  indent = 3.0\cm
  short-indent = 1.5\cm
}
```



Um Instrumentenbezeichnungen zu anderen Kontexten (wie etwa `ChordNames` or `FiguredBass`) hinzuzufügen, muss der `Instrument_name_engraver` dem entsprechenden Kontext hinzugefügt werden. Zu Einzelheiten siehe Abschnitt 32.4 [Umgebungs-Plugins verändern], Seite 586.

Die kurzen Instrumentenbezeichnungen können mitten in einer Partitur geändert werden. Die Lange Bezeichnung `instrumentName` wird nur bei ihrem ersten Auftreten gesetzt und spätere Änderungen nicht berücksichtigt.

```
\new Staff \with {
  instrumentName = "Flute"
  shortInstrumentName = "Flt."
}
{
  c1 c c c \break
  c1 c c c \break
  \set Staff.instrumentName = "Clarinet"
  \set Staff.shortInstrumentName = "Clt."
  c1 c c c \break
  c1 c c c \break
}
```

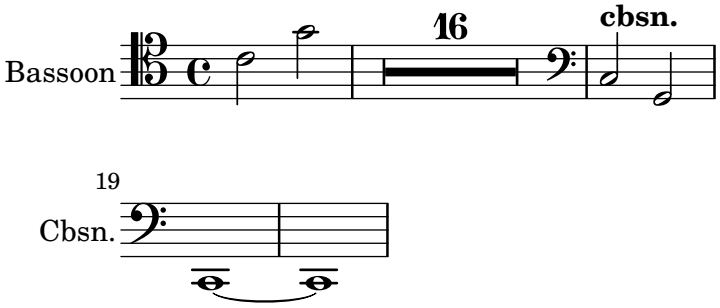


Wenn das Instrument gewechselt werden soll, kann der Befehl `\addInstrumentDefinition` in Begleitung von `\instrumentSwitch` benutzt werden, um eine detaillierte Auflistung aller notwendigen Änderungen für den Wechsel zu definieren. Der `\addInstrumentDefinition`-Befehl hat zwei Argumente: eine Identifikation und eine Assoziationsliste von Kontexteigenschaften und Werten, die für dieses Instrument benutzt werden müssen. Der Befehl muss sich auf der höchsten Ebene in der Eingabedatei befinden. `\instrumentSwitch` wird dann benutzt, um den Wechsel vorzunehmen:

```
\addInstrumentDefinition "contrabassoon"
  #`((instrumentTransposition . ,(ly:make-pitch -1 0 0))
     (shortInstrumentName . "Cbsn.")
     (clefGlyph . "clefs.F")
     (middleCPosition . 6)
     (clefPosition . 2)
     (instrumentCueName . ,(make-bold-markup "cbsn."))
     (midiInstrument . "bassoon"))

\new Staff \with {
  instrumentName = "Bassoon"
}
\relative c' {
  \clef tenor
  \compressEmptyMeasures
  c2 g'
  R1*16
```

```
  \instrumentSwitch "contrabassoon"
  c,,2 g \break
  c,1 ~ | 1
}
```



Siehe auch

Notationsreferenz: Abschnitt 25.5.3 [\paper-Variablen für Verschiebungen und Einrückungen], Seite 528, Abschnitt 32.4 [Umgebungs-Plugins verändern], Seite 586.

Schnipsel: Abschnitt “Staff notation” in *Schnipsel*.

Referenz der Interna: Abschnitt “InstrumentName” in *Referenz der Interna*, Abschnitt “PianoStaff” in *Referenz der Interna*, Abschnitt “Staff” in *Referenz der Interna*.

6.3.2 Andere Stimmen zitieren

Es kommt sehr oft vor, dass eine Orchesterstimme die gleichen Noten wie eine andere spielt. So können etwa die ersten und zweiten Geigen für eine bestimmte Passage die gleichen Noten haben. In LilyPond kann man das erreichen, indem eine Stimme von der anderen *zitiert*, sodass man die Noten für die zweite Stimme nicht noch einmal eingeben muss.

Der `\addQuote`-Befehl, auf höchster Ebene in der Datei, definiert einen Notenabschnitt, aus dem zitiert werden kann.
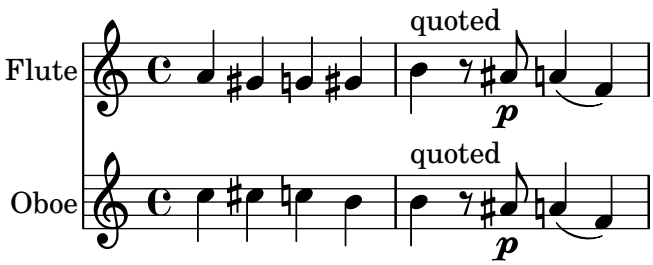
Der `\quoteDuring`-Befehl wird benutzt, um den Punkt anzuzeigen, an dem das Zitat beginnt. Er benötigt zwei Argumente: die Bezeichnung der zitierten Stimme, wie vorher mit `\addQuote` definiert, und einen musikalischen Ausdruck für die Dauer des Zitates.

```
fluteNotes = \relative {
  a'4 gis g gis | b4^"quoted" r8 ais\p a4( f)
}

oboeNotes = \relative {
  c''4 cis c b \quoteDuring "flute" { s1 }
}

\addQuote "flute" { \fluteNotes }

\score {
  <<
    \new Staff \with { instrumentName = "Flute" } \fluteNotes
    \new Staff \with { instrumentName = "Oboe" } \oboeNotes
  >>
}
```
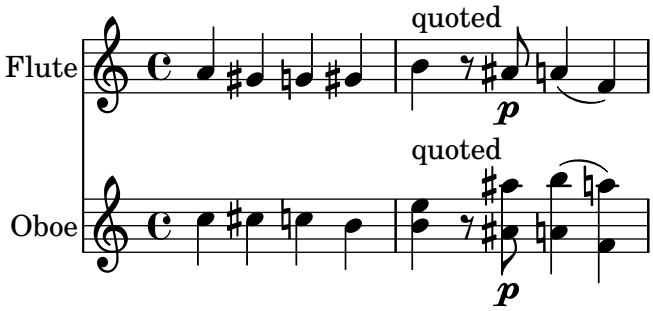


Wenn der musikalische Ausdruck, der mit dem \quoteDuring-Befehl benutzt wird, antelle von unsichtbare Noten oder Ganztaktpausen etwa Noten enthält, wird eine polyphone Stelle begonnen, was meistens nicht erwünscht ist:

```
fluteNotes = \relative {
  a'4 gis g gis | b4^"quoted" r8 ais\p a4( f)
}

oboeNotes = \relative {
  c''4 cis c b \quoteDuring "flute" { e4 r8 ais b4 a }
}

\addQuote "flute" { \fluteNotes }

\score {
  <<
    \new Staff \with { instrumentName = "Flute" } \fluteNotes
    \new Staff \with { instrumentName = "Oboe" } \oboeNotes
  >>
}
```
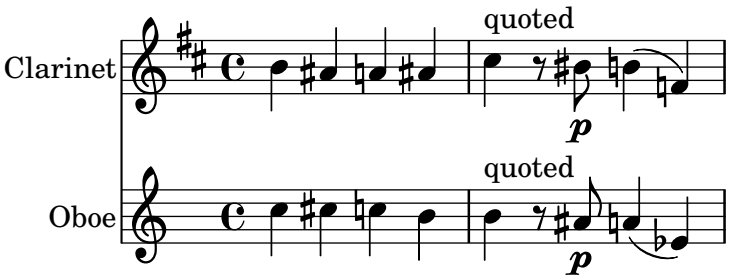


Der \quoteDuring-Befehl benützt die Einstellungen des \transposition-Befehls beider Stimmen, um Noten für die zitierende Stimme zu produzieren, die das gleiche klingende C wie die zitierte Stimme haben.

```
clarinetNotes = \relative c'' {
  \transposition bes
  \key d \major
  b4 ais a ais | cis4^"quoted" r8 bis\p b4( f)
}

oboeNotes = \relative {
  c''4 cis c b \quoteDuring "clarinet" { s1 }
}

\addQuote "clarinet" { \clarinetNotes }
```

```
\score {
  <<
    \new Staff \with { instrumentName = "Clarinet" } \clarinetNotes
    \new Staff \with { instrumentName = "Oboe" } \oboeNotes
  >>
}
```
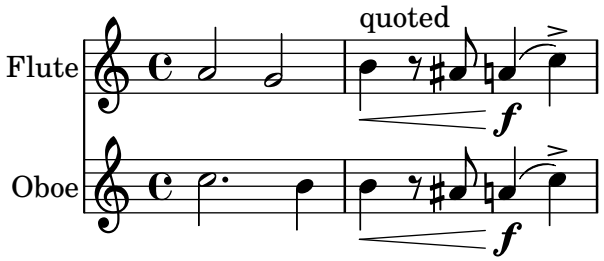


Standardmäßig werden mit den zitierten Noten auch alle Artikulationen, Dynamik, Beschriftungen usw. übernommen. Es ist aber möglich auszuwählen, welche Objekte der zitierten Noten dargestllt werden. Das geschieht mit der quotedEventTypes-Kontexteigenschaft.

```
fluteNotes = \relative {
  a'2 g2 |
  b4\<^"quoted" r8 ais a4\f( c->)
 }

oboeNotes = \relative {
  c''2. b4 |
  \quoteDuring "flute" { s1 }
}

\addQuote "flute" { \fluteNotes }

\score {
  <<
    \set Score.quotedEventTypes = #'(note-event articulation-event
                                     crescendo-event rest-event
                                     slur-event dynamic-event)
    \new Staff \with { instrumentName = "Flute" } \fluteNotes
    \new Staff \with { instrumentName = "Oboe" } \oboeNotes
  >>
 }
```



Zitate können auch mit Marken (engl. tag) versehen werden. Siehe auch Abschnitt 21.2.2 [Marken benutzen], Seite 491.

Siehe auch

Notationsreferenz: Abschnitt 1.3.4 [Transposition von Instrumenten], Seite 25, Abschnitt 21.2.2 [Marken benutzen], Seite 491.

Installierte Dateien: `scm/define-event-classes.scm`.

Schnipsel: Abschnitt “Staff notation” in *Schnipsel*.

Referenz der Interna: Abschnitt “Music classes” in *Referenz der Interna*, Abschnitt “Quote-Music” in *Referenz der Interna*, Abschnitt “Voice” in *Referenz der Interna*.

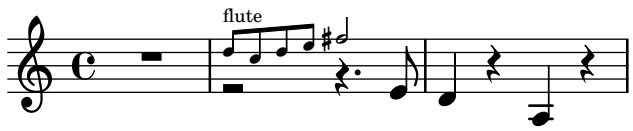
Bekannte Probleme und Warnungen

Nur der Inhalt der ersten Stimme innerhalb eines `\addQuote`-Befehls wird für das Zitat herangezogen. Wenn der zitierte Ausdruck also `\new` oder `\context Voice`-Befehle enthält, werden deren Inhalte nicht zitiert. Zitieren von Ziernoten und Vorschläge wird von LilyPond nicht unterstützt und kann dazu führen, dass LilyPond abstürzt. Wenn geschachtelte Triolen zitiert werden, ist das Notenbild unter Umständen sehr schlecht.

6.3.3 Stichnoten formatieren
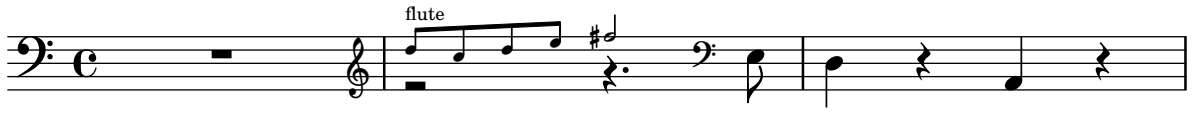
Die einfachste Art, Stichnoten zu erstellen, ist es, einen `CueVoice`-Kontext in der Stimme zu erstellen.

```
\relative {
  R1
  <<
    { e'2\rest r4. e8 }
    \new CueVoice {
      \stemUp d'8^"flute" c d e fis2
    }
  >>
  d,4 r a r
}
```
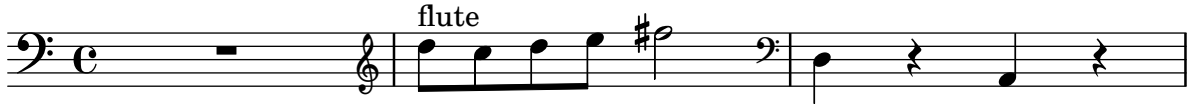


Der Befehl `\cueClef` kann auch in einem expliziten `CueVoice`-Kontext eingesetzt werden, wenn ein Schlüsselwechsel nötig ist. Hiermit wird ein Schlüssel entsprechender Größe für die Stichnoten ausgegeben. Der Befehl `\cueClefUnset` kann dann eingesetzt werden, um wieder zum ursprünglichen Schlüssel zurückzukehren, wiederum in der richtigen Größe:

```
\relative {
  \clef "bass"
  R1
  <<
    { e'2\rest r4. \cueClefUnset e,8 }
    \new CueVoice {
      \cueClef "treble" \stemUp d''8^"flute" c d e fis2
    }
  >>
  d,,4 r a r
}
```



Die Befehle \cueClef und \cueClefUnset können auch ohne eine CueVoice-Umgebung eingesetzt werden:

```
\relative {
  \clef "bass"
  R1
  \cueClef "treble"
  d''8^"flute" c d e fis2
  \cueClefUnset
  d,,4 r a r
}
```



Für kompliziertere Stichnotenbehandlung, etwa mit Transposition, oder um Stichnoten aus unterschiedlichen Stimmen einzufügen, können die Befehle \cueDuring oder \cueDuringWithClef eingesetzt werden. Sie stellen eine spezielle Form von \quoteDuring dar, siehe auch Abschnitt 6.3.2 [Andere Stimmen zitieren], Seite 205.

Die Syntax lautet:

\cueDuring #*Zitatbezeichnung* #*Richtung Noten*

sowie

\cueDuringWithClef #*Zitatbezeichnung* #*Richtung* #*Schlüssel* #*Noten*

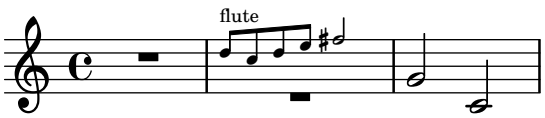
Die Noten der entprechenden Takten von *Zitatbezeichnung* wird dem CueVoice-Kontext hinzugefügt und erscheint gleichzeitig mit *Noten*, wodurch eine polyphone Situation entsteht. Die *Richtung* kann entweder UP oder DOWN sein, womit die zitierten Noten entweder als erste oder als zweite Stimme in einem System gesetzt werden.

```
fluteNotes = \relative {
  r2. c''4 | d8 c d e fis2 | g2 d |
}

oboeNotes = \relative c'' {
  R1
  \new CueVoice { \set instrumentCueName = "flute" }
  \cueDuring "flute" #UP { R1 }
  g2 c,
}

\addQuote "flute" { \fluteNotes }

\new Staff {
  \oboeNotes
}
```
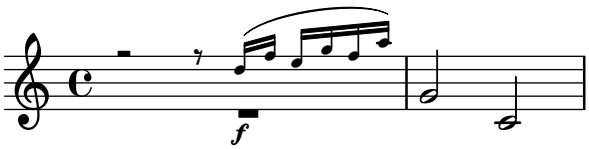


Es ist möglich anzupassen, welche Objekte der Notation von \cueDuring zitiert werden, indem man die quotedCueEventTypes-Eigenschaft verändert. Ihr Standardwert ist '(note-event rest-event tie-event beam-event +tuplet-span-event); somit werden also nur Noten, Pausen, Bindebögen, Balken und N-tolen zitiert, nicht aber Artikulationen, Dynamik, Beschriftung usw.

> **Achtung:** Wenn eine Voice-Umgebung mit \cueDuring beginnt, wie im folgenden Beispiel, muss die Voice-Umgebung explizit erstellt werden, weil sonst der gesamte musikalische Ausdruck zum CueVoice-Kontext gehören würde.

```
oboeNotes = \relative {
  r2 r8 d''16(\f f e g f a)
  g8 g16 g g2.
}
\addQuote "oboe" { \oboeNotes }

\new Voice \relative c'' {
  \set Score.quotedCueEventTypes = #'(note-event rest-event tie-event
                                      beam-event tuplet-span-event
                                      dynamic-event slur-event)
  \cueDuring "oboe" #UP { R1 }
  g2 c,
}
```
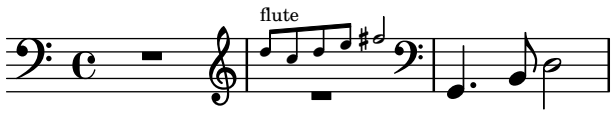


Die Bezeichnung des gerade spielenden Instruments in den Stichnoten kann gesetzt werden, indem man die instrumentCueName-Eigenschaft in einen temporären CueVoice-Kontext setzt. Die Platzierung und der Stil von instrumentCueName wird durch das \instrumentSwitch-Objekt kontrolliert, siehe Abschnitt 6.3.1 [Instrumentenbezeichnungen], Seite 201. Wenn die Stichnoten einen Schlüsselwechsel erfordern, kann dieser manuell hervorgerufen werden, aber der originale Schlüssel muss auch manuell am Ende der Stichnoten wieder hergestellt werden.

```
fluteNotes = \relative {
  r2. c''4 d8 c d e fis2 g2 d2
}

bassoonNotes = \relative c {
  \clef bass
  R1
  \clef treble
  \new CueVoice { \set instrumentCueName = "flute" }
  \cueDuring "flute" #UP { R1 }
  \clef bass
  g4. b8 d2
}

\addQuote "flute" { \fluteNotes }
```

```
\new Staff {
  \bassoonNotes
}
```
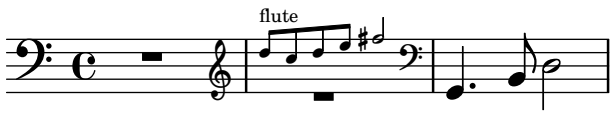


Alternativ kann auch die Funktion \cueDuringWithClef eingesetzt werden. Dieser Befehl erhält zusätzlcih ein Argument, das den Schlüsselwechsel anzeigt, den man für die Stichnoten braucht. Der originale Schlüssel wird automatisch wieder hergesetllt.

```
fluteNotes = \relative {
  r2. c''4 d8 c d e fis2 g2 d2
}

bassoonNotes = \relative c {
  \clef bass
  R1
  \new CueVoice { \set instrumentCueName = "flute" }
  \cueDuringWithClef "flute" #UP "treble" { R1 }
  g4. b8 d2
}

\addQuote "flute" { \fluteNotes }

\new Staff {
  \bassoonNotes
}
```



Genauso wie \quoteDuring berücksichtigt auch \cueDuring Transpositionen. Stichnoten werden auf den Tonhöhen erstellt, die für das Instrument geschrieben würden, in dessen Noten sie gesetzt werden, um die klingenden Töne des Quelleninstruments zu produzieren.

Um Stichnoten anders zu transponieren, muss \transposedCueDuring benutzt werden. Dieser Befehl braucht ein zusätzliches Argument, um (in absolutem Modus) die gedruckte Tonhöhe vorzugeben, mit der das das eingestrichene C dargestellt werden soll. Das ist nützlich, wenn man Stichnoten von einem Instrument mit einem vollständig anderen Register benutzt:

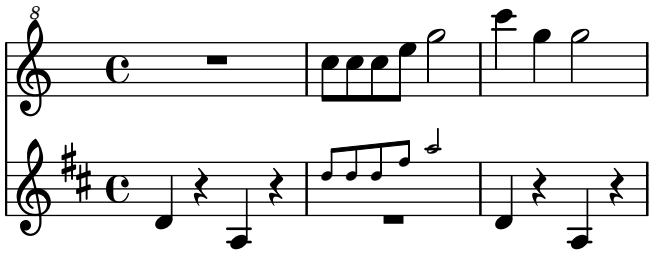
```
piccoloNotes = \relative {
  \clef "treble^8"
  R1
  c'''8 c c e g2
  c4 g g2
}

bassClarinetNotes = \relative c' {
  \key d \major
  \transposition bes,
  d4 r a r
  \transposedCueDuring "piccolo" #UP d { R1 }
```

```
  d4 r a r
}

\addQuote "piccolo" { \piccoloNotes }

<<
  \new Staff \piccoloNotes
  \new Staff \bassClarinetNotes
>>
```



Der \killCues-Befehl entfernt Stichnoten aus einem musikalischen Ausdruck, sodass derselbe musikalische Ausdruck für die Partitur und für eine Stimme mit Stichnoten eingesetzt werden kann. Der Befehl \killCues entfernt nur Noten und Ereignisse, die durch \cueDuring zitiert wurden. Andere Beschriftungen in Verbindung mit Stichnoten, wie etwa Schlüsselwechsel und Marken, die das Ursprungsinstrument anzeigen, können mit Marken versehen werden, um sie selektiv einzufügen, siehe Abschnitt 21.2.2 [Marken benutzen], Seite 491.

```
fluteNotes = \relative {
  r2. c''4 d8 c d e fis2 g2 d2
}
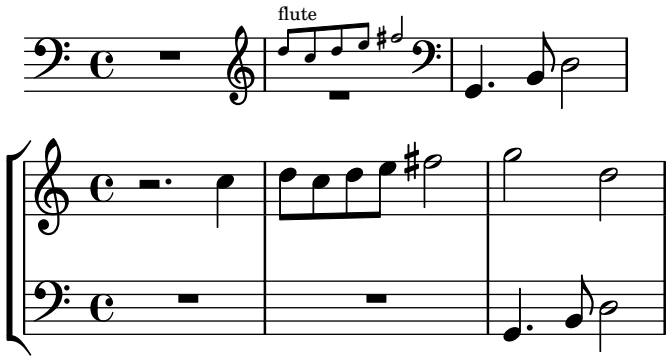
bassoonNotes = \relative c {
  \clef bass
  R1
  \tag #'part {
    \clef treble
    \new CueVoice { \set instrumentCueName = "flute" }
  }
  \cueDuring "flute" #UP { R1 }
  \tag #'part \clef bass
  g4. b8 d2
}

\addQuote "flute" { \fluteNotes }

\new Staff {
  \bassoonNotes
}

\new StaffGroup <<
  \new Staff {
    \fluteNotes
  }
  \new Staff {
    \removeWithTag #'part { \killCues { \bassoonNotes } }
```

```
  }
>>
```



Alternativ können Schlüsselwechsel und Instrumentenbezeichnungen auch in eine Instrument-Definition unter Einsatz des Befehls `\addInstrumentDefinition` zusammengefasst werden, siehe Abschnitt 6.3.1 [Instrumentenbezeichnungen], Seite 201.

Siehe auch

Notationsreferenz: Abschnitt 6.3.2 [Andere Stimmen zitieren], Seite 205, Abschnitt 1.3.4 [Transposition von Instrumenten], Seite 25, Abschnitt 6.3.1 [Instrumentenbezeichnungen], Seite 201, Abschnitt 21.2.2 [Marken benutzen], Seite 491, Abschnitt 1.3.1 [Notenschlüssel], Seite 19, Abschnitt 9.6.3 [Musikalische Stichnoten], Seite 295.

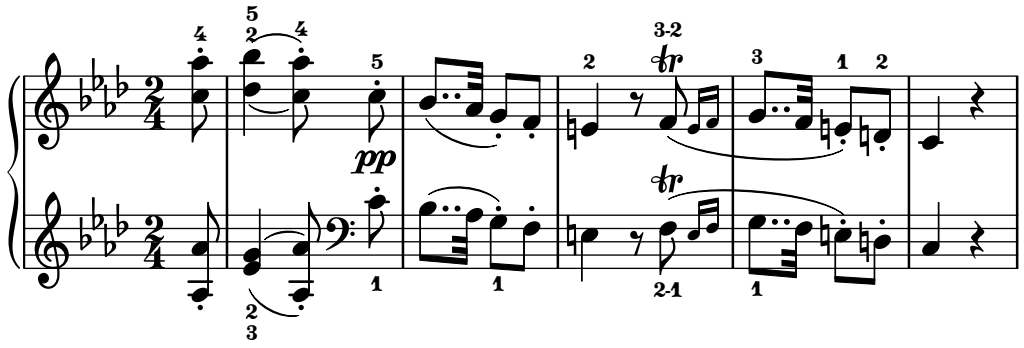
Schnipsel: Abschnitt “Staff notation” in *Schnipsel*.

Referenz der Interna: Abschnitt “CueVoice” in *Referenz der Interna*, Abschnitt “Voice” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Zusammenstöße können bei Benutzung von `\cueDuring` zwischen Pausen der Hauptstimme und den Stichnoten des `CueVoice`-Kontexts auftreten. Wenn `\cueDuringWithClef` oder `\transposedCueDuring` eingesetzt wird, muss das zusätzliche Argument nach dem Zitat und der Richtung erscheinen.

7 Anmerkungen



Dieser Abschnitt zeigt die verschiedenen Möglichkeiten, die Erscheinung der Noten zu ändern und analytische bzw. pädagogische Anmerkungen anzubringen.

7.1 Innerhalb des Systems
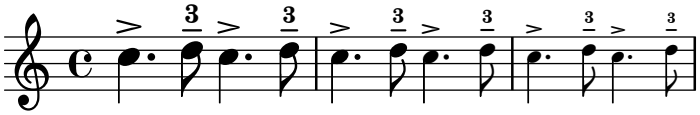
Dieser Abschnitt zeigt, wie man Elemente hervorhebt, die sich innerhalb des Notensystems befinden.

7.1.1 Auswahl der Notations-Schriftgröße

Die Schriftgröße von Notationselementen kann geändert werden. Damit wird allerdings nicht die Größe von veränderlichen Symbolen, wie Balken oder Bögen, geändert.
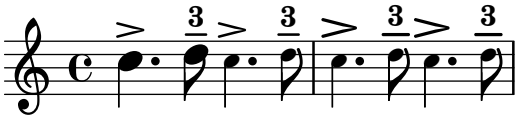
> **Achtung:** Für Schriftgröße von Text, siehe Abschnitt 8.2.2 [Überblick über die wichtigsten Textbeschriftungsbefehle], Seite 234.

```
\huge
c4.-> d8---3
\large
c4.-> d8---3
\normalsize
c4.-> d8---3
\small
c4.-> d8---3
\tiny
c4.-> d8---3
\teeny
c4.-> d8---3
```



Intern wird hiermit die `fontSize`-Eigenschaft gesetzt. Sie wird für alle Layout-Objekte definiert. Der Wert von `font-size` ist eine Zahl, die die Größe relativ zur Standardgröße für die aktuelle Systemhöhe angibt. Jeder Vergrößerungsschritt bedeutet etwa eine Vergrößerung um 12% der Schriftgröße. Mit sechs Schritten wird die Schriftgröße exakt verdoppelt. Die Scheme-Funktion `magstep` wandelt einen Wert von `font-size` in einen Skalierungsfaktor um. Die `font-size`-Eigenschaft kann auch direkt gesetzt werden, so dass sie sich nur auf bestimmte Layoutobjekte bezieht.

```
\set fontSize = #3
c4.-> d8---3
\override NoteHead.font-size = #-4
c4.-> d8---3
\override Script.font-size = #2
c4.-> d8---3
\override Stem.font-size = #-5
c4.-> d8---3
```



Schriftgrößenänderungen werden erreicht, indem man die Design-Schriftgröße nimmt, die der gewünschten am nächsten kommt, und sie dann skaliert. Die Standard-Schriftgröße (für `font-size = #0`) hängt von der Standard-Systemhöhe ab. Für ein Notensystem von 20pt wird eine Schriftgröße von 11pt ausgewählt.

Die `font-size`-Eigenschaft kann nur für die Layoutobjekte gesetzt werden, die Schrift-Dateien benutzen. Das sind die, welche die `font-interface`-Layoutschnittstelle unterstützen.

Vordefinierte Befehle

`\teeny`, `\tiny`, `\small`, `\normalsize`, `\large`, `\huge`.

Siehe auch
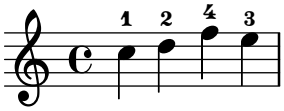
Schnipsel: Abschnitt “Editorial annotations” in *Schnipsel*.

Referenz der Interna: Abschnitt “font-interface” in *Referenz der Interna*.

7.1.2 Fingersatzanweisungen
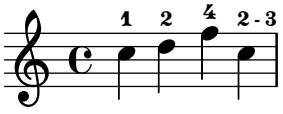
Fingersatzanweisungen können folgenderweise notiert werden: ‘`Note`’-*Zahl*

```
\relative { c''4-1 d-2 f-4 e-3 }
```
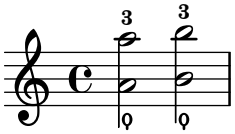


Für Fingerwechsel muss eine Textbeschriftung (markup) benutzt werden:

```
c4-1 d-2 f-4 c^\markup { \finger "2 - 3" }
```
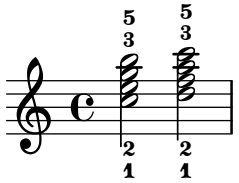


Mit dem Daumen-Befehl (`\thumb`) können die Noten bezeichnet werden, die mit dem Daumen (etwa auf dem Cello) gespielt werden sollen.

```
\relative { <a'_\thumb a'-3>2 <b_\thumb b'-3> }
```



Fingersätze für Akkorde können auch zu einzelnen Noten hinzugefügt werden, indem sie innerhalb der Akkord-Klammer direkt an die Noten angefügt werden.

```
\relative {
  <c''-1 e-2 g-3 b-5>2 <d-1 f-2 a-3 c-5>
}
```



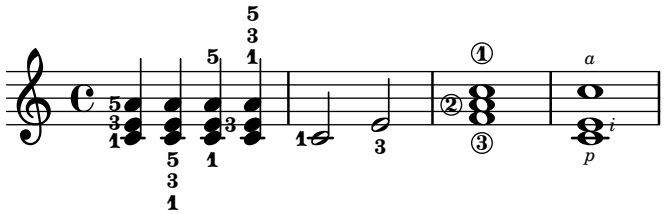
Fingersatzanweisungen können manuell oberhalb des Systems gesetzt werden, siehe Abschnitt 35.2 [Richtung und Platzierung], Seite 611.

Ausgewählte Schnipsel

Position von Fingersatz in Akkorden kontrollieren

Die Position von Fingersatzzahlen kann exakt kontrolliert werden.

```
\relative c' {
  \set fingeringOrientations = #'(left)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(down)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(down right up)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(up)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(left)
  <c-1>2
  \set fingeringOrientations = #'(down)
  <e-3>2
  \set stringNumberOrientations = #'(up left down)
  <f\3 a\2 c\1>1
  \set strokeFingerOrientations = #'(down right up)
  <c\rightHandFinger 1 e\rightHandFinger 2 c'\rightHandFinger 4 >
}
```
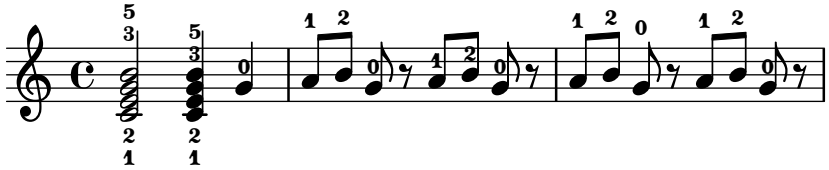


Fingersatz auch innerhalb des Systems setzen

Normalerweise werden vertikal orientierte Fingersatzzahlen außerhalb des Systems gesetzt. Das kann aber verändert werden.

```
\relative c' {
  <c-1 e-2 g-3 b-5>2
  \override Fingering.staff-padding = #'()
  <c-1 e-2 g-3 b-5>4 g'-0
  a8[-1 b]-2 g-0 r
  \override Fingering.add-stem-support = ##f
```

```
  a[-1 b]-2 g-0 r
  \override Fingering.add-stem-support = ##t
  a[-1 b]-2 g-0 r
  \override Fingering.add-stem-support = #only-if-beamed
  a[-1 b]-2 g-0 r
}
```



Siehe auch

Notationsreferenz: Abschnitt 35.2 [Richtung und Platzierung], Seite 611.

Schnipsel: Abschnitt “Editorial annotations” in *Schnipsel*.

Referenz der Interna: Abschnitt “FingeringEvent” in *Referenz der Interna*, Abschnitt “fingering-event” in *Referenz der Interna*, Abschnitt “Fingering_engraver” in *Referenz der Interna*, Abschnitt “New_fingering_engraver” in *Referenz der Interna*, Abschnitt “Fingering” in *Referenz der Interna*.
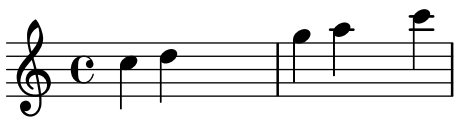
Bekannte Probleme und Warnungen

Standardmäßig ist eine Zahl größer als 9 nicht unterstützt, wenn man die Schreibweise `Note-Zahl` einsetzt.

7.1.3 Unsichtbare Noten
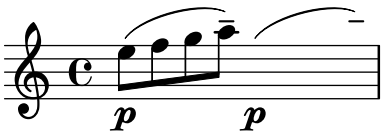
Versteckte (oder unsichtbare oder transparente) Noten können sinnvoll sein, wenn man Notation für den Theorieunterricht oder Kompositionsübungen erstellen will.

```
\relative {
  c''4 d
  \hideNotes
  e4 f
  \unHideNotes
  g a
  \hideNotes
  b
  \unHideNotes
  c
}
```



Notenköpfe, Hälse, Fähnchen und Pausen sind unsichtbar. Balken sind unsichtbar, wenn sie auf einer unsichtbaren Note beginnen. Objekte, die an unsichtbare Noten angehängt werden, sind trotzdem noch sichtbar.

```
\relative c'' {
  e8(\p f g a)--
  \hideNotes
  e8(\p f g a)--
}
```



Vordefinierte Befehle

`\hideNotes`, `\unHideNotes`.

Siehe auch

Handbuch zum Lernen: Abschnitt “Sichtbarkeit und Farbe von Objekten” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 2.2.2 [Unsichtbare Pausen], Seite 59, Abschnitt 35.6 [Sichtbarkeit von Objekten], Seite 619, Abschnitt 6.2.3 [Systeme verstecken], Seite 198.
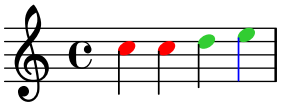
Schnipsel: Abschnitt “Editorial annotations” in *Schnipsel*.

Referenz der Interna: Abschnitt “Note_spacing_engraver” in *Referenz der Interna*, Abschnitt “NoteSpacing” in *Referenz der Interna*.

7.1.4 Farbige Objekte

Einzelnen Objekten können einfach eigene Farben zugewiesen werden. Gültige Farben-Bezeichnungen sind aufgelistet in Abschnitt A.7 [Liste der Farben], Seite 656.

```
\override NoteHead.color = #red
c''4 c''
\override NoteHead.color = #(x11-color 'LimeGreen)
d''
\override Stem.color = #blue
e''
```
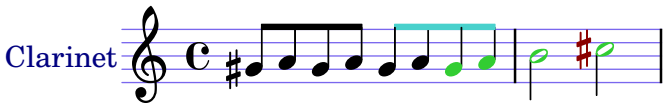


Die ganze Farbpalette, die für X11 definiert ist, kann mit der Scheme-Funktion `x11-color` benutzt werden. Diese Funktion hat ein Argument: entweder ein Symbol in der Form `'FooBar` oder eine Zeichenkette in der Form `"FooBar"`. Die erste Form ist schneller zu schreiben und effizienter. Mit der zweiten Form ist es allerdings möglich, auch Farbbezeichnungen einzusetzen, die aus mehr als einem Wort bestehen.

Wenn `x11-color` die angegebene Farbbezeichnung nicht kennt, wird Schwarz eingesetzt.

```
\relative c'' {
  \override Staff.StaffSymbol.color = #(x11-color 'SlateBlue2)
  \set Staff.instrumentName = \markup {
    \with-color #(x11-color 'navy) "Clarinet"
  }

  gis8 a
  \override Beam.color = #(x11-color "medium turquoise")
```
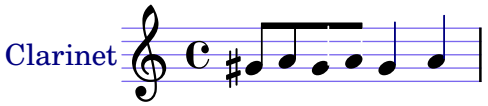
```
  gis a
  \override Accidental.color = #(x11-color 'DarkRed)
  gis a
  \override NoteHead.color = #(x11-color "LimeGreen")
  gis a
  % this is deliberate nonsense; note that the stems remain black
  \override Stem.color = #(x11-color 'Boggle)
  b2 cis
}
```



Exakte RGB-Farben können mit Hilfe der Scheme-Funktion `rgb-color` definiert werden.

```
\relative c'' {
  \override Staff.StaffSymbol.color = #(x11-color 'SlateBlue2)
  \set Staff.instrumentName = \markup {
    \with-color #(x11-color 'navy) "Clarinet"
  }

  \override Stem.color = #(rgb-color 0 0 0)
  gis8 a
  \override Stem.color = #(rgb-color 1 1 1)
  gis8 a
  \override Stem.color = #(rgb-color 0 0 0.5)
  gis4 a
}
```



Siehe auch

Notationsreferenz: Abschnitt A.7 [Liste der Farben], Seite 656, Abschnitt 34.4 [Der `\tweak`-Befehl], Seite 605.

Schnipsel: Abschnitt “Editorial annotations” in *Schnipsel*.

Bekannte Probleme und Warnungen

Eine X11-Farbe hat nicht notwendigerweise exakt denselben Farbton wie eine ähnlich genannte normale Farbe.
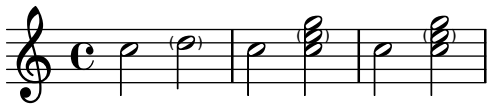
Nicht alle X11-Farben lassen sich am Webbrowser erkennen, d. h. der Unterschied etwa zwischen `LimeGreen` und `ForestGreen` wird eventuell nicht dargestellt. Für die Benutzung im Internet wird die Benutzung von einfachen Farben nahegelegt (z. B. `blue`, `green`, `red`).

Noten in Akkorden können nicht mit `\override` eingefärbt werden, dazu muss `\tweak` benutzt werden. Siehe auch Abschnitt 34.4 [Der `\tweak`-Befehl], Seite 605.

7.1.5 Klammern
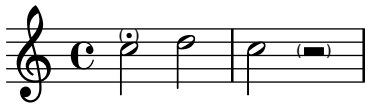
Objekte können in Klammern gesetzt werden, indem vor ihnen der Befehl `\parenthesize` geschrieben wird. Wenn ein Akkord in Klammern gesetzt wird, wirkt sich das auf jede Noten im Akkord aus. Innerhalb von einem Akkord gesetzte Befehle wirken sich auf einzelne Noten aus.

```
\relative {
  c''2 \parenthesize d
  c2 \parenthesize <c e g>
  c2 <c \parenthesize e g>
}
```



Auch andere Objekte als Noten können in Klammern gesetzt werden. Wenn Artikulationszeichen in Klammern gesetzt werden sollen, braucht man ein Minuszeichen vor dem `\parenthesize`-Befehl.

```
\relative {
  c''2-\parenthesize -. d
  c2 \parenthesize r
}
```



Siehe auch

Schnipsel: Abschnitt “Editorial annotations” in *Schnipsel*.

Referenz der Interna: Abschnitt “Parenthesis_engraver” in *Referenz der Interna*, Abschnitt “Parentheses” in *Referenz der Interna*, Abschnitt “parentheses-interface” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Wenn man einen Akkord einklammert, wird um jede Note eine eigene Klammer gesetzt, anstatt den gesamten Akkord in eine große Klammer zu fassen.

7.1.6 Hälse

Immer, wenn das Programm eine Note findet, wird automatisch ein Notenhals (Abschnitt “Stem” in *Referenz der Interna*) -Objekt erzeugt. Auch für ganze Noten und Pausen werden sie erzeugt, aber unsichtbar gemacht.

Hälse können manuell gesetzt werden, um nach oben oder unten zu zeigen, siehe Abschnitt 35.2 [Richtung und Platzierung], Seite 611.

Vordefinierte Befehle
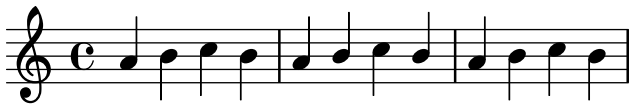
`\stemUp` (Hälse nach oben), `\stemDown` (Hälse nach unten), `\stemNeutral` (Hälse je nach Notenposition).

Ausgewählte Schnipsel

Standardrichtung für Hälse auf der Mittellinie

Die Richtung von Hälsen auf der mittleren Linie kann mit der `Stem`-Eigenschaft `neutral-direction` gesetzt werden.

```
\relative c'' {
  a4 b c b
  \override Stem.neutral-direction = #up
  a4 b c b
  \override Stem.neutral-direction = #down
  a4 b c b
}
```



Siehe auch

Notationsreferenz: Abschnitt 35.2 [Richtung und Platzierung], Seite 611.

Schnipsel: Abschnitt “Editorial annotations” in *Schnipsel*.

Referenz der Interna: Abschnitt “Stem_engraver” in *Referenz der Interna*, Abschnitt “Stem” in *Referenz der Interna*, Abschnitt “stem-interface” in *Referenz der Interna*.

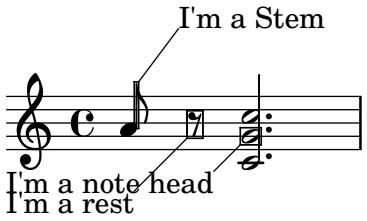
7.2 Außerhalb des Notensystems

Dieser Abschnitt zeigt, wie man Elemente im System von außerhalb des Systems hervorhebt.

7.2.1 Erklärungen in Ballonform

Notationselemente können bezeichnet und markiert werden, indem um sie eine rechteckige Blase gezeichnet wird. Dies ist vor allem dazu da, Notation zu erklären.

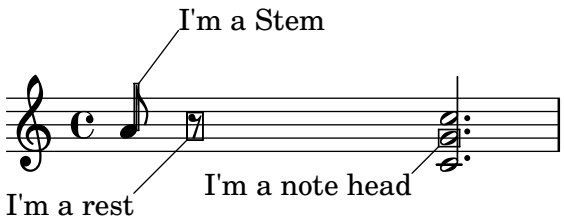
```
\new Voice \with { \consists Balloon_engraver }
\relative c'' {
  \balloonGrobText #'Stem #'(3 . 4) \markup { "I'm a Stem" }
  a8
  \balloonGrobText #'Rest #'(-4 . -4) \markup { "I'm a rest" }
  r
  <c, g'-\balloonText #'(-2 . -2) \markup { "I'm a note head" } c>2.
}
```



Es gibt zwei Funktionen, `balloonGrobText` und `balloonText`; die erste wird auf gleiche Art wie ein `\once \override` eingesetzt umd Text an einen Grob zu hängen, die zweite funktioniert wie ein `\tweak` und wird üblicherweise innerhalb von Akkorden eingesetzt, um Text an einzelne Noten zu hängen.

Textblasen beeinflussen normalerweise die Positionierung der Notation, aber das kann geändert werden.

```
\new Voice \with { \consists Balloon_engraver }
{
  \balloonLengthOff
  \balloonGrobText #'Stem #'(3 . 4) \markup { "I'm a Stem" }
  a8
  \balloonGrobText #'Rest #'(-4 . -4) \markup { "I'm a rest" }
  r
  \balloonLengthOn
  <c, g'-\balloonText #'(-2 . -2) \markup { "I'm a note head" } c>2.
}
```



Vordefinierte Befehle

`\balloonLengthOn`, `\balloonLengthOff`.

Siehe auch

Schnipsel: Abschnitt “Editorial annotations” in *Schnipsel*.

Referenz der Interna: Abschnitt “Balloon_engraver” in *Referenz der Interna*, Abschnitt “BalloonText” in *Referenz der Interna*, Abschnitt “balloon-interface” in *Referenz der Interna*.

7.2.2 Gitternetzlinien

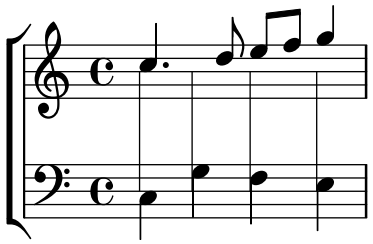
Vertikale Linien können zwischen Systemen gesetzt werden, die mit den Noten synchronisiert sind.

Der `Grid_point_engraver` muss benutzt werden, um die Endpunkte der Linien zu definieren, und der `Grid_line_span_engraver` wird benutzt, um dann die Linien zu setzen. Der Standard ist, dass die Gitterlinien unter den Noten und zur linken Seite des Notenkopfes gesetzt werden. Sie reichen von der Mitte eines Systems bis zur Mitte des anderen. Mit `gridInterval` wird die Dauer zwischen den Linien festgesetzt.

```
\layout {
  \context {
    \Staff
    \consists Grid_point_engraver
    gridInterval = #1/4
  }
  \context {
    \Score
    \consists Grid_line_span_engraver
  }
}

\score {
  \new ChoirStaff <<
    \new Staff \relative {
      \stemUp
      c''4. d8 e8 f g4
```

```
    }
    \new Staff \relative {
      \clef bass
      \stemDown
      c4 g' f e
    }
  >>
}
```



Ausgewählte Schnipsel

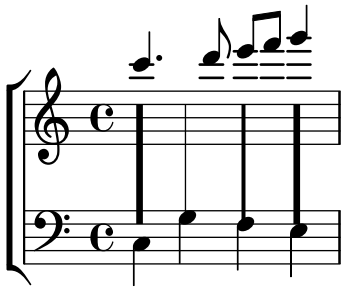
Gitternetzlinien: Aussehen verändern

Die Erscheinung der Gitternetzlinien kann durch einige Eigenschaften geändert werden.

```
\new ChoirStaff <<
  \new Staff {
    \relative c'' {
      \stemUp
      c'4. d8 e8 f g4
    }
  }
  \new Staff {
    \relative c {
      % this moves them up one staff space from the default position
      \override Score.GridLine.extra-offset = #'(0.0 . 1.0)
      \stemDown
      \clef bass
      \once \override Score.GridLine.thickness = 5.0
      c4
      \once \override Score.GridLine.thickness = 1.0
      g'4
      \once \override Score.GridLine.thickness = 3.0
      f4
      \once \override Score.GridLine.thickness = 5.0
      e4
    }
  }
>>

\layout {
  \context {
    \Staff
    % set up grids
    \consists "Grid_point_engraver"
    % set the grid interval to one quarter note
```

```
    gridInterval = #1/4
    }
  \context {
    \Score
    \consists "Grid_line_span_engraver"
    % this moves them to the right half a staff space
    \override NoteColumn.X-offset = -0.5
  }
}
```



Siehe auch
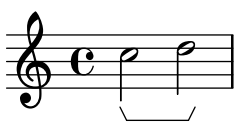
Schnipsel: Abschnitt “Editorial annotations” in *Schnipsel*.

Referenz der Interna: Abschnitt “Grid_line_span_engraver” in *Referenz der Interna*, Abschnitt “Grid_point_engraver” in *Referenz der Interna*, Abschnitt “GridLine” in *Referenz der Interna*, Abschnitt “GridPoint” in *Referenz der Interna*, Abschnitt “grid-line-interface” in *Referenz der Interna*, Abschnitt “grid-point-interface” in *Referenz der Interna*.

7.2.3 Analyseklammern

Klammern über dem System werden in der Musikanalyse benutzt, um strukturelle Einheiten der Musik zu markieren. Einfache horizontale Klammern werden von LilyPond unterstützt.

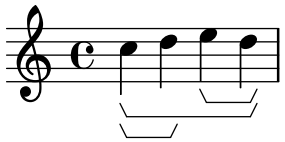
```
\layout {
  \context {
    \Voice
    \consists Horizontal_bracket_engraver
  }
}
\relative {
  c''2\startGroup
  d\stopGroup
}
```



Analyseklammern können verschachtelt sein.

```
\layout {
  \context {
    \Voice
    \consists Horizontal_bracket_engraver
  }
}
```

```
\relative {
  c''4\startGroup\startGroup
  d4\stopGroup
  e4\startGroup
  d4\stopGroup\stopGroup
}
```
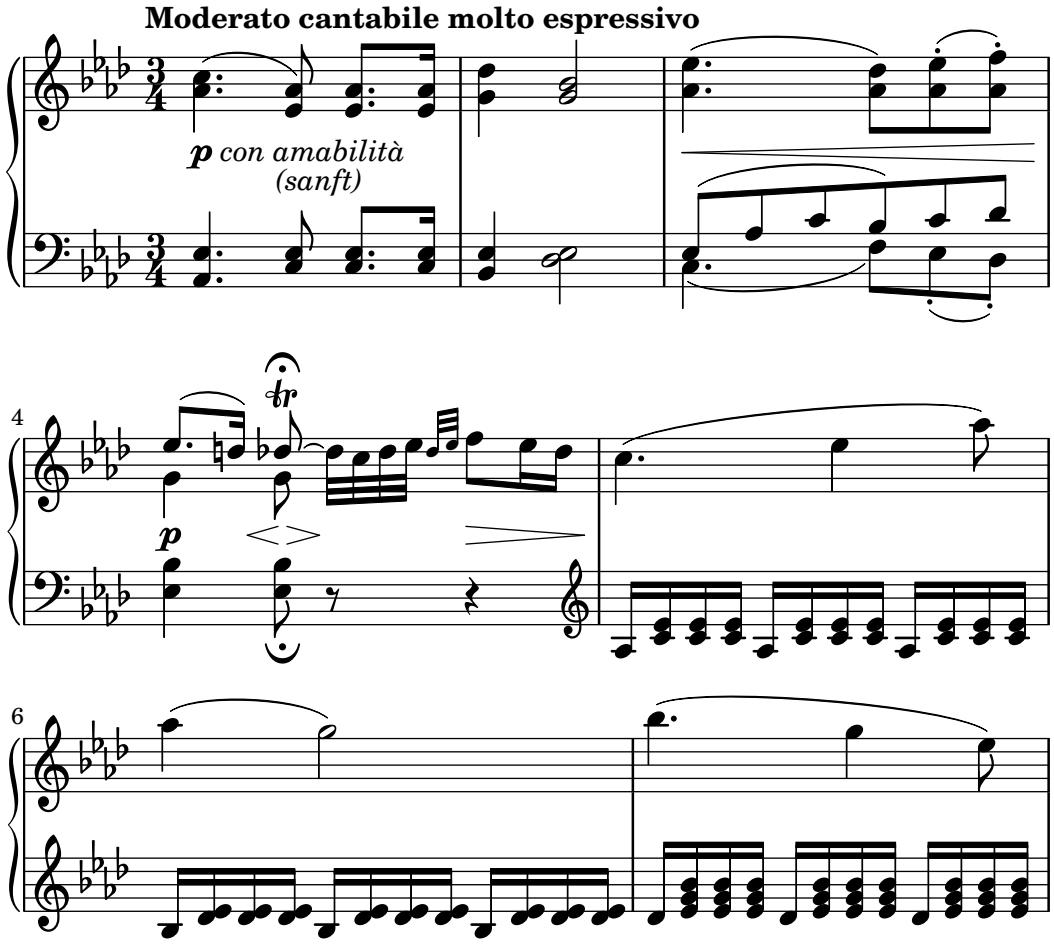


Siehe auch

Schnipsel: Abschnitt “Editorial annotations” in *Schnipsel*.

Referenz der Interna: Abschnitt “Horizontal_bracket_engraver” in *Referenz der Interna*, Abschnitt “HorizontalBracket” in *Referenz der Interna*, Abschnitt “horizontal-bracket-interface” in *Referenz der Interna*, Abschnitt “Staff” in *Referenz der Interna*.

8 Text



Dieser Abschnitt erklärt, wie man Text (mit vielfältiger Formatierung) in Partituren einfügt.

Einige Textelemente, die hier nicht behandelt werden, finden sich in anderen Abschnitten: Kapitel 9 [Notation von Gesang], Seite 253, Kapitel 20 [Titel], Seite 464.

8.1 Text eingeben

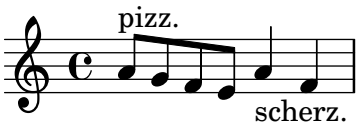
Dieser Abschnitt zeigt verschiedene Arten, wie Text in die Partitur eingefügt werden kann.

> **Achtung:** Wenn man Zeichen mit Akzenten und Umlaute oder besondere Zeichen (wie etwa Text mit anderen Alphabeten) eingeben möchte, kann man die Zeichen einfach direkt in die Datei einfügen. Die Datei muss als UTF-8 gespeichert werden. Für mehr Information siehe Abschnitt 21.3.1 [Zeichenkodierung], Seite 501.

8.1.1 Textarten
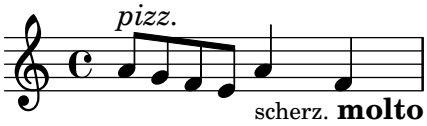
Am einfachsten kann Text mit geraden Anführungsstrichen in eine Partitur eingefügt werden, wie das folgende Beispiel zeigt. Derartiger Text kann manuell über oder unter dem Notensystem platziert werden, die Syntax hierzu ist beschrieben in Abschnitt 35.2 [Richtung und Platzierung], Seite 611.

```
\relative { a'8^"pizz." g f e a4-"scherz." f }
```
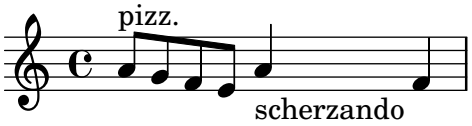


Diese Syntax ist eine Kurzform, komplexere Formatierungen können einem Text hinzugefügt werden, wenn man explizit den `\markup`-Befehl mit darauf folgenden geschweiften Klammern einsetzt, wie beschrieben in Abschnitt 8.2 [Text formatieren], Seite 233.

```
\relative {
  a'8^\markup { \italic pizz. } g f e
  a4_\markup { \tiny scherz. \bold molto } f }
```



Standardmäßig haben Textbeschriftungen keinen Einfluss auf die Positionierung der Noten. Man kann aber auch bestimmen, dass die Breite des Textes mit berücksichtigt wird. Im nächsten Beispiel fordert der erste Text keinen Platz, während der zweite die Note nach rechts verschiebt. Das Verhalten wird mit dem Befehl `\textLengthOn` (Textlänge an) erreicht, rückgängig kann es mit dem Befehl `\textLengthOff` gemacht werden.

```
\relative {
  a'8^"pizz." g f e
  \textLengthOn
  a4_"scherzando" f
}
```



Neben Textbeschriftungen können auch Artikulationen an Noten angehängt werden. Siehe auch Abschnitt 3.1.1 [Artikulationszeichen und Verzierungen], Seite 120.

Zu weiterer Information zu der relativen Anordnung von Textbeschriftungen und Artikulationen, siehe Abschnitt “Positionierung von Objekten” in *Handbuch zum Lernen*.

Vordefinierte Befehle

`\textLengthOn`, `\textLengthOff`.

Siehe auch

Handbuch zum Lernen: Abschnitt “Positionierung von Objekten” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 8.2 [Text formatieren], Seite 233, Abschnitt 35.2 [Richtung und Platzierung], Seite 611, Abschnitt 3.1.1 [Artikulationszeichen und Verzierungen], Seite 120.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

Referenz der Interna: Abschnitt “TextScript” in *Referenz der Interna*.

Bekannte Probleme und Warnungen
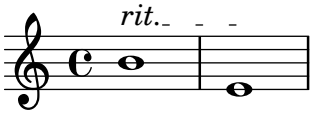
Eine Überprüfung, ob sich auch alle Textbeschriftungen und Gesangstext innerhalb der Ränder der Noten befinden, braucht verhältnismäßig viel Rechenaufwand. Sollten Sie aber etwas bessere Leistung bevorzugen, schreiben Sie in Ihre Datei:

```
\override Score.PaperColumn.keep-inside-line = ##f
```

8.1.2 Text mit Verbindungslinien

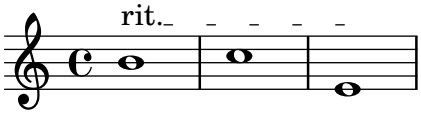
Einige Aufführungsanweisungen, etwa *rallentando* oder *accelerando*, werden als Text geschrieben, gefolgt von einer gestrichelten Linie, die anzeigt, wie weit sich die Anweisung auswirkt. Solche Objekte, „Strecker“ (engl. spanners) genannt, können von einer Note bis zu einer anderen mit folgender Anweisung erstellt werden:

```
\relative {
  \override TextSpanner.bound-details.left.text = "rit."
  b'1\startTextSpan
  e,\stopTextSpan
}
```



Der Text wird durch Objekteigenschaften beeinflusst. In den Standardeinstellungen wird er kursiv ausgegeben, aber eine andere Formatierung kann erreicht werden, indem man `\markup`-Blöcke einsetzt, wie beschrieben in Abschnitt 8.2 [Text formatieren], Seite 233.

```
\relative {
  \override TextSpanner.bound-details.left.text =
    \markup { \upright "rit." }
  b'1\startTextSpan c
  e,\stopTextSpan
}
```



Auch der Stil der Linie kann ähnlich wie der Text mit den Objekteigenschaften geändert werden. Diese Syntax ist beschrieben in Abschnitt 35.7 [Linienstile], Seite 624. Textstrecker sind Teil des `Dynamic`-Kontextes, siehe Abschnitt “Dynamics” in *Referenz der Interna*.

Vordefinierte Befehle

`\textSpannerUp`, `\textSpannerDown`, `\textSpannerNeutral`.

Ausgewählte Schnipsel
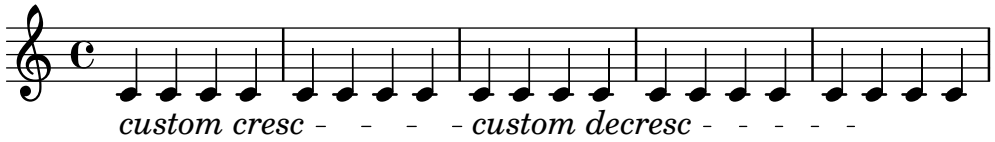
Dynamics spanner with custom text

Postfix functions for custom crescendo text spanners. The spanners should start on the first note of the measure. One has to use `-\mycresc`, otherwise the spanner start will rather be assigned to the next note.

```
% Two functions for (de)crescendo spanners where you can explicitly
% give the spanner text.
mycresc =
#(define-music-function (mymarkup) (markup?)
   (make-music 'CrescendoEvent
               'span-direction START
               'span-type 'text
               'span-text mymarkup))
mydecresc =
#(define-music-function (mymarkup) (markup?)
```

```
   (make-music 'DecrescendoEvent
               'span-direction START
               'span-type 'text
               'span-text mymarkup))

\relative c' {
  c4-\mycresc "custom cresc" c4 c4 c4 |
  c4 c4 c4 c4 |
  c4-\mydecresc "custom decresc" c4 c4 c4 |
  c4 c4 c4 c4 |
  c4 c4\! c4 c4
}
```



Siehe auch

Notationsreferenz: Abschnitt 35.7 [Linienstile], Seite 624, Abschnitt 3.1.2 [Dynamik], Seite 123, Abschnitt 8.2 [Text formatieren], Seite 233.
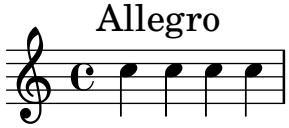
Schnipsel: Abschnitt “Text” in *Schnipsel*, Abschnitt “Expressive marks” in *Schnipsel*.

Referenz der Interna: Abschnitt “TextSpanner” in *Referenz der Interna*

8.1.3 Textartige Zeichen
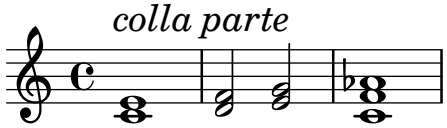
Verschiedene Textelemente können der Partitur hinzugefügt werden, indem man die Syntax für Zeichen einsetzen, wie beschrieben in Abschnitt 2.5.4 [Übungszeichen], Seite 110:

```
\relative {
  c''4
  \mark "Allegro"
  c c c
}
```



Diese Syntax ermöglicht es, beliebigen Text über eine Taktlinie zu platzieren, weitere Formatierungsmöglichkeiten sind mit dem \markup-Befehl gegeben, wie beschrieben in Abschnitt 8.2 [Text formatieren], Seite 233:

```
\relative {
  <c' e>1
  \mark \markup { \italic { colla parte } }
  <d f>2 <e g>
  <c f aes>1
}
```



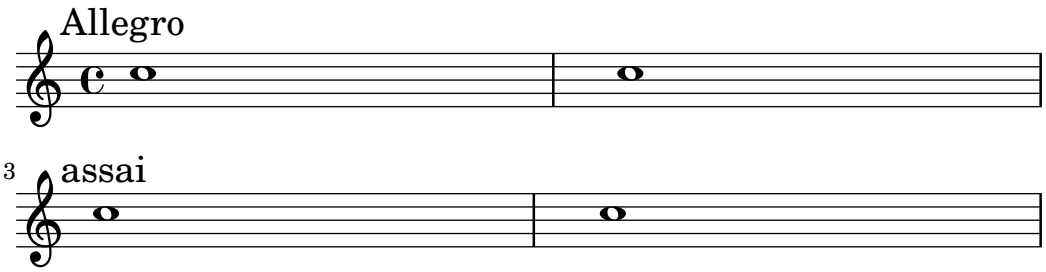
Diese Syntax ermöglicht es auch, besondere Zeichen einzufügen, wie etwa Coda-, Segno- oder Fermatenzeichen, indem das entsprechende Symbol mit dem Befehl `\musicglyph` angegeben wird, wie beschrieben in Abschnitt 8.2.5 [Musikalische Notation innerhalb einer Textbeschriftung], Seite 243:

```
\relative {
  <bes' f>2 <aes d>
  \mark \markup { \musicglyph "scripts.ufermata" }
  <e g>1
}
```



Derartige Objekte werden über dem höchsten System einer Partitur gesetzt – abhängig davon, ob sie mitten im Takt oder an seinem Ende notiert werden, werden sie zwischen Noten oder über der Taktlinie gesetzt. Wenn sie an einem Zeilenumbruch angegeben werden, wird das Zeichen zu Beginn der nächsten Zeile ausgegeben.

```
\mark "Allegro"
c1 c
\mark "assai" \break
c  c
```



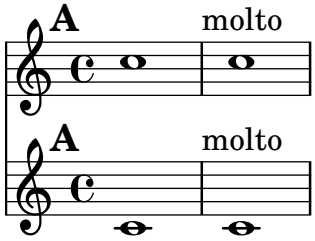
Ausgewählte Schnipsel

Zeichen über jedem System ausgeben

Normalerweise werden Textzeichen nur über dem obersten Notensystem gesetzt. Sie können aber auch über jedem System ausgegeben werden.

```
\score {
  <<
    \new Staff { \mark \default c''1 \textMark "molto" c'' }
    \new Staff { \mark \default c'1 \textMark "molto" c' }
  >>
  \layout {
    \context {
      \Score
      \remove Mark_engraver
      \remove Text_mark_engraver
      \remove Staff_collecting_engraver
    }
    \context {
      \Staff
      \consists Mark_engraver
```

```
      \consists Text_mark_engraver
      \consists Staff_collecting_engraver
    }
  }
}
```



Siehe auch

Notationsreferenz: Abschnitt 2.5.4 [Übungszeichen], Seite 110, Abschnitt 8.2 [Text formatieren], Seite 233, Abschnitt 8.2.5 [Musikalische Notation innerhalb einer Textbeschriftung], Seite 243, Abschnitt A.8 [Die Emmentaler-Schriftart], Seite 658.

Schnipsel: Abschnitt "Text" in *Schnipsel*.

Referenz der Interna: Abschnitt "TextMarkEvent" in *Referenz der Interna*, Abschnitt "Text_mark_engraver" in *Referenz der Interna*, Abschnitt "TextMark" in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Wenn ein Zeichen am Ende des letzten Taktes einer Partitur gesetzt wird (wenn also keine nächste Zeile mehr kommt), wird das Zeichen nicht ausgegeben.

8.1.4 Separater Text

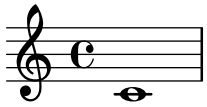
Eine `\markup`-Umgebung kann auch für sich alleine existieren, außerhalb einer `\score`-Umgebung, als ein Ausdruck auf der höchsten Ebene. Diese Syntax ist beschrieben in Abschnitt 19.5 [Die Dateistruktur], Seite 461.

```
\markup {
  Morgen, morgen, und morgen...
}
```

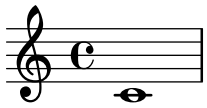
Morgen, morgen, und morgen...

Damit kann Text unabhängig von den Noten gesetzt werden. Das bietet sich vor allem in Situationen an, in denen mehrere Stücke in einer Datei vorkommen, wie beschrieben in Abschnitt 19.2 [Mehrere Partituren in einem Buch], Seite 458.

```
\score {
  c'1
}
\markup {
  Morgen, übermorgen, und überübermorgen...
}
\score {
  c'1
}
```



Morgen, übermorgen, und überübermorgen...



Unabhängige Textabschnitte können über mehrere Seiten reichen, so dass man Textdokumente oder Bücher ausschließlich mit LilyPond setzen kann. Einzelheiten zu den vielfältigen Möglichkeiten finden sich in Abschnitt 8.2.6 [Textbeschriftung über mehrere Seiten], Seite 246.

Vordefinierte Befehle

`\markup`, `\markuplist`.

Ausgewählte Schnipsel

Isolierter Text in zwei Spalten

Isolierter Text kann in mehreren Spalten mit `\markup`-Befehlen angeordnet werden:

```
\markup {
  \fill-line {
    \hspace #1
    \column {
      \line { O sacrum convivium }
      \line { in quo Christus sumitur, }
      \line { recolitur memoria passionis ejus, }
      \line { mens impletur gratia, }
      \line { futurae gloriae nobis pignus datur. }
      \line { Amen. }
    }
    \hspace #2
    \column \italic {
      \line { O sacred feast }
      \line { in which Christ is received, }
      \line { the memory of His Passion is renewed, }
      \line { the mind is filled with grace, }
      \line { and a pledge of future glory is given to us. }
      \line { Amen. }
    }
    \hspace #1
  }
}
```

O sacrum convivium
in quo Christus sumitur,
recolitur memoria passionis ejus,
mens impletur gratia,
futurae gloriae nobis pignus datur.
Amen.

O sacred feast
in which Christ is received,
the memory of His Passion is renewed,
the mind is filled with grace,
and a pledge of future glory is given to us.
Amen.

Siehe auch

Notationsreferenz: Abschnitt 8.2 [Text formatieren], Seite 233, Abschnitt 19.5 [Die Dateistruktur], Seite 461, Abschnitt 19.2 [Mehrere Partituren in einem Buch], Seite 458, Abschnitt 8.2.6 [Textbeschriftung über mehrere Seiten], Seite 246.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

Referenz der Interna: Abschnitt “TextScript” in *Referenz der Interna*.

8.2 Text formatieren

Dieser Abschnitt zeigt grundlegende und fortgeschrittene Formatierung von Text, wobei der Textbeschriftungsmodus (\markup benutzt wird.

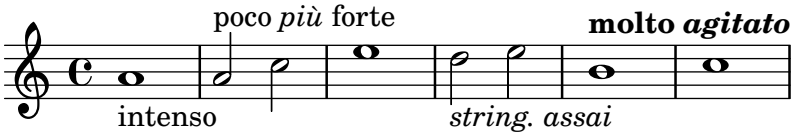
8.2.1 Textbeschriftung (Einleitung)

Eine \markup-Umgebung wird benutzt, um Text mit einer großen Anzahl von Formatierungsmöglichkeiten (im „markup-Modus“) zu setzen.

Die Syntax für Textbeschriftungen ähnelt der normalen Syntax von LilyPond: ein \markup-Ausdruck wird in geschweifte Klammern eingeschlossen ({... }). Ein einzelnes Wort wird als ein Minimalausdruck erachtet und muss deshalb nicht notwendigerweise eingeklammert werden.
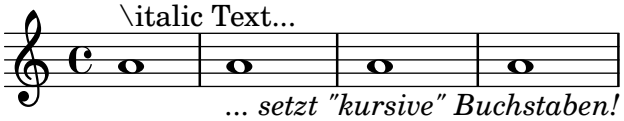
Anders als Text in Anführungsstrichen können sich in einer Textbeschriftungsumgebung (\markup) geschachtelte Ausdrücke oder weitere Textbefehle befinden, eingeführt mit einem Backslash (\). Derartige Befehle beziehen sich nur auf den ersten der folgenden Ausdrücke.

```
\relative {
  a'1-\markup intenso
  a2^\markup { poco \italic più forte  }
  c e1
  d2_\markup { \italic "string. assai" }
  e
  b1^\markup { \bold { molto \italic  agitato } }
  c
}
```



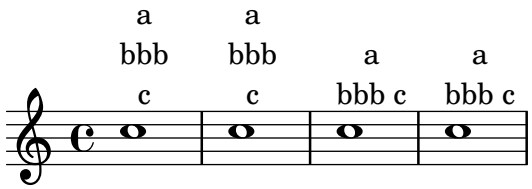
Eine \markup-Umgebung kann auch Text in Anführungszeichen beinhalten. Derartige Zeichenketten werden als ein Textausdruck angesehen, und darum werden innerhalb von ihnen Befehle oder Sonderzeichen (wie \ oder #) so ausgegeben, wie sie eingeben werden. Doppelte Anführungsstriche können gesetzt werden, indem man ihnen einen Backslash voranstellt.

```
\relative {
  a'1^"\\italic Text..."
  a_\markup { \italic "... setzt \"kursive\" Buchstaben!" }
  a a
}
```



Damit eine Anzahl von Wörtern als ein einziger Ausdruck behandelt wird, müssen alle Wörter zwischen geraden Anführungszeichen (Shift+2) stehen oder ihnen muss ein Befehl vorangestellt werden. Die Art, wie die Ausdrücke definiert sind, wirkt sich darauf aus, wie sie übereinander gestapelt, mittig und aneinander ausgerichtet werden. Im folgenden Beispiel verhält sich der zweite \markup-Ausdruck genauso wie der erste:
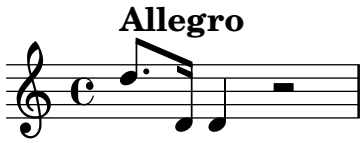
```
\relative c'' {
  c1^\markup { \center-column { a bbb c } }
  c1^\markup { \center-column { a { bbb c } } }
  c1^\markup { \center-column { a \line { bbb c } } }
  c1^\markup { \center-column { a "bbb c" } }
}
```



Textbeschriftung kann auch durch Variablen definiert werden. Diese Variablen können dann direkt an Noten angefügt werden:

```
allegro = \markup { \bold \large Allegro }

{
  d''8.^\allegro
  d'16 d'4 r2
}
```



Eine ausführliche Liste der \markup-Befehle findet sich in Abschnitt A.10 [Textbeschriftungsbefehle], Seite 678.

Siehe auch

Notationsreferenz: Abschnitt A.10 [Textbeschriftungsbefehle], Seite 678.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

Installierte Dateien: `scm/markup.scm`.
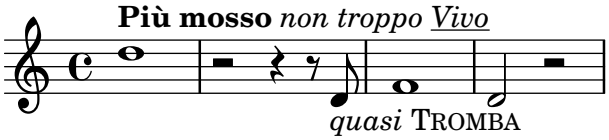
Bekannte Probleme und Warnungen

Syntaxfehler im Textbeschriftungsmodus können sehr verwirrend sein.

8.2.2 Überblick über die wichtigsten Textbeschriftungsbefehle

Einfache Änderungen des Schriftartschnitts können im Textbeschriftungsmodus vorgenommen werden:

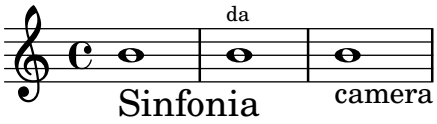
```
\relative {
  d''1^\markup {
    \bold { Più mosso }
    \italic { non troppo \underline Vivo }
  }
  r2 r4 r8
  d,_\markup { \italic quasi \smallCaps Tromba }
  f1 d2 r
}
```



Die Schriftgröße kann auf verschiedene Arten verändert werden, relativ zur globalen Notensystemgröße:

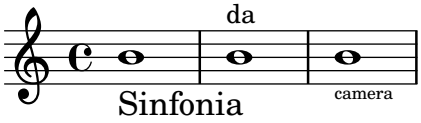
Sie kann auf eine vordefinierte Größe gesetzt werden:

```
\relative b' {
  b1_\markup { \huge Sinfonia }
  b1^\markup { \teeny da }
  b1-\markup { \normalsize camera }
}
```



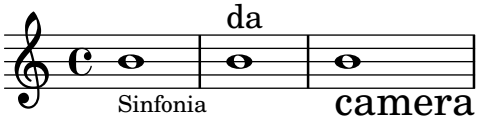
Sie kann relativ zum vorherigen Wert gesetzt werden:

```
\relative b' {
  b1_\markup { \larger Sinfonia }
  b1^\markup { \smaller da }
  b1-\markup { \magnify #0.6  camera }
}
```



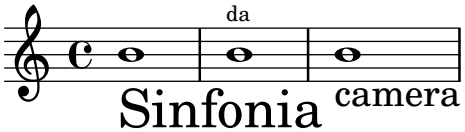
Sie kann vergrößert oder verkleinert werden relativ zum Wert, der von der globalen Notensystemgröße vorgegeben wird:

```
\relative b' {
  b1_\markup { \fontsize #-2 Sinfonia }
  b1^\markup { \fontsize #1 da }
  b1-\markup { \fontsize #3 camera }
}
```



Sie kann auch auf eine bestimmte Punktegröße festgelegt werden, unabhängig von der globalen Notensystemgröße:

```
\relative b' {
  b1_\markup { \abs-fontsize #20 Sinfonia }
  b1^\markup { \abs-fontsize #8 da }
  b1-\markup { \abs-fontsize #14 camera }
}
```



Text kann auch hoch- bzw. tiefgestellt gesetzt werden. Die so markierten Buchstaben werden automatisch in einer kleineren Schriftgröße gesetzt, aber die normale Schriftgröße kann auch eingesetzt werden:

```
\markup {
  \column {
    \line { 1 \super st movement }
    \line { 1 \normal-size-super st movement
      \sub { (part two) }  }
  }
}
```

1 st movement
1 st movement (part two)

Der Textbeschriftungsmodus stellt eine einfache Möglichkeit zur Verfügung unterschiedliche Schriftschnitte anzuwählen. Ohne besondere Einstellungen wird automatisch eine Schriftart mit Serifen ausgewählt. Das Beispiel unten zeigt die Verwendung der eigenen Zahlenschriftart von LilyPond, den Einsatz von serifenloser Schriftart und von Schreibmaschinenschriftart. Die letzte Zeile zeigt, dass sich die Standardeinstellung mit dem Befehl `\serif` wieder herstellen lässt.

```
\markup {
  \column {
    \line { Act \number 1 }
    \line { \sans { Scene I. } }
    \line { \typewriter { Verona. An open place. } }
    \line { Enter \serif Valentine and Proteus. }
  }
}
```

Act 1
Scene I.
Verona. An open place.
Enter Valentine and Proteus.

Einige dieser Schriftarten, etwa die Zahlenschriftart oder die Schriftart für Dynamikzeichen, stellen nicht alle Zeichen zur Verfügung, wie beschrieben in Abschnitt 3.1.3 [Neue Lautstärkezeichen], Seite 128, und Abschnitt 4.1.2 [Manuelle Wiederholungszeichen], Seite 152.

Einige Schriftartbefehle können ungewollte Leerzeichen innerhalb von Wörtern hervorrufen. Das kann vermieden werden, indem die einzelnen Elemente mit dem Befehl `\concat` zu einem Element verschmolzen werden:

```
\markup {
  \column {
    \line {
      \concat { 1 \super st }
      movement
    }
    \line {
      \concat { \dynamic p , }
      \italic { con dolce espressione }
    }
  }
}
```

1st movement
***p**, con dolce espressione*

Eine ausführliche Liste der unterschiedlichen Befehl zur Beeinflussung der Schriftarten findet sich in Abschnitt A.10.1 [Font markup], Seite 678.

Es ist auch möglich, eigene Schriftfamilien zu definieren, wie erklärt in Abschnitt 8.3 [Schriftarten], Seite 247.

Vordefinierte Befehle

`\teeny`, `\tiny`, `\small`, `\normalsize`, `\large`, `\huge`, `\smaller`, `\larger`.

Siehe auch

Notationsreferenz: Abschnitt A.10.1 [Font markup], Seite 678, Abschnitt 3.1.3 [Neue Lautstärkezeichen], Seite 128, Abschnitt 4.1.2 [Manuelle Wiederholungszeichen], Seite 152, Abschnitt 8.3 [Schriftarten], Seite 247.

Installierte Dateien: `scm/define-markup-commands.scm`.

Schnipsel: Abschnitt "Text" in *Schnipsel*.

Referenz der Interna: Abschnitt "TextScript" in *Referenz der Interna*.

Bekannte Probleme und Warnungen

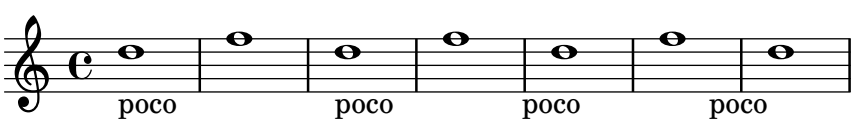
Wenn die Befehle `\teeny`, `\tiny`, `\small`, `\normalsize`, `\large` und `\huge` eingesetzt werden, erhält man schlechte Zeilenabstände verglichen mit `\fontsize`.

8.2.3 Textausrichtung

Dieser Abschnitt zeigt, wie man Text im Textbeschriftungsmodus eingibt. Textobjekte können auch als eine Einheit verschoben werden, wie beschrieben in Abschnitt "Verschieben von Objekten" in *Handbuch zum Lernen*.

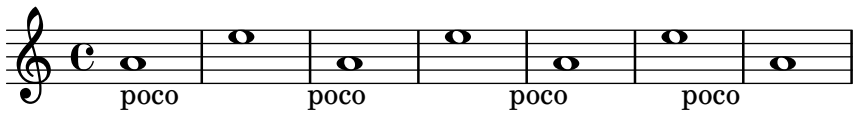
Textbeschriftungsobjekte können auf verschiedene Weise ausgerichtet werden. Standardmäßig wird ein Textobjekt an seiner linken Ecke ausgerichtet, darum wird das erste und zweite Objekt gleichermaßen an der linken Ecke ausgerichtet.

```
\relative {
  d''1-\markup { poco }
  f
  d-\markup { \left-align poco }
  f
  d-\markup { \center-align { poco } }
  f
  d-\markup { \right-align poco }
}
```



Die horizontale Ausrichtung kann mit einer Zahl auf einen exakten Wert festgelegt werden:

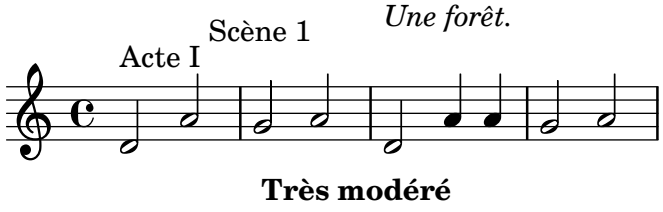
```
\relative {
  a'1-\markup { \halign #-1 poco }
  e'
  a,-\markup { \halign #0 poco }
  e'
  a,-\markup { \halign #0.5 poco }
  e'
  a,-\markup { \halign #2 poco }
}
```



Manche Objekte haben eigene Ausrichtungsvorgänge und werden deshalb nicht von diesen Befehlen beeinflusst. Es ist möglich, solche Objekte als eine Einheit anzusprechen und zu bewegen, wie gezeigt in Abschnitt 8.1.3 [Textartige Zeichen], Seite 229.

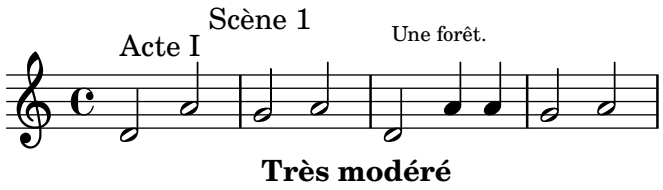
Die vertikale Ausrichtung ist etwas schwieriger. Textelemente können komplett verschoben werden, es ist aber auch möglich, nur einen Teil innerhalb der Textbeschriftung zu bewegen. In diesem Fall muss dem zu verschiebenden Objekt ein Ankerpunkt zugewiesen werden, welcher entweder ein anderes Textelement oder ein unsichtbares Objekt sein kann (im Beispiel mit `\null` erstellt). Der letzte Text im Beispiel hat keinen Anker und wird deshalb auch nicht verschoben.

```
\relative {
  d'2^\markup {
    Acte I
    \raise #2 { Scène 1 }
  }
  a'
  g_\markup {
    \null
    \lower #4 \bold { Très modéré }
  }
  a
  d,^\markup {
    \raise #4 \italic { Une forêt. }
  }
  a'4 a g2 a
}
```



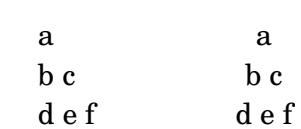
Einige Befehle können sowohl die horizontale als auch die vertikale Ausrichtung von Textobjekten beeinflussen. Jedes Objekt, das auf diese Weise verschoben wird, benötigt einen Anker:

```
\relative {
  d'2^\markup {
    Acte I
    \translate #'(-1 . 2) "Scène 1"
  }
  a'
  g_\markup {
    \null
    \general-align #Y #3.2 \bold "Très modéré"
  }
  a
  d,^\markup {
    \null
    \translate-scaled #'(-1 . 2) \teeny "Une forêt."
  }
  a'4 a g2 a
}
```



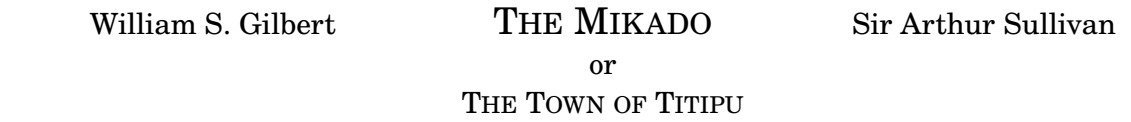
Ein Textbeschriftungsobjekt kann mehrere Zeilen beinhalten. Im folgenden Beispiel wird jeder Ausdruck innerhalb von \markup auf einer eigenen Zeile gesetzt, entweder linksbündig oder zentriert:

```
\markup {
  \column {
    a
    "b c"
    \line { d e f }
  }
  \hspace #10
  \center-column {
    a
    "b c"
    \line { d e f }
  }
}
```



Eine Anzahl an Ausdrücken innerhalb von \markup kann auch gestreckt werden, so dass die gesamte Seitenbreite benutzt wird. Wenn nur ein Objekt vorhanden ist, wird es zentriert gesetzt. Die Ausdrücke selber können wiederum mehrzeilig sein und andere Textbeschriftungsbefehle beinhalten.

```
\markup {
  \fill-line {
    \line { William S. Gilbert }
    \center-column {
      \huge \smallCaps "The Mikado"
      or
      \smallCaps "The Town of Titipu"
    }
    \line { Sir Arthur Sullivan }
  }
}
\markup {
  \fill-line { 1885 }
}
```



Längere Texte können auch automatisch umgebrochen werden, wobei es möglich ist, die Zeilenbreite zu bestimmen. Der Text ist entweder linksbündig oder im Blocksatz, wie das nächste Beispiel illustriert:

```
\markup {
  \column {
    \line  \smallCaps { La vida breve }
    \line \bold { Acto I }
    \wordwrap \italic {
      (La escena representa el corral de una casa de
      gitanos en el Albaicín de Granada.  Al fondo una
      puerta por la que se ve el negro interior de
      una Fragua, iluminado por los rojos resplandores
      del fuego.)
    }
    \hspace #0

    \line \bold { Acto II }
    \override #'(line-width . 50)
    \justify \italic {
      (Calle de Granada.  Fachada de la casa de Carmela
      y su hermano Manuel con grandes ventanas abiertas
      a través de las que se ve el patio
      donde se celebra una alegre fiesta)
    }
  }
}
```

La vida breve

Acto I

(La escena representa el corral de una casa de gitanos en el Albaicín de Granada. Al fondo una puerta por la que se ve el negro interior de una Fragua, iluminado por los rojos resplandores del fuego.)

Acto II

(Calle de Granada. Fachada de la casa de Carmela y su hermano Manuel con grandes ventanas abiertas a través de las que se ve el patio donde se celebra una alegre fiesta)

Eine vollständige Liste der Textausrichtungsbefehle findet sich in Abschnitt A.10.2 [Markup for text alignment], Seite 691.

Siehe auch

Handbuch zum Lernen: Abschnitt “Verschieben von Objekten” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt A.10.2 [Markup for text alignment], Seite 691, Abschnitt 8.1.3 [Textartige Zeichen], Seite 229.

Installierte Dateien: `scm/define-markup-commands.scm`.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

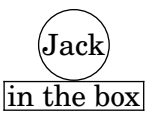
Referenz der Interna: Abschnitt “TextScript” in *Referenz der Interna*.

8.2.4 Graphische Notation innerhalb einer Textbeschriftung

Verschiedene graphische Objekte können im Textbeschriftungsmodus eingefügt werden.

Mit bestimmten Textbeschriftungsbefehlen kann man Textelementen Graphik hinzufügen, wie das nächste Beispiel zeigt:

```
\markup \fill-line {
  \center-column {
    \circle Jack
    \box "in the box"
    \null
    \line {
      Erik Satie
      \hspace #3
      \bracket "1866 - 1925"
    }
    \null
    \rounded-box \bold Prelude
  }
}
```

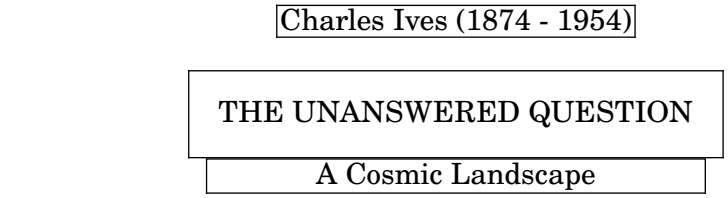


Erik Satie [1866 - 1925]

Prelude

Es kann nötig sein, einem Text mehr Platz einzuräumen. Das geschieht mit verschiedenen Befehlen, wie das folgende Beispiel zeigt. Eine ausführliche Übersicht findet sich in Abschnitt A.10.2 [Markup for text alignment], Seite 691.

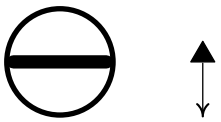
```
\markup \fill-line {
  \center-column {
    \box "Charles Ives (1874 - 1954)"
    \null
    \box \pad-markup #2 "THE UNANSWERED QUESTION"
    \box \pad-x #8 "A Cosmic Landscape"
    \null
  }
}
\markup \column {
  \line {
    \hspace #10
    \box \pad-to-box #'(-5 . 20) #'(0 . 5)
      \bold "Largo to Presto"
  }
  \pad-around #3
      "String quartet keeps very even time,
Flute quartet keeps very uneven time."
}
```



Andere graphische Elemente oder Symbole können gesetzt werden, ohne dass man Text benötigt. Wie mit allen Textbeschriftungen können Objekte innerhalb von `\markup` kombiniert werden.

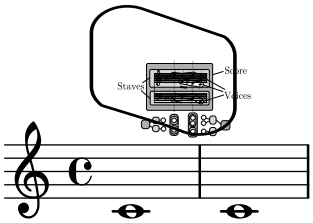
```
\markup {
  \combine
    \draw-circle #4 #0.4 ##f
    \filled-box #'(-4 . 4) #'(-0.5 . 0.5) #1
  \hspace #5

  \center-column {
    \triangle ##t
    \combine
      \draw-line #'(0 . 4)
      \arrow-head #Y #DOWN ##f
  }
}
```



Fortgeschrittene graphische Möglichkeiten bietet unter Anderem eine Funktion, mit der man externe Graphiken im Encapsulated PostScript (*eps*) -Format einbinden kann oder aber Graphiken direkt in den Quelltext unter Verwendung von PostScript-Code notiert. In diesem Fall kann es nötig sein, die Größe der Zeichnung explizit anzugeben, wie im Beispiel unten gezeigt:

```
c'1^\markup {
  \combine
    \epsfile #X #10 "./context-example.eps"
    \with-dimensions #'(0 . 6) #'(0 . 10)
    \postscript "
      -2 3 translate
      2.7 2 scale
      newpath
      2 -1 moveto
      4 -2 4 1 1 arct
      4 2 3 3 1 arct
      0 4 0 3 1 arct
      0 0 1 -1 1 arct
      closepath
      stroke"
  }
c'
```



Eine ausführliche Liste der Graphik-Befehle findet sich in Abschnitt A.10.3 [Graphical markup], Seite 710.

Siehe auch

Notationsreferenz: Abschnitt A.10.3 [Graphical markup], Seite 710, Kapitel 7 [Anmerkungen], Seite 214, Abschnitt A.10.2 [Markup for text alignment], Seite 691.

Installierte Dateien: `scm/define-markup-commands.scm`, `scm/stencil.scm`.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

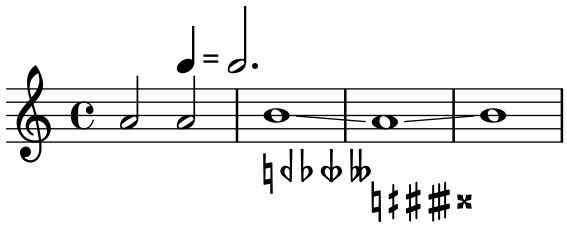
Referenz der Interna: Abschnitt “TextScript” in *Referenz der Interna*.

8.2.5 Musikalische Notation innerhalb einer Textbeschriftung

Auch Musikobjekte können innerhalb der Textbeschriftungsumgebung gesetzt werden.
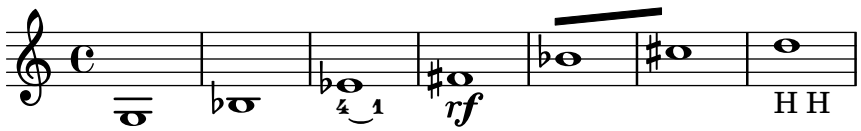
Noten und Versetzungszeichen lassen sich mit `\markup` einfügen:

```
a'2 a'^\markup {
  \note {4} #1
  =
  \note-by-number #1 #1 #1.5
}
b'1_\markup {
  \natural \semiflat \flat
  \sesquiflat \doubleflat
}
\glissando
a'1_\markup {
  \natural \semisharp \sharp
  \sesquisharp \doublesharp
}
\glissando b'
```



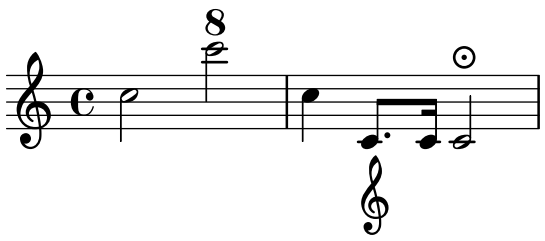
Andere Notationsobjekte können auch eingefügt werden:

```
g1 bes
ees-\markup {
  \finger 4
  \tied-lyric "~"
  \finger 1
}
fis_\markup { \dynamic rf }
bes^\markup {
  \beam #8 #0.1 #0.5
}
cis
d-\markup {
  \markalphabet #8
  \markletter #8
}
```



Allgemeiner gesagt kann jedes verfügbare Notationssymbol unabhängig von der Notation als ein Textbeschriftungsobjekt eingefügt werden, wie unten gezeigt. Eine vollständige Liste der verfügbaren Symbole findet sich in Abschnitt A.8 [Die Emmentaler-Schriftart], Seite 658.

```
\relative {
  c''2
  c'^\markup { \musicglyph "eight" }
  c,4
  c,8._\markup { \musicglyph "clefs.G_change" }
  c16
  c2^\markup { \musicglyph "timesig.neomensural94" }
}
```



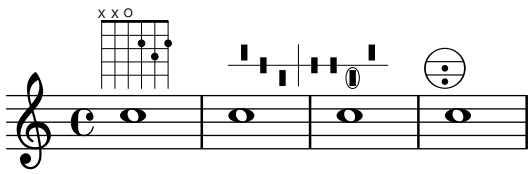
Eine andere Möglichkeit, andere als Textsymbole zu schreiben, findet sich in Abschnitt 8.3.1 [Was sind Schriftarten], Seite 247. Diese Methode bietet sich an, um Klammern unterschiedlicher Größe zu setzen.

Der Textbeschriftungsmodus unterstützt auch Diagramme für bestimmte Instrumente:

```
\relative {
  c''1^\markup {
    \fret-diagram-terse "x;x;o;2;3;2;"
  }
  c^\markup {
    \harp-pedal "^-v|--ov^"
  }
  c
  c^\markup {
    \combine
      \musicglyph "accordion.discant"
      \combine

\raise #0.5 \musicglyph "accordion.dot"
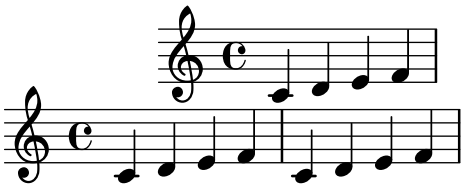
\raise #1.5 \musicglyph "accordion.dot"
  }
}
```



Derartige Digramme sind dokumentiert in Abschnitt A.10.6 [Instrument-specific markup], Seite 734.

Sogar eine ganze Partitur kann in ein Textbeschriftungsobjekt eingefügt werden. In diesem Fall muss die eingefügte `\score`-Umgebung eine `\layout`-Umgebung haben, wie in diesem Beispiel:

```
\relative {
  c'4 d^\markup {
    \score {
      \relative { c'4 d e f }
      \layout { }
    }
  }
  e f |
  c d e f
}
```



Eine vollständige Liste der Musiksymbol-Befehle findet sich in Abschnitt A.10.4 [Markup for music and musical symbols], Seite 721.

Siehe auch

Notationsreferenz: Abschnitt A.10.4 [Markup for music and musical symbols], Seite 721, Abschnitt A.8 [Die Emmentaler-Schriftart], Seite 658, Abschnitt 8.3.1 [Was sind Schriftarten], Seite 247.

Installierte Dateien: `scm/define-markup-commands.scm`, `scm/fret-diagrams.scm`, `scm/harp-pedals.scm`.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

Referenz der Interna: Abschnitt “TextScript” in *Referenz der Interna*.

8.2.6 Textbeschriftung über mehrere Seiten

Normale Textbeschriftungsobjekte können nicht getrennt werden, aber mit einer spezifischen Umgebung ist es möglich, Text auch über mehrere Seiten fließen zu lassen:

```
\markuplist {
  \justified-lines {
    A very long text of justified lines.
    ...
  }
  \wordwrap-lines {
    Another very long paragraph.
    ...
  }
  ...
}
```

A very long text of justified lines. ...

Another very long paragraph. ...

...

Die Syntax braucht eine Liste von Textbeschriftungen folgender Art:

- das Resultat eines Beschriftungslistenbefehls,

- eine Textbeschriftungsliste,
- eine Liste von Beschriftungslisten.

Eine vollständige Liste der Beschriftungslistenbefehle findet sich in Abschnitt A.11 [Textbeschriftungslistenbefehle], Seite 756.

Siehe auch

Notationsreferenz: Abschnitt A.11 [Textbeschriftungslistenbefehle], Seite 756.

Erweitern: Abschnitt “Neue Definitionen von Beschriftungslistenbefehlen” in *Extending*.

Installierte Dateien: `scm/define-markup-commands.scm`.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

Referenz der Interna: Abschnitt “TextScript” in *Referenz der Interna*.

Vordefinierte Befehle

`\markuplist`.

8.3 Schriftarten

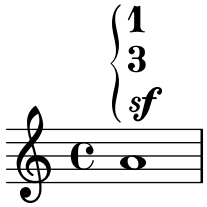
Dieser Abschnitt zeigt, wie Schriftarten eingesetzt werden können und wie man sie in Partituren ändern kann.

8.3.1 Was sind Schriftarten

Schriftarten werden von mehreren Bibliotheken verwaltet. FontConfig wird benützt, um die vorhandenen Schriftarten des Systems zu erkennen, die gewählte Schriftart wird dann mit Pango verarbeitet.

Notationsschriftarten können als eine Ansammlung von besonderen Zeichen erklärt werden, wobei die Sonderzeichen in verschiedene Familien klassifiziert werden. Die Syntax des folgenden Beispiels ermöglicht es, direkt auf verschiedene nicht textuelle Sonderzeichen der `feta`-Glyphe zuzugreifen. Das ist die Standardschriftart für Notationselemente in LilyPond.

```
a'1^\markup {
  \vcenter {
    \override #'(font-encoding . fetaBraces)
    \lookup "brace120"
    \override #'(font-encoding . fetaText)
    \column { 1 3 sf }
    \override #'(font-encoding . fetaMusic)
    \lookup "noteheads.s0petrucci"
  }
}
```



Außer den verschiedenen Klammern, die in `fetaBraces` in verschiedenen Größen enthalten sind, lassen sich alle dieses Symbole auch mit einer einfacheren Syntax notieren. Sie ist beschrieben in Abschnitt 8.2.5 [Musikalische Notation innerhalb einer Textbeschriftung], Seite 243.
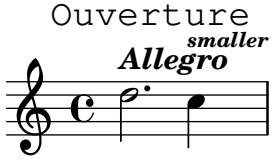
Wenn man die Klammern von `fetaBraces` benutzt, wird die Größe der Klammer durch einen numeralen Part in der Bezeichnung des Glyphs bestimmt. Als Wert kann eine Ganzzahl

von 0 bis 575 benutzt werden, wobei 0 die kleinste Klammern ergibt. Der optimale Wert muss durch Ausprobieren herausgefunden werden. Diese Glyphen sind alle linke Klammern, rechte Klammern lassen sich durch eine Drehung herstellen, siehe Abschnitt 35.8 [Drehen von Objekten], Seite 625.

Drei Textschriftarten sind verfügbar (auf Englisch `family` genannt): mit `roman` eine Schriftart mit Serifen (Standard ist New Century Schoolbook), mit `sans` eine serifenlose (gerade) Schriftart und mit `typewriter` eine Schreibmaschinenschrift, in welcher die Buchstaben alle die gleiche Weite haben. Die aktuelle Schriftart von `sans` und `typewriter` wird durch Pango entsprechend den Systemvorgaben gewählt.

Jede Familie kann verschiedene Schriftschnitte besitzen. Im Englischen wird unterschieden zwischen `shape` für kursive Schnitte und `series` für fette Schnitte. Im folgenden Beispiel wird demonstriert, wie man die verschiedenen Eigenschaften auswählen kann. Der Wert, der `font-size` übergeben wird, entspricht der geforderten Änderung in Bezug auf die Standardschriftgröße.

```
\override Score.RehearsalMark.font-family = #'typewriter
\mark \markup "Ouverture"
\override Voice.TextScript.font-shape = #'italic
\override Voice.TextScript.font-series = #'bold
d''2.^\markup "Allegro"
\override Voice.TextScript.font-size = #-3
c''4^smaller
```



Eine ähnliche Syntax kann im Textbeschriftungsmodus eingesetzt werden, hier bietet es sich aber an, die einfacheren Befehle zu verwenden, die erklärt wurden in Abschnitt 8.2.2 [Überblick über die wichtigsten Textbeschriftungsbefehle], Seite 234:

```
\markup {
  \column {
    \line {
      \override #'((font-shape . italic) (font-size . 4))
      Idomeneo,
    }
    \line {
      \override #'(font-family . typewriter)
      {
        \override #'(font-series . bold)
        re
        di
      }
      \override #'(font-family . sans)
      Creta
    }
  }
}
```



Auch wenn es einfach ist, zwischen den vordefinierten Schriftarten umzuschalten, kann man auch eigene Schriftarten verwenden, wie erklärt in folgenden Abschnitten: Abschnitt 8.3.2 [Schriftarten für einen Eintrag], Seite 249, und Abschnitt 8.3.3 [Schriftart des gesamten Dokuments], Seite 249.

Siehe auch
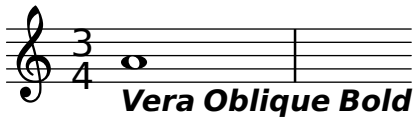
Notationsreferenz: Abschnitt A.8 [Die Emmentaler-Schriftart], Seite 658, Abschnitt 35.8 [Drehen von Objekten], Seite 625, Abschnitt 8.2.5 [Musikalische Notation innerhalb einer Textbeschriftung], Seite 243, Abschnitt 8.2.2 [Überblick über die wichtigsten Textbeschriftungsbefehle], Seite 234, Abschnitt A.10.1 [Font markup], Seite 678.

8.3.2 Schriftarten für einen Eintrag

Jede Schriftart, die über das Betriebssystem installiert ist und von FontConfig erkannt wird, kann in einer Partitur eingefügt werden. Dazu verwendet man folgende Syntax:

```
\override Staff.TimeSignature.font-name = "Bitstream Charter"
\override Staff.TimeSignature.font-size = #2
\time 3/4

a'1_\markup {
  \override #'(font-name . "Bitstream Vera Sans,sans-serif, Oblique Bold")
    { Vera Oblique Bold }
}
```



Mit folgendem Befehl erhält man eine Liste aller verfügbaren Schriftarten des Betriebssystems:

```
lilypond -dshow-available-fonts
```

Siehe auch

Notationsreferenz: Abschnitt 8.3.1 [Was sind Schriftarten], Seite 247, Abschnitt 8.3.3 [Schriftart des gesamten Dokuments], Seite 249.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

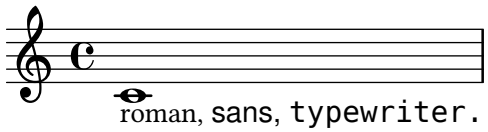
8.3.3 Schriftart des gesamten Dokuments

Es ist auch möglich, die Schriftarten für die gesamte Partitur zu ändern. In diesem Fall müssen die Familien `roman`, `sans` und `typewriter` in genaus dieser Reihenfolge entsprechend der Syntax unten definiert werden. Einzelheiten zu Schriftarten in Abschnitt 8.3.1 [Was sind Schriftarten], Seite 247.

```
\paper  {
  myStaffSize = #20
  property-defaults.fonts.serif = "Linux Libertine O"
  property-defaults.fonts.sans = "Nimbus Sans, Nimbus Sans L"
  property-defaults.fonts.typewriter = "DejaVu Sans Mono"
}

\relative c'{
  c1-\markup {
    roman,
```

```
    \sans sans,
    \typewriter typewriter. }
}
```

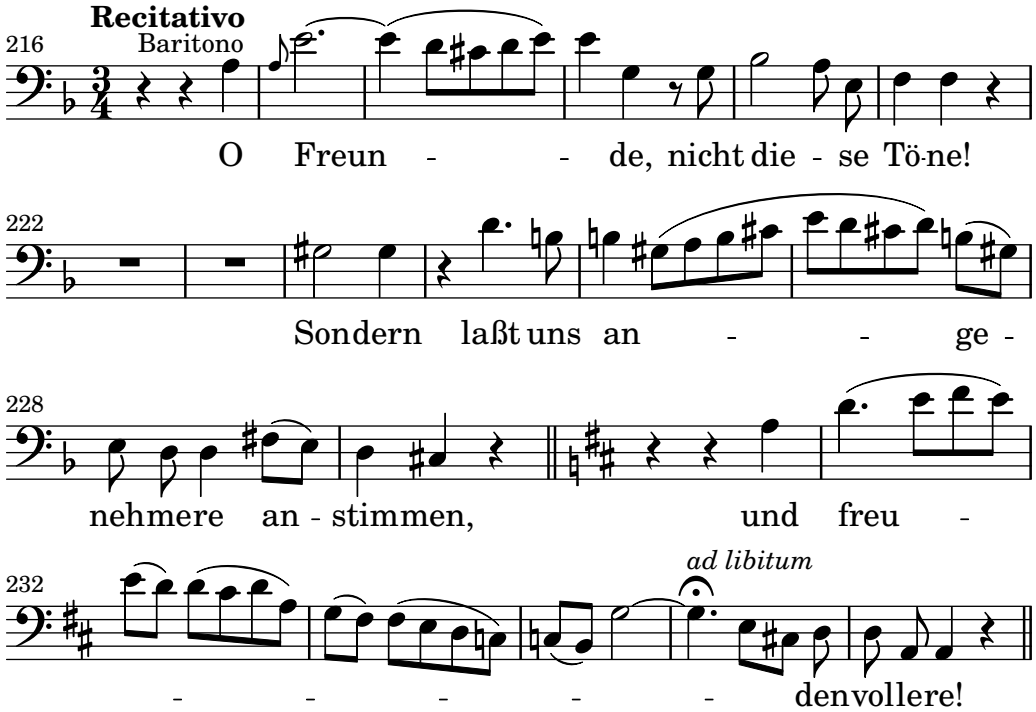


Siehe auch

Notationsreferenz: Abschnitt 8.3.1 [Was sind Schriftarten], Seite 247, Abschnitt 8.3.2 [Schriftarten für einen Eintrag], Seite 249, Abschnitt 8.2.2 [Überblick über die wichtigsten Textbeschriftungsbefehle], Seite 234, Abschnitt A.10.1 [Font markup], Seite 678.

Notation für spezielle Zwecke

9 Notation von Gesang



Dieser Abschnitt erklärt, wie Vokalmusik gesetzt werden kann und die Silben von Gesangstext an den Noten ausgerichtet werden.

9.1 Übliche Notation für Vokalmusik

Dieser Abschnitt erklärt Eigenheiten und Probleme, die die meisten Arten an Vokalmusik gemeinsam haben.

9.1.1 Referenz für Vokalmusik

Dieser Abschnitt, wo man Lösungen zu den Problemen finden kann, die bei der Notation von Gesang mit Text auftreten können.

- Die meisten Vokalmusikstile benutzen Text für den Gesangstext. Eine Einleitung hierzu findet sich in Abschnitt “Einfache Lieder setzen” in *Handbuch zum Lernen*.
- Vokalmusik braucht oft die Benutzung von Textbeschriftung (dem `markup`-Modus) für den Gesangstext oder andere Textelemente (Namen von Figuren usw.). Die entsprechende Syntax ist beschrieben in Abschnitt 8.2.1 [Textbeschriftung (Einleitung)], Seite 233.
- ‚Ambitus‘ können zu Beginn der Stimmen hinzugefügt werden, dies findet sich erklärt in Abschnitt 1.3.6 [Tonumfang], Seite 33.
- Dynamikbezeichnung werden normalerweise unter das Notensystem platziert, aber in Chormusik werden sie normalerweise über das Notensystem notiert, um Platz für den Text zu schaffen. Siehe Abschnitt 9.5.2 [Partiturbeispiele für Chormusik], Seite 290.

Siehe auch

Glossar: Abschnitt “ambitus” in *Glossar*.

Handbuch zum Lernen: Abschnitt “Einfache Lieder setzen” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 8.2.1 [Textbeschriftung (Einleitung)], Seite 233, Abschnitt 1.3.6 [Tonumfang], Seite 33, Abschnitt 9.5.2 [Partiturbeispiele für Chormusik], Seite 290.

Snippets: Abschnitt “Vocal music” in *Schnipsel*.

9.1.2 Eingabe von Text

Gesangstext muss in einem speziellen Modus notiert werden. Der Gesangstextmodus kann mit der Umgebung `\lyricmode` angegeben werden, oder indem `\addlyrics` bzw. `\lyricsto` eingesetzt wird. In diesem Modus kann Text mit Akzenten und Satzzeichen notiert werden, und das Programm liest `d` nicht als die Tonhöhe *D*, sondern als eine Silbe Text. Anders gesagt: Silben werden wie Noten notiert, aber die Tonhöhen werden durch Text ersetzt.

Beispielsweise:

```
\lyricmode { Gern4 hätt'4 ich4 dich4 lieb!2 }
```

Es gibt zwei generelle Methoden, die horizontale Orientierung der Textsilben anzugeben, entweder indem ihre Dauer angegeben wird, wie oben in dem Beispiel, oder indem die Silben automatisch an den Noten ausgerichtet werden. Dazu muss entweder `\addlyrics` oder `\lyricsto` eingesetzt werden. Die erste Methode ist beschrieben in Abschnitt 9.1.5 [Manuelle Silbendauern], Seite 260, die zweite in Abschnitt 9.1.4 [Automatische Silbendauern], Seite 257.

Ein Wort oder eine Silbe beginnt mit einem alphabetischen Zeichen (inklusive einige andere Zeichen, siehe unten) und endet mit einem Leerzeichen oder einer Zahl. Die folgenden Zeichen in der Silbe können beliebig sein, außer Leerzeichen und Zahlen.

Jedes Zeichen, das nicht Leerzeichen noch Zahl ist, wird als Bestandteil der Silbe angesehen. Eine Silbe kann also auch mit `}` enden, was oft zu dem Fehler

```
\lyricmode { lah- lah}
```

führen kann. Hier wird `}` als Teil der letzten Silbe gerechnet, so dass die öffnende Klammer keine schließende Klammer hat und die Eingabedatei nicht funktioniert. Klammern sollten deshalb immer von Leerzeichen umgeben sein.

```
\lyricmode { lah lah lah }
```

Auch ein Punkt, der auf eine Silbe folgt, wird in die Silbe inkorporiert. Infolgedessen müssen auch um Eigenschaftsbezeichnungen Leerzeichen gesetzt werden. Ein Befehl heißt also *nicht*:
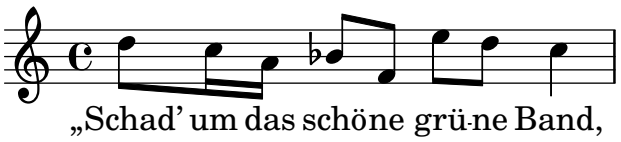
```
\override Score.LyricText.font-shape = #'italic
```

sondern

```
\override Score.LyricText.font-shape = #'italic
```
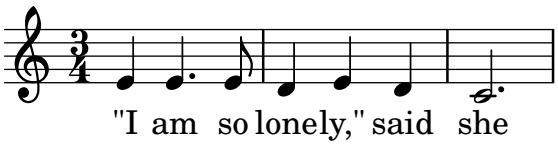
Punkte, Gesangstext mit Akzenten, verschiedene lateinische und nicht-lateinische Zeichen sowie auch etwa Sonderzeichen (wie ein Herz-Symbol) können direkt in die Notationsdatei geschrieben werden. Es muss dabei sichergestellt werden, dass die Datei in der UTF-8-Kodierung gespeichert wird. Zu mehr Information siehe Abschnitt 21.3 [Sonderzeichen], Seite 501.

```
\relative { d''8 c16 a bes8 f e' d c4 }
\addlyrics { „Schad' um das schö -- ne grü -- ne Band, }
```



Normale Anführungszeichen können im Gesangstext auch benutzt werden, aber sie müssen mit einem Backslash und weiteren Anführungszeichen begleitet werden:

```
\relative { \time 3/4 e'4 e4. e8 d4 e d c2. }
\addlyrics { "\"I" am so lone -- "ly,\"" said she }
```



Die vollständige Definition des Anfangs eines Wortes in LilyPond ist etwas komplizierter. Ein Wort im Gesangstextmodus beginnt mit einem alphabetischen Zeichen, `_`, `?`, `!`, `:`, `'`, den Kontrollzeichen `^A` bis `^F`, `^Q` bis `^W`, `^Y`, `^^`, einem beliebigen 8-bit-Zeichen mit einem ASCII-Code größer als 127 oder einer zwei-Zeichen-Kombination aus einem Backslash, gefolgt von `` ` ``, `'`, `"` oder `^`.

Man hat sehr gut Möglichkeiten, die Erscheinung des Gesangstextes zu beeinflussen, wenn man dafür Textbeschriftungsbefehle einsetzt. Siehe hierzu Abschnitt 8.2 [Text formatieren], Seite 233.
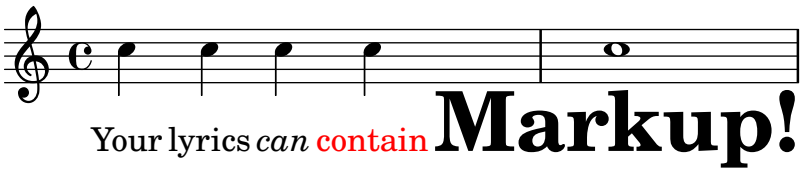
Ausgewählte Schnipsel

Silben im Gesangstext formatieren

Textbeschriftungsmodus kann eingesetzt werden, um individuelle Silben im Gesangstext zu formatieren.

```
mel = \relative c'' { c4 c c c c1 }
lyr = \lyricmode {
  Your lyrics \markup { \italic can }
  \markup { \with-color #red contain }
  \markup { \fontsize #8 \bold Markup! }
}

<<
  \new Voice = "melody" \mel
  \new Lyrics \lyricsto "melody" \lyr
>>
```



Siehe auch

Handbuch zum Lernen: Abschnitt “Lieder” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 9.1.4 [Automatische Silbendauern], Seite 257, Abschnitt 8.3 [Schriftarten], Seite 247, Abschnitt 35.1 [Eingabe-Modi], Seite 610, Abschnitt 9.1.5 [Manuelle Silbendauern], Seite 260, Abschnitt 8.2 [Text formatieren], Seite 233, Abschnitt 21.3 [Sonderzeichen], Seite 501.

Referenz der Interna: Abschnitt “LyricText” in *Referenz der Interna*.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

9.1.3 Text an einer Melodie ausrichten

Gesangstext wird gesetzt, wenn er in einem `Lyrics`-Kontext ausgewertet wird, siehe Abschnitt 32.1 [Was sind Kontexte?], Seite 580.

```
\new Lyrics \lyricmode { ... }
```

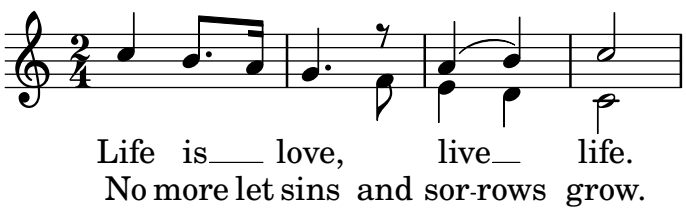
Gesangstext kann an einer Melodie auf zwei Arten ausgerichtet werden:

- Gesangstext kann automatisch ausgerichtet werden, wobei die Dauern von einer Notenstimme oder (in besonderen Umständen) von einer assoziierten Melodie übernommen werden. Das geschieht mit `\addlyrics`, `\lyricsto` oder indem man die `associatedVoice`-Eigenschaft einsetzt. Mehr Informationen in Abschnitt 9.1.4 [Automatische Silbendauern], Seite 257.

```
<<
  \new Staff <<
    \time 2/4
    \new Voice = "one" \relative {
      \voiceOne
      c''4 b8. a16 g4. r8 a4 ( b ) c2
    }
    \new Voice = "two" \relative {
      \voiceTwo
       s2 s4. f'8 e4 d c2
    }
  >>

% takes durations and alignment from notes in "one"
  \new Lyrics \lyricsto "one" {
    Life is __ _ love, live __ life.
  }

% takes durations and alignment from notes in "one" initially
% then switches to "two"
  \new Lyrics \lyricsto "one" {
    No more let
    \set associatedVoice = "two"  % must be set one syllable early
    sins and sor -- rows grow.
  }
>>
```



Die erste Strophe zeigt, wie normalerweise Text gesetzt wird.

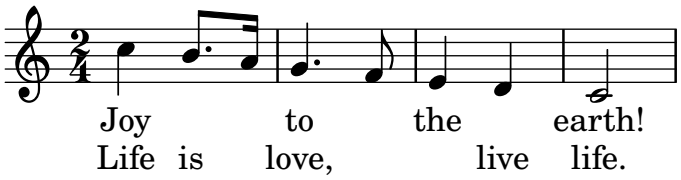
Die zweite Strophe zeigt, wie die Stimme (`Voice`), von der die Dauern entnommen werden, geändert werden kann. Das ist nützlich, wenn der Text unterschiedlicher Strophen die Noten auf unterschiedliche Weise füllt und alle Dauern als `Voice`-Kontexte vorhanden sind. Zu mehr Details siehe Abschnitt 9.3 [Strophen], Seite 281.

- Gesangstext kann unabhängig von den Notendauern platziert werden, indem man die Dauern der Silben explizit vorgibt und den Text innerhalb von `\lyricmode` notiert:

```
<<
  \new Voice = "one" \relative {
    \time 2/4
    c''4 b8. a16 g4. f8 e4 d c2
  }

% uses previous explicit duration of 2;
  \new Lyrics \lyricmode {
    Joy to the earth!
  }

% explicit durations, set to a different rhythm
  \new Lyrics \lyricmode {
    Life4 is love,2. live4 life.2
  }
>>
```



Die erste Strophe wird nicht an den Noten ausgerichtet, weil die Dauern nicht angegeben sind, und der erste Wert 2 für alle Silben benutzt wird.

Die zweite Strophe zeigt, dass die Silben sehr unabhängig von den Noten notiert werden können. Das ist nützlich, wenn der Text von verschiedenen Strophen die Noten auf unterschiedliche Weise füllt, aber die Dauer nicht in einem Noten-Kontext vorhanden ist. Mehr Details finden sich in Abschnitt 9.1.5 [Manuelle Silbendauern], Seite 260. Diese Technik ist auch nützlich, wenn man Dialog zu einer Musik setzt; ein Beispiel hierzu findet sich in Abschnitt 9.6.5 [Dialog zur Musik], Seite 299.

Wenn Text auf diese Weise eingegeben wird, werden die Silben links an den Noten ausgerichtet, können aber auch zentriert werden, indem man eine assoziierte Stimme angibt, wenn eine existiert. Zu Einzelheiten siehe Abschnitt 9.1.5 [Manuelle Silbendauern], Seite 260.

Siehe auch

Handbuch zum Lernen: Abschnitt “Text an einer Melodie ausrichten” in *Handbuch zum Lernen*.

Notation Reference: Abschnitt 32.1 [Was sind Kontexte?], Seite 580, Abschnitt 9.1.4 [Automatische Silbendauern], Seite 257, Abschnitt 9.3 [Strophen], Seite 281, Abschnitt 9.1.5 [Manuelle Silbendauern], Seite 260, Abschnitt 9.6.5 [Dialog zur Musik], Seite 299.

Referenz der Interna: Abschnitt “Lyrics” in *Referenz der Interna*.

9.1.4 Automatische Silbendauern

Die Silben des Gesangstextes können automatisch an einer Melodie ausgerichtet werden. Das ist auf drei Arten möglich:

- indem man einen benannten `Voice`-Kontext mit der Melodie durch `\lyricsto` zuweist,
- indem man den Gesangstext mit `\addlyrics` beginnt und direkt nach dem `Voice`-Kontext mit der Melodie notiert,
- indem man die `associatedVoice`-Eigenschaft definiert, sodass man die Ausrichtung des Gesangstextes zwischen verschiedenen benannten `Voice`-Kontexten gewechselt werden.

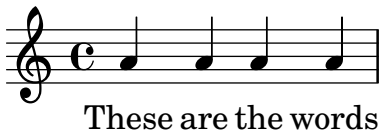
In allen drei Methoden können Bindestriche zwischen den Silben oder Fülllinien hinter einem Wortende gezogen werden. Zu Einzelheiten siehe Abschnitt 9.1.8 [Fülllinien und Trennstriche], Seite 265.

Der `Voice`-Kontext, der die Melodie enthält, an der der Text ausgerichtet werden soll, darf noch nicht „gestorben“ sein, weil sonst aller Text danach verloren geht. Das kann passieren, wenn es Strecken gibt, in denen die Stimme pausiert. Zu Methoden, wie man Kontexte am Leben erhält, siehe Abschnitt 32.3 [Kontexte am Leben halten], Seite 583.

`\lyricsto` Benutzen

Gesangstext kann an einer Melodie automatisch ausgerichtet werdne, indem man den beannten `Voice`-Kontext mit der Melodie durch den Befehl `\lyricsto` angibt:

```
<<
  \new Voice = "melody" {
    a4 a a a
  }
  \new Lyrics \lyricsto "melody" {
    These are the words
  }
>>
```



Damit wird der Text an den Noten des benannten `Voice`-Kontextes ausgerichtet, der schon vorher existieren muss. Aus diesem Grund wird der `Voice`-Kontext normalerweise zuerst definiert, gefolgt vom `Lyrics`-Kontext. Der Gesangstext selber folgt dem `\lyricsto`-Befehl. Der `\lyricsto`-Befehl ruft den Gesangstextmodus automatisch auf, sodass man `\lyricmode` in diesem Fall auslassen kann. Standardmäßig werden die Silben unter den Noten angeordnet. Für andere Optionen siehe Abschnitt 9.2.2 [Gesangstext vertikal verschieben], Seite 267.

`\addlyrics` benutzen

Der `\addlyrics`-Befehl ist eigentlich nur eine Abkürzung für eine etwas kompliziertere LilyPond-Struktur, den man manchmal aus Bequemlichkeit einsetzen kann.
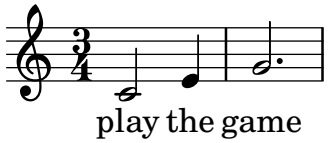
```
{ Noten }
\addlyrics { Gesangstext }
```

bedeutet das Gleiche wie

```
\new Voice = "bla" { Noten }
\new Lyrics \lyricsto "bla" { Gesangstext }
```
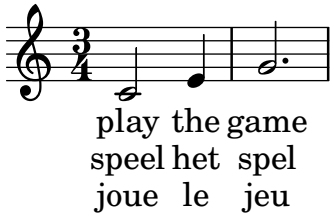
Hier ein Beispiel:

```
{
  \time 3/4
  \relative { c'2 e4 g2. }
  \addlyrics { play the game }
}
```



Weitere Strophen können mit weiteren `\addlyrics`-Abschnitten hinzugefügt werden:

```
{
  \time 3/4
  \relative { c'2 e4 g2. }
  \addlyrics { play the game }
  \addlyrics { speel het spel }
  \addlyrics { joue le jeu }
}
```



Der Befehl `\addlyrics` kann keine polyphonen Situationen bewältigen. In diesen Fällen sollen man `\lyricsto` benutzen.

`associatedVoice` benutzen
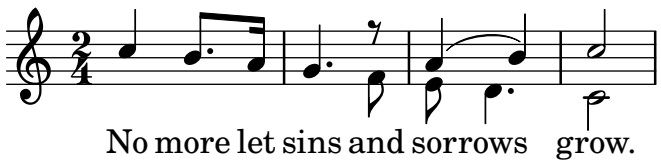
Die Melodie, an die der Gesangstext ausgerichtet wird, kann durch Setzen der `associatedVoice`-Eigenschaft geändert werden:

```
\set associatedVoice = "lala"
```

Der Wert der Eigenschaft (hier `"lala"`) ist die Bezeichnung eines `Voice`-Kontextes. Aus technischen Gründen muss der `\set`-Befehl eine Silbe vor der Silbe gesetzt werden, auf die er wirken soll.

Ein Beispiel demonstriert das:

```
<<
  \new Staff <<
    \time 2/4
    \new Voice = "one" \relative {
      \voiceOne
      c''4 b8. a16 g4. r8 a4 ( b ) c2
    }
    \new Voice = "two" \relative {
      \voiceTwo
       s2 s4. f'8 e8 d4. c2
    }
  >>
% takes durations and alignment from notes in "one" initially
% then switches to "two"
  \new Lyrics \lyricsto "one" {
    No more let
    \set associatedVoice = "two"  % must be set one syllable early
    sins and sor -- rows grow.
  }
>>
```



Siehe auch

Notationsreferenz: Abschnitt 9.1.8 [Fülllinien und Trennstriche], Seite 265, Abschnitt 32.3 [Kontexte am Leben halten], Seite 583, Abschnitt 9.2.2 [Gesangstext vertikal verschieben], Seite 267.
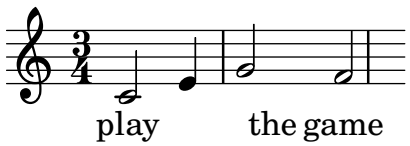
9.1.5 Manuelle Silbendauern

In komplexer Vokalmusik kann es nötig sein, den Gesangstext vollkommen unabhängig von den Noten zu positionieren. In diesem Fall sollte man nicht `\addlyrics` bzw. `\lyricsto` benutzen, und auch keine `associatedVoice` definieren. Die Silben werden wie Noten notiert – indem die Tonhöhen durch den Text der Silbe ersetzt werden – und die Dauer jeder Silbe muss angegeben werden.
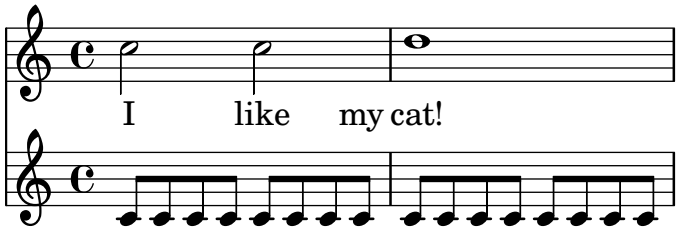
Standardmäßig werden die Silben links am entsprechenden musikalischen Moment ausgerichtet. Bindestriche zwischen den Silben können wie üblich gezogen werden, aber Fülllinien hinter dem Wortende können nicht gezogen werden, wenn es keine mit dem Text verknüpfte Melodie gibt.

Hier zwei Beispiele:

```
<<
  \new Voice = "melody" {
    \time 3/4
    c2 e4 g2 f
  }
  \new Lyrics \lyricmode {
    play1 the4 game4
  }
>>
```



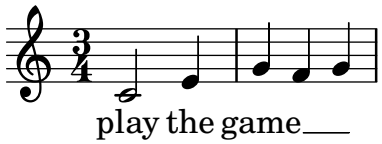

```
<<
  \new Staff {
    \relative {
      c''2 c2
      d1
    }
  }
  \new Lyrics {
    \lyricmode {
      I2 like4. my8 cat!1
    }
  }
  \new Staff {
    \relative {
      c'8 c c c c c c c
      c8 c c c c c c c
    }
  }
>>
```



Diese Technik ist nützlich, wenn man Dialog zur Musik schreiben will, siehe Abschnitt 9.6.5 [Dialog zur Musik], Seite 299.

Um Silben an den Noten des entsprechenden musikalischen Moments zu zentrieren, muss `associatedVoice` auf die Bezeichnung des Stimmen-Kontext eingestellt werden, in dem sich die Noten befinden. Wenn `associatedVoice` definiert ist, können doppelte Bindestriche zwischen Silben und doppelte Unterstriche hinter Wörtern für Fülllinien benutzt werden:

```
<<
  \new Voice = "melody" {
    \time 3/4
    c2 e4 g f g
  }
  \new Lyrics \lyricmode {
    \set associatedVoice = "melody"
    play2 the4 game2. __
  }
>>
```



Siehe auch
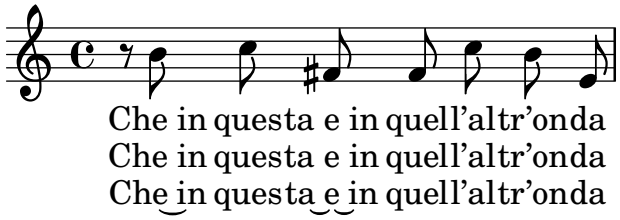
Notationsreferenz: Abschnitt 9.6.5 [Dialog zur Musik], Seite 299, Abschnitt 32.3 [Kontexte am Leben halten], Seite 583.

Referenz der Interna: Abschnitt “Lyrics” in *Referenz der Interna*, Abschnitt “Voice” in *Referenz der Interna*.

9.1.6 Mehrere Silben zu einer Note

Um mehr als eine Silbe zu einer Note zuzuordnen, können die Silben mit geraden Anführungszeichen (`"`) umgeben werden oder ein Unterstrich (`_`) benutzt werden, um ein Leerzeichen zwischen Silben zu setzen. Mit der Tilde (`~`) kann ein Bindebogen gesetzt werden.

```
{
  \relative { \autoBeamOff
    r8 b' c fis, fis c' b e, }
  \addlyrics { Che_in ques -- ta_e_in quel -- l'al -- tr'on -- da }
  \addlyrics { "Che in" ques -- "ta e in" quel -- l'al -- tr'on -- da }
  \addlyrics { Che~in ques -- ta~e~in quel -- l'al -- tr'on -- da }
}
```



Siehe auch

Referenz der Interna: Abschnitt "LyricCombineMusic" in *Referenz der Interna*.

9.1.7 Mehrere Noten zu einer Silbe

Öfters, insbesondere in Alter Musik, wird eine einzige Silbe zu mehreren Noten gesungen, was als Melisma bezeichnet wird. Die Silbe eines Melismas wird normalerweise links an der ersten Note des Melismas ausgerichtet.

Melismen können direkt im Gesangstext definiert werden, indem ein Unterstrich (_) für jede Note notiert wird, die übersprungen werden soll.
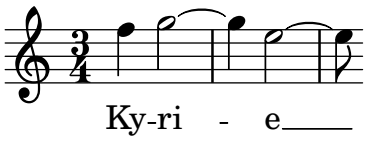
Wenn ein Melisma an einer Silbe auftritt, die nicht die letzte eines Wortes ist, wird diese Silbe mit der folgenden durch wiederholte Trennstriche verbunden. Dies wird notiert, indem man zwei Trennstriche (--) nach der Silbe notiert.

Wenn ein Melisma an der letzten Silbe eines Wortes auftritt, wird eine Fülllinie vom Ende der Silbe bis zur letzten Note des Melismas gezeichnet. Das wird durch zwei Unterstriche (__) nach der Silbe notiert.

Es gibt fünf Arten, auf die ein Melisma angezeigt werden kann:
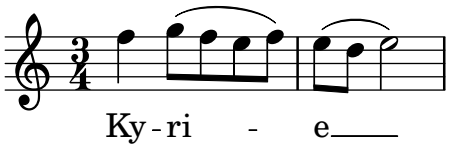
- Melismen werden automatisch zu Noten erstellt, die übergebunden sind:

```
<<
  \new Voice = "melody" \relative {
    \time 3/4
    f''4 g2 ~ |
    4 e2 ~ |
    8
  }
  \new Lyrics \lyricsto "melody" {
    Ky -- ri -- e __
  }
>>
```



- Melismen können automatisch aus den Noten erstellt werden, indem man Legatobögen über den Noten eines Melismas notiert. Auf diese Weise wird Gesangstext üblicherweise notiert:
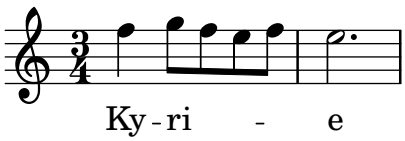
```
<<
  \new Voice = "melody" \relative {
    \time 3/4
    f''4 g8 ( f e f )
    e8 ( d e2 )
  }
  \new Lyrics \lyricsto "melody" {
    Ky -- ri -- e __
  }
>>
```



Dabei ist zu beachten, dass Phrasierungsbogen die Erstellung eines Melsima nicht beeinflussen.

- Noten werden als ein Melisma betrachtet, wenn sie manuell mit einem Balken versehen werden, vorausgesetzt, dass die automatische Bebalkung ausgeschaltet ist. Siehe Abschnitt 2.4.2 [Einstellung von automatischen Balken], Seite 84.

```
<<
  \new Voice = "melody" \relative {
    \time 3/4
    \autoBeamOff
    f''4 g8[ f e f]
    e2.
  }
  \new Lyrics \lyricsto "melody" {
    Ky -- ri -- e
  }
>>
```
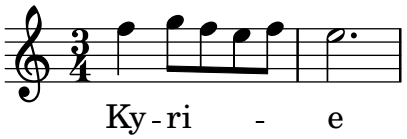


Diese Methode eignet sich natürlich nicht für Noten, die länger als Achtel sind.

- Eine Gruppe von Noten ohne Legatobogen werden als Melisma betrachtet, wenn sie zwischen `\melisma` und `\melismaEnd` eingeschlossen sind:
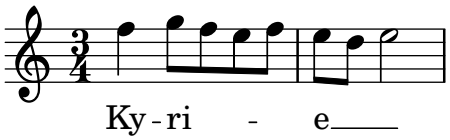
```
<<
  \new Voice = "melody" \relative {
    \time 3/4
    f''4 g8
    \melisma
    f e f
    \melismaEnd
    e2.
  }
  \new Lyrics \lyricsto "melody" {
    Ky -- ri -- e
```
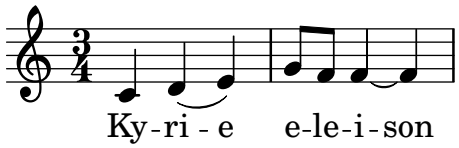
```
  }
>>
```



- Ein Melisma kann auch ausschließlich im Gesangstext notiert werden, indem man einzelne Unterstriche (_) für jede Note eingibt, die zum Melisma hinzugefügt werden soll.

```
<<
  \new Voice = "melody" \relative {
    \time 3/4
    f''4 g8 f e f
    e8 d e2
  }
  \new Lyrics \lyricsto "melody" {
    Ky -- ri -- _ _ _ e __ _ _
  }
>>
```



Man kann durchaus auch Binde- und Legatobögen sowie manuelle Balken benutzen, ohne dass sie Melismen bezeichnen, wenn `melismaBusyProperties` aufgerufen wird:

```
<<
  \new Voice = "melody" \relative {
    \time 3/4
    \set melismaBusyProperties = #'()
    c'4 d ( e )
    g8 [ f ] f4 ~ 4
  }
  \new Lyrics \lyricsto "melody" {
    Ky -- ri -- e e -- le -- i -- son
  }
>>
```
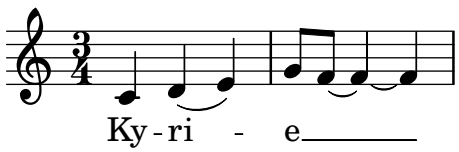


Andere Einstellungen für `melismaBusyProperties` können bneutzt werden, um nur eine Auswahl von Binde-, Legatobögen und Balken zur automatischen Melismenerkennung gelten zu lassen. Siehe `melismaBusyProperties` in Abschnitt “Tunable context properties” in *Referenz der Interna*.

Alternativ kann auch `ignoreMelismata` auf wahr gesetzt werden, wenn alle Melisma-Bezeichnungen ignoriert werden sollen, siehe Abschnitt 9.3.4 [Strophen mit unterschiedlichem Rhythmus], Seite 282.

Wenn ein Melisma während einer Passage benötigt wird, in der `melismaBusyProperties` aktiviert ist, kann es durch einen einzelnen Unterstrich im Gesangstext für jede Note des Melismas angegeben werden:

```
<<
  \new Voice = "melody" \relative {
    \time 3/4
    \set melismaBusyProperties = #'()
    c'4 d ( e )
    g8 [ f ] ~ 4 ~ f
  }
  \new Lyrics \lyricsto "melody" {
    Ky -- ri -- _ e __ _ _ _
  }
>>
```



Vordefinierte Befehle

`\autoBeamOff`, `\autoBeamOn`, `\melisma`, `\melismaEnd`.

Siehe auch

Glossar: Abschnitt “melisma” in *Glossar*.

Handbuch zum Lernen: Abschnitt “Text an einer Melodie ausrichten” in *Handbuch zum Lernen*.

Notation Reference: Abschnitt 32.3 [Kontexte am Leben halten], Seite 583, Abschnitt 9.1.4 [Automatische Silbendauern], Seite 257, Abschnitt 2.4.2 [Einstellung von automatischen Balken], Seite 84, Abschnitt 9.3.4 [Strophen mit unterschiedlichem Rhythmus], Seite 282.

Internals Reference: Abschnitt “Tunable context properties” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Fülllinien zu Melismen werden nicht automatisch erkannt, sondern müssen manuell gesetzt werden.

9.1.8 Fülllinien und Trennstriche

Wenn die letzte Silbe eines Wortes auf ein Melisma fällt, wird das Melisma oft mit einer langen horizontalen Linie angezeigt, die nach dem Wort beginnt und mit der letzten Note des Melismas endet. Derartige Fülllinien werden mit einem doppelten Unterstrich (`__`) eingegeben, wobei beachtet werden muss, dass er von Leerzeichen umgeben ist.

> **Achtung:** Melismen werden mit Fülllinien angezeigt, die als doppelter Unterstrich notiert sind. Kurze Melismen können auch notiert werden, indem eine Note übersprungen wird. Hierzu wird ein einfacher Unterstrich notiert und keine Fülllinie gezogen.

Zentrierte Bindestriche zwischen den einzelnen Silben werden mit einem doppelten Bindestrich (`--`) eingegeben, wobei beachtet werden muss, dass er von Leerzeichen umgeben ist. Der Bindestrich wird zwischen den Silben zentriert und seine Länge dem Notenabstand angepasst.

In sehr eng notierter Musik können die Bindestriche ganz wegfallen. Dieses Verhalten kann aber auch unterbunden werden, wenn den Eigenschaften `minimum-distance` (minimaler Abstand zwischen Silben) und `minimum-length` (Wert, unterhalb von dem Bindestriche wegfallen) andere Werte erhalten. Beide sind Eigenschaften von `LyricHyphen`.

Siehe auch
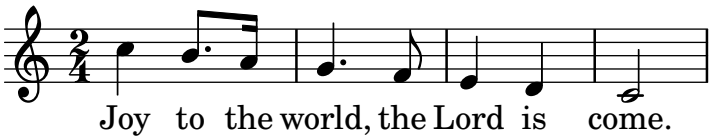
Referenz der Interna: Abschnitt "LyricExtender" in *Referenz der Interna*, Abschnitt "LyricHyphen" in *Referenz der Interna*.

9.2 Techniken für die Gesangstextnotation

9.2.1 Mit Gesangstexten und Bezeichnern arbeiten

Um Variablen zu definieren, die Gesangstext beinhalten, muss die `\lyricmode`-Umgebung benutzt werden.

```
musicOne = \relative {
  c''4 b8. a16 g4. f8 e4 d c2
}
verseOne = \lyricmode {
  Joy to the world, the Lord is come.
}
\score {
  <<
    \new Voice = "one" {
      \time 2/4
      \musicOne
    }
    \new Lyrics \lyricsto "one" {
      \verseOne
    }
  >>
}
```
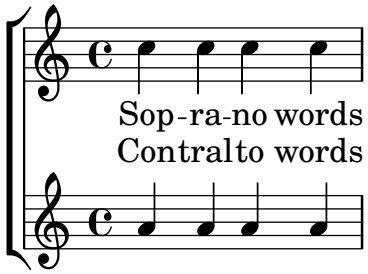


Dauern müssen nicht angegeben werden, wenn die Variable im Zusammenhang mit `\addlyrics` oder `\lyricsto` aufgerufen wird.

Für eine andere Anordnung oder kompliziertere Situationen bietet es sich an, zuerst Systeme und Gesangstextumgebungen zu definieren, dann die Hierarchie von Systemen und Gesangstextzeilen aufzustellen ohne den Gesangstext selber aufzurufen, und dann den Gesangstext mit `\context` darunter aufzurufen. Das stellt sicher, dass die Stimmen, die durch `\lyricsto` angefordert werden, auch immer schon definiert sind. Beispielsweise:

```
sopranoMusic = \relative { c''4 c c c }
contraltoMusic = \relative { a'4 a a a }
sopranoWords = \lyricmode { Sop -- ra -- no words }
contraltoWords = \lyricmode { Con -- tral -- to words }

\score {
  \new ChoirStaff <<
```

```
    \new Staff {
      \new Voice = "sopranos" {
        \sopranoMusic
      }
    }
    \new Lyrics = "sopranos"
    \new Lyrics = "contraltos"
    \new Staff {
      \new Voice = "contraltos" {
        \contraltoMusic
      }
    }
    \context Lyrics = "sopranos" {
      \lyricsto "sopranos" {
        \sopranoWords
      }
    }
    \context Lyrics = "contraltos" {
      \lyricsto "contraltos" {
        \contraltoWords
      }
    }
  >>
}
```



Siehe auch

Notationsreferenz: Abschnitt 9.2.2 [Gesangstext vertikal verschieben], Seite 267.
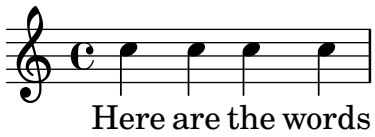
Referenz der Interna: Abschnitt “LyricCombineMusic” in *Referenz der Interna*, Abschnitt “Lyrics” in *Referenz der Interna*.

9.2.2 Gesangstext vertikal verschieben

Abhängig von der Art der Musik kann der Gesangstext über oder unter einem Notensystem oder zwischen zwei Systemen positioniert werden. Es ist am einfachsten, den Text unter das verknüpfte System zu positionieren, was man erreicht, indem man den `Lyrics`-Kontext direkt unter dem System definiert:

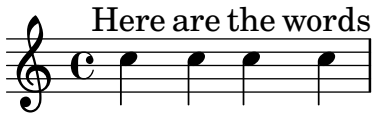
```
\score {
  <<
    \new Staff {
      \new Voice = "melody" {
        \relative { c''4 c c c }
      }
    }
    \new Lyrics {
```

```
      \lyricsto "melody" {
        Here are the words
      }
    }
  >>
}
```



Gesangstext kann auch über dem System positioniert werden, indem man eine der beiden folgenden Methoden benutzt. Die einfachste (und bevorzugte) Methode ist es, die gleiche Syntax wie oben gezeigt einzusetzen und die Position des Gesangstextes explizit anzugeben:

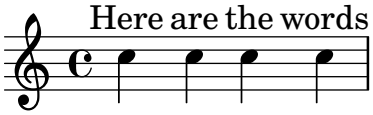
```
\score {
  <<
    \new Staff = "staff" {
      \new Voice = "melody" {
        \relative { c''4 c c c }
      }
    }
    \new Lyrics \with { alignAboveContext = "staff" } {
      \lyricsto "melody" {
        Here are the words
      }
    }
  >>
}
```



Alternativ kann auch ein zweistufiger Prozess benutzt werden. Zuerst wird der Gesangstextkontext definiert (ohne jeglichen Inhalt), bevor Stimm- und Systemkontexte definiert wurden. Dann wird der `\lyricsto`-Befehl nach der Definition der Stimme, auf die er verweist, mit `\context` angegeben:

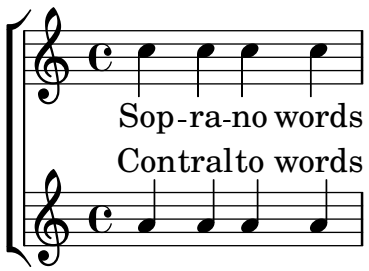
```
\score {
  <<
    \new Lyrics = "lyrics" \with {
      % lyrics above a staff should have this override
      \override VerticalAxisGroup.staff-affinity = #DOWN
    }
    \new Staff {
      \new Voice = "melody" {
        \relative { c''4 c c c }
      }
    }
    \context Lyrics = "lyrics" {
      \lyricsto "melody" {
        Here are the words
      }
```

```
    }
  >>
}
```



Wenn zwei Stimmen sich auf unterschiedlichen Systemen befinden, kann der Text zwischen den Systemen platziert werden, wobei beide der Methoden einesetzt werden können. Hier ein Beispiel für die zweite Methode:

```
\score {
  \new ChoirStaff <<
    \new Staff {
      \new Voice = "sopranos" {
        \relative { c''4 c c c }
      }
    }
    \new Lyrics = "sopranos"
    \new Lyrics = "contraltos" \with {
      % lyrics above a staff should have this override
      \override VerticalAxisGroup.staff-affinity = #DOWN
    }
    \new Staff {
      \new Voice = "contraltos" {
        \relative { a'4 a a a }
      }
    }
    \context Lyrics = "sopranos" {
      \lyricsto "sopranos" {
        Sop -- ra -- no words
      }
    }
    \context Lyrics = "contraltos" {
      \lyricsto "contraltos" {
        Con -- tral -- to words
      }
    }
  >>
}
```



Andere Kombinationen von Gesangstext und System können erstellt werdne, indem man die gegebenen Beispiele modifiziert oder auch die Abschnitt “Vokalensemble” in *Handbuch zum Lernen*-Vorlagen im Handbuch zum Lernen heranzieht.

Siehe auch

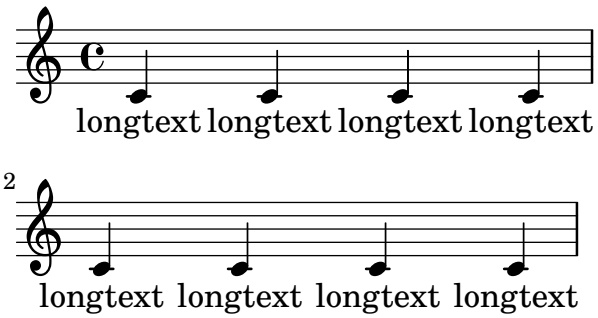
Handbuch zum Lernen: Abschnitt “Vokalensemble” in *Handbuch zum Lernen*.

Notation Reference: Abschnitt 32.7 [Reihenfolge des Kontextlayouts], Seite 595, Abschnitt 32.2 [Kontexte erstellen und referenzieren], Seite 582.

9.2.3 Silben horizontal verschieben
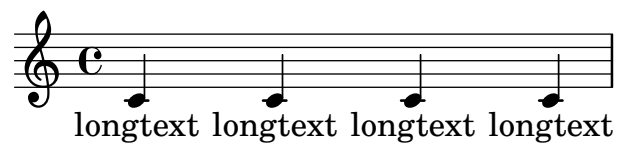
Um den Abstand zwischen Silben zu vergrößern, kann die `minimum-distance`-Eigenschaft des `LyricSpace`-Objekts gesetzt werden:
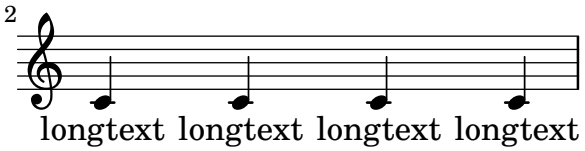
```
\relative c' {
  c c c c
  \override Lyrics.LyricSpace.minimum-distance = #1.0
  c c c c
}
\addlyrics {
  longtext longtext longtext longtext
  longtext longtext longtext longtext
}
```



Damit diese Einstellung für alle Gesangtextzeilen in einer Partitur wirkt, muss sie im `layout`-Block vorgenommen werden.

```
\score {
  \relative {
  c' c c c
  c c c c
  }
  \addlyrics {
  longtext longtext longtext longtext
  longtext longtext longtext longtext
  }
  \layout {
    \context {
      \Lyrics
      \override LyricSpace.minimum-distance = #1.0
    }
  }
}
```





Ausgewählte Schnipsel
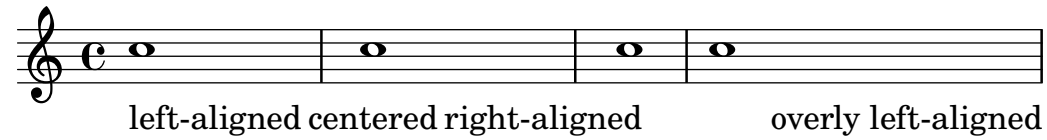
Ausrichtung von Gesangstext

Die horizontale Ausrichtung von Gesangstext kann eingestellt werden, indem man die `self-alignment-X`-Eigenschaft des `LyricText`-Objekts verändert. `#-1` bedeutet links, `#0` bedeutet mittig und `#1` bedeutet rechts, man kann aber genauso gut auch `#LEFT`, `#CENTER` und `#RIGHT` benutzen.

```
\layout {
  ragged-right = ##f
}

\relative c'' {
  c1 c c c
}

\addlyrics {
  \once \override LyricText.self-alignment-X = #LEFT
  "left-aligned"
  \once \override LyricText.self-alignment-X = #CENTER
  "centered"
  \once \override LyricText.self-alignment-X = 1
  "right-aligned"
  \once \override LyricText.self-alignment-X = #-1.5
  "overly left-aligned"
}
```



Eine Überprüfung, mit der sichergestellt wird, dass kein Text in die Seitenränder ragt, verlangt zusätzliche Computerzeit. Um den Notensatz etwas zu beschleunigen, kann die Überprüfung abgestellt werden:

```
\override Score.PaperColumn.keep-inside-line = ##f
```

Damit Gesangstext auch nicht mit Taktlinien zusammenstößt, kann folgende Einstellung gesetzt werden:

```
\layout {
  \context {
    \Lyrics
      \consists Bar_engraver
      \consists Separating_line_group_engraver
      \hide BarLine
  }
}
```
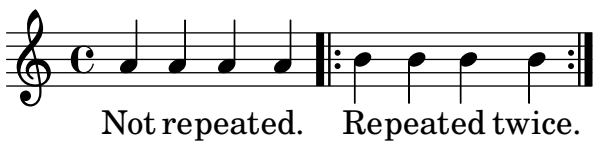
9.2.4 Gesangstext und Wiederholungen

Einfache Wiederholungen

Wiederholungen von *Musik* ist vollständig an anderer Stelle behandelt, siehe Kapitel 4 [Wiederholungszeichen], Seite 144. Dieser Abschnitt erklärt, wie man Gesangstext zu wiederholten Noten hinzufügt.

Gesangstext zu einem Abschnitt, der wiederholt wird, muss in der gleichen Wiederholungskonstruktion wie die Noten enthalten sein, wenn der Text sich nicht ändert:
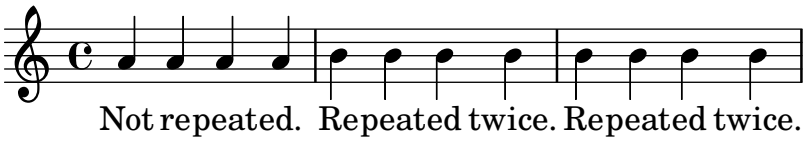
```
\score {
  <<
    \new Staff {
      \new Voice = "melody" {
        \relative {
          a'4 a a a
          \repeat volta 2 { b4 b b b }
        }
      }
    }
    \new Lyrics {
      \lyricsto "melody" {
        Not re -- peat -- ed.
        \repeat volta 2 { Re -- peat -- ed twice. }
      }
    }
  >>
}
```



Der Text wird dann richtig erweitert, wenn die Wiederholung mit `\unfoldRepeats` ausgeklappt wird:

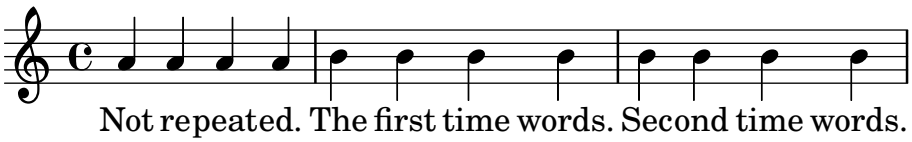
```
\score {
  \unfoldRepeats {
    <<
      \new Staff {
        \new Voice = "melody" {
          \relative {
            a'4 a a a
            \repeat volta 2 { b4 b b b }
          }
        }
      }
      \new Lyrics {
        \lyricsto "melody" {
          Not re -- peat -- ed.
          \repeat volta 2 { Re -- peat -- ed twice. }
        }
      }
    >>
  }
```

```
}
```



Wenn der wiederholte Abschnitt unterschiedlichen Text hat und ausgeklappt werden soll, müssen alle Wörter notiert werden:

```
\score {
  <<
    \new Staff {
      \new Voice = "melody" {
        \relative {
          a'4 a a a
          \repeat unfold 2 { b4 b b b }
        }
      }
    }
    \new Lyrics {
      \lyricsto "melody" {
        Not re -- peat -- ed.
        The first time words.
        Sec -- ond time words.
      }
    }
  >>
}
```
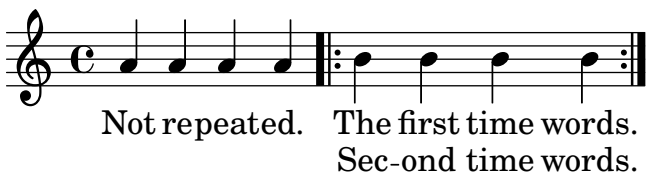


Wenn der Text in einer Wiederholung mit `volta` (also mit punktiertem Doppelstrich) unterschiedlich ist, muss der Text jeder Wiederholung in einem eigenen `Lyrics`-Kontext notiert werden, der korrekt in parrallelen Abschnitten geschachtelt wird:

```
\score {
  <<
    \new Staff {
      \new Voice = "melody" {
        \relative {
          a'4 a a a
          \repeat volta 2 { b4 b b b }
        }
      }
    }
    \new Lyrics \lyricsto "melody" {
      Not re -- peat -- ed.
      <<

{ The first time words. }
```

```
\new Lyrics {
  \set associatedVoice = "melody"
  Sec -- ond time words.

}
      >>
    }
  >>
}
```



Neue Strophen können auf die gleiche Art hinzugefügt werden:

```
\score {
  <<
    \new Staff {
      \new Voice = "singleVoice" {
        \relative {
  a'4 a a a
  \repeat volta 3 { b4 b b b }
          c4 c c c

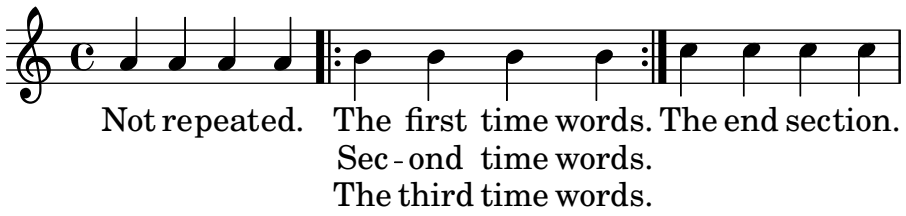
}
      }
    }
    \new Lyrics \lyricsto "singleVoice" {
      Not re -- peat -- ed.
      <<
        { The first time words. }

\new Lyrics {
  \set associatedVoice = "singleVoice"
  Sec -- ond time words.

}

\new Lyrics {
  \set associatedVoice = "singleVoice"
  The third time words.

}
      >>
      The end sec -- tion.
    }
  >>
}
```



Wenn diese Konstruktion jedoch innerhalb eines Mehrsystemkontexts eingebettet ist, wie etwa ein `ChoirStaff`, werden die Texte der zweiten und dritten Strophe unter dem untersten System ausgegeben werden.

Um sie richtig zu positionieren, kann `alignBelowContext` eingesetzt werden:

```
\score {
  <<
    \new Staff {
      \new Voice = "melody" {
        \relative {
  a'4 a a a
  \repeat volta 3 { b4 b b b }
          c4 c c c

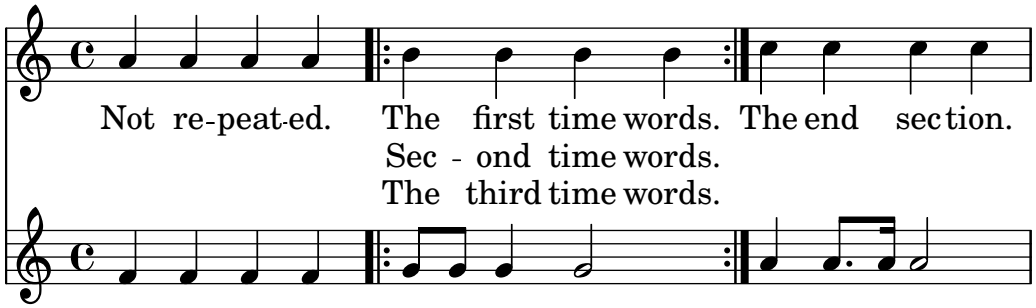
}
      }
    }
    \new Lyrics = "firstVerse" \lyricsto "melody" {
      Not re -- peat -- ed.
      <<
        { The first time words. }

\new Lyrics = "secondVerse"
        \with { alignBelowContext = "firstVerse" } {
  \set associatedVoice = "melody"
  Sec -- ond time words.

}

\new Lyrics = "thirdVerse"
        \with { alignBelowContext = "secondVerse" } {
  \set associatedVoice = "melody"
  The third time words.

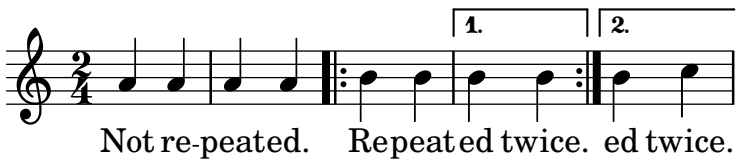
}
      >>
      The end sec -- tion.
    }
    \new Voice = "harmony" {
      \relative {
        f'4 f f f \repeat volta 2 { g8 g g4 g2 } a4 a8. a16 a2
      }
    }
  >>
}
```



Wiederholungen mit alternativen Endungen

Wenn der Text des wiederholten Abschnitts der gleiche ist, kann die gleiche Struktur für Gesangstext und Noten eingesetzt werden.

```
\score {
  <<
    \new Staff {
      \time 2/4
      \new Voice = "melody" {
        \relative {
          a'4 a a a
          \repeat volta 2 { b4 b }
          \alternative { { b b } { b c } }
        }
      }
    }
    \new Lyrics {
      \lyricsto "melody" {
        Not re -- peat -- ed.
        \repeat volta 2 { Re -- peat -- }
        \alternative { { ed twice. } { ed twice. } }
      }
    }
  >>
}
```

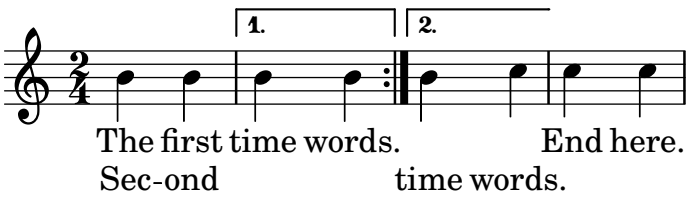


Aber wenn der wiederholte Abschnitt unterschiedlichen Text hat, kann keine `repeat`-Konstruktion um den Text eingesetzt werden und man muss manuell mit `\skip`-Befehlen die Noten überspringen, auf die sich der alternative Text nicht bezieht.

Achtung: hier darf kein Unterstrich (_) benutzt werden, um Noten zu überspringen, weil das ein Melisma anzeigen würde und die vorhergehende Silbe dazu veranlasst, links ausgerichtet zu werden.

Achtung: Der \skip-Befehl muss von einer Zahl gefolgt werden, aber diese Zahl wird ignoriert, wenn der Gesangstext seine Dauern von einer assoziierten Melodie ableitet, die mit \addlyrics oder \lyricsto angefügt wird. Jeder \skip-Befehl überspringt eine einzelne Note beliebiger Dauer, unabhängig vom Wert der auf den Befehl folgenden Zahl.

```
\score {
  <<
    \new Staff {
      \time 2/4
      \new Voice = "melody" {
        \relative {
          \repeat volta 2 { b'4 b }
          \alternative { { b b } { b c } }
          c4 c
        }
      }
    }
    \new Lyrics {
      \lyricsto "melody" {
        The first time words.
        \repeat unfold 2 { \skip 1 }
        End here.
      }
    }
    \new Lyrics {
      \lyricsto "melody" {
        Sec -- ond
        \repeat unfold 2 { \skip 1 }
        time words.
      }
    }
  >>
}
```

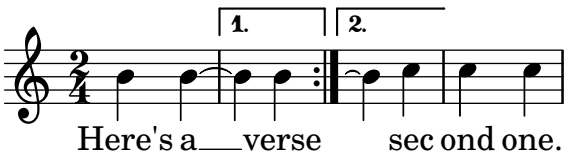


Wenn eine Note zu zwei oder mehr alternativen Endungen übergebunden wird, wird ein Bindebogen benutzt, um die Note in die erste alternative Endung überzubinden, und ein \repeatTie benutzt, um in die zweite und alle weiteren Klammern zu überbinden. Diese Struktur erzeugt problematische Ausrichtungen, wenn ein Gesangstext hinzu kommt und verlängert die alternativen Klammern, sodass es besser sein kann, die übergebundenen Noten vollständig in die Klammern aufzunehmen, um ein besseres Resultat zu erhalten.

Der Bindebogen erstellt ein Melisma zur ersten Klammer, aber nicht zur zweiten und allen weiteren Klammern, sodass es nötig ist, die automatische Erstellung von Melismen für die

Klammer-Abschnitte zu deaktivieren und manuell die Noten mit \skip zu überspringen, um eine richtige Ausrichtung des Textes zu erreichen.

```
\score {
  <<
    \new Staff {
      \time 2/4
      \new Voice = "melody" {
        \relative {
          \set melismaBusyProperties = #'()
          \repeat volta 2 { b'4 b ~}
          \alternative { { b b } { b \repeatTie c } }
          \unset melismaBusyProperties
          c4 c
        }
      }
    }
    \new Lyrics {
      \lyricsto "melody" {
        \repeat volta 2 { Here's a __ }
        \alternative {
          { \skip 1 verse }
          { \skip 1 sec }
        }
        ond one.
      }
    }
  >>
}
```

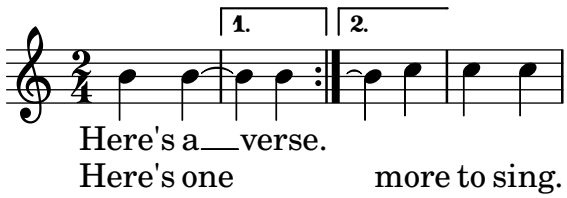


Wenn \unfoldRepeats in einem Abschnitt eingesetzt wird, der den \repeatTie-Befehl enthält, sollte der \repeatTie entfernt werden, damit nicht beide Bindestriche ausgegeben werden.

Wenn der wiederholte Abschnitt unterschiedlichen Text hat, kann \repeat nicht um den Gesangstext benutzt werden, und \skip-Befehle müssen manuell eingegeben werden:
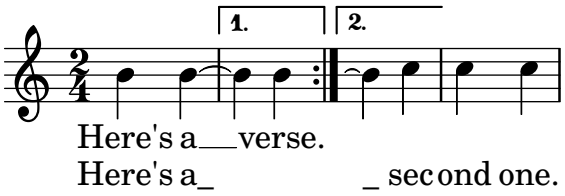
```
\score {
  <<
    \new Staff {
      \time 2/4
      \new Voice = "melody" {
        \relative {
          \repeat volta 2 { b'4 b ~}
          \alternative { { b b } { b \repeatTie c } }
          c4 c
        }
      }
    }
    \new Lyrics {
```

```
      \lyricsto "melody" {
        Here's a __ verse.
        \repeat unfold 2 { \skip 1 }
      }
    }
    \new Lyrics {
      \lyricsto "melody" {
        Here's one
        \repeat unfold 2 { \skip 1 }
        more to sing.
      }
    }
  >>
}
```



Wenn Sie Bindestriche und Fülllinien zwischen Wiederholung und Klammer benutzen wollen, müssen sie manuell notiert werden:

```
\score {
  <<
    \new Staff {
      \time 2/4
      \new Voice = "melody" {
        \relative {
          \repeat volta 2 { b'4 b ~}
          \alternative { { b b } { b \repeatTie c } }
          c4 c
        }
      }
    }
    \new Lyrics {
      \lyricsto "melody" {
        Here's a __ verse.
        \repeat unfold 2 { \skip 1 }
      }
    }
    \new Lyrics {
      \lyricsto "melody" {
        Here's "a_"
        \skip 1
        "_" sec -- ond one.
      }
    }
  >>
}
```
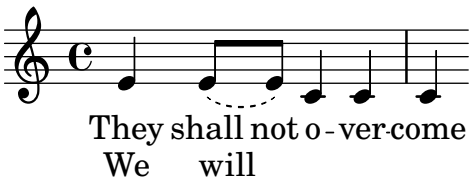


Siehe auch

Notationsreferenz: Abschnitt 32.3 [Kontexte am Leben halten], Seite 583, Kapitel 4 [Wiederholungszeichen], Seite 144.

9.2.5 Getrennte Texte

Wenn nur Text und Rhythmus von zwei Texten unterschiedlich sind, während die Noten gleich bleiben, kann man die automatische Melisma-Erkennung kurzzeitig ausschalten und das Melisma im Text anzeigen:
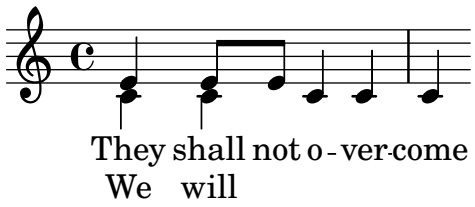
```
\score {
  <<
    \new Voice = "melody" {
      \relative c' {
        \set melismaBusyProperties = #'()
        \slurDown
        \slurDashed
        e4 e8 ( e ) c4 c |
        \unset melismaBusyProperties
        c
      }
    }
    \new Lyrics \lyricsto "melody" {
      They shall not o -- ver -- come
    }
    \new Lyrics \lyricsto "melody" {
      We will _
    }
  >>
}
```



Wenn sich sowohl Noten als auch Worte unterscheiden, kann es besser sein, die unterschiedlichen Noten und den Text zu notieren, indem man `Voice`-Kontexte benennt und den Text an die enstprechenden Kontexte anhängt:

```
\score {
  <<
    \new Voice = "melody" {
      \relative {
        <<
          {
            \voiceOne
```

```
            e'4 e8 e
          }
          \new Voice = "splitpart" {
            \voiceTwo
            c4 c
          }
        >>
        \oneVoice
        c4 c |
        c
      }
    }
    \new Lyrics \lyricsto "melody" {
      They shall not o -- ver -- come
    }
    \new Lyrics \lyricsto "splitpart" {
      We will
    }
  >>
}
```

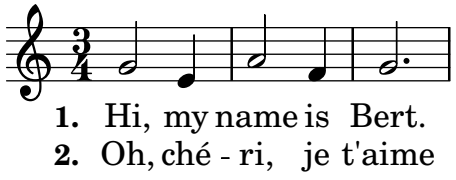


9.3 Strophen

9.3.1 Strophennummern hinzufügen

Strophennummerierung kann hinzugefügt werden:

```
\new Voice \relative {
  \time 3/4 g'2 e4 a2 f4 g2.
} \addlyrics {
  \stanza "1. "
  Hi, my name is Bert.
} \addlyrics {
  \stanza "2. "
  Oh, ché -- ri, je t'aime
}
```



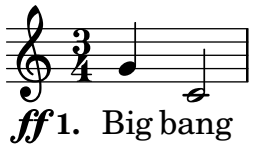
Die Zahl wird direkt vor die erste Silbe gesetzt.

9.3.2 Lautstärkebezeichnung zu Strophen hinzufügen

Dynamikzeichen können zur Strophennummer hinzugefügt werden. In LilyPond muss alles, was vor einer Strophe gesetzt wird, im `StanzaNumber`-Objekt definiert werden, also auch Dynamikbezeichnung. Aus technischen Gründen muss die Strophe außerhalb von `lyricmode` gesetzt werden:

```
text = {
  \stanza \markup { \dynamic "ff" "1. " }
  \lyricmode {
    Big bang
  }
}

<<
  \new Voice = "tune" {
    \time 3/4
    g'4 c'2
  }
\new Lyrics \lyricsto "tune" \text
>>
```

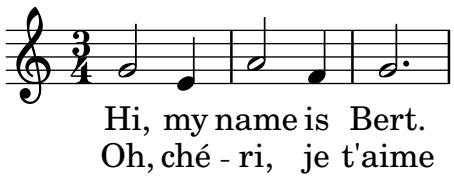


ff 1. Big bang

9.3.3 Sängernamen zu Strophen hinzufügen

Namen von Sängern können auch eingefügt werden. Sie werden zu Beginn der Zeile gesetzt, ähnlich wie eine Instrumentenbezeichnung. Sie werden mit der `vocalName`-Eigenschaft erstellt. Eine Kurzversion kann mit `shortVocalName` definiert werden.

```
\new Voice \relative {
  \time 3/4 g'2 e4 a2 f4 g2.
} \addlyrics {
  \set vocalName = "Bert "
  Hi, my name is Bert.
} \addlyrics {
  \set vocalName = "Ernie "
  Oh, ché -- ri, je t'aime
}
```



Bert Hi, my name is Bert.
Ernie Oh, ché - ri, je t'aime

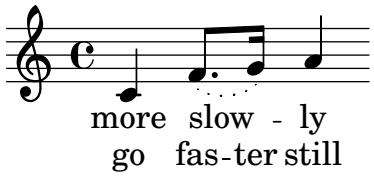
9.3.4 Strophen mit unterschiedlichem Rhythmus

Oft haben unterschiedliche Strophen eines Liedes leicht unterschiedliche Silbenzahlen und werden darum auf andere Art zur Melodie gesungen. Derartige Variationen können mit `\lyricsto` bewältigt werden.

Melismen ignorieren

Teilweise wird zu einer Silbe ein Melisma in einer Strophe gesungen, während in einer anderen jede Note eine Silbe erhält. Eine Möglichkeit ist, dass die Strophe mit mehr Text das Melisma ignoriert. Das wird mit der `ignoreMelismata`-Eigenschaft im `Lyrics`-Kontext vorgenommen.

```
<<
  \relative \new Voice = "lahlah" {
    \set Staff.autoBeaming = ##f
    c'4
    \slurDotted
    f8.[( g16])
    a4
  }
  \new Lyrics \lyricsto "lahlah" {
    more slow -- ly
  }
  \new Lyrics \lyricsto "lahlah" {
    go
    \set ignoreMelismata = ##t
    fas -- ter
    \unset ignoreMelismata
    still
  }
>>
```



Bekannte Probleme und Warnungen
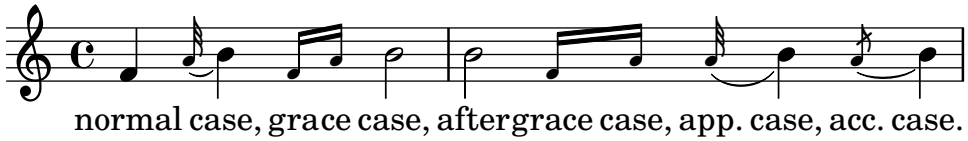
Anders als die meisten `\set`-Befehle funktioniert `\set ignoreMelismata` nicht zusammen mit `\once`. Es ist notwendig, explizit `\set` und `\unset` zu verwenden, um den Text einzugrenzen, für den Melismen ignoriert werden sollen.

Silben zu Verzierungsnoten hinzufügen

Normalerweise werden Verzierungsnoten (z.B. durch `\grace`) bei `\lyricsto` keine Silben zugeordnet. Dieses Verhalten kann geändert werden, wie das folgende Beispiel zeigt.

```
<<
  \new Voice = melody \relative {
    f'4 \appoggiatura a32 b4
    \grace { f16 a16 } b2
    \afterGrace b2 { f16[ a16] }
    \appoggiatura a32 b4
    \acciaccatura a8 b4
  }
  \new Lyrics
  \lyricsto melody {
    normal
    \set includeGraceNotes = ##t
    case,
```
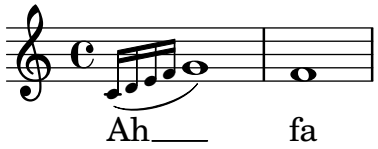
```
    gra -- ce case,
    after -- grace case,
    \set ignoreMelismata = ##t
    app. case,
    acc. case.
  }
>>
```



Bekannte Probleme und Warnungen

Wie bei `associatedVoice` muss `includeGraceNotes` spätestens eine Silbe vor derjenigen gesetzt werden, die unter einer Verzierungsnote stehen soll. Im Fall, dass eine Verzierungsnote die erste des Musikstückes ist, kann ein `\with`- oder `\context`-Block verwendet werden:

```
<<
  \new Voice = melody \relative c' {
    \grace { c16( d e f }
    g1) f
  }
  \new Lyrics \with { includeGraceNotes = ##t }
  \lyricsto melody {
    Ah __ fa
  }
>>
```

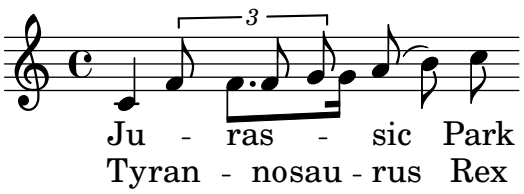


Zu einer alternativen Melodie umschalten

Es ist auch möglich, die Silben von verschiedenen Textzeilen an unterschiedlichen Melodien auszurichten. Das wird mit der `associatedVoice`-Eigenschaft vorgenommen:

```
<<
  \relative \new Voice = "lahlah" {
    \set Staff.autoBeaming = ##f
    c'4
    <<
      \new Voice = "alternative" {
        \voiceOne
        \tuplet 3/2 {
          % show associations clearly.
          \override NoteColumn.force-hshift = #-3
          f8 f g
        }
      }
      {
        \voiceTwo
        f8.[ g16]
```

```
      \oneVoice
    } >>
  a8( b) c
}
\new Lyrics \lyricsto "lahlah" {
  Ju -- ras -- sic Park
}
\new Lyrics \lyricsto "lahlah" {
  % Tricky: need to set associatedVoice
  % one syllable too soon!
  \set associatedVoice = "alternative" % applies to "ran"
  Ty --
  ran --
  no --
  \set associatedVoice = "lahlah" % applies to "rus"
  sau -- rus Rex
} >>
```



Der Text der ersten Strophe wird an der Stimme „lahlah“ ausgerichtet, aber die zweite Strophe wird zuerst zum `lahlah`-Kontext gesetzt und dann zur `alternative`-Melodie für die Silben „ran“ bis „sau“:

```
\set associatedVoice = "alternative" % applies to "ran"
Ty --
ran --
no --
\set associatedVoice = "lahlah" % applies to "rus"
sau -- rus Rex
```

Hier ist `alternative` die Bezeichnung des `Voice`-Kontexts mit der Triole.

Der `\set associatedVoice`-Befehl tritt eine Silbe zu früh auf, aber das ist in diesem Fall richtig.

> **Achtung:** Der `\set associatedVoice`-Befehl muss eine Silbe *vor* der Silbe notiert werden, auf welcher der Wechsel zur neuen Stimme stattfinden soll. Anders gesagt: der Wechsel der assoziierten Stimme geschieht eine Silber später, als man erwarten würde. Das geschieht aus technischen Gründen – es handelt sich nicht um einen Fehler.

9.3.5 Die Strophen am Ende ausdrucken

Manchmal soll nur eine Strophe mit der Melodie gesetzt werden und die weiteren Strophen als Text unter den Noten hinzugefügt werden. Dazu wird der Text in einer `markup`-Umgebung außerhalb der `\score`-Umgebung gesetzt. Es gibt zwei Arten, die Zeilen auszurichten, wie das Beispiel zeigt:
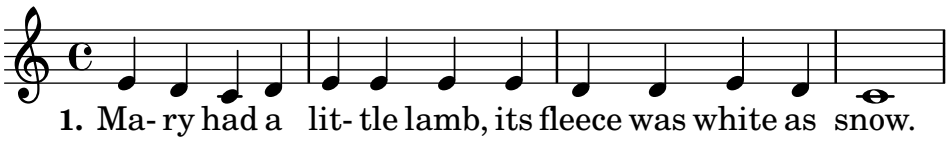
```
melody = \relative {
e' d c d | e e e e |
d d e d | c1 |
}

text = \lyricmode {
\stanza "1." Ma- ry had a lit- tle lamb,
its fleece was white as snow.
}

\score{ <<
  \new Voice = "one" { \melody }
  \new Lyrics \lyricsto "one" \text
>>
  \layout { }
}
\markup { \column{
  \line{ Verse 2. }
  \line{ All the children laughed and played }
  \line{ To see a lamb at school. }
  }
}
\markup{
  \wordwrap-string "
  Verse 3.

  Mary took it home again,

  It was against the rule."
}
```



Verse 2.
All the children laughed and played
To see a lamb at school.

Verse 3.
Mary took it home again,
It was against the rule.

9.3.6 Die Strophen am Ende in mehreren Spalten drucken

Wenn in einem Lied sehr viele Strophen vorkommen, werden sie oft in mehreren Spalten unter den Noten gesetzt. Eine nach außen versetzte Zahl zeigt die Strophennummer an. Dieses Beispiel zeigt eine Methode, diese Art von Notensatz zu produzieren.
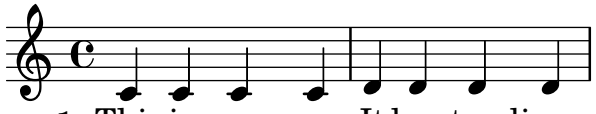
```
melody = \relative {
  c'4 c c c | d d d d
}

text = \lyricmode {
  \stanza "1." This is verse one.
  It has two lines.
}

\score{ <<
    \new Voice = "one" { \melody }
    \new Lyrics \lyricsto "one" \text
   >>
  \layout { }
}

\markup {
  \fill-line {
    \hspace #0.1 % moves the column off the left margin;
     % can be removed if space on the page is tight
     \column {
      \line { \bold "2."
        \column {
          "This is verse two."
          "It has two lines."
        }
      }
      \combine \null \vspace #0.1 % adds vertical spacing between verses
      \line { \bold "3."
        \column {
          "This is verse three."
          "It has two lines."
        }
      }
    }
    \hspace #0.1 % adds horizontal spacing between columns;
    \column {
      \line { \bold "4."
        \column {
          "This is verse four."
          "It has two lines."
        }
      }
      \combine \null \vspace #0.1 % adds vertical spacing between verses
      \line { \bold "5."
        \column {
          "This is verse five."
          "It has two lines."
        }
      }
    }
  \hspace #0.1 % gives some extra space on the right margin;
```

```
  % can be removed if page space is tight
  }
}
```



1. This is verse one. It has two lines.

2. This is verse two.
It has two lines.

3. This is verse three.
It has two lines.

4. This is verse four.
It has two lines.

5. This is verse five.
It has two lines.

Siehe auch

Referenz der Interna: Abschnitt “LyricText” in *Referenz der Interna*, Abschnitt “Stanza-Number” in *Referenz der Interna*.

9.4 Lieder

9.4.1 Verweise für Lieder

Lieder werden normalerweise auf drei Systemen notiert: das oberste System für den Sänger und zwei Systeme für die Klavierbegleitung darunter. Der Gesangstext der ersten Strophe wird direkt unter dem ersten System ausgegeben. Wenn es nur eine kleine Anzahl weiterer Strophen gibt, können sie sofort unter der ersten gesetzt werden, aber wenn es viele Strophen gibt, werden die zweite und alle weiteren Strophen als Text unter den Noten gesetzt.

Alle Notationselemente, die für die Notation von Liedern benötigt werden, werden woanders beschrieben:

- Um das Systemlayout einzurichten, siehe Abschnitt 6.1 [Systeme anzeigen lassen], Seite 183.
- Zur Notation von Klaviermusik siehe Kapitel 10 [Tasteninstrumente und andere Instrumente mit mehreren Systemen], Seite 314.
- Zur Notation von Gesangstext zu einer Notenzeile siehe Abschnitt 9.1 [Übliche Notation für Vokalmusik], Seite 253.
- Zur Platzierung des Gesangstext siehe Abschnitt 9.2.2 [Gesangstext vertikal verschieben], Seite 267.
- Zur Notation von Strophen siehe Abschnitt 9.3 [Strophen], Seite 281.
- Lieder werden oft auch mit Akkorden gesetzt, die als Symbole über dem Notensystem notiert werden. Das wird in Abschnitt 15.2 [Akkorde anzeigen], Seite 402, beschrieben.
- Zur Notation von Bunddiagrammen für die Akkorde einer Gitarrenbegleitung oder anderer Bundinstrumente siehe „Bund-Diagramm-Beschriftung“ in Abschnitt 12.1 [Übliche Notation für Saiteninstrumente mit Bünden], Seite 327.

Siehe auch

Handbuch zum Lernen: Abschnitt “Lieder” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 9.1 [Übliche Notation für Vokalmusik], Seite 253, Abschnitt 15.2 [Akkorde anzeigen], Seite 402, Abschnitt 6.1 [Systeme anzeigen lassen], Seite 183, Kapitel 10 [Tasteninstrumente und andere Instrumente mit mehreren Systemen], Seite 314, Abschnitt 9.2.2 [Gesangstext vertikal verschieben], Seite 267, Abschnitt 9.3 [Strophen], Seite 281.

Schnipsel: Abschnitt “Vocal music” in *Schnipsel*.

9.4.2 Liedblätter
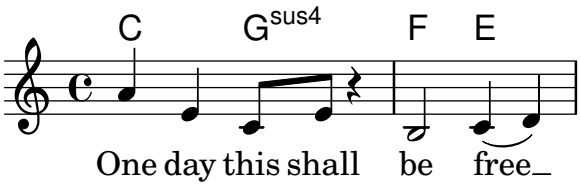
Liedblätter können erstellt werden, indem man Gesangstext mit Akkorden im Akkord-Modus kombiniert; die Syntax ist erklärt in Kapitel 15 [Notation von Akkorden], Seite 397.

Ausgewählte Schnipsel

Ein einfaches Liedblatt

Ein Liedblatt besteht aus Akkordbezeichnungen, einer Melodie und dem Liedtext:

```
<<
  \chords { c2 g:sus4 f e }
  \new Staff \relative c'' {
    a4 e c8 e r4
    b2 c4( d)
  }
  \addlyrics { One day this shall be free __ }
>>
```



Siehe auch

Notationsreferenz: Kapitel 15 [Notation von Akkorden], Seite 397.

9.5 Chormusik

Dieser Abschnitt zeigt Eigenheiten der Notation von Chormusik. Hierzu gehören Hymnen, mehrstimmige Lieder, Oratorien, Kantaten usw.

9.5.1 Verweise für Chormusik

Chormusik wird normalerweise auf zwei, drei oder vier Systemen innerhalb einer `ChoirStaff`-Gruppe notiert. Begleitung wird darunter als `PianoStaff`-Klaviersystem gesetzt, oft auch in kleinerer Größe, wenn es sich um eine Übungshilfe für ein *a capella*-Chorwerk handelt. Die Noten jeder Stimme werden in einem `Voice`-Kontext notiert, welche entweder einzeln auf einem eigenen Notensytem notiert werden oder zu zweit auf dem gleichen System gesetzt werden.

Gesangstext wird in `Lyrics`-Kontext gesetzt, entweder unter dem zugehörigen System oder ein Text über dem System, der andere darunter, wenn das System die Noten von zwei Stimmen enthält.

Einige häufig anzutreffende Sachverhalte für Chormusik sind woanders behandelt:

- Eine Einleitung, um SATB-Chorpartituren zu erstellen, findet sich in Abschnitt "Vierstimmige SATB-Partitur" in *Handbuch zum Lernen*.
- Einige Vorlagen, die sich für unterschiedliche Chormusik eignen, finden sich im Handbuch zum Lernen, siehe Abschnitt "Vokalensemble" in *Handbuch zum Lernen*.
- Zu Information über `ChoirStaff` und `PianoStaff` siehe Abschnitt 6.1.2 [Systeme gruppieren], Seite 185.
- Besondere Notenköpfe, wie etwa für die „Sacred Harp"-Notation benutzt, finden sich in Abschnitt 1.4.3 [Notenköpfe mit besonderen Formen], Seite 39.

- Wenn zwei Vokalstimmen sich ein System teilen, werden Hälse, Bögen usw. der oberen Stimme nach oben gerichtet und die der unteren Stimme nach unten. Dieses Verhalten erreicht man mit `\voiceOne` und `\voiceTwo`. Siehe Abschnitt 5.2.1 [Mehrstimmigkeit in einem System], Seite 166.

Vordefinierte Befehle

`\oneVoice`, `\voiceOne`, `\voiceTwo`.

Siehe auch

Handbuch zum Lernen: Abschnitt “Vierstimmige SATB-Partitur” in *Handbuch zum Lernen*, Abschnitt “Vokalensemble” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 6.1.2 [Systeme gruppieren], Seite 185, Abschnitt 32.7 [Reihenfolge des Kontextlayouts], Seite 595, Abschnitt 1.4.3 [Notenköpfe mit besonderen Formen], Seite 39, Abschnitt 5.2.1 [Mehrstimmigkeit in einem System], Seite 166.

Referenz der Interna: Abschnitt “ChoirStaff” in *Referenz der Interna*, Abschnitt “Lyrics” in *Referenz der Interna*, Abschnitt “PianoStaff” in *Referenz der Interna*.

9.5.2 Partiturbeispiele für Chormusik

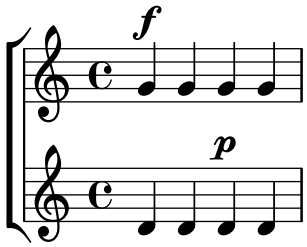
Chormusik auf vier Systemen, mit oder ohne Klavierbegleitung, wird meistens mit zwei Systemgruppen pro Seite gesetzt. Abhängig von der Seitengröße kann das erfordern, dass die Standardgröße der Notensysteme geändert wird. Die folgenden Einstellungen sollten in Betracht gezogen werden:

- Die globale Systemgröße kann verändert werden, um die Größe aller Elemente einer Partitur zu ändern. Siehe Abschnitt 26.2 [Die Notensystemgröße einstellen], Seite 534.
- Die Abstände zwischen den Systemen, den Systemgruppen und den Gesangstexten können alle einzeln eingestellt werden. Siehe Kapitel 28 [Vertikale Abstände], Seite 545.
- Die Dimensionen der vertikalen Layout-Variablen können angezeigt werden, um beim Anpassen der vertikalen Platzverteilung zu helfen. Das und andere Möglichkeiten, die Noten auf weniger Seiten zu zwingen, finden sich in Kapitel 30 [Die Musik auf weniger Seiten zwingen], Seite 573.
- Wenn die Anzahl der Systemgruppen pro Seite zwischen einer und mehreren wechselt, wird dies üblicherweise mit einem Trennzeichen zwichen den Systemgruppen angezeigt. Siehe Abschnitt 6.1.4 [Systeme trennen], Seite 190.
- Zu Details für andere Eigenschaften der Seitenformatierung siehe Kapitel 25 [Seitenlayout], Seite 521.

Dynamikzeichen werden in den Grundeinstellungen unter dem System notiert, aber in Chormusik werden sie oft über dem System gesetzt um nicht mit dem Gesangstext zu kollidieren. Der vordefiniert Befehl `\dynamicUp` erledigt das für einen `Voice`-Kontext auf einem eigenen Notensystem. Wenn mehrere `Voice`-Kontexte vorhanden sind, müsste man den Befehl in jedem einzeln setzen. Um alle Dynamikzeichen in einer Partitur über den Systemen zu setzen, kann eine erweiterte Form genutzt werden, wie das Beispiel zeigt:

```
\score {
  \new ChoirStaff <<
    \new Staff {
      \new Voice {
        \relative { g'4\f g g g }
      }
    }
    \new Staff {
```

```
      \new Voice {
        \relative { d'4 d d\p d }
      }
    }
  >>
  \layout {
    \context {
      \Score
      \override DynamicText.direction = #UP
      \override DynamicLineSpanner.direction = #UP
    }
  }
}
```



Vordefinierte Befehle

`\dynamicUp`.

Siehe auch

Notationsreferenz: Abschnitt 26.2 [Die Notensystemgröße einstellen], Seite 534, Kapitel 28 [Vertikale Abstände], Seite 545, Abschnitt 30.1 [Abstände anzeigen lassen], Seite 573, Abschnitt 30.2 [Abstände verändern], Seite 574, Kapitel 26 [Partiturlayout], Seite 532, Abschnitt 27.8 [Eine zusätzliche Stimme für Umbrüche benutzen], Seite 542, Kapitel 25 [Seitenlayout], Seite 521, Abschnitt 6.1.4 [Systeme trennen], Seite 190, Kapitel 30 [Die Musik auf weniger Seiten zwingen], Seite 573.

Referenz der Interna: Abschnitt “VerticalAxisGroup” in *Referenz der Interna*, Abschnitt “StaffGrouper” in *Referenz der Interna*.

9.5.3 Geteilte Stimmen

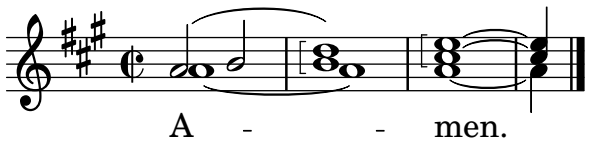
Using a bracket to clarify divisi

The `\nonArpeggiato` command can be used to indicate the division of voices where there are no stems to provide the information. This is often seen in choral music.

```
\include "english.ly"

\score {
  \relative c'' {
    \key a \major
    \time 2/2
    <<
      \new Voice = "upper" <<
        {
          \voiceOne
          a2( b2
```

```
            <b d>1\nonArpeggiato)
            <cs e>\nonArpeggiato ~
            <cs e>4
            \fine
          }
          \addlyrics { \lyricmode { A -- men. } }
        >>
        \new Voice = "lower" {
          \voiceTwo
          a1 ~
          a
          a ~
          a4
          \fine
        }
      >>
    }
  }
```



Siehe auch

Notationsreferenz: Abschnitt 3.3 [Ausdrucksbezeichnungen als Linien], Seite 136.

9.6 Oper und Musical

Noten, Text und Dialoge von Oper und Singspielen werden normalerweise auf eine der folgenden Weisen notiert:

- Eine *Aufführungspartitur* enthält alle Orchester- und Gesangsstimmen sowie Libretto-Stichworte der gesprochenen Abschnitte.
- *Orchesterstimmen* enthalten die Noten für einzelne Instrumente des Orchesters oder der Band.
- Ein *Klavierauszug* enthält alle Gesangsstimmen mit Klavierbegleitung. Die Begleitung ist normalerweise ein Auszug der Orchesterstimmen, in dem often das originale Instrument bezeichnet ist. Klavierauszüge enthalten teilweise auch Regieanweisungen und Libretto-Stichworte.
- Ein *Vocal Book* enthält nur die Gesangstimmen (ohne Begleitung), teilweise zusammen mit dem Libretto.
- Ein *Libretto* enthält die ausführlichen gesprochenen Abschnitte, wie man sie oft in Musicals oder Operetten findet, sowie den gesungenen Text. Normalerweise sind auch Regieanweisungen enthalten. LilyPond kann eingesetzt werden, um Libretti zu setzen, aber es kann günstiger sein, dafür eine andere Methode zu benutzen, da sie keine Noten enthalten.

Die Abschnitte der LilyPond-Dokumentation, die nützlich zum Setzen von Oper und Musical sind, sind in den Verweisen unten aufgezeigt. Darauf folgen einige Abschnitte, die bestimmte Techniken behandeln, die besonders spezifisch für das Setzen von Singspiel-Partituren sind.

9.6.1 Verweise für Oper und Musical

- Eine große Partitur hat viele gruppierte Systeme und Gesangstext. Arten, Notensysteme zu gruppieren, finden sich in Abschnitt 6.1.2 [Systeme gruppieren], Seite 185. Wie Gruppen geschachtelt werden, findet sich in Abschnitt 6.1.3 [Verschachtelte Notensysteme], Seite 188.
- Die Ausgabe von leeren Notensystemen in Paritur und Klavierauszug wird oft verhindert. Um eine komprimierte Paritur ohne leere Systeme zu erstellen siehe Abschnitt 6.2.3 [Systeme verstecken], Seite 198.
- Wie Orchetserstimmen notiert werden, ist dokumentiert in Abschnitt 6.3 [Orchesterstimmen erstellen], Seite 201. Andere Abschnitte des Kapitels „Spezielle Notation“ können auch relevant sein, abhängig von der benutzten Orchestration. Viele Instrumente sind transponierend, siehe Abschnitt 1.3.4 [Transposition von Instrumenten], Seite 25.
- Wenn die Anzahl der Notensystemgruppen pro Seite sich ändert, wird normalerweise zwischen zwei Systemgruppen ein Trenner gesetzt. Siehe Abschnitt 6.1.4 [Systeme trennen], Seite 190.
- Zu Einzelheiten der Seitenformatierung siehe Kapitel 25 [Seitenlayout], Seite 521.
- Stichworte der Dialoge und Regieanweisungen können mit Textbeschriftungen eingefügt werden. Siehe Abschnitt 20.3 [Fußnoten erstellen], Seite 476. Ausführliche Regieanweisungen können mit Abschnitten von alleinstehenden Textbeschriftungen zwischen zwei `\score`-Umgebungen gesetzt werden. Siehe Abschnitt 8.1.4 [Separater Text], Seite 231.

Siehe auch

Glossar: Abschnitt “Frenched score” in *Glossar*, Abschnitt “Frenched staves” in *Glossar*, Abschnitt “transposing instrument” in *Glossar*.

Notationsreferenz: Abschnitt 6.1.2 [Systeme gruppieren], Seite 185, Abschnitt 6.2.3 [Systeme verstecken], Seite 198, Abschnitt 1.3.4 [Transposition von Instrumenten], Seite 25, Abschnitt 6.1.3 [Verschachtelte Notensysteme], Seite 188, Kapitel 25 [Seitenlayout], Seite 521, Abschnitt 6.1.4 [Systeme trennen], Seite 190, Abschnitt 1.2.2 [Transponieren], Seite 12, Abschnitt 6.3 [Orchesterstimmen erstellen], Seite 201, Abschnitt 8.1 [Text eingeben], Seite 226, Abschnitt 20.3 [Fußnoten erstellen], Seite 476.

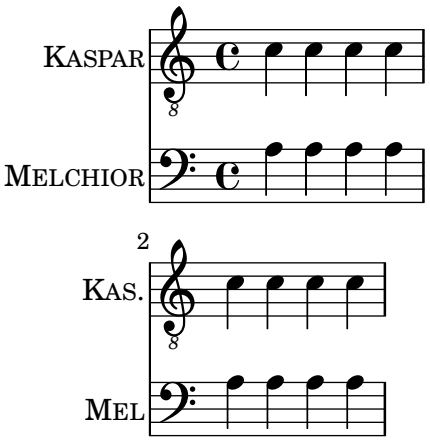
Schnipsel: Abschnitt “Vocal music” in *Schnipsel*.

9.6.2 Namen von Figuren

Namen von Figuren werden normalerweise links des Notensystems angezeigt, wenn auf dem System nur die Stimme einer Figure notiert ist:
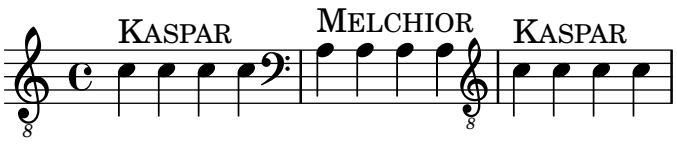
```
\score {
  <<
    \new Staff {
      \set Staff.vocalName = \markup \smallCaps Kaspar
      \set Staff.shortVocalName = \markup \smallCaps Kas.
      \relative {
        \clef "G_8"
        c'4 c c c
        \break
        c4 c c c
      }
    }
    \new Staff {
      \set Staff.vocalName = \markup \smallCaps Melchior
      \set Staff.shortVocalName = \markup \smallCaps Mel
```

```
      \clef "bass"
      \relative {
        a4 a a a
        a4 a a a
      }
    }
  >>
}
```



Wenn zwei oder mehr Figuren sich ein System teilen, wird der Name normalerweise über dem System immer dann gesetzt, wenn der kommende Abschnitt von der Figur gesungen werden soll. Das kann man mit Textbeschriftungen vornehmen. Oft wird eine bestimmte Schriftart hierfür benutzt.

```
\relative c' {
  \clef "G_8"
  c4^\markup \fontsize #1 \smallCaps Kaspar
  c c c
  \clef "bass"
  a4^\markup \fontsize #1 \smallCaps Melchior
  a a a
  \clef "G_8"
  c4^\markup \fontsize #1 \smallCaps Kaspar
  c c c
}
```
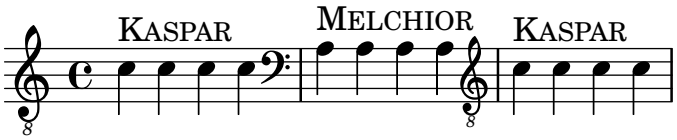


Wenn sehr viele Figurenwechsel vorkommen, kann es auch einfacher sein, „Instrument“-Definitionen für jeden Namen auf oberster Dateiebene zu definieren, sodass `\instrumentSwitch` der Wechsel der Figur angezeigt werden kann.

```
\addInstrumentDefinition "kaspar"
  #`((instrumentTransposition . ,(ly:make-pitch -1 0 0))
     (shortInstrumentName . "Kas.")
     (clefGlyph . "clefs.G")
     (clefTransposition . -7)
     (middleCPosition . 1)
```

```
      (clefPosition . -2)
      (instrumentCueName . ,(markup #:fontsize 1 #:smallCaps "Kaspar"))
      (midiInstrument . "voice oohs"))

\addInstrumentDefinition "melchior"
  #`((instrumentTransposition . ,(ly:make-pitch 0 0 0))
      (shortInstrumentName . "Mel.")
      (clefGlyph . "clefs.F")
      (clefTransposition . 0)
      (middleCPosition . 6)
      (clefPosition . 2)
      (instrumentCueName . ,(markup #:fontsize 1 #:smallCaps "Melchior"))
      (midiInstrument . "voice aahs"))

\relative c' {
  \instrumentSwitch "kaspar"
  c4 c c c
  \instrumentSwitch "melchior"
  a4 a a a
  \instrumentSwitch "kaspar"
  c4 c c c
}
```



Siehe auch

Notationsreferenz: Abschnitt 6.3.1 [Instrumentenbezeichnungen], Seite 201, Kapitel 8 [Text], Seite 226, Abschnitt A.10 [Textbeschriftungsbefehle], Seite 678.

LilyPond Erweitern: Abschnitt “Beschriftungskonstruktionen in Scheme” in *Extending*.

Referenz der Interna: Abschnitt “Scheme-Funktionen” in *Referenz der Interna*.

9.6.3 Musikalische Stichnoten

Stichnoten können in Klavierauszügen, Vocal Books und Orchesterstimmen eingesetzt werdne, um anzuziegen, was für Noten eine andere Stimme direkt vor dem eigenen Einsatz spielt. Stichnoten und Instrumentenbezeichnungen werden auch im Klavierauszug eingesetzt, um anzuzeigen, welches Instrument in der Orchesterfassung den Abschnitt spielt. Das hilft dem Dirigenten, wenn man keine große Partitur parat hat.

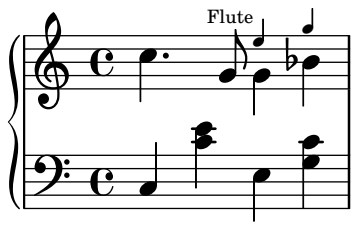
Der grundlegende Mechanismus, um Stichnoten einzufügen, findet sich in Abschnitt 6.3.2 [Andere Stimmen zitieren], Seite 205, und Abschnitt 6.3.3 [Stichnoten formatieren], Seite 208. Aber wenn man viele Stichnoten etwa in einen Klavierauszug einfügen will, um dem Dirigenten zu helfen, muss man sehr sorgfältig mit der Positionierung der Instrumentenbezeichnungen sein. Im folgenden Beispiel gibt es dazu einige Hilfestellungen.

```
flute = \relative {
  s4 s4 e'' g
}
\addQuote "flute" { \flute }

pianoRH = \relative {
```

```
    c''4. g8
    % position name of cue-ing instrument just before the cue notes,
    % and above the staff
    \new CueVoice {
      \override InstrumentSwitch.self-alignment-X = #RIGHT
      \set instrumentCueName = "Flute"
    }
    \cueDuring "flute" #UP { g4 bes4 }
  }
  pianoLH = \relative { c4 <c' e> e, <g c> }

  \score {
    \new PianoStaff <<
      \new Staff {
        \pianoRH
      }
      \new Staff {
        \clef "bass"
        \pianoLH
      }
    >>
  }
```



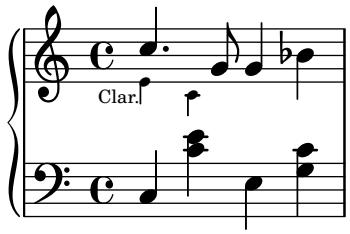
Wenn ein transponierendes Instrument zitiert wird, sollte die Orchsterstimme die Tonart angeben, damit die Transposition der Stichnoten automatisch geschehen kann. Das Beispiel unten zeigt, wie man das vornimmt. Die Noten im Beipiel sind sehr tief auf dem System, sodass `DOWN` (nach unten) im `\cueDuring` definiert ist, damit die Hälse nach unten zeigen. Die Instrumentbezeichnung wird auch unter dem System platziert. Auch die Stimme für die rechte Hand des Klaviers ist explizit definiert. Das ist wichtig, weil die Stichnoten dieses Beispiels direkt am Anfang des ersten Taktes beginnen und sonst die gesamte Rechte Hand der Klaviernoten als `CueVoice`-(Stichnoten)-Kontext definiert werden würde!

```
  clarinet = \relative c' {
    \transposition bes
    fis4 d d c
  }
  \addQuote "clarinet" { \clarinet }

  pianoRH = \relative c'' {
    \transposition c'
    % position name of cue-ing instrument below the staff
    \new CueVoice {
      \override InstrumentSwitch.self-alignment-X = #RIGHT
      \override InstrumentSwitch.direction = #DOWN
      \set instrumentCueName = "Clar."
    }
```

```
  \cueDuring "clarinet" #DOWN { c4. g8 }
  g4 bes4
}
pianoLH = \relative { c4 <c' e> e, <g c> }

\score {
  <<
    \new PianoStaff <<
      \new Staff {
        \new Voice {
          \pianoRH
        }
      }
      \new Staff {
        \clef "bass"
        \pianoLH
      }
    >>
  >>
}
```



Aus diesen zwei Beispielen wird klar, dass es sehr viele Probleme bereiten kann, ausgiebig Stichnoten in einen Klavierauszug einzufügen, und die Noten für das Klavier würden unleserlich. Im folgenden Schnipsel wird jedoch gezeigt, wie man eine Musikfunktion definiert, die Tipparbeit erspart und die Klaviernoten klarer macht.

Ausgewählte Schnipsel

Orchesterstichnoten zu einem Klavierauszug hinzufügen

Hier wird gezeigt, wie man das Hinzufügen von vielen Orchsterstichnoten zu einem Klavierauszug hinzufügen kann. Die musikalische Funktion `\cueWhile` braucht vier Argumente: Die Noten, von denen die Stichnoten formatiert werden sollen, definiert durch `\addQuote`, die Bezeichnung, die mit den Noten angegeben werden soll, dann entweder `#UP` (hoch) oder `#DOWN` (runter) zur Angabe von entweder `\voiceOne` mit der Bezeichnung über dem System oder `\voiceTwo` mit der Bezeichnung unter dem System, und schließlich die Klaviermusik, die parallel zu den Stichnoten gespielt werden soll. Die Bezeichnung des Stichnoteninstruments wird links der Stichnoten positioniert. Viele Abschnitte können zitiert werden, aber sie dürfen sich nicht überlappen.

```
cueWhile =
#(define-music-function
   (instrument name dir music)
   (string? string? ly:dir? ly:music?)
   #{
     \cueDuring $instrument #dir {
       \once \override TextScript.self-alignment-X = #RIGHT
       \once \override TextScript.direction = $dir
```

```
        <>-\markup { \tiny #name }
        $music
      }
    #})
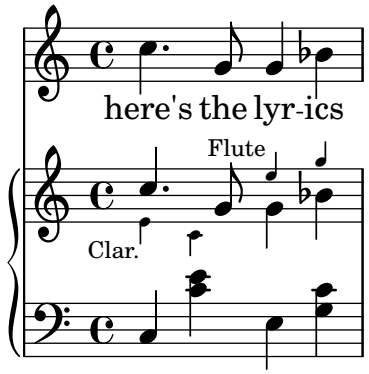
flute = \relative c'' {
  \transposition c'
  s4 s4 e g
}
\addQuote "flute" { \flute }

clarinet = \relative c' {
  \transposition bes
  fis4 d d c
}
\addQuote "clarinet" { \clarinet }

singer = \relative c'' { c4. g8 g4 bes4 }
words = \lyricmode { here's the lyr -- ics }

pianoRH = \relative c'' {
  \transposition c'
  \cueWhile "clarinet" "Clar." #DOWN { c4. g8 }
  \cueWhile "flute" "Flute" #UP { g4 bes4 }
}
pianoLH = \relative c { c4 <c' e> e, <g c> }

\score {
  <<
    \new Staff {
      \new Voice = "singer" {
        \singer
      }
    }
    \new Lyrics {
      \lyricsto "singer"
      \words
    }
    \new PianoStaff <<
      \new Staff {
        \new Voice {
          \pianoRH
        }
      }
      \new Staff {
        \clef "bass"
        \pianoLH
      }
    >>
  >>
}
```



Siehe auch

Glossar: Abschnitt “cue-notes” in *Glossar*.

Notationsreferenz: Abschnitt 36.1 [Objekte ausrichten], Seite 627, Abschnitt 35.2 [Richtung und Platzierung], Seite 611, Abschnitt 6.3.3 [Stichnoten formatieren], Seite 208, Abschnitt 6.3.2 [Andere Stimmen zitieren], Seite 205, Kapitel 37 [Musikfunktionen benutzen], Seite 637.

Schnipsel: Abschnitt “Vocal music” in *Schnipsel*.

Referenz der Interna: Abschnitt “InstrumentSwitch” in *Referenz der Interna*, Abschnitt “CueVoice” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

`\cueDuring` fügt automatisch einen `CueVoice`-Kontext ein, in dem alle Noten platziert werden. Das heißt, dass es nicht möglich ist, überlappende Abschnitte an Stichnoten mit dieser Technik zu haben. Überlappende Abschnitt könnten notiert werden, indem man explizit unterschiedliche `CueVoice`-Kontexte definiert und mit `\quoteDuring` die Noten ausschneidet und als Stichnoten einfügt.
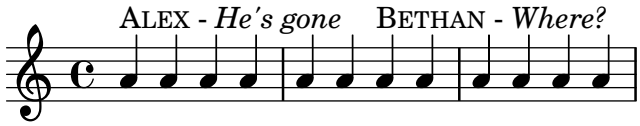
9.6.4 Gesprochene Musik

Effekte wie „Parlato“ bzw. „Sprechgesang“ erfordern, dass die Noten ohne Tonhöhe, aber mit dem notierten Rhythmus gesprochen werden. Solche Noten werden mit einem Kreuz als Notenkopf notiert, siehe hierzu Abschnitt 1.4.1 [Besondere Notenköpfe], Seite 36.

9.6.5 Dialog zur Musik

Dialoge zur Musik wird üblicherweise über den Notensystemen gesetzt, meistens in kursiver Schrift, wobei der Beginn der Phrasen mit einem musikalischen Moment verklammert ist.

```
\relative {
  a'4^\markup { \smallCaps { Alex - } \italic { He's gone } } a a a
  a4 a a^\markup { \smallCaps { Bethan - } \italic Where? } a
  a4 a a a
}
```



Für längere Abschnitt kann es nötig sein, die Noten zu dehnen, damit die Wörter besser passen. Es gibt keine Möglichkeit, das vollautomatisch von LilyPond erledigen zu lassen, und einige manuelle Änderungen am Seitenlayout sind nötig.

Für lange Phrasen und Passagen mit viel dicht gepackten Dialogen hilft es, einen `Lyrics`-Kontext zu benutzen. Der Kontext sollte nicht mit einer Stimme verknüpft sein, sondern jeder

Abschnitt des Dialogs sollte eine spezifische Dauer haben. Wenn es eine Pause im Dialog gibt, sollte das letzte Wort vom Rest getrennt werden und die Dauer zwischen ihnen aufgeteilt werdne, sodass die Noten darunter sich gut verteilen.
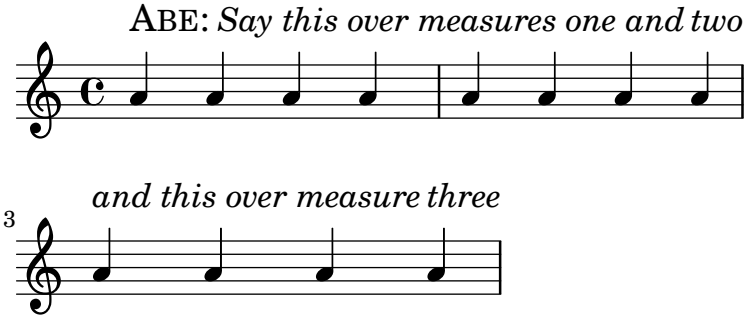
Wenn sich der Dialog über mehr als eine Zeile erstreckt, ist es nötig, manuell Umbrüche mit `\break` einzufügen und die Platzierung des Dialogs anzupassen, damit er nicht in den rechten Seitenrand läuft. Das letzte Wort des letzten Taktes einer Zeile sollte wie oben erklärt getrennt werden.

Hier ein Beispiel, das zeigt, wie das gesetzt werden kann.

```
music = \relative {
  \repeat unfold 3 { a'4 a a a }
}

dialogue = \lyricmode {
  \markup {
    \fontsize #1 \upright \smallCaps Abe:
    "Say this over measures one and"
  }4*7
  "two"4 |
  \break
  "and this over measure"4*3
  "three"4 |
}

\score {
  <<
    \new Lyrics \with {
      \override LyricText.font-shape = #'italic
      \override LyricText.self-alignment-X = #LEFT
    }
    { \dialogue }
    \new Staff {
      \new Voice { \music }
    }
  >>
}
```



Siehe auch

Notationsreferenz: Abschnitt 9.1.5 [Manuelle Silbendauern], Seite 260, Kapitel 8 [Text], Seite 226.

Referenz der Interna: Abschnitt “LyricText” in *Referenz der Interna*.

9.7 Psalmengesänge und Hymnen

Noten und Text für Psalmengesänge, Hymnen und Kirchengesänge haben eine spezifische Form in jeder Kirche. Auch wenn die Form sich unterscheidet, sind jedoch die typographischen Probleme sehr ähnlich und werden hier gesammelt behandelt.

9.7.1 Verweise für Psalmen und Hymnen

Wie der Gregorianische Choral in verschiedenen alten Notationsstilen gesetzt wird, findet sich in Kapitel 17 [Notation von alter Musik], Seite 420.

Siehe auch

Notationreferenz: Kapitel 17 [Notation von alter Musik], Seite 420.

Schnkpsel: Abschnitt “Vocal music” in *Schnipsel*.

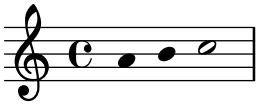
9.7.2 Kirchengesang notieren

Moderne Kirchengesänge benutzen eine Notation mit einer wechselnden Anzahl von Notationselementen der Notation alter Musik. Einige dieser Elemente und Methoden werden hier vorgestellt.

Kirchengesänge werden oft mit Viertelnoten ohne Hälse notiert, um die Tonhöhen darzustellen, während der Rhythmus sich am Rhythmus der gesprochenen Worte orientiert.

```
stemOff = { \hide Staff.Stem }
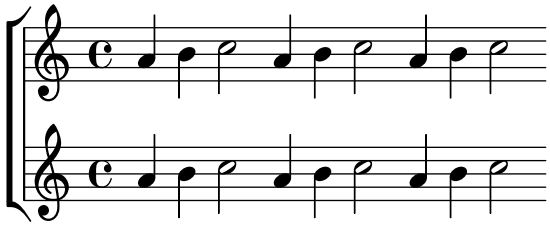
\relative c' {
  \stemOff
  a'4 b c2 |
}
```



Kirchengesänge verzichten üblicherweise auf die Taktstriche oder setzen gekürzte oder punktierte Taktstriche ein. Um Taktstriche auszulassen, kann der `Bar_engraver` entfert werden.
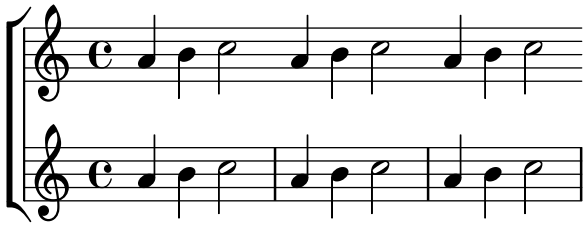
```
\score {
  \new StaffGroup <<
    \new Staff {
      \relative {
        a'4 b c2 |
        a4 b c2 |
        a4 b c2 |
      }
    }
    \new Staff {
      \relative {
        a'4 b c2 |
        a4 b c2 |
        a4 b c2 |
      }
    }
  >>
  \layout {
    \context {
      \Staff
```

```
      \remove Bar_engraver
    }
  }
}
```



Taktstriche können auf nur für ein System entfernt werden:
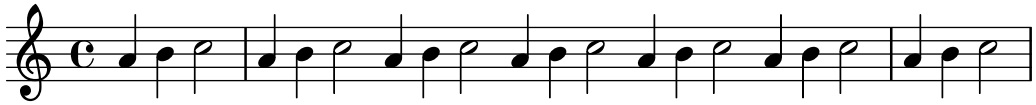
```
\score {
  \new ChoirStaff <<
    \new Staff
    \with { \remove Bar_engraver } {
      \relative {
        a'4 b c2 |
        a4 b c2 |
        a4 b c2 |
      }
    }
    \new Staff {
      \relative {
        a'4 b c2 |
        a4 b c2 |
        a4 b c2 |
      }
    }
  >>
}
```



Um Taktstriche nur von einem Abschnitt zu entfernen, kann die Musik als Kadenz notiert werdne. Wenn der Abschnitt lang ist, müssen unsichtbare Taktstriche mit `\bar ""` eingefügt werden, wo Zeilenumbrüche stattfinden sollen.
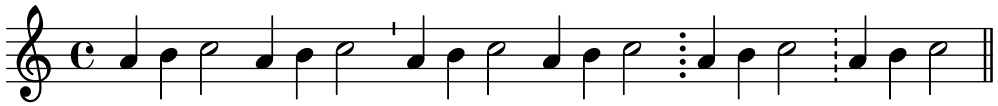
```
\relative a' {
  a4 b c2 |
  \cadenzaOn
  a4 b c2
  a4 b c2
  \bar ""
  a4 b c2
  a4 b c2
  \cadenzaOff
```

```
  a4 b c2 |
  a4 b c2 |
}
```



Pausen können als modifizierte Taktlinien notiert werden:

```
\relative a' {
  a4
  \cadenzaOn
  b c2
  a4 b c2
  \bar "'"
  a4 b c2
  a4 b c2
  \bar ";"
  a4 b c2
  \bar "!"
  a4 b c2
  \bar "||"
}
```
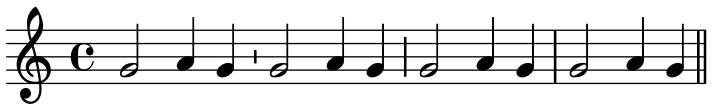


Alternativ werden die Pausenzeichen der Notation des Gregorianischen Chorals eingesetzt, obwohl die Noten selber modern sind. Das erreicht man durch Veränderung des `\breathe`-Zeichens:

```
divisioMinima = {
  \once \override BreathingSign.stencil = #ly:breathing-sign::divisio-minima
  \once \override BreathingSign.Y-offset = #0
  \breathe
}
divisioMaior = {
  \once \override BreathingSign.stencil = #ly:breathing-sign::divisio-maior
  \once \override BreathingSign.Y-offset = #0
  \breathe
}
divisioMaxima = {
  \once \override BreathingSign.stencil = #ly:breathing-sign::divisio-maxima
  \once \override BreathingSign.Y-offset = #0
  \breathe
}
finalis = {
  \once \override BreathingSign.stencil = #ly:breathing-sign::finalis
  \once \override BreathingSign.Y-offset = #0
  \breathe
}
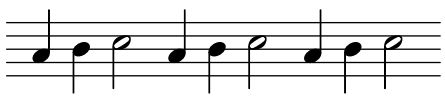
\score {
  \relative {
```

```
    g'2 a4 g
    \divisioMinima
    g2 a4 g
    \divisioMaior
    g2 a4 g
    \divisioMaxima
    g2 a4 g
    \finalis
  }
  \layout {
    \context {
      \Staff
      \remove Bar_engraver
    }
  }
}
```



Im Choral wird oft die Taktangabe und teilweise auch der Schlüssel weggelassen.

```
\score {
  \new Staff {
    \relative {
      a'4 b c2 |
      a4 b c2 |
      a4 b c2 |
    }
  }
  \layout {
    \context {
      \Staff
      \remove Bar_engraver
      \remove Time_signature_engraver
      \remove Clef_engraver
    }
  }
}
```



Gesänge für Psalmen der Anglikanischen Kirche werden normalerweise entweder *einfach*, mit 7 Takten, oder *doppelt*, mit zwei 7-Takt-Phrasen, notiert. Jede Siebener-Gruppe ist in zwei Hälften geteilt, die den Hälften jeder Strophe entsprechen, normalerweise durch eine Doppellinie getrennt. Nur halbe und ganze Noten werden genutzt. Der erste Takt jeder Hälfte hat immer eine ganze Note. Das ist der „Rezitationston“. Gesänge werden üblicherweise auf der Seite zentriert.

```
SopranoMusic = \relative {
  g'1 | c2 b | a1 | \bar "||"
  a1 | d2 c | c b | c1 | \bar "||"
}

AltoMusic = \relative {
  e'1 | g2 g | f1 |
  f1 | f2 e | d d | e1 |
}

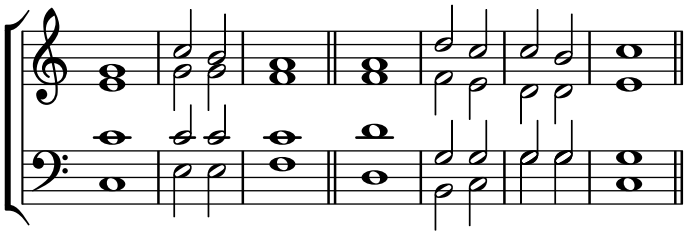
TenorMusic = \relative {
  c'1 | c2 c | c1 |
  d1 | g,2 g | g g | g1 |
}

BassMusic =  \relative {
  c1 | e2 e | f1 |
  d1 | b2 c | g' g | c,1 |
}

global = {
  \time 2/2
}

% Use markup to center the chant on the page
\markup {
  \fill-line {
    \score {  % centered
      <<
        \new ChoirStaff <<
          \new Staff <<
            \global
            \clef "treble"
            \new Voice = "Soprano" <<
              \voiceOne
              \SopranoMusic
            >>
            \new Voice = "Alto" <<
              \voiceTwo
              \AltoMusic
            >>
          >>
          \new Staff <<
            \clef "bass"
            \global
            \new Voice = "Tenor" <<
              \voiceOne
              \TenorMusic
            >>
            \new Voice = "Bass" <<
              \voiceTwo
              \BassMusic
```

```
            >>
          >>
        >>
      >>
      \layout {
        \context {
          \Score
          \override SpacingSpanner.base-shortest-duration = \musicLength 2
        }
        \context {
          \Staff
          \remove Time_signature_engraver
        }
      }
    }  % End score
  }
}  % End markup
```



Einige andere Herangehensweisen derartige Gesänge zu notieren, finden sich im ersten der folgenden Schnipsel.

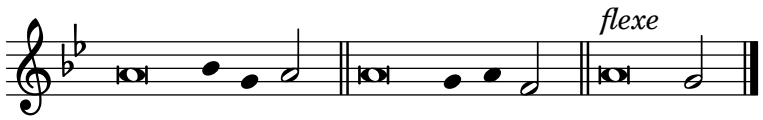
Ausgewählte Schnipsel

Chant or psalm notation

This form of notation is used for psalm chant, where verses are not always of the same length.

```
stemOff = \hide Staff.Stem
stemOn  = \undo \stemOff

\score {
  \new Staff \with { \remove "Time_signature_engraver" }
  {
    \key g \minor
    \cadenzaOn
    \stemOff a'\breve bes'4 g'4
    \stemOn a'2 \section
    \stemOff a'\breve g'4 a'4
    \stemOn f'2 \section
    \stemOff a'\breve^\markup { \italic flexe }
    \stemOn g'2 \fine
  }
}
```



Cantica und andere liturgische Texte können freier gesetzt werdne, und können auch Elementer der Notation alter Musik benutzen. Oft weren die Worte unter den Noten und an ihnen ausgerichtet gesetzt. In diesem Fall werden die Noten entsprechend der Ausdehnung des Textes und nicht der Notenlänge gesetzt.

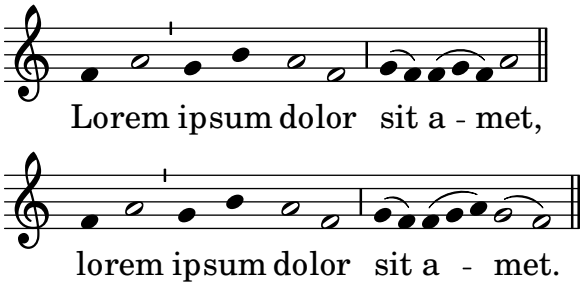
Ancient notation template – modern transcription of Gregorian music

This example demonstrates how to do modern transcription of Gregorian music. Gregorian music has no measure, no stems; it uses only half and quarter note heads, and special marks, indicating rests of different length.

```
chant = \relative c' {
  \set Score.timing = ##f
  f4 a2 \divisioMinima
  g4 b a2 f2 \divisioMaior
  g4( f) f( g f) a2 \finalis \break
  f4 a2 \divisioMinima
  g4 b a2 f2 \divisioMaior
  g4( f) f( g a) g2( f) \finalis
}

verba = \lyricmode {
  Lo -- rem ip -- sum do -- lor sit a -- met,
  lo -- rem ip -- sum do -- lor sit a -- met.
}

\score {
  \new GregorianTranscriptionStaff <<
    \new GregorianTranscriptionVoice = "melody" \chant
    \new GregorianTranscriptionLyrics = "one" \lyricsto melody \verba
  >>
}
```



Siehe auch

Handbuch zum Lernen: Abschnitt “Sichtbarkeit und Farbe von Objekten” in *Handbuch zum Lernen*, Abschnitt “Vokalensemble” in *Handbuch zum Lernen*.

Notationsreferenz: Kapitel 17 [Notation von alter Musik], Seite 420, Abschnitt 35.6 [Sichtbarkeit von Objekten], Seite 619, Abschnitt 2.5 [Takte], Seite 97, Abschnitt 2.3.4 [Musik ohne Metrum], Seite 73, Abschnitt 32.4 [Umgebungs-Plugins verändern], Seite 586, Abschnitt 17.4 [Gregorianischen Choral setzen], Seite 431.

9.7.3 Einen Psalm notieren

Der Text zu einem Anglikanischen Psalm wird normalerweise in separaten Versen zentriert under den Noten gesetzt.

Einfache Gesänge (mit sieben Takten) werden für jeden Vers wiederholt. Doppelte Gesänge (mit 14 Takten) werden für jeweils zwei Verse wiederholt. Zeichen zwischen den Wörtern zeigen an, wie man sie auf die Melodie anpasst. Jeder Vers wird in zwei Hälften geteilt. Ein Doppelpunkt wird benutzt, um die Teilung anzuzeigen. Das enstpricht einem doppelten Taktstrich in den Noten. Die Worte vor dem Doppelpunkt werden zu den ersten drei Takten gesungen, die Worte nach dem Doppelpunkt zu den vier letzten Takten.

Einfache Taktstriche (oder in einigen Psalmen ein umgedrehtes Komma) werden zwischen Wörtern eingefügt, um anzuzeigen, wie die Taktstriche der Noten positioniert werden. Im Beschriftungsmodus kann ein einfacher Taktstrich mit | notiert werden.

```
\markup {
  \fill-line {
    \column {
      \left-align {
        \line { O come let us sing | unto the | Lord : let }
        \line { us heartily rejoice in the | strength of | our }
        \line { sal- | -vation. }
      }
    }
  }
}
```

O come let us sing | unto the | Lord : let
us heartily rejoice in the | strength of | our
sal- | -vation.

Andere Symbole benötigen möglicherweise Zeichen aus den `fetaMusic`-Schriftarten. Siehe Abschnitt 8.3 [Schriftarten], Seite 247.

```
tick = \markup {
  \raise #1 \fontsize #-5 \musicglyph "scripts.rvarcomma"
}
\markup {
  \fill-line {
    \column {
      \left-align {
        \line { O come let us sing \tick unto the \tick Lord : let }
        \line {
          us heartily rejoice in the \tick strength of \tick our
        }
        \line { sal \tick vation. }
      }
    }
  }
}
```

O come let us sing ' unto the ' Lord : let
us heartily rejoice in the ' strength of ' our
sal ' vation.

Wenn in einem Takt nur eine ganze Note notiert ist, werden alle Worte dieses Taktes auf dieser Note im Sprechrhythmus gesungen. Wenn im Takt zwei Noten notiert sind, gibt es normalerweise auch nur eine oder zwei Silben. Wenn mehr Silben auf einen Takt gesungen werden sollen, wird mit einem Punkt angegeben, an welcher Stelle die Note gewechselt werden soll.

```
dot = \markup {
  \raise #0.7 \musicglyph "dots.dot"
}
tick = \markup {
  \raise #1 \fontsize #-5 \musicglyph "scripts.rvarcomma"
}
\markup {
  \fill-line {
    \column {
      \left-align {
        \line {
          O come let us sing \tick unto \dot the \tick Lord : let
        }
        \line {
          us heartily rejoice \bold \italic in the \tick strength of \tick our
        }
        \line { sal \tick vation. }
      }
    }
  }
}
```

O come let us sing ' unto • the ' Lord : let
us heartily rejoice in the ' strength of ' our
sal ' vation.

In einigen Psaltern wird ein Asterisk benutzt, um eine Pause in einem rezitierten Abschnitt anzuzeigen, und betonte oder verlängerte Silben werden mit fettem Text angezeigt:

```
dot = \markup {
  \raise #0.7 \musicglyph "dots.dot"
}
tick = \markup {
  \raise #1 \fontsize #-5 \musicglyph "scripts.rvarcomma"
}
\markup {
  \fill-line {
    \column {
      \left-align {
        \line { Today if ye will hear his voice * }
        \line {
          \concat { \bold hard en }
          | not your | hearts : as \bold \italic in the pro-
        }
        \line { vocation * and as \bold \italic in the \bold day of tempt- | }
        \line { -ation | \bold \italic in the | wilderness. }
      }
    }
  }
}
```

Today if ye will hear his voice *
harden | not your | hearts : as in the pro-
vocation * and as in the **day** of tempt- |
-ation | in the | wilderness.

Andere Psalter setzen einen Akzent über die Silbe, um eine Betonung anzuzeigen:

```
tick = \markup {
  \raise #2 \fontsize #-5 \musicglyph "scripts.rvarcomma"
}
\markup {
  \fill-line {
    \column {
      \left-align {
        \line {
          O come let us \concat {
            si \combine \tick ng
          }
          | unto the | Lord : let
        }
        \line {
          us heartily \concat {
            rejo \combine \tick ice
          }
          in the | strength of | our
        }
        \line { sal- | -vation. }
      }
    }
  }
}
```

O come let us sing′ | unto the | Lord : let
us heartily rejoice′ in the | strength of | our
sal- | -vation.

Der Einsatz von Beschriftung, um den Text zu zentrieren und die Zeilen in Spalten zu formatieren, findet sich in Abschnitt 8.2 [Text formatieren], Seite 233.

Die meisten dieser Elemente werden in einem der beiden Strophen der „Psalm“-Vorlage demonstriert, siehe Abschnitt “Vokalensemble” in *Handbuch zum Lernen*.

Siehe auch

Handbuch zum : Abschnitt “Vokalensemble” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 8.3 [Schriftarten], Seite 247, Abschnitt 8.2 [Text formatieren], Seite 233.

9.7.4 Unvollständige Takte in Hymnen

Hymnen beginnen und enden oft jede Zeile der Noten mit einem unvollständigen Takt, sodass jede Notenzeile exakt mit eine Textzeile übereinstimmt. Dazu setzt man den `\partial`-Befehl zu Beginn der Musik ein und `\bar "|"` oder `\bar "||"`, um die schließende Taktlinie am Ende der Zeile zu setzen.

Hymnus-Vorlage

Dieses Beispiel zeigt eine Möglichkeit, eine Hymnusmelodie zu setzen, in der jede Zeile mit einem Auftakt beginnt und einem unvollständigen Takt abschließt. Es zeigt auch, wie man die Strophen als allein stehenden Text unter die Noten hinzufügt.

```
Timeline = {
  \time 4/4
  \tempo 4=96
  \partial 2
  s2 | s1 | s2 \breathe s2 | s1 | s2 \caesura \break
  s2 | s1 | s2 \breathe s2 | s1 | s2 \fine
}

SopranoMusic = \relative g' {
  g4 g | g g g g | g g g g | g g g g | g2
  g4 g | g g g g | g g g g | g g g g | g2
}

AltoMusic = \relative c' {
  d4 d | d d d d | d d d d | d d d d | d2
  d4 d | d d d d | d d d d | d d d d | d2
}

TenorMusic = \relative a {
  b4 b | b b b b | b b b b | b b b b | b2
  b4 b | b b b b | b b b b | b b b b | b2
}

BassMusic = \relative g {
  g4 g | g g g g | g g g g | g g g g | g2
  g4 g | g g g g | g g g g | g g g g | g2
}

global = {
 \key g \major
}

\score {  % Start score
  \new PianoStaff <<  % Start pianostaff
    \new Staff <<  % Start Staff = RH
      \global
      \clef "treble"
      \new Voice = "Soprano" <<  % Start Voice = "Soprano"
        \Timeline
        \voiceOne
        \SopranoMusic
      >>  % End Voice = "Soprano"
      \new Voice = "Alto" <<  % Start Voice = "Alto"
        \Timeline
        \voiceTwo
        \AltoMusic
      >>  % End Voice = "Alto"
```

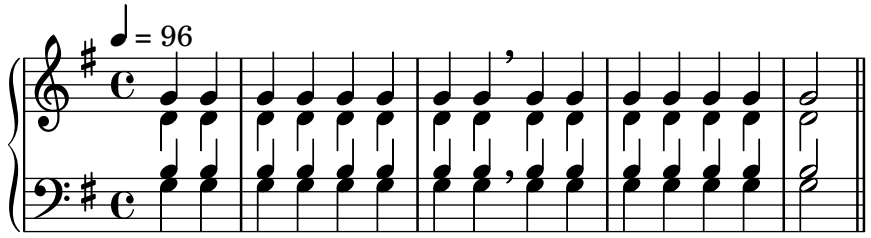
```
    >>  % End Staff = RH

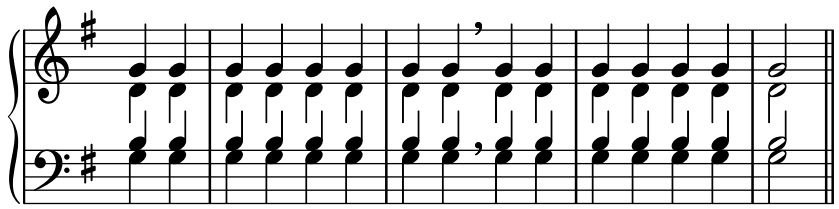
    \new Staff <<  % Start Staff = LH
      \global
      \clef "bass"
      \new Voice = "Tenor" <<  % Start Voice = "Tenor"
        \Timeline
        \voiceOne
        \TenorMusic
      >>  % End Voice = "Tenor"
      \new Voice = "Bass" <<  % Start Voice = "Bass"
        \Timeline
        \voiceTwo
        \BassMusic
      >>  % End Voice = "Bass"
    >>  % End Staff = LH
  >>  % End pianostaff
}  % End score

\markup \fill-line {
  \left-column {
    "This is line one of the first verse"
    "This is line two of the same"
    \null
    "And here's line one of the second verse"
    "And the next line of the same"
  }
}

\layout {
  \context {
    \Score
    caesuraType = #'((bar-line . "||"))
    fineBarType = "||"
  }
}

\paper {  % Start paper block
  indent = 0         % don't indent first system
  line-width = 130   % shorten line length to suit music
  tagline = ##f      % Don't print tag line, can be removed
}  % End paper block
```





This is line one of the first verse
This is line two of the same

And here's line one of the second verse
And the next line of the same
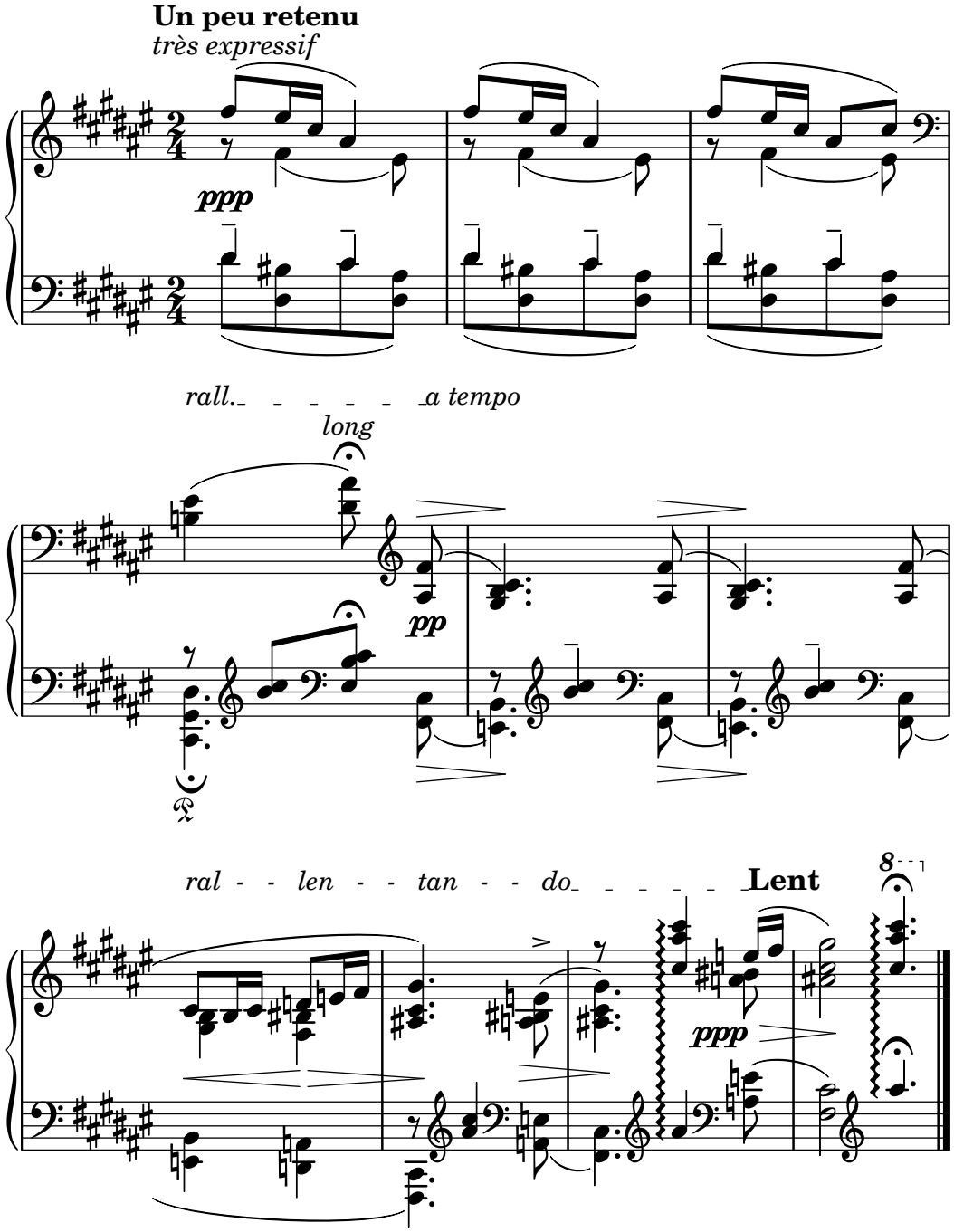
9.8 Alte Vokalmusik

Alte Vokalmusik ist unterstützt, wie erklärt in Kapitel 17 [Notation von alter Musik], Seite 420.

Siehe auch

Notationsreferenz: Kapitel 17 [Notation von alter Musik], Seite 420.

10 Tasteninstrumente und andere Instrumente mit mehreren Systemen



Dieser Abschnitt behandelt verschiedene Notationsaspekte, die typischerweise in Noten für Tasteninstrumente und andere Instrumente auf mehreren Notensystemen auftreten, wie etwa Harfe und Vibraphon. Hier wird die gesamte Gruppe von Instrumenten, die auf mehreren Systemen notiert werden, als „Tasteninstrumente“ bezeichnet, auch wenn einige von ihnen keine Tasten aufweisen.

10.1 Übliche Notation für Tasteninstrumente

Dieser Abschnitt zeigt allgemeine Eigenschaften des Notensatzes, die für die meisten Instrumente mit mehreren Systemen benötigt werden.

10.1.1 Referenz für Tasteninstrumente

Tasteninstrumente werden normalerweise auf einem Klaviersystem notiert. Es besteht aus zwei Notensystemen, die durch eine Klammer verbunden sind. Die gleiche Notation wird auch für andere Tasteninstrumente sowie Harfen verwendet. Orgelmusik wird normalerweise auf zwei Systemen innerhalb eines Klaviersystems notiert, denen noch ein drittes normales Notensystem für die Pedaltöne hinzugefügt wird.

Die Systeme eines Klaviersystems sind ziemlich unabhängig, aber Stimmen können bei Bedarf zwischen den Systemen wechseln.

Einige häufige Besonderheiten von Notation für Tasteninstrumenten wird an anderen Stellen besprochen:

- Noten für Tasteninstrumente haben oft mehrere Stimmen und die Anzahl der Stimmen kann sich häufig ändern. Das ist beschrieben in Abschnitt 5.2.3 [Auflösung von Zusammenstößen], Seite 170.
- Noten für Tasteninstrumente kann auch parallel, Takt für Takt notiert werden, wie gezeigt in Abschnitt 5.2.5 [Musik parallel notieren], Seite 179.
- Dynamikbezeichnung kann in einem `Dynamics`-Kontext notiert werden, der zwischen zwei `Staff`-Kontexten steht und dann horizontal zwischen diesen beiden zentriert wird; siehe Abschnitt 3.1.2 [Dynamik], Seite 123.
- Fingersatz wird erklärt in Abschnitt 7.1.2 [Fingersatzanweisungen], Seite 215.
- Orgelpedal-Zeichen werden als Artikulationszeichen notiert, siehe Abschnitt A.13 [Liste der Artikulationszeichen], Seite 760.
- Vertikale Rasterlinien können erstellt werden, siehe Abschnitt 7.2.2 [Gitternetzlinien], Seite 222.
- Noten für Tasteninstrumente beinhalten oft *Laissez vibrer*-Bögen und Bindebögen mit Arpeggio oder Tremolo, siehe hierzu Abschnitt 2.1.4 [Bindebögen], Seite 53.
- Arpeggios können auch zwischen den Systemen verbunden werden, siehe hierzu Abschnitt 3.3.2 [Arpeggio], Seite 138.
- Tremolo-Zeichen finden sich in Abschnitt 4.2.2 [Tremolo-Wiederholung], Seite 158.
- Viele der Optimierungen, die für Tastenmusik nötig sein können, sind demonstriert in Abschnitt “Beispiel aus dem Leben” in *Handbuch zum Lernen*.
- Unsichtbare Noten können eingesetzt werden, um Überbindungen zwischen Stimmen zu setzen, siehe Abschnitt “Andere Benutzung von Optimierungen” in *Handbuch zum Lernen*.

Siehe auch

Handbuch zum Lernen: Abschnitt “Beispiel aus dem Leben” in *Handbuch zum Lernen*, Abschnitt “Andere Benutzung von Optimierungen” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 6.1.2 [Systeme gruppieren], Seite 185, Abschnitt 6.3.1 [Instrumentenbezeichnungen], Seite 201, Abschnitt 5.2.3 [Auflösung von Zusammenstößen], Seite 170, Abschnitt 5.2.5 [Musik parallel notieren], Seite 179, Abschnitt 7.1.2 [Fingersatzanweisungen], Seite 215, Abschnitt A.13 [Liste der Artikulationszeichen], Seite 760, Abschnitt 7.2.2 [Gitternetzlinien], Seite 222, Abschnitt 2.1.4 [Bindebögen], Seite 53, Abschnitt 3.3.2 [Arpeggio], Seite 138, Abschnitt 4.2.2 [Tremolo-Wiederholung], Seite 158.

Schnipsel: Abschnitt “Keyboard and other multi-staff instruments” in *Schnipsel*.

Referenz der Interna: Abschnitt “PianoStaff” in *Referenz der Interna*.

10.1.2 Notensysteme manuell verändern

Stimmen können mit dem Befehl

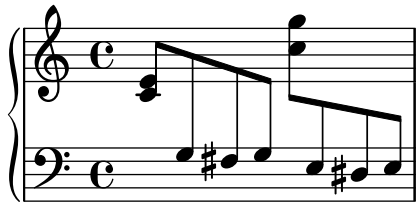
```
\change Staff = Systembezeichnung
```

manuell erzielt werden. Die Zeichenkette *Systembezeichnung* ist die Bezeichnung des Systems. Damit wird die aktuelle Stimme vom aktuellen System zu dem System mit der *Systembezeichnung* gewechselt. Üblicherweise ist die Systembezeichnung `"up"` oder `"down"`, `"RH"` oder `"LH"`.

Das System, zu dem die Stimme wechseln soll, muss zum Zeitpunkt des Wechsels existieren. Wenn notwendig, müssen Systeme „künstlich am Leben gehalten werden“, siehe Abschnitt 32.3 [Kontexte am Leben halten], Seite 583.

Balken zwischen den Systemen werden automatisch erstellt:

```
\new PianoStaff <<
  \new Staff = "up" {
    <e' c'>8
    \change Staff = "down"
    g8 fis g
    \change Staff = "up"
    <g'' c''>8
    \change Staff = "down"
    e8 dis e
    \change Staff = "up"
  }
  \new Staff = "down" {
    \clef bass
    % keep staff alive
    s1
  }
>>
```

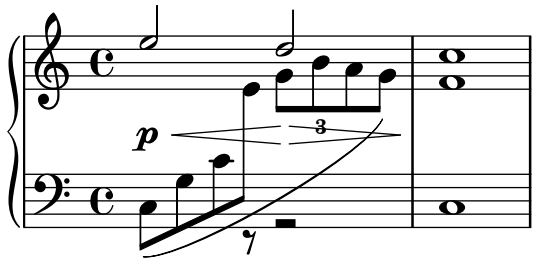


Wenn die Balken verändert werden müssen, sollte zuerst die Richtung des Balkens beeinflusst werden. Die Balkenposition wird dann von der Mitte des Systems gemessen, dass näher am Balken ist. Ein einfaches Beispiel ist gezeigt in Abschnitt “Überlappende Notation in Ordnung bringen” in *Handbuch zum Lernen*.

Bei Stimmen, die zwischen den Systemen wechseln, kann es zu überlappender Notation kommen:

```
\new PianoStaff <<
  \new Staff = "up" {
    \voiceOne
    % Make space for fingering in the cross-staff voice
    \once\override DynamicLineSpanner.staff-padding = #3.4
    e''2\p\< d''\>
    c''1\!
  }
  \new Staff = "down" <<
    {
      \clef bass
      s4. e,8\rest g,2\rest
      c1
    } \\ {
```

```
        c8\( g c'
        \change Staff = "up"
        e' g' b'-3 a' g'\)
        f'1
      }
    >>
>>
```



Die Hälse und Bögen überlappen sich mit der dazwischenstehenden Dynamik-Zeile, weil die automatische Zusammenstoßauflösung für Balken, Bögen und andere Strecker, die Noten zwischen unterschiedlichen Systemen verbinden, ausgeschaltet ist. Das gilt auch für Hälse und Artikulationszeichen, wenn ihre Positionierung durch einen Strecker zwischen Systemen verändert wird. Die resultierenden Zusammenstöße müssen manuell aufgelöst werden, wo es nötig ist, dabei kann man die Methoden anwenden, die in Abschnitt "Überlappende Notation in Ordnung bringen" in *Handbuch zum Lernen* gezeigt werden.

Siehe auch

Handbuch zum Lernen: Abschnitt "Überlappende Notation in Ordnung bringen" in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 7.1.6 [Hälse], Seite 220, Abschnitt 2.4.1 [Automatische Balken], Seite 81, Abschnitt 32.3 [Kontexte am Leben halten], Seite 583.

Schnipsel: Abschnitt "Keyboard and other multi-staff instruments" in *Schnipsel*.

Referenz der Interna: Abschnitt "Beam" in *Referenz der Interna*, Abschnitt "ContextChange" in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Die Zusammenstoßauflösung für Balken funktioniert nicht für Balken, die direkt am Ende eines Systems enden. In diesem Fall muss man manuelle Balken einsetzen.
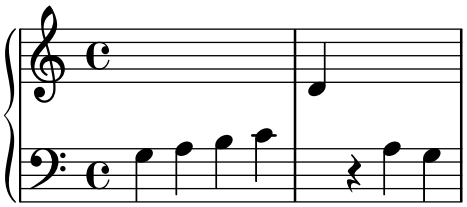
10.1.3 Automatischer Systemwechsel

Stimmen können angewiesen werden, automatisch zwischen dem oberen und unteren System zu wechseln. Die Syntax hierfür lautet:

```
\autoChange ...Noten...
```

Damit werden zwei Notensysteme innerhalb des aktiven Klaviersystems erstellt, die „oben" (`up`) und „unten" (`down`) genannt werden. Auf dem unteren System wird als Standard der Bassschlüssel gesetzt. Der Wechsel wird automatisch basierend auf der Tonhöhe der Note vorgenommen (als Wechselpunkt gilt das eingestrichene C). Dabei wird die Richtung auch über Pausen hinweg im Voraus bestimmt.

```
\new PianoStaff {
  \autoChange {
    g4 a b c'
    d'4 r a g
  }
}
```



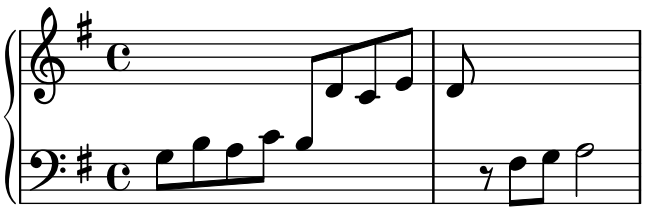
Ein \relative-Abschnitt, der sich außerhalb des \autoChange-Abschnittes befindet, hat keinen Einfluss auf die Notenhöhen.

Wenn individuelle Kontrolle über die einzelnen Systeme benötigt wird, können sie manuell mit den Bezeichnungen "up" und "down" erstellt werden. Der \autoChange-Befehl wechselt dann die Stimme zwischen den Systemen.

> **Achtung:** Wenn Systeme manuell erstellt werden, **müssen** sie genau die Bezeichnungen "up" und "down" bekommen, damit die automatische Wechselfunktion sie erkennen kann.

Systeme müssen etwa manuell erstellt werden, damit die Tonart im unteren System gesetzt werden kann:

```
\new PianoStaff <<
  \new Staff = "up" {
    \new Voice = "melodieEins" {
      \key g \major
      \autoChange \relative {
        g8 b a c b d c e
        d8 r fis, g a2
      }
    }
  }
  \new Staff = "down" {
    \key g \major
    \clef bass
  }
>>
```



Siehe auch

Notationsreferenz: Abschnitt 10.1.2 [Notensysteme manuell verändern], Seite 315.

Schnipsel: Abschnitt “Keyboard and other multi-staff instruments” in *Schnipsel*.

Bekannte Probleme und Warnungen

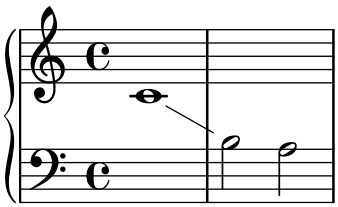
Die Aufteilung auf die Systeme geschieht nicht unbedingt an optimaler Stelle. Für bessere Qualität müssen die Wechsel manuell eingestellt werden.

Akkorde werde nicht über die Systeme verteilt, sie werden dem System zugewiesen, auf dem sich ihre erste Note befinden würde.

10.1.4 Stimmführungslinien

Immer, wenn eine Stimme von einem Klaviersystem zu dem anderen wechselt, kann automatisch eine Linie zur Verdeutlichung des Stimmenverlaufs ausgegeben werden:

```
\new PianoStaff <<
  \new Staff = "one" {
    \showStaffSwitch
    c'1
    \change Staff = "two"
    b2 a
  }
  \new Staff = "two" {
    \clef bass
    s1*2
  }
>>
```



Vordefinierte Befehle

`\showStaffSwitch`, `\hideStaffSwitch`.

Siehe auch

Schnipsel: Abschnitt “Keyboard and other multi-staff instruments” in *Schnipsel*.

Referenz der Interna: Abschnitt “Note_head_line_engraver” in *Referenz der Interna*, Abschnitt “VoiceFollower” in *Referenz der Interna*.

10.1.5 Hälse über beide Systeme

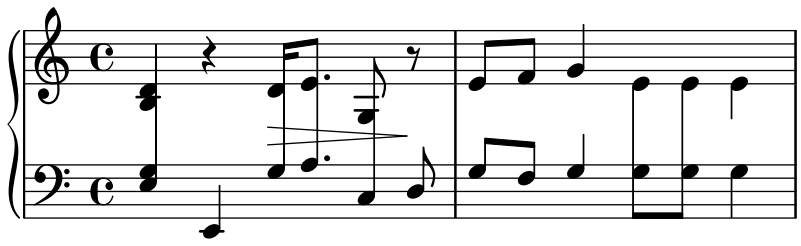
Akkorde, die über zwei Systeme reichen, können mit dem `Span_stem_engraver` erstellt werden. Man muss dabei sicherstellen, dass die automatische Bebalkung die Noten nicht auf dem einen System mit Balken versieht, wenn es auf dem anderen nicht nötig wäre.

```
\layout {
  \context {
    \PianoStaff
    \consists "Span_stem_engraver"
  }
}

\new PianoStaff <<
  \new Staff {
    <b d'>4 r d'16\> e'8. g8 r\! |
```

```
    e'8 f' g'4
      \voiceTwo
      % Down to lower staff
      \crossStaff { e'8 e'8 } e'4 |
  }

  \new Staff {
    \clef bass
    \voiceOne
    % Up to upper staff
    \crossStaff { <e g>4 e, g16 a8. c8 } d |
    g8 f g4 \voiceTwo g8 g g4 |
  }
>>
```



Ausgewählte Schnipsel

Indicating cross-staff chords with a bracket

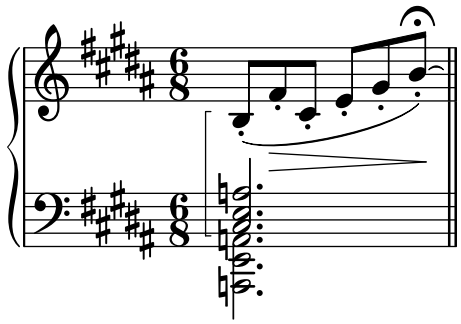
A non-arpeggiato bracket can indicate that notes on two different staves are to be played with the same hand. In order to do this, the `PianoStaff` must be set to accept cross-staff brackets.

The following example typesets measure 65 of Debussy's prelude *Les collines d'Anacapri*.

```
\new PianoStaff <<
  \once \set PianoStaff.connectChordBrackets = ##t

  \new Staff \relative c' {
    \key b \major
    \time 6/8
    b8-.(\nonArpeggiato fis'-.\> cis-.
      e-. gis-. b-.)\!\fermata^\laissezVibrer
    \section
  }

  \new Staff \relative c' {
    \clef bass
    \key b \major
    << { <a e cis>2.\nonArpeggiato } \\
       { <a, e a,>2. } >>
    \section
  }
>>
```



Siehe auch

Schnipsel: Abschnitt “Keyboard and other multi-staff instruments” in *Schnipsel*.

Referenz der Interna: Abschnitt “Stem” in *Referenz der Interna*.
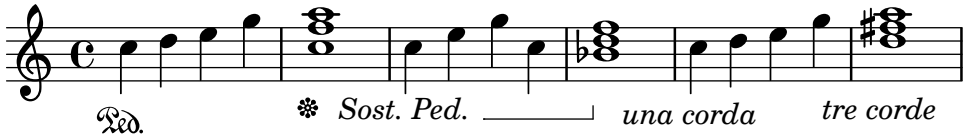
10.2 Klavier

Dieser Abschnitt zeigt Eigenheiten der Notation von Klavermusik

10.2.1 Klavierpedal

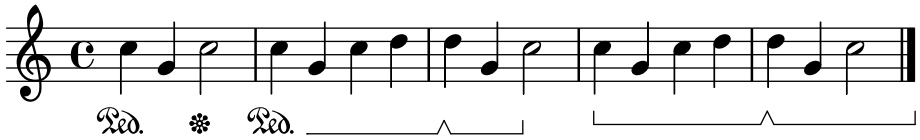
Klaviere besitzen üblicherweise drei Pedale, die den Klang beeinflussen: das Haltepedal (rechts), das Una-corda-Pedal (links) und das Sostenuto-Pedal (in der Mitte). Die englischen Begriffe hierzu lauten *sustain*, *una corda* und *sostenuto*. Vibraphone und Celestas haben ebenfalls ein Haltepedal.

```
\relative {
  c''4\sustainOn d e g
  <c, f a>1\sustainOff
  c4\sostenutoOn e g c,
  <bes d f>1\sostenutoOff
  c4\unaCorda d e g
  <d fis a>1\treCorde
}
```



Die Pedalbezeichnung kann auf drei Arten vorgenommen werden: mit Text, Klammern oder einer Mischung aus beidem. Das Haltepedal und das Una-corda-Pedal benutzen als Standard die Textdarstellung, während das Sostenuto-Pedal den gemischten Stil benutzt:

```
\relative {
  c''4\sustainOn g c2\sustainOff
  \set Staff.pedalSustainStyle = #'mixed
  c4\sustainOn g c d
  d\sustainOff\sustainOn g, c2\sustainOff
  \set Staff.pedalSustainStyle = #'bracket
  c4\sustainOn g c d
  d\sustainOff\sustainOn g, c2
  \bar "|."
}
```



Die Platzierung der Befehle entspricht der Bewegung der Pedale während des Spielens. Um das Pedal bis zur letzten Taktlinie zu halten, muss der letzte Pedal-hoch-Befehl weggelassen werden.

Pedalbezeichnungen können innerhalb eines `Dynamics`-Kontextes notiert werden, sodass sie an einer horizontalen Linie ausgerichtet werden.

Siehe auch

Notationsreferenz: Abschnitt 2.1.4 [Bindebögen], Seite 53.

Schnipsel: Abschnitt “Keyboard and other multi-staff instruments” in *Schnipsel*.

Referenz der Interna: Abschnitt “SustainPedal” in *Referenz der Interna*, Abschnitt “SustainPedalLineSpanner” in *Referenz der Interna*, Abschnitt “SustainEvent” in *Referenz der Interna*, Abschnitt “SostenutoPedal” in *Referenz der Interna*, Abschnitt “SostenutoPedalLineSpanner” in *Referenz der Interna*, Abschnitt “SostenutoEvent” in *Referenz der Interna*, Abschnitt “UnaCordaPedal” in *Referenz der Interna*, Abschnitt “UnaCordaPedalLineSpanner” in *Referenz der Interna*, Abschnitt “UnaCordaEvent” in *Referenz der Interna*, Abschnitt “PianoPedalBracket” in *Referenz der Interna*, Abschnitt “Piano_pedal_engraver” in *Referenz der Interna*.

10.3 Akkordeon

Dieser Abschnitt behandelt Notation, die nur für Akkordeonmusik benötigt wird.

10.3.1 Diskant-Symbole

Akkordeons werden oft mit mehreren Reihen an Zungen gebaut, welche Unisono oder eine Oktave höher bzw. tiefer erklingen. Jedes Akkordeon hat eigene Bezeichnungen für die Register (engl. shift) wie etwa *Oboe*, *Bandonium* usw. Eine Anzahl an Symbolen wird benutzt um die Wechsel anzuzeigen.

Siehe auch

Schnipsel: Abschnitt “Keyboard and other multi-staff instruments” in *Schnipsel*.

10.4 Harfe

Dieser Abschnitt zeigt Eigenheiten der Notation für Harfe.

10.4.1 Referenzen für Harfe

Einige übliche Notationseigenheiten für Harfe sind woanders behandelt:

- Glissando ist die üblichste Harfentechnik, siehe Abschnitt 3.3.1 [Glissando], Seite 136.
- Ein *Bisbigliando* wird als ein Tremolo notiert, siehe Abschnitt 4.2.2 [Tremolo-Wiederholung], Seite 158.
- Flageolettöne werden hier beschrieben: Abschnitt 11.1.3 [Flageolett], Seite 325.
- Für Arpeggio und non-arpeggio, siehe Abschnitt 3.3.2 [Arpeggio], Seite 138.
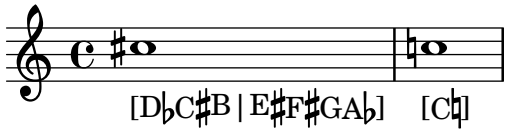
Siehe auch

Notationsreferenz: Abschnitt 4.2.2 [Tremolo-Wiederholung], Seite 158, Abschnitt 3.3.1 [Glissando], Seite 136, Abschnitt 3.3.2 [Arpeggio], Seite 138, Abschnitt 11.1.3 [Flageolett], Seite 325.

10.4.2 Harfenpedal

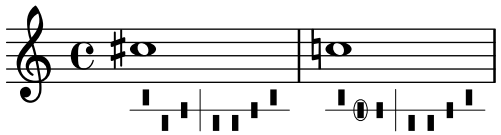
Harfe haben sieben Saiten in einer Oktave, die entweder als normaler Ton, oder aber erhöht bzw. erniedrigt klingen können. Bei einer Hakenharfe kann man jede Saite einzeln einstellen, bei Pedalharfen aber wird jede Saite mit der gleichen Notenbezeichnung von einem einzigen Pedal kontrolliert. Vom Spieler aus gesehen von rechts nach links sind die Pedale: D, C und H für die linke und E, F, G und A für die rechte Seite. Die Position des Pedals kann mit Textbeschriftungselementen:

```
\textLengthOn
cis''1_\markup \concat \vcenter {
  [D \flat C \sharp B|E \sharp F \sharp G A \flat] }
c''!1_\markup \concat \vcenter {
  [ C \natural ] }
```



oder Pedaldiagrammen angezeigt werden:

```
\textLengthOn
cis''1_\markup { \harp-pedal "^v-|vv-^" }
c''!1_\markup { \harp-pedal "^o--|vv-^" }
```

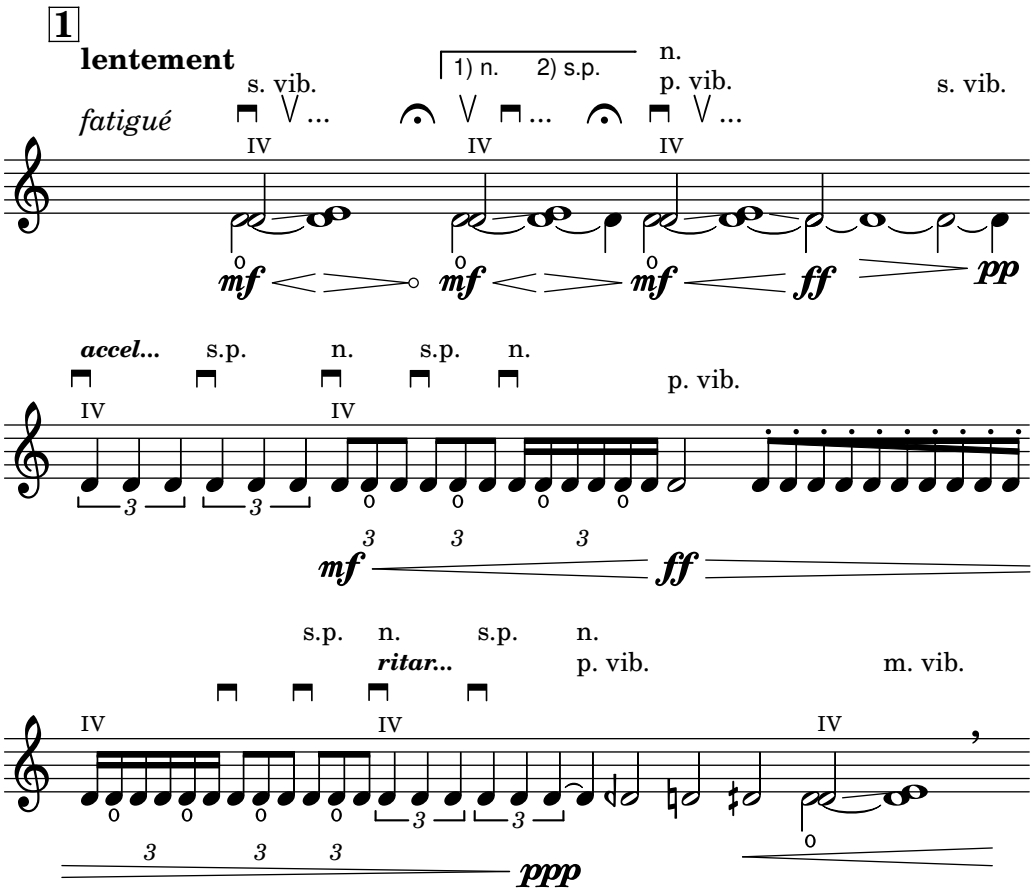


Der `\harp-pedal`-Befehl braucht eine Anzahl an Zeichen, von welchen `^` die höchste Pedalposition (erniedrigte Tonhöhe), `-` die mittlere Pedalposition (normale Tonhöhe, `v` die tiefste Pedalposition (erhöhter Ton) anzeigt. `|` ist ein Trenner. Ein `o` vor der Definition umrandet das Symbol.

Siehe auch

Notationsreferenz: Abschnitt 8.1.1 [Textarten], Seite 226, Abschnitt A.10.6 [Instrument-specific markup], Seite 734.

11 Bundlose Saiteninstrumente



Dieser Abschnitt stellt Information und Referenzen zur Verfügung, die beim Setzen von Noten für Saiteninstrumente ohne Bund herangezogen werden können.

11.1 Übliche Notation für bundlose Saiteninstrumente

Es gibt wenige Spezifikationen für die Notation von Saiteninstrumenten ohne Bünde. Die Noten werden auf einem System notiert und meistens ist auch nur eine Stimme erforderlich. Zwei Stimmen können für Doppelgriff- oder Divisi-Stellen erforderlich sein.

11.1.1 Hinweise für bundlose Saiteninstrumente

Die meisten Notationseigenschaften, die für Orchesterstreicher eingesetzt werden, sind an anderer Stelle beschrieben:

- Textanweisungen wie „pizz.“ oder „arco“ werden als einfacher Text eingefügt, siehe Abschnitt 8.1.1 [Textarten], Seite 226.
- Fingersatz, auch das Zeichen für den Daumen, ist erklärt in Abschnitt 7.1.2 [Fingersatzanweisungen], Seite 215.
- Doppelgriffe werden normalerweise als Akkord notiert, siehe hierzu Abschnitt 5.1.1 [Noten mit Akkorden], Seite 160. Anweisungen, wie Akkorde gespielt werden sollen, können auch hinzugefügt werden, siehe Abschnitt 3.3.2 [Arpeggio], Seite 138.
- Eine Vorlage für Streichquartett findet sich in Abschnitt “Streichquartett” in *Handbuch zum Lernen*. Andere sind als Schnipsel zur Verfügung gestellt.

Siehe auch

Handbuch zum Lernen: Abschnitt “Streichquartett” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 8.1.1 [Textarten], Seite 226, Abschnitt 7.1.2 [Fingersatzanweisungen], Seite 215, Abschnitt 5.1.1 [Noten mit Akkorden], Seite 160, Abschnitt 3.3.2 [Arpeggio], Seite 138.
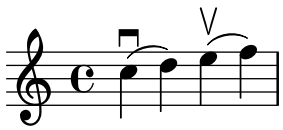
Schnipsel: Abschnitt “Unfretted string instruments” in *Schnipsel*.

11.1.2 Bezeichnung des Bogens

Hinweise zur Bogenfügung können als Artikulationen erstellt werden, wie beschrieben in Abschnitt 3.1.1 [Artikulationszeichen und Verzierungen], Seite 120.
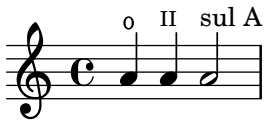
Die Befehle `\upbow` und `\downbow` werden mit Legatobögen in folgender Weise eingesetzt:

```
\relative { c''4(\downbow d) e(\upbow f) }
```



und das nächste Beispiel zeigt drei Arten, eine offene A-Saite auf der Geige anzuzeigen:

```
a4 \open
a^\markup { \teeny "II" }
a2^\markup { \small "sul A" }
```



Vordefinierte Befehle

`\downbow`, `\upbow`, `\open`.

Siehe auch
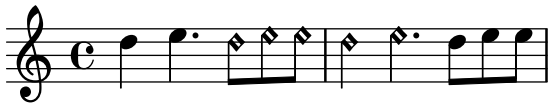
Notation Reference: Abschnitt 3.1.1 [Artikulationszeichen und Verzierungen], Seite 120, Abschnitt 3.2.1 [Legatobögen], Seite 130.

11.1.3 Flageolett

Natürliches Flageolett
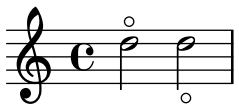
Flageolett-Töne können auf verschiedene Arten notiert werden. Üblicherweise werden sie mit einem Rautenkopf notiert, wenn ein Ton angezeigt werde, bei dem die Saite berührt wird, wo sie sonst abgegriffen würde.

```
\relative d'' {
  d4 e4.
  \harmonicsOn
  d8 e e
  d4 e4.
  \harmonicsOff
  d8 e e
}
```



Alternativ kann auch eine normale Noten die Tonhöhe anzeigen, die erklingen soll, wobei ein kleiner Kreis angibt, dass es sich um einen Flageolett-Ton handelt:
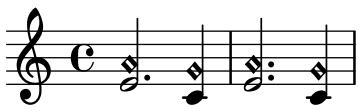
```
d''2^\flageolet d''_\flageolet
```



Künstliches Flageolett

Künstliche Flageoletttöne werden mit zwei Noten notiert, von denen einen einen normalen Notenkopf besitzt und die Griffposition des Fingers angibt, während die andere in Rautenform die Position des leicht aufgesetzten Fingers anzeigt.

```
\relative e' {
  <e a\harmonic>2.  <c g'\harmonic>4
  \set harmonicDots = ##t
  <e a\harmonic>2.  <c g'\harmonic>4
}
```



Achtung: `\harmonic` **muss** innerhalb einer Akkordkonstruktion gesetzt werden, auch wenn nur eine Note gesetzt wird. Normalerweise würde `\harmonicsOn` in dieser Situation benutzt.

Siehe auch

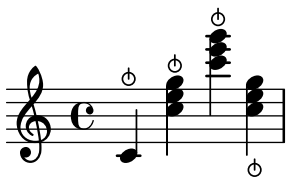
Glossar: Abschnitt “harmonics” in *Glossar*.

Notationsreferenz: Abschnitt 1.4.1 [Besondere Notenköpfe], Seite 36, Abschnitt 11.1.1 [Hinweise für bundlose Saiteninstrumente], Seite 324.
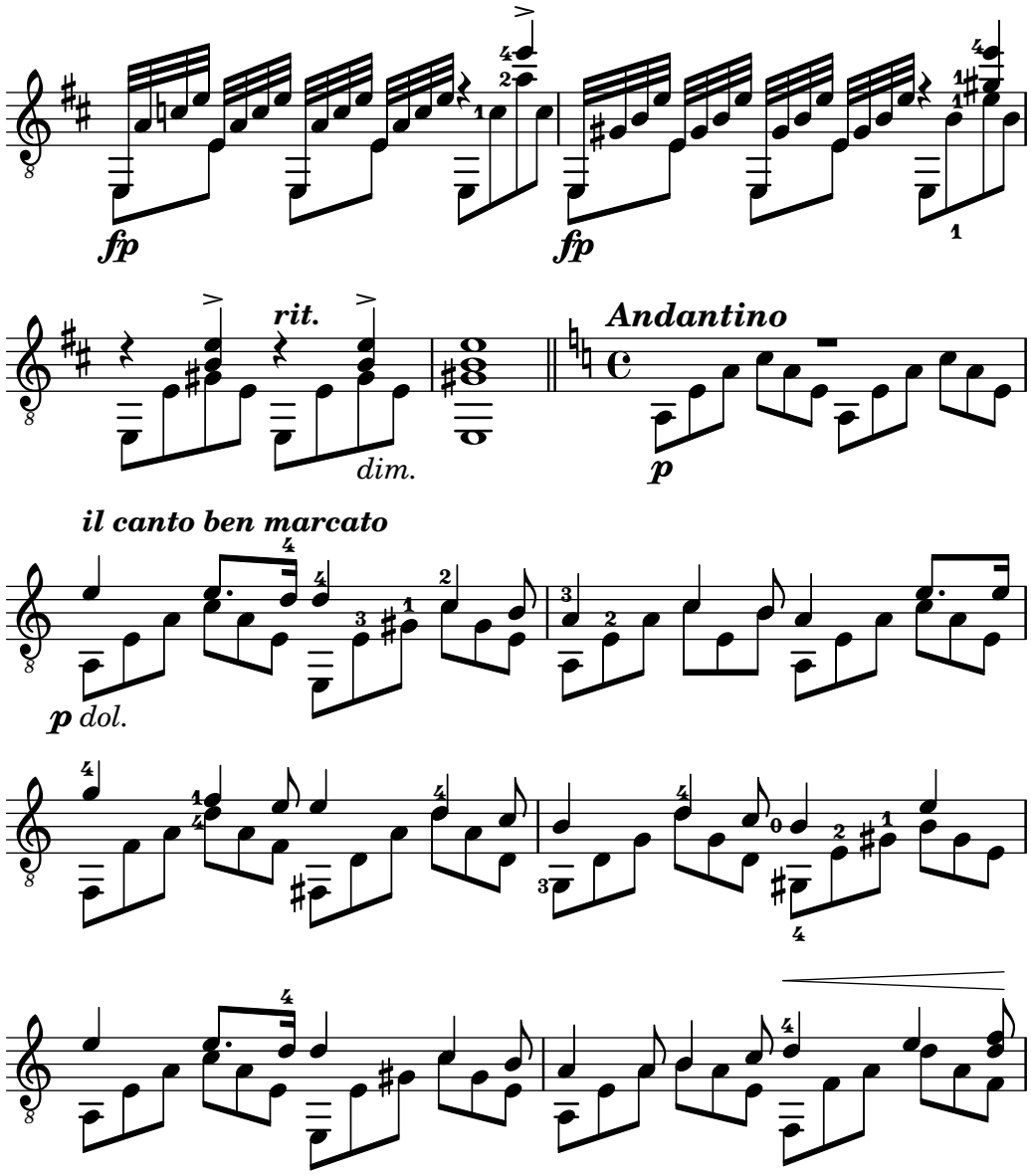
11.1.4 Bartók-Pizzicato

Ein Knallpizzicato, auch als Bartók-Pizzicato bekannt, ist ein hartes Pizzicato, bei dem man die Saite nach oben (und nicht seitlich) zieht, sodass sie beim Schwingen das Griffbrett berührt.

```
\relative {
  c'4\snappizzicato
  <c' e g>4\snappizzicato
  <c' e g>4^\snappizzicato
  <c, e g>4_\snappizzicato
}
```



12 Saiteninstrumente mit Bünden



Dieser Abschnitt erklärt bestimmte Eigenheiten der Notation für Saiteninstrumente mit Bünden.

12.1 Übliche Notation für Saiteninstrumente mit Bünden

Dieser Abschnitt zeigt Besonderheiten der Notation, die allen Bundinstrumenten eigen ist.

12.1.1 Referenz für Saiteninstrumente mit Bünden

Noten für Bundinstrumente wird normalerweise auf einem einzelnen System notiert, entweder als traditionelles Notensystem oder in Tabulaturform. Manchmal werden beide Arten miteinander verbunden, und besonders in populärer Musik ist es üblich, über dem traditionellen System Griffsymbole zu setzen. Gitarre und Banjo sind transponierende Instrumente, die eine Oktave tiefer klingen als sie notiert werden. Partituren für diese Instrumente sollten den „Tenorschlüssel“ (`"treble_8"` bzw. `\transposition c`) benutzen, um korrekte MIDI-Dateien zu erhalten. Einige Spezifika für Instrumente mit Bünden sind an anderer Stelle erklärt:

- Fingersatz kann notiert werden, siehe Abschnitt 7.1.2 [Fingersatzanweisungen], Seite 215.

- Anweisungen für *Laissez vibrer*-Bögen und Bögen zwischen Arpeggios und Tremolos sind beschrieben in Abschnitt 2.1.4 [Bindebögen], Seite 53.
- Hinweise, wie mehrere Stimmen gesetzt werden können, finden sich in Abschnitt 5.2.3 [Auflösung von Zusammenstößen], Seite 170.
- Instructions for indicating harmonics can be found in Abschnitt 11.1.3 [Flageolett], Seite 325.

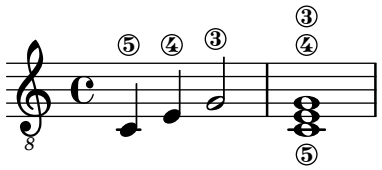
Siehe auch

Notationsreferenz: Abschnitt 7.1.2 [Fingersatzanweisungen], Seite 215, Abschnitt 2.1.4 [Bindebögen], Seite 53, Abschnitt 5.2.3 [Auflösung von Zusammenstößen], Seite 170, Abschnitt 6.3.1 [Instrumentenbezeichnungen], Seite 201, Abschnitt 5.2.5 [Musik parallel notieren], Seite 179, Abschnitt 3.3.2 [Arpeggio], Seite 138, Abschnitt A.13 [Liste der Artikulationszeichen], Seite 760, Abschnitt 1.3.1 [Notenschlüssel], Seite 19, Abschnitt 1.3.4 [Transposition von Instrumenten], Seite 25.

12.1.2 Seitennummerbezeichnung

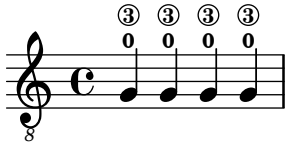
Die Nummer der Saite, auf der gespielt werden soll, kann angezeigt werden, indem `\Zahl` an eine Note gehängt wird:

```
\clef "treble_8"
c4\5 e\4 g2\3
<c\5 e\4 g\3>1
```



Wenn Fingersatz und Saitennummer zusammen benutzt werden, wird ihre Position anhand der Reihenfolge, mit der sie im Code auftauchen, *nur* entschieden, wenn sie in einem expliziten Akkord auftreten: Fingersatz, der außen an einen Akkord oder an einzelne Noten *außerhalb* eines Akkords gehängt wird, wird nach anderen Regeln positioniert.

```
\clef "treble_8"
g4\3-0
g-0\3
<g\3-0>
<g-0\3>
```

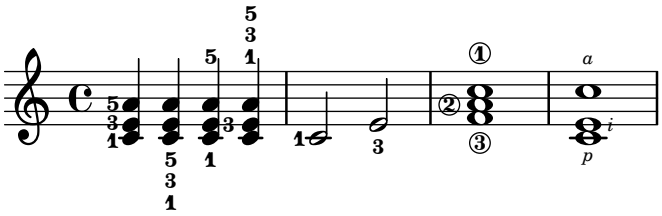


Ausgewählte Schnipsel

Position von Fingersatz in Akkorden kontrollieren

Die Position von Fingersatzzahlen kann exakt kontrolliert werden.
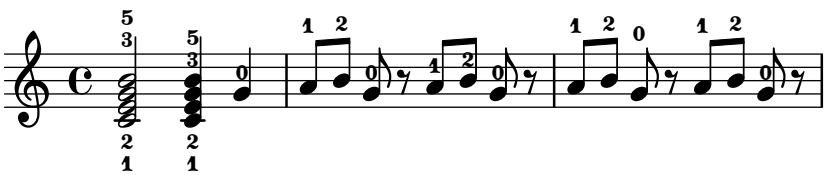
```
\relative c' {
  \set fingeringOrientations = #'(left)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(down)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(down right up)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(up)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(left)
  <c-1>2
  \set fingeringOrientations = #'(down)
  <e-3>2
  \set stringNumberOrientations = #'(up left down)
  <f\3 a\2 c\1>1
  \set strokeFingerOrientations = #'(down right up)
  <c\rightHandFinger 1 e\rightHandFinger 2 c'\rightHandFinger 4 >
}
```



Fingersatz auch innerhalb des Systems setzen

Normalerweise werden vertikal orientierte Fingersatzzahlen außerhalb des Systems gesetzt. Das kann aber verändert werden.

```
\relative c' {
  <c-1 e-2 g-3 b-5>2
  \override Fingering.staff-padding = #'()
  <c-1 e-2 g-3 b-5>4 g'-0
  a8[-1 b]-2 g-0 r
  \override Fingering.add-stem-support = ##f
  a[-1 b]-2 g-0 r
  \override Fingering.add-stem-support = ##t
  a[-1 b]-2 g-0 r
  \override Fingering.add-stem-support = #only-if-beamed
  a[-1 b]-2 g-0 r
}
```



Siehe auch

Notationsreferenz: Abschnitt 7.1.2 [Fingersatzanweisungen], Seite 215.

Schnipsel: Abschnitt “Fretted string instruments” in *Schnipsel*.

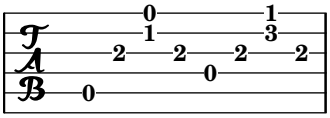
Referenz der Interna: Abschnitt “StringNumber” in *Referenz der Interna*, Abschnitt “Fingering” in *Referenz der Interna*.

12.1.3 Standardtabulaturen

Musik für gezupfte Saiteninstrumente wird oft notiert, indem man eine Finger/Berührungsnotation bzw. Tabulatur benutzt. Im Gegensatz zur traditionellen Notation werden hier Tonhöhen nicht mit Notenköpfen notiert, sondern mit Zahlen (oder buchstabenartigen Symbolen in historischen Tabulaturen). Die Notenlinien einer Tabulatur zeigen die Saite an, auf der eine Note gespielt werden soll, und eine Zahl auf einer Notenlinie zeigt an, welcher Bund für eine Note gespielt werden muss. Die Zahlen werden vertikal übereinander geschrieben, wenn sie gleichzeitig gespielt werden sollen.

Standardmäßig ist Saite 1 die höchste Saite und entspricht der höchsten Notenlinie des `TabStaff` (der Tabulatur). Die voreingestellte Saitenstimmung der Tabulatur ist die normale Gitarrenstimmung (mit 6 Saiten). Die Noten werden als Tabulatur ausgegeben, wenn man den `TabStaff`-Kontext und darin den `TabVoice`-Kontext benutzt. Ein kalligraphischer Tabulaturschlüssel wird automatisch hinzugefügt.

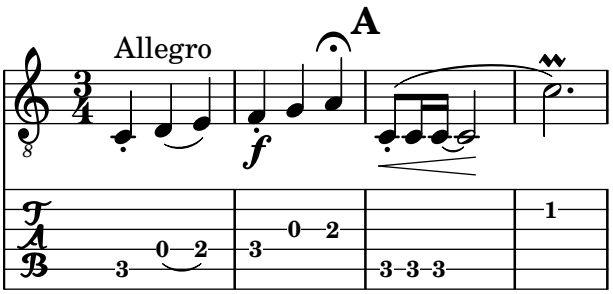
```
\new TabStaff \relative {
  a,8 a' <c e> a
  d,8 a' <d f> a
}
```



Standard-Tabulaturen haben weder Symbole, die Notendauern anzeigen, noch andere musikalische Symbole wie etwa Ausdrucksbezeichnungen.

```
symbols = {
  \time 3/4
  c4-.^"Allegro" d( e)
  f4-.\f g a^\fermata
  \mark \default
  c8_.\<\( c16 c~ 2\!
  c'2.\prall\)
}

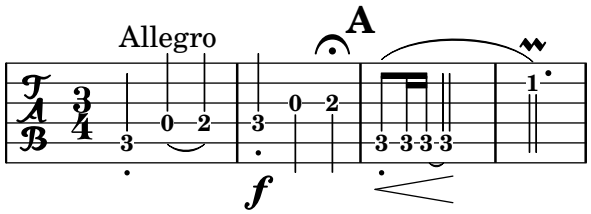
\score {
  <<
    \new Staff { \clef "G_8" \symbols }
    \new TabStaff { \symbols }
  >>
}
```



Wenn alle musikalischen Symbole, die in der traditionellen Notation eingesetzt werden, auch in der Tabulatur gedruckt werden sollen, muss man den Befehl `\tabFullNotation` in einem `TabStaff`-Kontext hinzufügen. Dabei ist zu beachten, dass halbe Noten in einer Tabulatur mit zwei Hälsen dargestellt werden, um sie von Viertelnoten zu unterscheiden.

```
symbols = {
  \time 3/4
  c4-.^"Allegro" d( e)
  f4-.\f g a^\fermata
  \mark \default
  c8_.\<\( c16 c~ 2\!
  c'2.\prall\)
}

\score {
  \new TabStaff {
    \tabFullNotation
    \symbols
  }
}
```

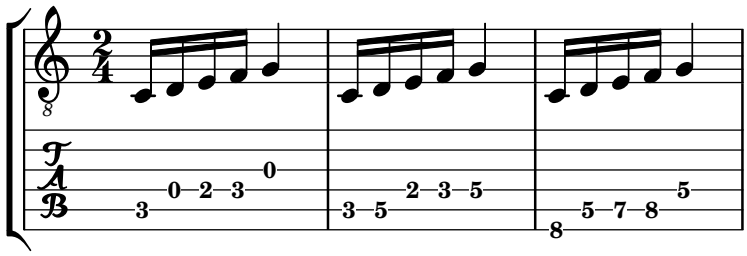


Normalerweise werden Tonhöhen der tiefstmöglichen Spielposition auf dem Bundbrett zugewiesen (erste Lage). Offene Saiten werden automatisch bevorzugt. Wenn man eine bestimmte Tonhöhe auf einer bestimmten Saite gespielt haben will, kann man eine Saitennummeranweisung zur Tonhöhe hinzufügen. Wenn man die Saitenzahlanweisung nicht in der traditionellen Notation sehen will, kann man den entsprechenden Stencil mit `\override` verändern. Es ist jedoch sehr viel bequemer, die Spielposition unter Benutzung von `minimumFret` zu definieren. Der Standardwert von `minimumFret` beträgt 0.

Auch wenn `minimumFret` gesetzt ist, werden offene Saiten immer benützt, wenn es möglich ist. Dieses Verhalten kann verändert werden, indem `restrainOpenStrings` auf `#t` gesetzt wird.

```
\layout { \omit Voice.StringNumber }
\new StaffGroup <<
   \new Staff \relative {
     \clef "treble_8"
     \time 2/4
     c16 d e f g4
     c,16\5 d\5 e\4 f\4 g4\4
     c,16 d e f g4
   }
   \new TabStaff \relative {
     c16 d e f g4
     c,16\5 d\5 e\4 f\4 g4\4
     \set TabStaff.minimumFret = #5
     \set TabStaff.restrainOpenStrings = ##t
     c,16 d e f g4
   }
```

```
>>
```



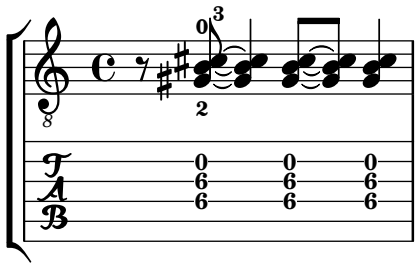
Akkord-Konstruktionen können mit dem Akkord-Wiederholungssymbol q wiederholt werden. In Verbindung mit Tabulaturen verhält sich diese Wiederholung jedoch seltsam, weil sie Saiten- und Fingerzahlen entfernt. Darum sollte man

```
\chordRepeats #'(string-number-event fingering-event)
```

explizit für musikalische Ausdrücke in Tabulaturen aufrufen, wenn Akkordwiederholungen gewünscht sind. Der Befehl ist so wichtig, dass er durch `\tabChordRepeats` zur Verfügung gestellt wird.

```
guitar = \relative {
  r8 <gis-2 cis-3 b-0>~ q4 q8~ 8 q4
}

\new StaffGroup <<
  \new Staff {
    \clef "treble_8"
    \guitar
  }
  \new TabStaff {
    \tabChordRepeats \guitar
  }
>>
```

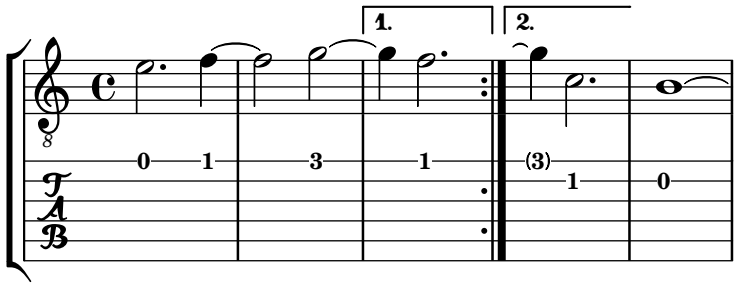


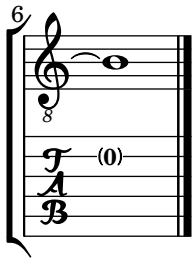
Bindestriche über einen Zeilenumbruch werden standardmäßig in Klammern gesetzt. Das gilt auch für die zweite Klammer einer Wiederholung.

```
ties = \relative {
  \repeat volta 2 {
    e'2. f4~
    2 g2~
  }
  \alternative {
     { g4 f2. }
     { g4\repeatTie c,2. }
  }
  b1~
```

```
  \break
  b1
  \bar "|."
}

\score {
  <<
    \new StaffGroup  <<
      \context Staff {
        \clef "treble_8"
        \ties
      }
      \context TabStaff {
        \ties
      }
    >>
  >>
  \layout {
  indent = #0
  ragged-right = ##t
  }
}
```



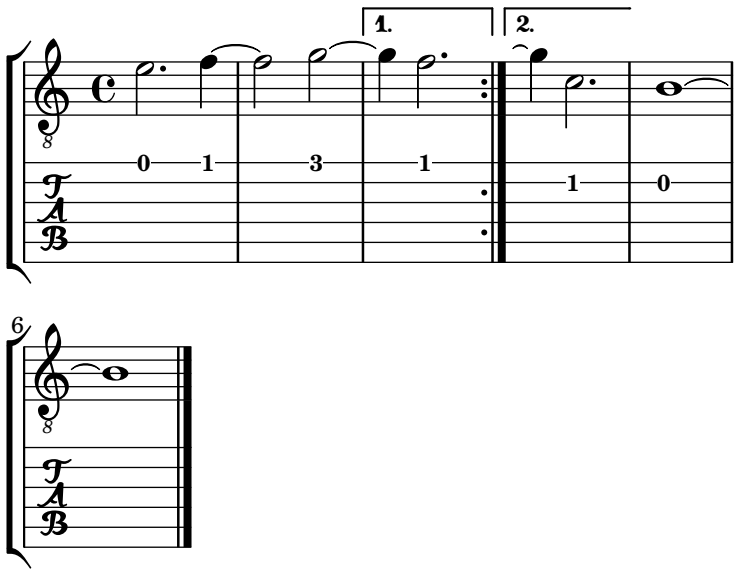


Der Befehl \hideSplitTiedTabNotes hebt das Verhalten auf, dass Bundnummern in Klammern gesetzt werden:

```
ties = \relative {
  \repeat volta 2 {
    e'2. f4~
    2 g2~ }
  \alternative {
    { g4 f2. }
    { g4\repeatTie c,2. }
  }
  b1~
  \break
```

```
  b1
  \bar "|."
}

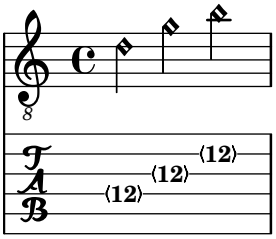
\score {
  <<
    \new StaffGroup  <<
      \context Staff {
        \clef "treble_8"
        \ties
      }
      \context TabStaff {
      \hideSplitTiedTabNotes
        \ties
      }
    >>
  >>
  \layout {
  indent = #0
  ragged-right = ##t
  }
}
```



Flageolett (engl. *harmonic*) kann zur Tabulaturnotation als klingende Tonhöhe hinzugefügt werden:

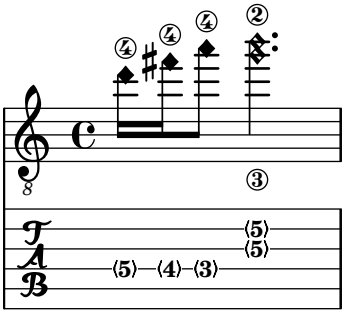
```
\layout { \omit Voice.StringNumber }
firstHarmonic = {
  d'4\4\harmonic
  g'4\3\harmonic
  b'2\2\harmonic
}
\score {
  <<
    \new Staff {
      \clef "treble_8"
```

```
      \firstHarmonic
    }
    \new TabStaff { \firstHarmonic }
  >>
}
```



Dabei ist zu beachten, dass der Befehl \harmonic immer an einzelne Noten angehängt werden muss (die sich auch innerhalb eines Akkordes befinden können). Flageolett ist nur sinnvoll für offene Saiten im 12. Bund. Alle anderen Flageolett-Töne sollten von LilyPond errechnet werden. Das wird erreicht, indem man den Bund angibt, wo der Finger der Greifhand die Saite berühren soll.

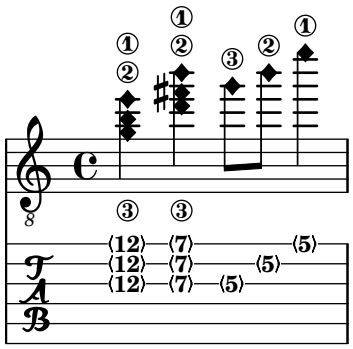
```
fretHarmonics = {
  \harmonicByFret #5 d16\4
  \harmonicByFret #4 d16\4
  \harmonicByFret #3 d8\4
  \harmonicByFret #5 <g\3 b\2>2.
}
\score {
  <<
    \new Staff {
      \clef "treble_8"
      \fretHarmonics
    }
    \new TabStaff { \fretHarmonics }
  >>
}
```



Alternativ können Flageolett-Töne auch errechnet werden, indem man das Verhältnis der Saitenlängen über und unter dem Flageolett-Finger definiert:

```
ratioHarmonics = {
  \harmonicByRatio #1/2 <g\3 b\2 e'\1>4
  \harmonicByRatio #1/3 <g\3 b\2 e'\1>4
  \harmonicByRatio #1/4 { g8\3 b8\2 e'4\1 }
}
\score {
```

```
  <<
    \new Staff {
      \clef "treble_8"
      \ratioHarmonics
    }
    \new TabStaff { \ratioHarmonics }
  >>
}
```

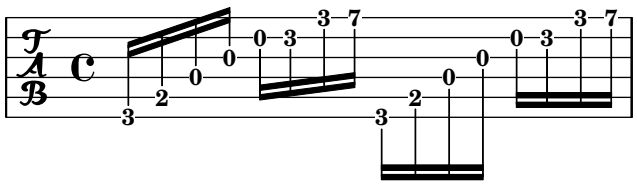


Ausgewählte Schnipsel

Hals- und Balkenverhalten in einer Tabulatur

Die Richtung von Hälsen wird in Tabulaturen genauso wie in normaler Notation eingestellt. Balken können horizontal eingestellt werden, wie das Beispiel zeigt.

```
\new TabStaff {
  \relative c {
    \tabFullNotation
    g16 b d g b d g b
    \stemDown
    \override Beam.concaveness = 10000
    g,,16 b d g b d g b
  }
}
```



Polyphonie in einer Tabulatur

Polyphonie kann in einer Tabulatur (`TabStaff`) genauso wie in einem normalen Notensystem erstellt werden.

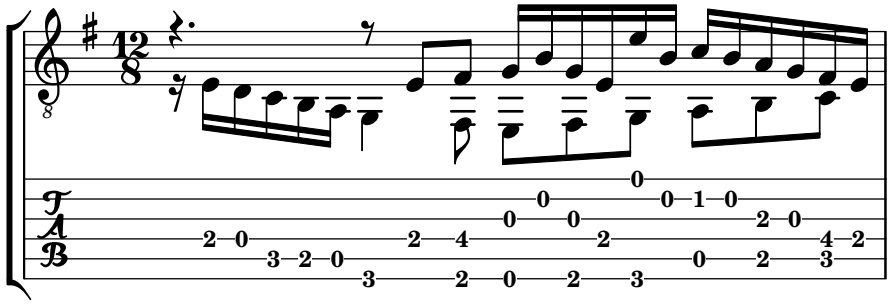
```
upper = \relative c' {
  \time 12/8
  \key e \minor
  \voiceOne
  r4. r8 e, fis g16 b g e e' b c b a g fis e
}

lower = \relative c {
```

```
  \key e \minor
  \voiceTwo
  r16 e d c b a g4 fis8 e fis g a b c
}

\score {
  \new StaffGroup = "tab with traditional" <<
    \new Staff = "guitar traditional" <<
      \clef "treble_8"
      \new Voice = "upper" \upper
      \new Voice = "lower" \lower
    >>

    \new TabStaff = "guitar tab" <<
      \new TabVoice = "upper" \upper
      \new TabVoice = "lower" \lower
    >>
  >>
}
```



Referenz für Flageolett von offenen Saiten

Referenz für Flageolett von offenen Saiten:

```
openStringHarmonics = {
  \textSpannerDown
  \override TextSpanner.staff-padding = 3
  \override TextSpanner.dash-fraction = 0.3
  \override TextSpanner.dash-period = 1

  % first harmonic
  \override TextSpanner.bound-details.left.text =
    \markup\small "1st harm. "
  \harmonicByFret 12 e,2\6\startTextSpan
  \harmonicByRatio #1/2 e,\6\stopTextSpan

  % second harmonic
  \override TextSpanner.bound-details.left.text =
    \markup\small "2nd harm. "
  \harmonicByFret 7 e,\6\startTextSpan
  \harmonicByRatio #1/3 e,\6
  \harmonicByFret 19 e,\6
  \harmonicByRatio #2/3 e,\6\stopTextSpan
  %\harmonicByFret 19 < e,\6 a,\5 d\4 >
```

```
  %\harmonicByRatio #2/3 < e,\6 a,\5 d\4 >

  % third harmonic
  \override TextSpanner.bound-details.left.text =
    \markup\small "3rd harm. "
  \harmonicByFret 5 e,\6\startTextSpan
  \harmonicByRatio #1/4 e,\6
  \harmonicByFret 24 e,\6
  \harmonicByRatio #3/4 e,\6\stopTextSpan
  \break

  % fourth harmonic
  \override TextSpanner.bound-details.left.text =
    \markup\small "4th harm. "
  \harmonicByFret 4 e,\6\startTextSpan
  \harmonicByRatio #1/5 e,\6
  \harmonicByFret 9 e,\6
  \harmonicByRatio #2/5 e,\6
  \harmonicByFret 16 e,\6
  \harmonicByRatio #3/5 e,\6\stopTextSpan

  % fifth harmonic
  \override TextSpanner.bound-details.left.text =
    \markup\small "5th harm. "
  \harmonicByFret 3 e,\6\startTextSpan
  \harmonicByRatio #1/6 e,\6\stopTextSpan
  \break

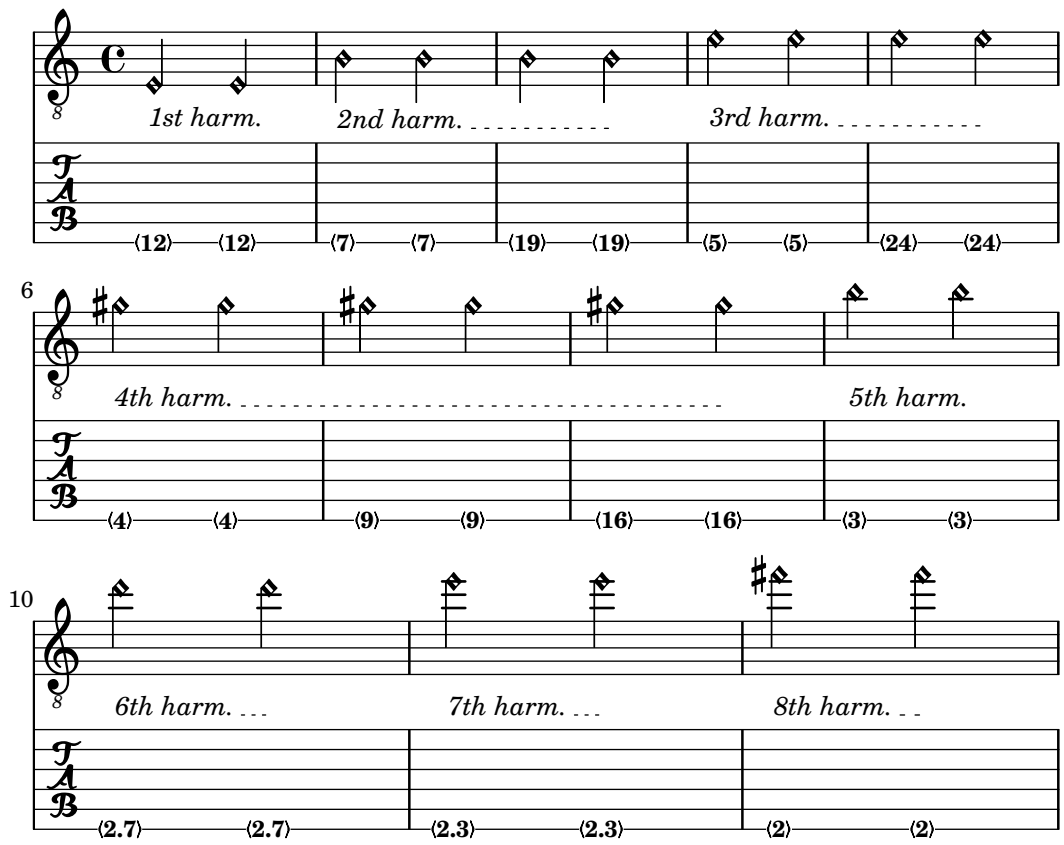
  % sixth harmonic
  \override TextSpanner.bound-details.left.text =
    \markup\small "6th harm. "
  \harmonicByFret 2.7 e,\6\startTextSpan
  \harmonicByRatio #1/7 e,\6\stopTextSpan

  % seventh harmonic
  \override TextSpanner.bound-details.left.text =
    \markup\small "7th harm. "
  \harmonicByFret 2.3 e,\6\startTextSpan
  \harmonicByRatio #1/8 e,\6\stopTextSpan

  % eighth harmonic
  \override TextSpanner.bound-details.left.text =
    \markup\small "8th harm. "
  \harmonicByFret 2 e,\6\startTextSpan
  \harmonicByRatio #1/9 e,\6\stopTextSpan
}

\score {
  <<
    \new Staff \with { \omit StringNumber } {
      \new Voice {
        \clef "treble_8"
```

```
        \openStringHarmonics
      }
    }
    \new TabStaff {
      \new TabVoice {
        \openStringHarmonics
      }
    }
  >>
}
```



Flageolett von Bundinstrumenten in einer Tabulatur

Flageolett für Bundinstrumente:

```
pinchedHarmonics = {
  \textSpannerDown
  \override TextSpanner.bound-details.left.text =
    \markup {\halign #-0.5 \teeny "PH" }
  \override TextSpanner.style = #'dashed-line
  \override TextSpanner.dash-period = 0.6
  \override TextSpanner.bound-details.right.attach-dir = 1
  \override TextSpanner.bound-details.right.text =
    \markup { \draw-line #'(0 . 1) }
  \override TextSpanner.bound-details.right.padding = -0.5
}

harmonics = {
  % artificial harmonics (AH)
```

```
  \textLengthOn
  <\parenthesize b b'\harmonic>4_\markup { \teeny "AH 16" }
  <\parenthesize g g'\harmonic>4_\markup { \teeny "AH 17" }
  <\parenthesize d' d''\harmonic>2_\markup { \teeny "AH 19" }

  % pinched harmonics (PH)
  \pinchedHarmonics
  <a'\harmonic>2\startTextSpan
  <d''\harmonic>4
  <e'\harmonic>4\stopTextSpan

  % tapped harmonics (TH)
  <\parenthesize g\4 g'\harmonic>4_\markup { \teeny "TH 17" }
  <\parenthesize a\4 a'\harmonic>4_\markup { \teeny "TH 19" }
  <\parenthesize c'\3 c''\harmonic>2_\markup { \teeny "TH 17" }

  % touch harmonics (TCH)
  a4( <e''\harmonic>2. )_\markup { \teeny "TCH" }
}

frettedStrings = {
  % artificial harmonics (AH)
  \harmonicByFret 4 g4\3
  \harmonicByFret 5 d4\4
  \harmonicByFret 7 g2\3

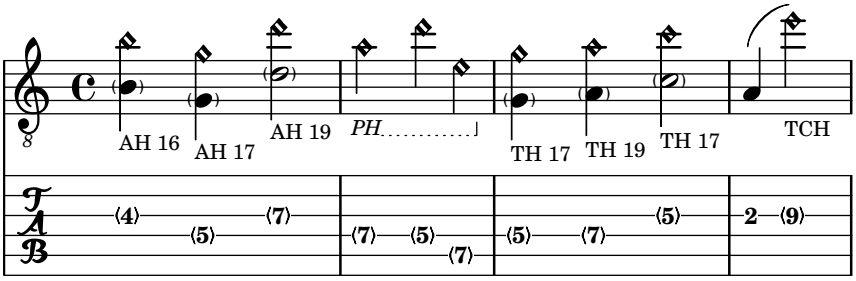
  % pinched harmonics (PH)
  \harmonicByFret 7 d2\4
  \harmonicByFret 5 d4\4
  \harmonicByFret 7 a4\5

  % tapped harmonics (TH)
  \harmonicByFret 5 d4\4
  \harmonicByFret 7 d4\4
  \harmonicByFret 5 g2\3

  % touch harmonics (TCH)
  a4 \harmonicByFret 9 g2.\3
}

\score {
  <<
    \new Staff
    \with { \omit StringNumber } {
      \new Voice {
        \clef "treble_8"
        \harmonics
      }
    }
    \new TabStaff {
      \new TabVoice {
        \frettedStrings
```

```
      }
    }
  >>
}
```



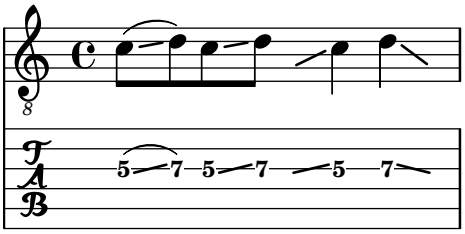
Gleiten (Glissando) in Tabulatur

Gleiten kann sowohl in normalem Notensystem als auch in Tabulaturen notiert werden:

```
slides = {
  c'8\3(\glissando d'8\3)
  c'8\3\glissando d'8\3
  \hideNotes
  \grace { g16\glissando }
  \unHideNotes
  c'4\3
  \afterGrace d'4\3\glissando {
  \stemDown \hideNotes
  g16 }
  \unHideNotes
}

\score {
  <<
    \new Staff { \clef "treble_8" \slides }
    \new TabStaff { \slides }
  >>

  \layout {
    \context {
      \Score
      \override Glissando.minimum-length = 4
      \override Glissando.springs-and-rods =
                  #ly:spanner::set-spacing-rods
      \override Glissando.thickness = 2
      \omit StringNumber
      % or:
      %\override StringNumber.stencil = ##f
    }
  }
}
```



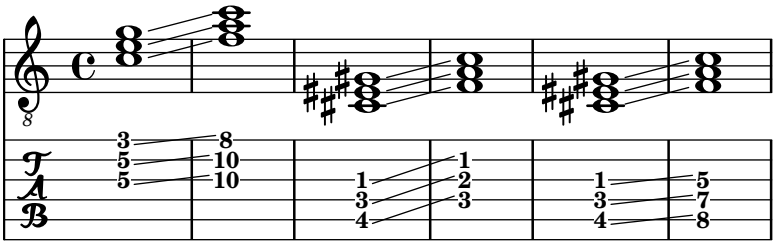
Akkordglissando in Tabulaturen

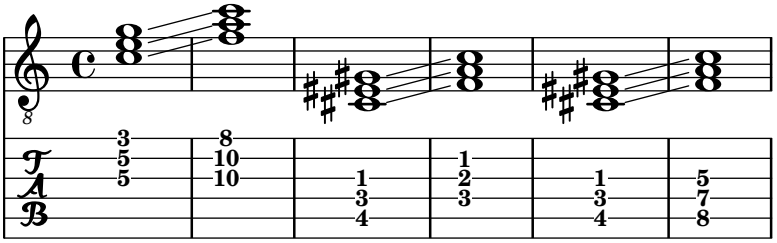
Gleiten von Akkorden kann sowohl im normalen Notensystem als auch in einer Tabulatur notiert werden. Saitennummern werden für Tabulaturen benötigt, weil die automatische Saitenberechnung unterschiedlich für Akkorde und einzelne Noten funktioniert.

```
myMusic = \relative c' {
  <c e g>1 \glissando <f a c>
  <cis, eis gis>1 \glissando <f a c>
  <cis eis gis>1 \glissando <f a c\3>
}

\score {
  <<
    \new Staff {
      \clef "treble_8"
      \omit StringNumber
      \myMusic
    }
    \new TabStaff \myMusic
  >>
}

\score {
  <<
    \new Staff {
      \clef "treble_8"
      \omit StringNumber
      \myMusic
    }
    \new TabStaff \with { \override Glissando.style = #'none } {
      \myMusic
    }
  >>
}
```





Siehe auch

Notationsreferenz: Abschnitt 7.1.6 [Hälse], Seite 220, Abschnitt 5.1.2 [Akkord-Wiederholungen], Seite 162, Abschnitt 4.1.3 [Ausgeschriebene Wiederholungen], Seite 154, Abschnitt 11.1.3 [Flageolett], Seite 325, Abschnitt 3.3.1 [Glissando], Seite 136.

Schnipsel: Abschnitt “Fretted string instruments” in *Schnipsel*.

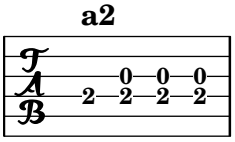
Referenz der Interna: Abschnitt “TabNoteHead” in *Referenz der Interna*, Abschnitt “TabStaff” in *Referenz der Interna*, Abschnitt “TabVoice” in *Referenz der Interna*, Abschnitt “Beam” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Akkorde werden nicht gesondert behandelt, sodass die Saitenauswahlfunktion eventuell die selbe Saite für zwei Töne eines Akkordes auswählen kann.

Damit die Kombination von Stimmen (`\partCombine`) richtig funktioniert, müssen speziell erstellte Stimmen innerhalb des Tabulatursystems (`TabStaff`) benutzt werden:

```
melodia = \partCombine { e4 g g g } { e4 e e e }
<<
  \new TabStaff <<
    \new TabVoice = "one" s1
    \new TabVoice = "two" s1
    \new TabVoice = "shared" s1
    \new TabVoice = "solo" s1
    { \melodia }
  >>
>>
```



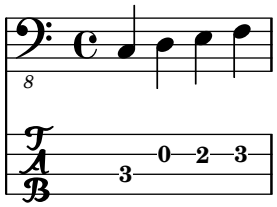
Spezialeffekte für Gitarre beschränken sich auf Flageolett und Slide.

12.1.4 Angepasste Tabulaturen

LilyPond errechnet automatisch den Bund für eine Note auf Grundlage der Saite, zu welcher der Ton zugeordnet ist. Um das tun zu können, muss die Stimmung der Saiten angegeben werden. Die Stimmung wird in der `StringTunings`-Eigenschaften bestimmt.

LilyPond hat vordefinierte Stimmungen für Banjo, Mandoline, Gitarre, Bassgitarre, Ukulele, Geige, Bratsche, Cello und Kontrabass. Für diese Stimmungen wird automatisch die richtige Transposition eingesetzt. Das nächste Beispiel ist für Bassgitarre, welche eine Oktave niedriger erklingt, als sie geschrieben ist:

```
<<
  \new Voice \with {
    \omit StringNumber
  } {
    \clef "bass_8"
    \relative {
      c,4 d e f
    }
  }
  \new TabStaff \with {
    stringTunings = #bass-tuning
  } {
    \relative {
      c,4 d e f
    }
  }
>>
```



Die Standardstimmung ist die Gitarrenstimmung (guitar-tuning) in der EADGHE-Stimmung. Andere vordefinierte Stimmung sind: guitar-open-g-tuning, mandolin-tuning und banjo-open-g-tuning. Die vordefinierten Stimmungen finden sich in ly/string-tunings-init.ly.

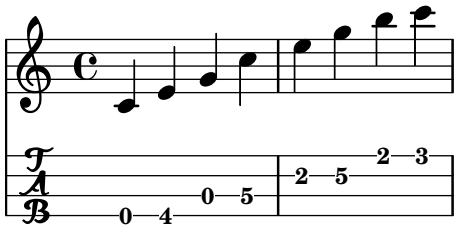
Jede beliebige Stimmung kann erstellt werden. Die Funktion \stringTuning kann benutzt werden, um eine Saitenstimmung zu definieren und als den Wert von stringTunings für den aktuellen Kontext zu bestimmen.

Als Argument braucht die Funktion eine Akkordkonstruktion, die die Tonhöhen jeder Saite der Stimmung angibt. Die Akkordkonstruktion muss im absoluten Oktavenmodus angegeben werden, siehe Abschnitt 1.1.1 [Absolute Oktavenbezeichnung], Seite 3. Die Saite mit der höchsten Zahl (normalerweise die tiefste Seite) muss im Akkord zuerst geschrieben werden. Eine Stimmung für ein viersaitiges Instrument mit den Tonhöhen a'', d'', g' und c' kann folgenderweise erstellt werden:

```
mynotes = {
  c'4 e' g' c'' |
  e''4 g'' b'' c'''
}

<<
  \new Staff {
    \clef treble
    \mynotes
  }
  \new TabStaff {
    \set Staff.stringTunings = \stringTuning <c' g' d'' a''>
    \mynotes
  }
```

```
>>
```



Die stringTunings-Eigenschaft wird auch von FretBoards benutzt, um automatische Bunddiagramme zu errechnen.

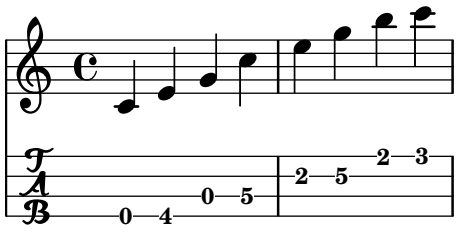
Saitensitmmungen werden als Teil des Hash-Schlüsselwertes für vordefinierte Bunddiagramme eingesetzt (siehe auch Abschnitt 12.1.6 [Vordefinierte Bund-Diagramme], Seite 355.

Das vorherige Beispiel könnte auch folgenderweise geschrieben werden:

```
custom-tuning = \stringTuning <c' g' d'' a''>

mynotes = {
  c'4 e' g' c'' |
  e''4 g'' b'' c'''
}

<<
  \new Staff {
    \clef treble
    \mynotes
  }
  \new TabStaff {
    \set TabStaff.stringTunings = #custom-tuning
    \mynotes
  }
>>
```



Intern ist die Stimmung eine Scheme-Liste von Tonhöhen der Saiten, eine für jede Saite, geordnet von Saitennummer 1 bis n, wobei 1 die höchste Saite der Tabulatur ist und n die unterste. Normalerweise wird so die Stimmung vom höchsten bis zum tiefsten Ton angegeben, aber bei einige Instrumente (etwa Ukulele) werden die Saiten nicht aufgrund der Tonhöhe angeordnet.

Die Tonhöhe einer Saite in einer Seitenstimmungsliste ist ein Tonhöhenobjekt für LilyPond. Tonhöhenobjekte werden mit der Scheme-Funktion `+ly:make-pitch` erstellt (siehe Abschnitt "Scheme-Funktionen" in *Referenz der Interna*).

`\stringTuning` erstellt derartige Objekte aus der Akkord-Eingabe.

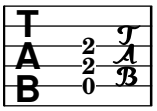
LilyPond errechnet automatisch die Linienanzahl für die Tabulatur und die Zahl der Saiten in dem automatisch erstellten `FretBoard` (Bunddiagramm) aus der Anzahl der Elemente von `stringTunings`.

Um für alle TabStaff-Kontexte die selbe Standardstimmung zu benutzen, kann man benutzen:

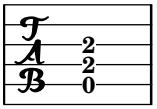
```
\layout {
  \context {
    \TabStaff
    stringTunings = \stringTuning <c' g' d'' a''>
  }
}
```

Auch ein moderner Tabulatur-Schlüssel kann verwendet werden:

```
\new TabStaff {
  \clef moderntab
  <a, e a>1
  \break
  \clef tab
  <a, e a>1
}
```



2



Der moderne Tabulatur-Schlüssel unterstützt Tabulaturen von 4 bis 7 Saiten.

Siehe auch

Notationsreferenz: Abschnitt 1.1.1 [Absolute Oktavenbezeichnung], Seite 3, Abschnitt 12.1.6 [Vordefinierte Bund-Diagramme], Seite 355.

Installierte Dateien: `ly/string-tunings-init.ly` `scm/tablature.scm`.

Schnipsel: Abschnitt “Fretted string instruments” in *Schnipsel*.

Referenz der Interna: Abschnitt “Scheme-Funktionen” in *Referenz der Interna*, Abschnitt “Tab_note_heads_engraver” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

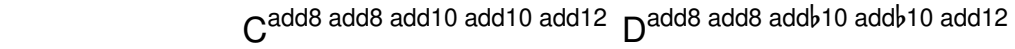
Automatische Tabulatur-Berechnung funktioniert in den meisten Fällen nicht korrekt bei Instrumenten, deren Saitenstimmung nicht monotonisch fortschreitet, wie etwa Ukulele.

12.1.5 Bund-Diagramm-Beschriftung

Bunddiagramme können zu Notation als Textbeschriftung hinzugefügt werden. Die Beschriftung enthält Information zu dem gewünschten Bunddiagramm. Es gibt drei unterschiedliche Darstellungsarten: normal, knapp und ausführlich. Die drei Arten erzeugen die gleiche Ausgabe, aber mit jeweils mehr oder weniger Einzelheiten. Einzelheiten zur Syntax der unterschiedlichen Beschriftungsbefehle, mit denen die Bunddiagramme definiert werden, findet sich in Abschnitt A.10.6 [Instrument-specific markup], Seite 734.

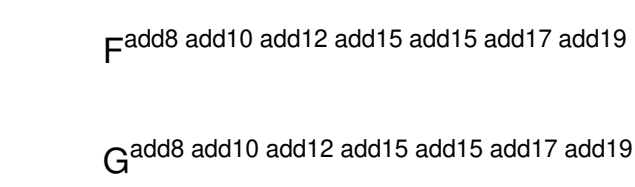
Die Standard-Bunddiagrammbeschriftung beinhaltet die Saitennummer und die Bundnummer für jeden Punkt, der notiert werden soll. Zusätzlich können offenen und nicht gespielte (schwingende) Saiten angezeigt werden.

```
<<
  \context ChordNames {
    \chordmode {
      c1 d:m
    }
  }
  \context Staff {
    \clef "treble_8"
    <c e g c' e'>1^\markup {
      \fret-diagram "6-x;5-3;4-2;3-o;2-1;1-o;"
    }
    <d a d' f'>1^\markup {
      \fret-diagram "6-x;5-x;4-o;3-2;2-3;1-1;"
    }
  }
>>
```



Barré kann hinzugefügt werden:

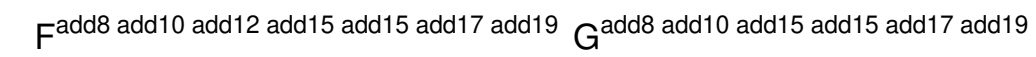
```
<<
  \context ChordNames {
     \chordmode {
       f1 g
     }
  }
  \context Staff {
    \clef "treble_8"
    <f, c f a c' f'>1^\markup {
      \fret-diagram "c:6-1-1;6-1;5-3;4-3;3-2;2-1;1-1;"
    }
    <g, d g b d' g'>1^\markup {
      \fret-diagram "c:6-1-3;6-3;5-5;4-5;3-4;2-3;1-3;"
    }
  }
>>
```



Die Größe des Bunddiagrammes und die Anzahl der Bünde im Diagramm kann geändert werden:

```
<<
  \context ChordNames {
     \chordmode {
       f1 g
     }
  }
  \context Staff {
    \clef "treble_8"
```

```
    <f, c f a c' f'>1^\markup {
      \fret-diagram "s:1.5;c:6-1-1;6-1;5-3;4-3;3-2;2-1;1-1;"
    }
    <g, b, d g b g'>1^\markup {
      \fret-diagram "h:6;6-3;5-2;4-o;3-o;2-o;1-3;"
    }
  }
>>
```



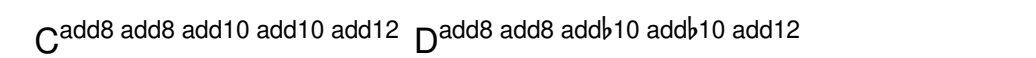
Die Anzahl der Saiten in einem Bunddiagramm kann geändert werden, um sie für andere Instrumente anzupassen, wie etwas Banjo oder Ukulele.

```
<<
  \context ChordNames {
    \chordmode {
      a1
    }
  }
  \context Staff {
    % An 'A' chord for ukulele
    a'1^\markup {
      \fret-diagram "w:4;4-2-2;3-1-1;2-o;1-o;"
    }
  }
>>
```



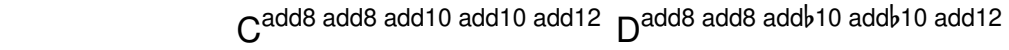
Fingersatz kann auch angezeigt werden, und die Position der Fingersatzzahlen kann kontrolliert werden.

```
<<
  \context ChordNames {
    \chordmode {
      c1 d:m
    }
  }
  \context Staff {
    \clef "treble_8"
    <c e g c' e'>1^\markup {
      \fret-diagram "f:1;6-x;5-3-3;4-2-2;3-o;2-1-1;1-o;"
    }
    <d a d' f'>1^\markup {
      \fret-diagram "f:2;6-x;5-x;4-o;3-2-2;2-3-3;1-1-1;"
    }
  }
>>
```



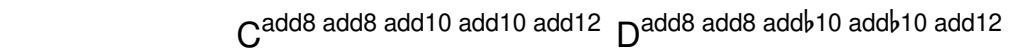
Die Größe und Position der Punkte kann geändert werden:

```
<<
  \context ChordNames {
    \chordmode {
      c1 d:m
    }
  }
  \context Staff {
    \clef "treble_8"
    <c e g c' e'>1^\markup {
      \fret-diagram "d:0.35;6-x;5-3;4-2;3-o;2-1;1-o;"
    }
    <d a d' f'>1^\markup {
      \fret-diagram "p:0.2;6-x;5-x;4-o;3-2;2-3;1-1;"
    }
  }
>>
```



Die Beschriftungsfunktion `fret-diagram-terse` (knappe Version) lässt die Saitennummern aus: das Vorhandensein einer Saite wird durch ein Semikolon ausgedrückt. Für jede Saite des Diagramms muss ein Semikolon gesetzt werden. Das erste Semikolon entspricht der höchsten Saite, das letze der ersten Saite. Stumme und offene Saiten sowie Bundnummern können angezeigt werden.

```
<<
  \context ChordNames {
    \chordmode {
      c1 d:m
    }
  }
  \context Staff {
    \clef "treble_8"
    <c e g c' e'>1^\markup {
      \fret-diagram-terse "x;3;2;o;1;o;"
    }
    <d a d' f'>1^\markup {
      \fret-diagram-terse "x;x;o;2;3;1;"
    }
  }
>>
```



Barré kann im knappen Modus auch angezeigt werden:

```
<<
  \context ChordNames {
    \chordmode {
      f1 g
    }
  }
  \context Staff {
    \clef "treble_8"
```

```
    <f, c f a c' f'>1^\markup {
      \fret-diagram-terse "1-(;3;3;2;1;1-);"
    }
    <g, d g b d' g'>1^\markup {
      \fret-diagram-terse "3-(;5;5;4;3;3-);"
    }
  }
>>
```

F add8 add10 add12 add15 add15 add17 add19

2

G add8 add10 add12 add15 add15 add17 add19

Fingersatz kann im knappen Modus hinzugefügt werden:

```
<<
  \context ChordNames {
    \chordmode {
      c1 d:m
    }
  }
  \context Staff {
    \override Voice.TextScript.fret-diagram-details.finger-code = #'below-string
    \clef "treble_8"
    <c e g c' e'>1^\markup {
      \fret-diagram-terse "x;3-3;2-2;o;1-1;o;"
    }
    <d a d' f'>1^\markup {
      \fret-diagram-terse "x;x;o;2-2;3-3;1-1;"
    }
  }
>>
```



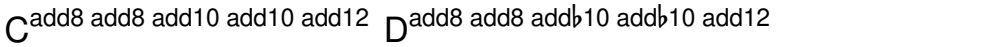
C add8 add8 add10 add10 add12 D add8 add8 add♭10 add♭10 add12

Andere Eigenschaften der Bunddiagramme müssen im knappen Modus mit `\override`-Befehlen angegeben werden.

Die Beschriftungsfunktion `fret-diagram-verbose` (ausführlicher Stil) ist in der Form eine Scheme-Liste. Jedes Element stellt ein Element dar, dass im Bunddiagramm gesetzt werden soll.

```
<<
    \context ChordNames {
      \chordmode {
        c1 d:m
      }
    }
  \context Staff {
    \clef "treble_8"
    <c e g c' e'>1^\markup {
      \fret-diagram-verbose #'(
        (mute 6)
```

```
        (place-fret 5 3)
        (place-fret 4 2)
        (open 3)
        (place-fret 2 1)
        (open 1)
      )
    }
    <d a d' f'>1^\markup {
      \fret-diagram-verbose #'(
        (mute 6)
        (mute 5)
        (open 4)
        (place-fret 3 2)
        (place-fret 2 3)
        (place-fret 1 1)
      )
    }
  }
>>
```



Fingersatz und Barré kann im ausführlichen Modus notiert werden. Nur im ausführlichen Modus kann ein Capo angezeigt werden, das auf dem Bunddiagramm plaziert wird. Die Capo-Anzeige ist ein dicker Strich, der alle Saiten bedeckt. Der Bund mit dem Capo ist der unterste Bund im Diagramm.

```
<<
    \context ChordNames {
      \chordmode {
        f1 g c
      }
    }
  \context Staff {
    \clef "treble_8"
    \override Voice.TextScript.fret-diagram-details.finger-code = #'below-string
    <f, c f a c' f'>1^\markup {
      \fret-diagram-verbose #'(
        (place-fret 6 1)
        (place-fret 5 3)
        (place-fret 4 3)
        (place-fret 3 2)
        (place-fret 2 1)
        (place-fret 1 1)
        (barre 6 1 1)
      )
    }
    <g, b, d g b g'>1^\markup {
      \fret-diagram-verbose #'(
        (place-fret 6 3 2)
        (place-fret 5 2 1)
        (open 4)
        (open 3)
```

```
        (open 2)
        (place-fret 1 3 3)
      )
    }
    <c e g c' e'>1^\markup {
      \fret-diagram-verbose #'(
        (capo 3)
        (mute 6)
        (place-fret 4 5 1)
        (place-fret 3 5 2)
        (place-fret 2 5 3)
      )
    }
  }
>>
```



F add8 add10 add12 add15 add15 add17 add19

2

G add8 add10 add15 add15 add17 add19 C add8 add8 add10 add10 add12

Alle anderen Bunddiagramm-Eigenschaften müssen im ausführlichen Modus mit mit `\override`-Befehlen angegeben werden.

Die graphische Erscheinung eines Bunddiagramms kann den Wünschen des Notensetzers angepasst werden. Hierzu werden die Eigenschaften des `fret-diagram-interface` (Bunddiagramm-Schnittstelle) eingesetzt. Einzelheiten hierzu in Abschnitt “fret-diagram-interface” in *Referenz der Interna*. Die Eigenschaften der Schnittstelle gehören dem `Voice.TextScript`-Kontext an.

Ausgewählte Schnipsel

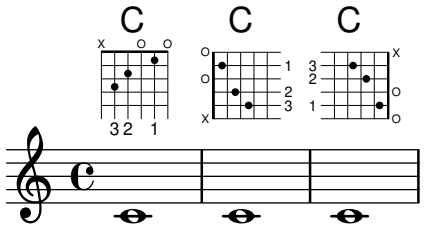
Changing fret orientations

Fret diagrams can be oriented in three ways. By default the top string or fret in the different orientations will be aligned.

```
\include "predefined-guitar-fretboards.ly"

<<
  \chords {
    c1
    c1
    c1
  }
  \new FretBoards \chordmode {
    c1
    \override FretBoard.fret-diagram-details.orientation =
      #'landscape
    c1
    \override FretBoard.fret-diagram-details.orientation =
      #'opposing-landscape
    c1
  }
```

```
  \new Voice {
    c'1
    c'1
    c'
  }
>>
```



Anpassung von Beschriftungs-Bunddiagrammen

Bunddiagramme können mit der Eigenschaft `'fret-diagram-details` angepasst werden. Bunddiagramme, die als Textbeschriftung eingefügt werden, können Veränderungen im `Voice.TextScript`-Objekt oder direkt in der Beschriftung vorgenommen werden.

```
<<
  \chords { c1 | c | c | d }

  \new Voice = "mel" {
    \textLengthOn
    % Set global properties of fret diagram
    \override TextScript.size = 1.2
    \override TextScript.fret-diagram-details.finger-code = #'in-dot
    \override TextScript.fret-diagram-details.dot-color = #'white

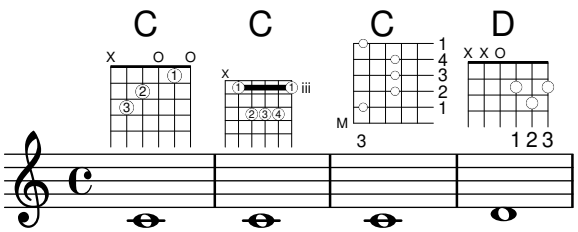
    %% C major for guitar, no barre, using defaults
       % terse style
    c'1^\markup { \fret-diagram-terse "x;3-3;2-2;o;1-1;o;" }

    %% C major for guitar, barred on third fret
       % verbose style
       % size 1.0
       % roman fret label, finger labels below string, straight barre
    c'1^\markup {
      % standard size
      \override #'(size . 1.0) {
        \override #'(fret-diagram-details . (
                     (number-type . roman-lower)
                     (finger-code . in-dot)
                     (barre-type . straight))) {
          \fret-diagram-verbose #'((mute 6)
                                   (place-fret 5 3 1)
                                   (place-fret 4 5 2)
                                   (place-fret 3 5 3)
                                   (place-fret 2 5 4)
                                   (place-fret 1 3 1)
                                   (barre 5 1 3))
        }
```

```
      }
    }

    %% C major for guitar, barred on third fret
       % verbose style
       % landscape orientation, arabic numbers, M for mute string
       % no barre, fret label down or left, small mute label font
    c'1^\markup {
      \override #'(fret-diagram-details . (
                   (finger-code . below-string)
                   (number-type . arabic)
                   (label-dir . -1)
                   (mute-string . "M")
                   (orientation . landscape)
                   (barre-type . none)
                   (xo-font-magnification . 0.4)
                   (xo-padding . 0.3))) {
        \fret-diagram-verbose #'((mute 6)
                                 (place-fret 5 3 1)
                                 (place-fret 4 5 2)
                                 (place-fret 3 5 3)
                                 (place-fret 2 5 4)
                                 (place-fret 1 3 1)
                                 (barre 5 1 3))
      }
    }

    %% simple D chord
       % terse style
       % larger dots, centered dots, fewer frets
       % label below string
    d'1^\markup {
      \override #'(fret-diagram-details . (
                   (finger-code . below-string)
                   (dot-radius . 0.35)
                   (dot-position . 0.5)
                   (fret-count . 3))) {
        \fret-diagram-terse "x;x;o;2-1;3-2;2-3;"
      }
    }
  }
>>
```



Siehe auch

Notationsreferenz: Abschnitt A.10.6 [Instrument-specific markup], Seite 734.

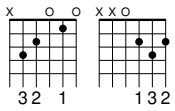
Schnipsel: Abschnitt “Fretted string instruments” in *Schnipsel*.

Referenz der Interna: Abschnitt “fret-diagram-interface” in *Referenz der Interna*.

12.1.6 Vordefinierte Bund-Diagramme

Bunddiagramme können mit dem `FretBoards`-Kontext angezeigt werden. Standardmäßig zeigt der `FretBoards`-Kontext Bunddiagramme an, die in einer Tabelle definiert sind:

```
\include "predefined-guitar-fretboards.ly"
\context FretBoards {
  \chordmode {
    c1 d
  }
}
```



Die vordefinierten Diagramme sind in der Datei `predefined-guitar-fretboards.ly` enthalten. Sie werden basierend auf der Tonhöhe eines Akkordes und dem Wert von `stringTunings` (Saitenstimmung), der gerade benutzt wird, gespeichert. `predefined-guitar-fretboards.ly` beinhaltet vordefinierte Diagramme für die Gitarrenstimmung (`guitar-tuning`). Anhand der Beispiele in dieser Datei können auch für andere Instrumente oder Stimmungen Diagramme definiert werden.

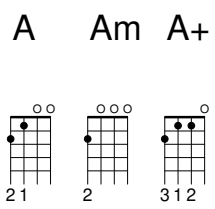
Bunddiagramme für die Ukulele finden sich in der Datei `predefined-ukulele-fretboards.ly`.

```
\include "predefined-ukulele-fretboards.ly"

myChords = \chordmode { a1 a:m a:aug }

\new ChordNames {
  \myChords
}

\new FretBoards {
  \set Staff.stringTunings = #ukulele-tuning
  \myChords
}
```



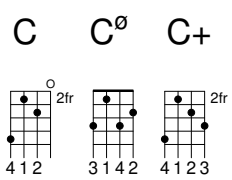
Bunddiagramme für Mandoline sind in der Datei `predefined-mandolin-fretboards.ly` enhalten:

```
\include "predefined-mandolin-fretboards.ly"

myChords = \chordmode { c1 c:m7.5- c:aug }

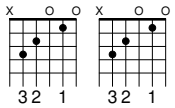
\new ChordNames {
  \myChords
}

\new FretBoards {
  \set Staff.stringTunings = #mandolin-tuning
  \myChords
}
```



Tonhöhen von Akkorden können entweder als Akkordkonstrukte oder im Akkordmodus notiert werden (siehe auch Abschnitt 15.1.1 [Überblick über den Akkord-Modus], Seite 397).

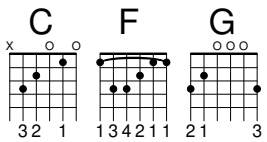
```
\include "predefined-guitar-fretboards.ly"
\context FretBoards {
  \chordmode { c1 }
  <c' e' g'>1
}
```



Oft wird sowohl eine Akkordbezeichnung als ein Bunddiagramm notiert. Das kann erreicht werden, indem ein `ChordNames`-Kontext parallel mit einem `FretBoards`-Kontext gesetzt wird und beiden Kontexten die gleichen Noten zugewiesen werden.

```
\include "predefined-guitar-fretboards.ly"
mychords = \chordmode{
  c1 f g
}

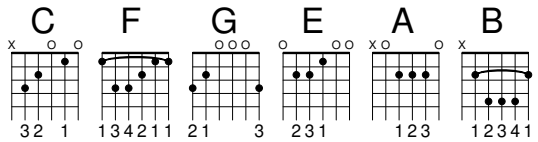
<<
  \context ChordNames {
    \mychords
  }
  \context FretBoards {
    \mychords
  }
>>
```



Vordefinierte Bunddiagramme können transponiert werden, solange ein Diagramm für den transponierten Akkord in der Bunddiagramm-Tabelle vorhanden ist.

```
\include "predefined-guitar-fretboards.ly"
mychords = \chordmode{
  c1 f g
}

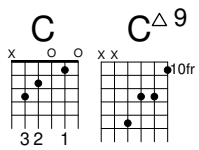
mychordlist = {
  \mychords
  \transpose c e { \mychords }
}
<<
  \context ChordNames {
    \mychordlist
  }
  \context FretBoards {
    \mychordlist
  }
>>
```



Die Tabelle der vordefinierten Bunddiagramme für Gitarre enthält acht Akkorde (Dur, Moll, übermäßig, vermindert, Dominantseptakkord, große Septime, kleine Septime und Dominantnonenakkord) für alle 17 Tonarten. Die Tabelle der vordefinierten Bunddiagramme für Ukulele enthält neben diesen Akkorden noch zusätzlich drei weitere (große Sext, Sekundakkord und Quartakkort). Eine vollständige Liste der vordefinierten Bunddiagramme findet sich in Abschnitt A.4 [Die vordefinierten Bund-Diagramme], Seite 647. Wenn in der Tabelle für einen Akkord kein Wert steht, wird ein Bunddiagramm vom `FretBoards`-Engraver errechnet, wobei die automatische Bunddiagrammfunktion zu Anwendung kommt. Siehe hierzu Abschnitt 12.1.7 [Automatische Bund-Diagramme], Seite 365.

```
\include "predefined-guitar-fretboards.ly"
mychords = \chordmode{
  c1 c:maj9
}

<<
  \context ChordNames {
    \mychords
  }
  \context FretBoards {
    \mychords
  }
>>
```



Bunddiagramme können zu der Tabelle hinzugefügt werden. Um ein Diagramm hinzuzufügen, muss der Akkord des Diagramms, die Stimmung und die Diagramm-Definition angegeben werden.

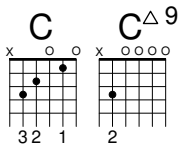
Dies geschieht normalerweise in der Tabelle *default-fret-table*. Die Diagramm-Definition kann entweder eine `fret-diagram-terse`-Definition oder eine `fret-diagram-verbose`-Liste sein.

```
\include "predefined-guitar-fretboards.ly"

\storePredefinedDiagram #default-fret-table
                        \chordmode { c:maj9 }
                        #guitar-tuning
                        "x;3-2;o;o;o;o;"

mychords = \chordmode {
  c1 c:maj9
}

<<
  \context ChordNames {
    \mychords
  }
  \context FretBoards {
    \mychords
  }
>>
```



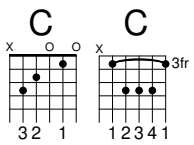
Unterschiedliche Bunddiagramme für den selben Akkord können gespeichert werden, indem unterschiedliche Oktaven für die Tonhöhe benutzt werden. Die unterschiedliche Oktave sollte mindestens zwei Oktaven über oder unter der Standardoktave liegen, die für transponierende Bunddiagramme eingesetzt wird.

```
\include "predefined-guitar-fretboards.ly"

\storePredefinedDiagram #default-fret-table
                        \chordmode { c'' }
                        #guitar-tuning
                        #(offset-fret 2 (chord-shape 'bes guitar-tuning))

mychords = \chordmode {
  c1 c''
}

<<
  \context ChordNames {
    \mychords
  }
  \context FretBoards {
    \mychords
  }
>>
```



Zusätzlich zu Bunddiagrammen speichert LilyPond auch eine interne Liste an Akkordformen. Die Akkordformen sind Bunddiagramme, die am Hals entlang verschoben werden können und dabei unterschiedliche Akkorde ergeben. Akkordformen können zu der internen Liste hinzugefügt werden und dann benutzt werden, um vordefinierte Bunddiagramme zu definieren. Weil sie auf verschiedenen Positionen auf dem Steg gelegt werden können, beinhalten vordefinierte Akkord üblicherweise keine leeren Saiten. Wie Bunddiagramme können auch Akkordformen entweder als `fret-diagram-terse`-Definition oder als `fret-diagram-verbose`-Liste erstellt werden.

```
\include "predefined-guitar-fretboards.ly"

% Add a new chord shape

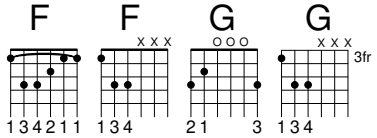
\addChordShape #'powerf #guitar-tuning "1-1;3-3;3-4;x;x;x;"

% add some new chords based on the power chord shape

\storePredefinedDiagram #default-fret-table
                        \chordmode { f'' }
                        #guitar-tuning
                        #(chord-shape 'powerf guitar-tuning)
\storePredefinedDiagram #default-fret-table
                        \chordmode { g'' }
                        #guitar-tuning
                        #(offset-fret 2 (chord-shape 'powerf guitar-tuning))

mychords = \chordmode{
  f1 f'' g g''
}

<<
  \context ChordNames {
    \mychords
  }
  \context FretBoards {
    \mychords
  }
>>
```



Die graphische Form eines Bunddiagramms kann entsprechend den eigenen Wünschen verändert werden, indem man die Eigenschaften der `fret-diagram-interface`-Schnittstelle verändert. Einzelheiten hierzu in Abschnitt “fret-diagram-interface” in *Referenz der Interna*. Die Schnittstelleneigenschaften eines vordefinierten Bunddiagrammes gehören dem `FretBoards.FretBoard`-Kontext an.

Ausgewählte Schnipsel

Bunddiagramme anpassen

Eigenschaften von Bunddiagrammen können in `'fret-diagram-details` verändert werden. Einstellungen mit dem `\override`-Befehl werden dem `FretBoards.FretBoard`-Objekt zugewiesen. Genauso wie `Voice` ist auch `FretBoards` ein Kontext der niedrigsten Ebene, weshalb der Kontext auch in dem Befehl weggelassen werden kann.

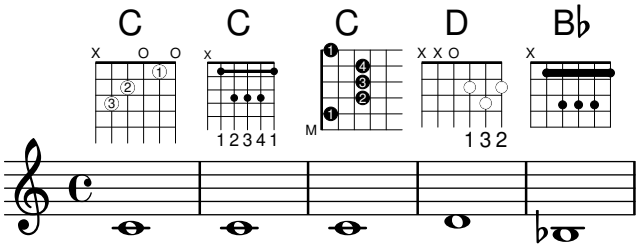
```
\include "predefined-guitar-fretboards.ly"

\storePredefinedDiagram #default-fret-table \chordmode { c' }
                        #guitar-tuning
                        "x;1-1-(;3-2;3-3;3-4;1-1-);"

% shorthand
oo = #(define-music-function
       (grob-path value)
       (list? scheme?)
       #{ \once \override $grob-path = #value #})

<<
  \new ChordNames {
    \chordmode { c1 | c | c | d | bes }
  }
  \new FretBoards {
    % Set global properties of fret diagram
    \override FretBoards.FretBoard.size = 1.2
    \override FretBoard.fret-diagram-details.finger-code = #'in-dot
    \override FretBoard.fret-diagram-details.dot-color = #'white
    \chordmode {
      c
      \oo FretBoard.size #1.0
      \oo FretBoard.fret-diagram-details.barre-type #'straight
      \oo FretBoard.fret-diagram-details.dot-color #'black
      \oo FretBoard.fret-diagram-details.finger-code #'below-string
      c'
      \oo FretBoard.fret-diagram-details.barre-type #'none
      \oo FretBoard.fret-diagram-details.number-type #'arabic
      \oo FretBoard.fret-diagram-details.orientation #'landscape
      \oo FretBoard.fret-diagram-details.mute-string "M"
      \oo FretBoard.fret-diagram-details.label-dir #LEFT
      \oo FretBoard.fret-diagram-details.dot-color #'black
      c'
      \oo FretBoard.fret-diagram-details.finger-code #'below-string
      \oo FretBoard.fret-diagram-details.dot-radius #0.35
      \oo FretBoard.fret-diagram-details.dot-position #0.5
      \oo FretBoard.fret-diagram-details.fret-count #3
      d
      \oo FretBoard.fret-diagram-details.barre-type #'straight
      \oo FretBoard.fret-diagram-details.finger-code #'none
      \oo FretBoard.fret-diagram-details.dot-radius #0.25
      \oo FretBoard.fret-diagram-details.dot-color #'black
```

```
      \oo FretBoard.fret-diagram-details.string-overhang #0.
      \oo FretBoard.fret-diagram-details.barre-thickness #2.
      bes
    }
  }
  \new Voice {
    c'1 | c' | c' | d' | bes
  }
>>
```



Setting up predefined fretboards for other instruments

Predefined fret diagrams can be added for new instruments in addition to the standard diagrams used for guitar. This file shows how this is done by defining a new string tuning and a few predefined fretboards for the Venezuelan *cuatro*.

This file also shows how fingerings can be included in the chords used as reference points for the chord lookup, and displayed in the fret diagram and the TabStaff, but not the music.

These fretboards are not transposable because they contain string information. This is planned to be corrected in the future.

```
% Add fretboards for the cuatro.
%
%   Note: This section could be put into a separate file
%         `predefined-cuatro-fretboards.ly`
%         and be \included into each of your compositions.

cuatroTuning = #`(,(ly:make-pitch 0 6 0)
                  ,(ly:make-pitch 1 3 SHARP)
                  ,(ly:make-pitch 1 1 0)
                  ,(ly:make-pitch 0 5 0))

dSix = { <a\4 b\1 d\3 fis\2> }
dMajor = { <a\4 d\1 d\3 fis \2> }
aMajSeven = { <a\4 cis\1 e\3 g\2> }
dMajSeven = { <a\4 c\1 d\3 fis\2> }
gMajor = { <b\4 b\1 d\3 g\2> }

\storePredefinedDiagram #default-fret-table \dSix
                        #cuatroTuning
                        "o;o;o;o;"
\storePredefinedDiagram #default-fret-table \dMajor
                        #cuatroTuning
                        "o;o;o;3-3;"
\storePredefinedDiagram #default-fret-table \aMajSeven
                        #cuatroTuning
```

```
                        "o;2-2;1-1;2-3;"
\storePredefinedDiagram #default-fret-table \dMajSeven
                        #cuatroTuning
                        "o;o;o;1-1;"
\storePredefinedDiagram #default-fret-table \gMajor
                        #cuatroTuning
                        "2-2;o;1-1;o;"

% End of potential include file `predefined-cuatro-fretboards.ly`.


#(set-global-staff-size 16)

primerosNames = \chordmode {
  d:6 d a:maj7 d:maj7
  g
}
primeros = {
  \dSix \dMajor \aMajSeven \dMajSeven
  \gMajor
}

\score {
  <<
    \new ChordNames {
      \set chordChanges = ##t
      \primerosNames
    }

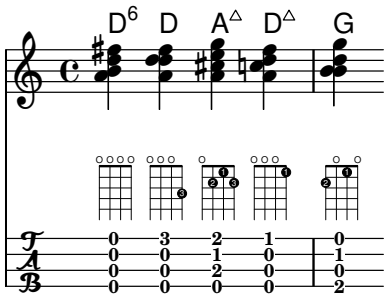
    \new Staff {
      \new Voice \with {
        \remove "New_fingering_engraver"
      }
      \relative c'' {
        \primeros
      }
    }

    \new FretBoards {
      \set Staff.stringTunings = #cuatroTuning
%      \override FretBoard
%        #'(fret-diagram-details string-count) = 4
      \override FretBoard.fret-diagram-details.finger-code = #'in-dot
      \primeros
    }

    \new TabStaff \relative c'' {
      \set TabStaff.stringTunings = #cuatroTuning
      \primeros
    }

  >>
```

```
  \layout {
    \context {
      \Score
      \override SpacingSpanner.base-shortest-duration =
        \musicLength 16
    }
  }
  \midi { }
}
```



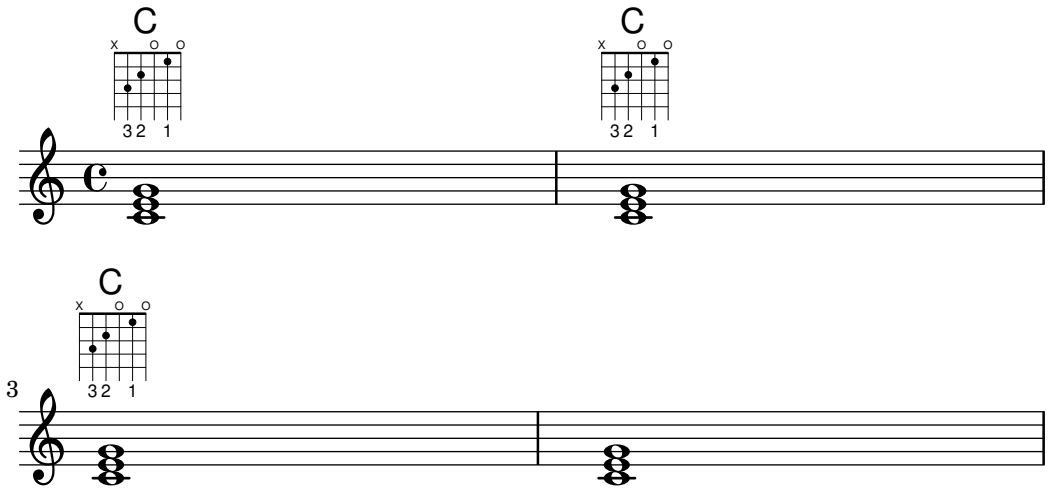
Chord changes for fretboards

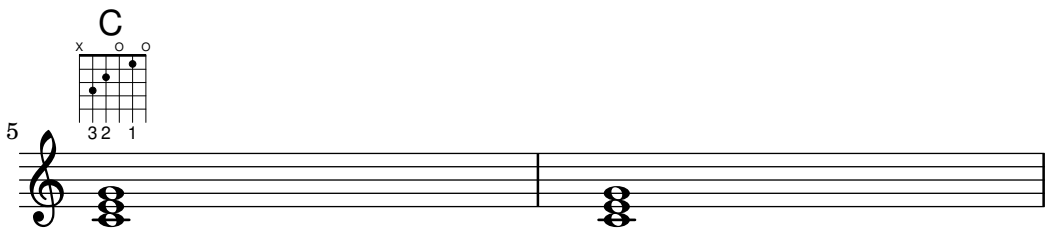
Fretboards can be set to display only when the chord changes, or at the beginning of a new line.

```
\include "predefined-guitar-fretboards.ly"

myChords = \chordmode {
  c1 c1 \break
  \set chordChanges = ##t
  c1 c1 \break
  c1 c1
}

<<
  \new ChordNames { \myChords }
  \new FretBoards { \myChords }
  \new Staff { \myChords }
>>
```





Alternative Bunddiagrammtabellen

Alternative Bunddiagrammtabellen können erstellt werden. Sie können benutzt werden, um alternative Bunddiagramme für einen bestimmten Akkord zu haben.

Damit eine alternative Bunddiagrammentabelle benutzt werden kann, muss die Tabelle zuerst erstellt werden. Dann werden die Bunddiagramme zur Tabelle hinzugefügt.

Die erstellte Bunddiagrammtabelle kann auch leer sein, oder sie kann aus einer existierenden Tabelle kopiert werden.

Die Tabelle, die eingesetzt wird, um vordefinierte Bunddiagramme anzuzeigen, wird mit der Eigenschaft \predefinedDiagramTable ausgewählt.

```
\include "predefined-guitar-fretboards.ly"

% Make a blank new fretboard table.
#(define custom-fretboard-table-one
   (make-fretboard-table))

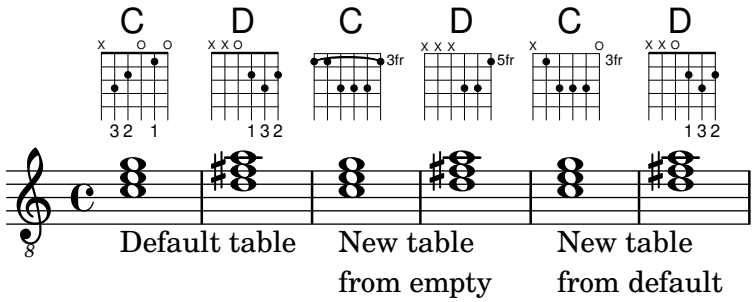
% Make a new fretboard table as a copy of `default-fret-table`.
#(define custom-fretboard-table-two
   (make-fretboard-table default-fret-table))

% Add a chord to `custom-fretboard-table-one`.
\storePredefinedDiagram #custom-fretboard-table-one
                        \chordmode {c}
                        #guitar-tuning
                        "3-(;3;5;5;5;3-);"

% Add a chord to `custom-fretboard-table-two`.
\storePredefinedDiagram #custom-fretboard-table-two
                        \chordmode {c}
                        #guitar-tuning
                        "x;3;5;5;5;o;"

<<
  \chords {
    c1 | d1 |
    c1 | d1 |
    c1 | d1 |
  }
  \new FretBoards {
    \chordmode {
      \set predefinedDiagramTable = #default-fret-table
      c1 | d1 |
      \set predefinedDiagramTable = #custom-fretboard-table-one
      c1 | d1 |
```

```
      \set predefinedDiagramTable = #custom-fretboard-table-two
      c1 | d1 |
    }
  }
  \new Staff {
    \clef "treble_8"
    <<
      \chordmode {
        c1 | d1 |
        c1 | d1 |
        c1 | d1 |
      }
      {
        s1_\markup "Default table" | s1 |
        s1_\markup \column { "New table" "from empty" } | s1 |
        s1_\markup \column { "New table" "from default" } | s1 |
      }
    >>
  }
>>
```



Siehe auch

Notationsreferenz: Abschnitt 12.1.4 [Angepasste Tabulaturen], Seite 343, Abschnitt 12.1.7 [Automatische Bund-Diagramme], Seite 365, Abschnitt 15.1.1 [Überblick über den Akkord-Modus], Seite 397, Abschnitt A.4 [Die vordefinierten Bund-Diagramme], Seite 647.

Installierte Dateien: `ly/predefined-guitar-fretboards.ly`, `ly/predefined-guitar-ninth-fretboards.ly`, `ly/predefined-ukulele-fretboards.ly`, `ly/predefined-mandolin-fretboards.ly`.

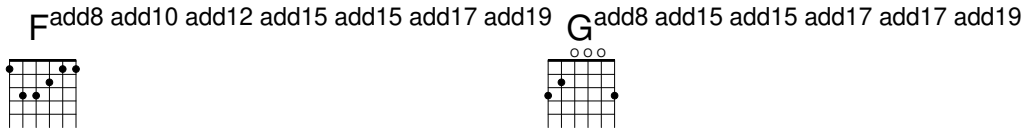
Schnipsel: Abschnitt “Fretted string instruments” in *Schnipsel*.

Referenz der Interna: Abschnitt “fret-diagram-interface” in *Referenz der Interna*.

12.1.7 Automatische Bund-Diagramme

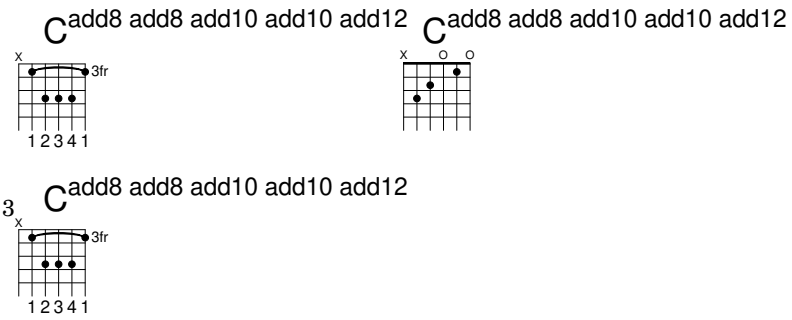
Bunddiagramme können automatisch aus notierten Noten erstellt werden. Hierzu wird der `FretBoards`-Kontext eingesetzt. Wenn keine vordefinierten Diagramme für die entsprechenden Noten mit der aktiven Saitenstimmung (`stringTunings`) vorhanden sind, errechnet der Kontext Saiten und Bünde die benutzt werden können, um die Noten zu spielen.

```
<<
  \context ChordNames {
    \chordmode {
      f1 g
    }
  }
  \context FretBoards {
    <f, c f a c' f'>1
    <g,\6 b, d g b g'>1
  }
  \context Staff {
    \clef "treble_8"
    <f, c f a c' f'>1
    <g, b, d g b' g'>1
  }
>>
```



Da in den Standardeinstellungen keine vordefinierten Diagramme geladen werden, ist die automatische Diagrammerstellung das Standardverhalten. Wenn die vordefinierten Diagramme eingesetzt werden, kann die automatische Berechnung an- und ausgeschaltet werden.

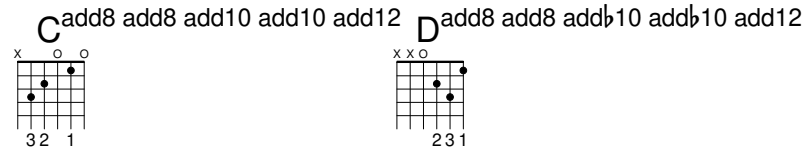
```
\storePredefinedDiagram #default-fret-table
                        <c e g c' e'>
                        #guitar-tuning
                        "x;3-1-(;5-2;5-3;5-4;3-1-1-);"
<<
  \context ChordNames {
    \chordmode {
      c1 c c
    }
  }
  \context FretBoards {
    <c e g c' e'>1
    \predefinedFretboardsOff
    <c e g c' e'>1
    \predefinedFretboardsOn
    <c e g c' e'>1
  }
  \context Staff {
    \clef "treble_8"
    <c e g c' e'>1
    <c e g c' e'>1
    <c e g c' e'>1
  }
>>
```



Manchmal kann die Berechnungsfunktion für Bunddiagramme kein passendes Diagramm finden. Das kann oft umgangen werden, indem man manuell einer Note eine bestimmte Saite zuweist. In vielen Fällen muss nur eine Note derart gekennzeichnet werden, der Rest wird dann entsprechend durch den `FretBoards`-Kontext behandelt.

Fingersatz kann zu FretBoard-Bunddiagrammen hinzugefügt werden.

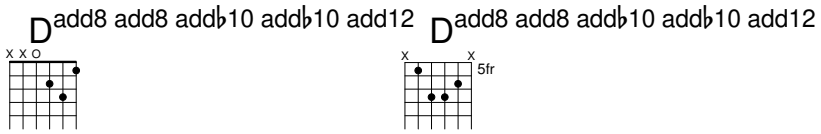
```
<<
  \context ChordNames {
    \chordmode {
      c1 d:m
    }
  }
  \context FretBoards {
    <c-3 e-2 g c'-1 e'>1
    <d a-2 d'-3 f'-1>1
  }
  \context Staff {
    \clef "treble_8"
    <c e g c' e'>1
    <d a d' f'>1
  }
>>
```



Der kleinste Bund, der benutzt wird, um Saiten und Bünde im FretBoard-Kontext zu errechnen, kann mit der `minimumFret`-Eigenschaft gesetzt werden.

```
<<
  \context ChordNames {
    \chordmode {
      d1:m d:m
    }
  }
  \context FretBoards {
    <d a d' f'>1
    \set FretBoards.minimumFret = #5
    <d a d' f'>1
  }
  \context Staff {
    \clef "treble_8"
```

```
    <d a d' f'>1
    <d a d' f'>1
  }
>>
```



Die Saiten und Bünde des `FretBoards`-Kontextes hängen von der `stringTunings`-Eigenschaft ab, die die gleiche Bedeutung wie im TabStaff-Kontext hat. Siehe auch Abschnitt 12.1.4 [Angepasste Tabulaturen], Seite 343, zu Information über die `stringTunings`Eigenschaft.

Die graphische Erscheinung eines Bunddiagrammes kann den Bedürfnissen angepasst werden, indem Eigenschaften der `fret-diagram-interface`-Schnittstelle verändert werden. Einzelheiten finden sich in Abschnitt “fret-diagram-interface” in *Referenz der Interna*. Die Schnittstelleneigenschaften eines `FretBoards`-Diagramms gehören dem `FretBoards.FretBoard`-Kontext an.

Vordefinierte Befehle

`\predefinedFretboardsOff`, `\predefinedFretboardsOn`.

Siehe auch

Notationsreferenz: Abschnitt 12.1.4 [Angepasste Tabulaturen], Seite 343.

Schnipsel: Abschnitt “Fretted string instruments” in *Schnipsel*.

Referenz der Interna: Abschnitt “fret-diagram-interface” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Automatische Bundberechnung funktioniert nicht richtig für Instrumente mit nicht-monotonischer Stimmung.

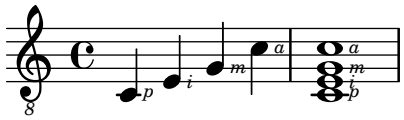
12.1.8 Fingersatz der rechten Hand

cindex Fingersatz der rechten Hand, Bundinstrumente

Fingersatz für die rechte Hand *p-i-m-a* muss mit `\rightHandFinger`, gefolgt von einer Zahl, notiert werden.

> **Achtung:** Wenn die Zahl in Scheme-Notation eingegeben wird, muss darauf geachtet werden, ein Leerzeichen anzuhängen vor dem schließenden > o. Ä.

```
\clef "treble_8"
c4\rightHandFinger #1
e\rightHandFinger #2
g\rightHandFinger #3
c'\rightHandFinger #4
<c\rightHandFinger #1 e\rightHandFinger #2
 g\rightHandFinger #3 c'\rightHandFinger #4 >1
```



Zur Erleichterung kann der Befehl `\rightHandFinger` zu ein paar Buchstaben abgekürzt werden, etwa `RH`.

```
RH=#rightHandFinger
```

Ausgewählte Schnipsel

Positionierung von Fingersatz der rechten Hand

Man kann die Positionierung von Fingersatz der rechten Hand besser kontrollieren, wenn eine bestimmte Eigenschaft gesetzt wird, wie das folgende Beispiel zeigt:

```
#(define RH rightHandFinger)

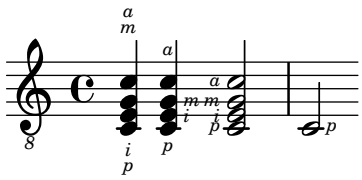
\relative c {
  \clef "treble_8"

  \set strokeFingerOrientations = #'(up down)
  <c\RH 1 e\RH 2 g\RH 3 c\RH 4 >4

  \set strokeFingerOrientations = #'(up right down)
  <c\RH 1 e\RH 2 g\RH 3 c\RH 4 >4

  \set strokeFingerOrientations = #'(left)
  <c\RH 1 e\RH 2 g\RH 3 c\RH 4 >2

  \set strokeFingerOrientations = #'(right)
  c\RH 1
}
```

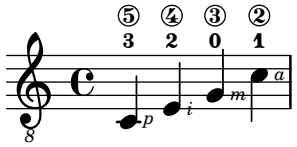


Fingersatz, Saitennummern und Fingersatz für die rechte Hand

Dieses Beispiel kombiniert Fingersatz für die linke Hand, Saitennummern und Fingersatz für die rechte Hand.

```
#(define RH rightHandFinger)

\relative c {
  \clef "treble_8"
  <c-3\5\RH 1 >4
  <e-2\4\RH 2 >4
  <g-0\3\RH 3 >4
  <c-1\2\RH 4 >4
}
```



Siehe auch

Schnipsel: Abschnitt “Fretted string instruments” in *Schnipsel*.

Referenz der Interna: Abschnitt “StrokeFinger” in *Referenz der Interna*.
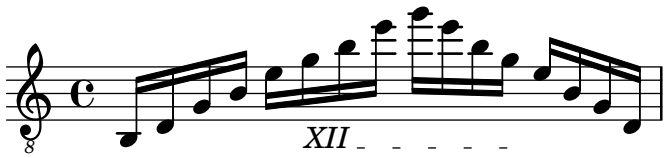
12.2 Gitarre

Die meisten der Besonderheiten von Gitarrennotation wurden im allgemeinen Abschnitt behandelt, aber es gibt noch einige, die hier gezeigt werden sollen. Teilweise soll ein Lead-sheet nur die Akkordsymbole und den Gesangstext enthalten. Da LilyPond ein Notensatzprogramm ist, wird es nicht für derartige Projekte empfohlen, die keine eigentliche Notation beinhalten. Anstatt dessen sollte ein Textbearbeitungsprogramm oder ein Satzprogramm wie GuitarTeX (für erfahrende Benutzer) eingesetzt werden.

12.2.1 Position und Barré anzeigen

Das Beispiel zeigt, wie man Griff- und Barréposition notieren kann.

```
\relative {
  \clef "treble_8"
  b,16 d g b e
  \textSpannerDown
  \override TextSpanner.bound-details.left.text = "XII "
  g16\startTextSpan
  b16 e g e b g\stopTextSpan
  e16 b g d
}
```



Siehe auch

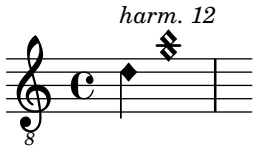
Notationsreferenz: Abschnitt 8.1.2 [Text mit Verbindungslinien], Seite 228.

Schnipsel: Abschnitt “Fretted string instruments” in *Schnipsel*, Abschnitt “Expressive marks” in *Schnipsel*.

12.2.2 Flageolett und gedämpfte Noten

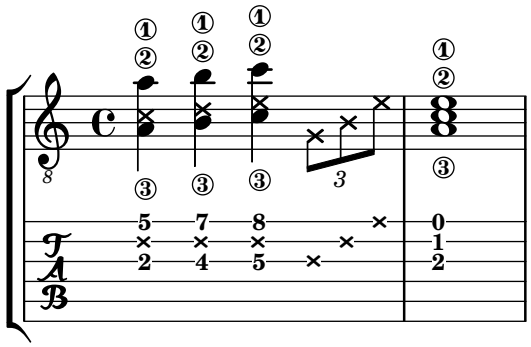
Besondere Notenköpfe können eingesetzt werden, um gedämpfte Noten oder Flageoletttöne anzuzeigen. Flageoletttöne werden normalerweise mit einem Text erklärt.

```
\relative {
  \clef "treble_8"
  \override Staff.NoteHead.style = #'harmonic-mixed
  d'^\markup { \italic { \fontsize #-2 { "harm. 12" }}} <g b>1
}
```



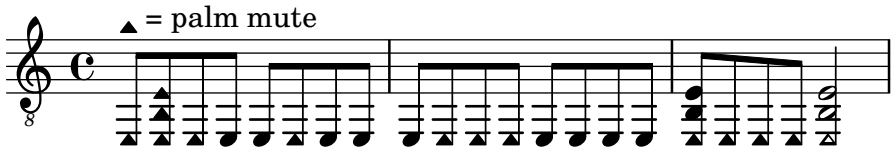
Gedämpfte oder gestoppte Noten werden in normalen und Tabulatur-Systemen unterstützt:

```
music = \relative {
  < a\3 \deadNote c\2 a'\1 >4
  < b\3 \deadNote d\2 b'\1 >
  < c\3 \deadNote e\2 c'\1 >
  \deadNotesOn
  \tuplet 3/2 { g8 b e }
  \deadNotesOff
  < a,\3 c\2 e\1 >1
}
\new StaffGroup <<
  \new Staff {
    \clef "treble_8"
    \music
  }
  \new TabStaff {
    \music
  }
>>
```



Eine andere Spieltechnik (insbesondere bei elektrischen Gitarren benutzt) ist *palm mute*. Hierbei wird die Saite teilweise durch die Handfläche der Schlaghand gedämpft. LilyPond unterstützt die Notation dieser Art von Technik, indem die Notenköpfe der so gedämpften Noten durch Dreiecke ersetzt werden.

```
\new Voice { % Warning: explicit Voice instantiation is
             %    required to have palmMuteOff work properly
             %    when palmMuteOn comes at the beginning of
             %    the piece.
  \relative c, {
    \clef "G_8"
    \palmMuteOn
    e8^\markup { \musicglyph "noteheads.s2do"  = palm mute }
    < e b' e > e
    \palmMuteOff
    e e  \palmMute e e e |
    e8 \palmMute { e e e } e e e e |
    < \palmMute e b' e >8 \palmMute { e e e } < \palmMute e b' e >2
  }
}
```



Siehe auch

Notationsreferenz: Abschnitt 1.4.1 [Besondere Notenköpfe], Seite 36, Abschnitt A.9 [Notenkopfstile], Seite 678.

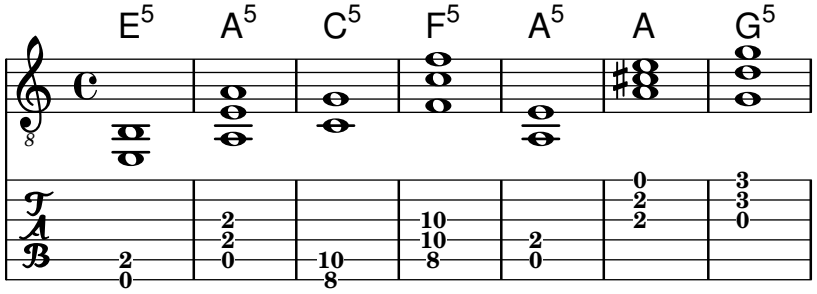
Schnipsel: Abschnitt “Fretted string instruments” in *Schnipsel*.

12.2.3 Powerakkorde anzeigen

Powerakkorde und ihre Symbole können im Akkordmodus oder als Akkordkonstruktionen gesetzt werden:

```
ChordsAndSymbols = {
  \chordmode {
    e,,1:5
    a,,:5.8
    \set TabStaff.restrainOpenStrings = ##t
    \set minimumFret = #8
    c,:5
    f,:5.8
  }
  \set minimumFret = #2
  \set restrainOpenStrings = ##f
  <a, e> <a cis' e'>
  <g d' g'>
}
\score {
  <<
    \new ChordNames {
    \ChordsAndSymbols
    }
    \new Staff {
      \clef "treble_8"
      \ChordsAndSymbols
```

```
    }
    \new TabStaff {
      \ChordsAndSymbols
    }
  >>
}
```



Siehe auch

Glossar: Abschnitt “power chord” in *Glossar*.

Notationsreferenz: Abschnitt 15.1.3 [Erweiterte und modifizierte Akkorde], Seite 400, Abschnitt 15.2.1 [Akkordbezeichnungen drucken], Seite 403.

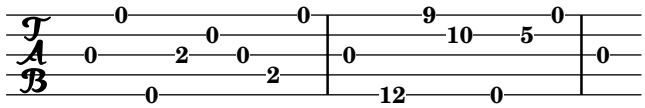
Schnipsel: Abschnitt “Fretted string instruments” in *Schnipsel*.

12.3 Banjo

12.3.1 Banjo-Tabulaturen

LilyPond hat grundlegende Unterstützung für fünfsaitige Banjo. Die Banjo-Tabulatur-Funktion sollte zum Notieren von Banjo-Tabulaturen verwendet werden, damit die richtigen Bund-Nummern für die fünfte Saite gesetzt werden:

```
\new TabStaff <<
  \set TabStaff.tablatureFormat = #fret-number-tablature-format-banjo
  \set TabStaff.stringTunings = #banjo-open-g-tuning
  {
    \stemDown
    g8 d' g'\5 a b g e d' |
    g4 d''8\5 b' a'\2 g'\5 e'\2 d' |
    g4
  }
>>
```



Eine Anzahl von üblichen Stimmungen für fünsaitiges Banjo sind in LilyPond vordefiniert: `banjo-c-tuning` (gCGBD), `banjo-modal-tuning` (gDGCD), `banjo-open-d-tuning` (aDF#AD) und `banjo-open-dm-tuning` (aDFAD).

Sie können für das viersaitige Banjo angepasst werden, indem die `four-string-banjo`-Funktion eingesetzt wird:

```
\set TabStaff.stringTunings = #(four-string-banjo banjo-c-tuning)
```

Siehe auch

Installierte Dateien: `ly/string-tunings-init.ly`.

Schnipsel: Abschnitt “Fretted string instruments” in *Schnipsel*.

13 Schlagzeug

13.1 Übliche Notation für Schlagzeug

Rhythmusnotation wird vor allem für Schlaginstrumente eingesetzt, aber hiermit kann auch der Rhythmus einer Melodie dargestellt werden.

13.1.1 Referenz für Schlagzeug

- Viele Schlagzeugmusik kann auf einem rhythmischen System notiert werden. Das wird gezeigt in Abschnitt 2.3.7 [Melodierhythmus anzeigen], Seite 79, und Abschnitt 6.1.1 [Neue Notensysteme erstellen], Seite 183.
- MIDI-Ausgabe wird behandelt in Abschnitt 23.6 [Schlagzeug in MIDI], Seite 514.

Siehe auch

Notationsreferenz: Abschnitt 2.3.7 [Melodierhythmus anzeigen], Seite 79, Abschnitt 6.1.1 [Neue Notensysteme erstellen], Seite 183, Abschnitt 23.6 [Schlagzeug in MIDI], Seite 514.

Schnipsel: Abschnitt “Percussion” in *Schnipsel*.

13.1.2 Grundlagen der Schlagzeugnotation

Schlagzeug-Noten können im `\drummode`-Modus notiert werden, der sich ähnlich verhält wie der Standardmodus für die Noteneingabe. Am einfachsten kann der `\drums`-Befehl benutzt werden, der sich um den richtigen Kontext und Eingabemodus kümmert:

```
\drums {
  hihat4 hh bassdrum bd
}
```



Das ist die Kurzschreibweise für:
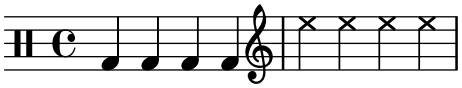
```
\new DrumStaff {
  \drummode {
    hihat4 hh bassdrum bd
  }
}
```



Jedes Schlagzeuginstrument hat einen langen Namen und eine Abkürzung, und beide können nebeneinander benutzt werden. Eine Liste der Notenbezeichnungen für Schlagzeug findet sich in Abschnitt A.14 [Schlagzeugnoten], Seite 763.

Beachten Sie, dass normale Tonhöhen (wie `cis4`) in einem `DrumStaff`-Kontext eine Fehlernachricht erzielen. Schlüssel für Schlagzeug werden automatisch hinzugefügt, aber sie können auch explizit gesetzt Werden. Auch andere Schlüssel können benutzt werden.

```
\drums {
  \clef percussion
  bd4 bd bd bd
  \clef treble
  hh4 hh hh hh
}
```



Es gibt einige Probleme mit der MIDI-Unterstützung für Schlagzeuginstrumente. Details finden sich in Abschnitt 23.6 [Schlagzeug in MIDI], Seite 514.

Siehe auch

Notationsreferenz: Abschnitt 23.6 [Schlagzeug in MIDI], Seite 514, Abschnitt A.14 [Schlagzeugnoten], Seite 763.
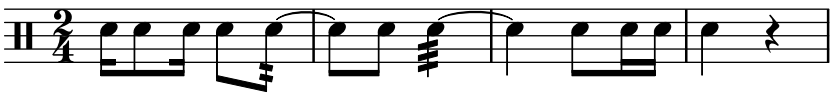
Installierte Dateien: `ly/drumpitch-init.ly`.

Schnipsel: Abschnitt “Percussion” in *Schnipsel*.

13.1.3 Trommelwirbel

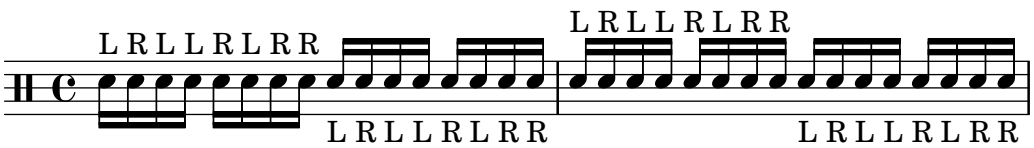
Trommelwirbel werden mit drei Balken durch den Notenhals notiert. Für Viertelnoten oder längere Noten werden die drei Balken explizit notiert, Achtel werden mit zwei Balken gezeigt (und der dritte ist der eigentliche Balken), und Trommelwirbel mit kürzeren Werten als Achtelnoten haben einen Balken zusätzlich zu den eigentlichen Balken der Noten. Dieses Verhalten wird mit der Tremolonotation erreicht, wie in Abschnitt 4.2.2 [Tremolo-Wiederholung], Seite 158, gezeigt.

```
\drums {
  \time 2/4
  sn16 sn8 sn16 sn8 sn8:32 ~
  sn8 sn8 sn4:32 ~
  sn4 sn8 sn16 sn16
  sn4 r4
}
```



Benutzung der Stöcke kann angezeigt werden, indem eine Beschriftung durch `"R"` oder `"L"` über oder unter der Note angefügt wird, Näheres in Abschnitt 35.2 [Richtung und Platzierung], Seite 611. Die `staff-padding`-Eigenschaft kann verändert werden, um eine Orientierung an einer gemeinsamen Linie zu ermöglichen.

```
\drums {
  \repeat unfold 2 {
    sn16^"L" sn^"R" sn^"L" sn^"L" sn^"R" sn^"L" sn^"R" sn^"R"
    \stemUp
    sn16_"L" sn_"R" sn_"L" sn_"L" sn_"R" sn_"L" sn_"R" sn_"R"
  }
}
```



Siehe auch

Notationsreferenz: Abschnitt 4.2.2 [Tremolo-Wiederholung], Seite 158.

Schnipsel: Abschnitt “Percussion” in *Schnipsel*.

13.1.4 Schlagzeug mit Tonhöhe

Bestimmte Schlagzeuginstrumente mit Tonhöhe (z. B. Xylophone, vibraphone und Pauken) werden auf normalen Systemen geschrieben. Das wird in anderen Abschnitten des Handbuchs behandelt.

Siehe auch

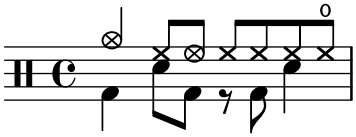
Notationsreferenz: Abschnitt 23.6 [Schlagzeug in MIDI], Seite 514.

Schnipsel: Abschnitt “Percussion” in *Schnipsel*.

13.1.5 Schlagzeugsysteme

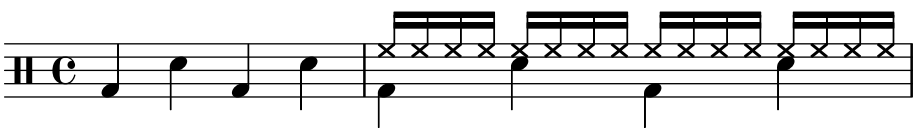
Ein Schlagzeug-System besteht üblicherweise aus einem Notensystem mit mehreren Linien, wobei jede Linie ein bestimmtes Schlagzeug-Instrument darstellt. Um die Noten darstellen zu können, müssen sie sich innerhalb von einem `DrumStaff`- und einem `DrumVoice`-Kontext befinden.

```
up = \drummode {
  crashcymbal4 hihat8 halfopenhihat hh hh hh openhihat
}
down = \drummode {
  bassdrum4 snare8 bd r bd sn4
}
\new DrumStaff <<
  \new DrumVoice { \voiceOne \up }
  \new DrumVoice { \voiceTwo \down }
>>
```



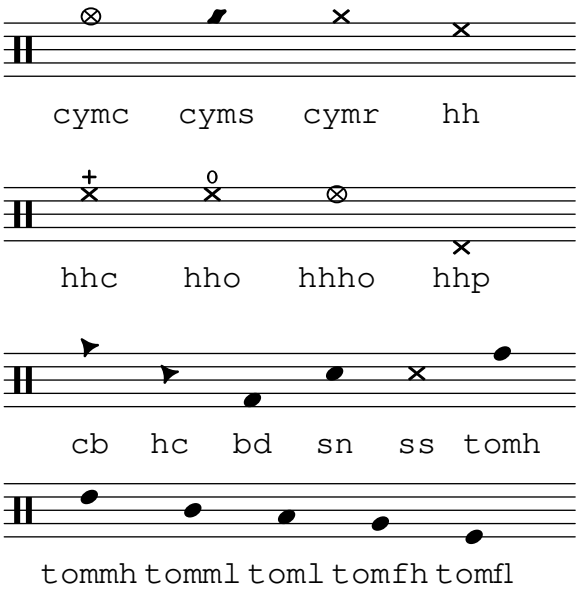
Das Beispiel zeigt ausdrücklich definierte mehrstimmige Notation. Die Kurznotation für mehrstimmige Musik, wie sie im Abschnitt Abschnitt “Ich höre Stimmen” in *Handbuch zum Lernen* beschrieben wird, kann auch verwendet werden.

```
\new DrumStaff <<
  \drummode {
    bd4 sn4 bd4 sn4
    << {
      \repeat unfold 16 hh16
    } \\ {
      bd4 sn4 bd4 sn4
    } >>
  }
>>
```



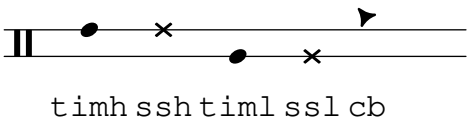
Es gibt auch weitere Layout-Einstellungen. Um diese zu verwenden, muss die Eigenschaft `drumStyleTable` im `DrumVoice`-Kontext entsprechend eingestellt werden. Folgende Variablen sind vordefiniert:

`drums-style`
: Das ist die Standardeinstellung. Hiermit wird ein typisches Schlagzeug-System auf fünf Notenlinien erstellt.

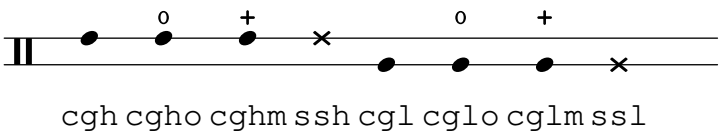


Die Schlagzeugdefinitionen unterstützen sechs unterschiedliche Tom Toms. Falls eine geringere Anzahl verwendet wird, kann man einfach die Tom Toms auswählen, deren Notation man haben will. Tom Toms auf den drei mittleren Linien werden mit den Bezeichnungen `tommh`, `tomml` und `tomfh` notiert.

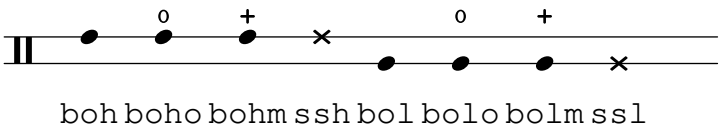
`timbales-style`
: Hiermit werden Timbale auf zwei Notenlinien gesetzt.



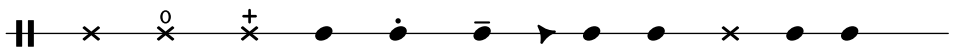
`congas-style`
: Hiermit werden Congas auf zwei Linien gesetzt.



`bongos-style`
: Hiermit werden Bongos auf zwei Linien gesetzt.



`percussion-style`
: Dieser Stil ist für alle einfachen Perkussionsinstrumente auf einer Notenlinie.



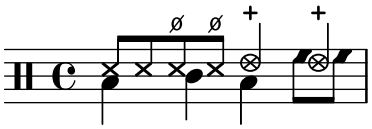
tri trio trim gui guis guil cb cl tamb cab mar hc

13.1.6 Eigene Schlagzeugsysteme

Wenn ihnen keine der vordefinierten Stile gefällt, können Sie auch eine eigene Liste der Positionen und Notenköpfe am Anfang ihrer Datei erstellen.

```
#(define mydrums '(
         (bassdrum        default   #f          -1)
         (snare           default   #f          0)
         (hihat           cross     #f          1)
         (halfopenhihat   cross     halfopen    1)
         (pedalhihat      xcircle   stopped     2)
         (lowtom          diamond   #f          3)))
up = \drummode { hh8 hh hhho hhho hhp4 hhp }
down = \drummode { bd4 sn bd toml8 toml }

\new DrumStaff <<
  \set DrumStaff.drumStyleTable = #(alist->hash-table mydrums)
  \new DrumVoice { \voiceOne \up }
  \new DrumVoice { \voiceTwo \down }
>>
```



Ausgewählte Schnipsel

Hier einige Beispiele:

Zwei Holzblöcke, notiert mit wbh (hoch) und wbl (tief)

```
% These lines define the position of the woodblocks in the stave;
% if you like, you can change it or you can use special note heads
% for the woodblocks.
#(define mydrums '((hiwoodblock default #f  3)
                   (lowoodblock default #f -2)))

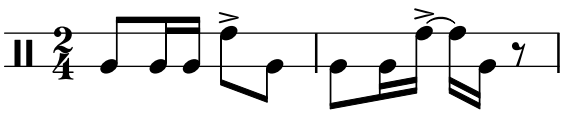
woodstaff = {
  % This defines a staff with only two lines.
  % It also defines the positions of the two lines.
  \override Staff.StaffSymbol.line-positions = #'(-2 3)

  % This is necessary; if not entered, the barline would be too short!
  \override Staff.BarLine.bar-extent = #'(-1.5 . 1.5)
}

\new DrumStaff {
  \set DrumStaff.drumStyleTable = #(alist->hash-table mydrums)

  % with this you load your new drum style table
  \woodstaff
```

```
  \drummode {
    \time 2/4
    wbl8 wbl16 wbl wbh8-> wbl |
    wbl8 wbl16 wbh-> ~ wbh wbl16 r8 |
  }
}
```



In diesem Spezialfalls muss die Länge der Taktlinie mit `\override Staff.BarLine.bar-extent #'(von . bis)` angepasst werden. Andernfalls wäre sie zu kurz. Die Position der beiden Linien muss auch definiert werden.

Tamburin, notiert mit `tamb`:

```
#(define mydrums '((tambourine default #f 0)))

tambustaff = {
  \override Staff.StaffSymbol.line-positions = #'( 0 )
  \override Staff.BarLine.bar-extent = #'(-1.5 . 1.5)
  \set DrumStaff.instrumentName = "Tambourine"
}

\new DrumStaff {
  \tambustaff
  \set DrumStaff.drumStyleTable = #(alist->hash-table mydrums)

  \drummode {
    \time 6/8
    tamb8. tamb16 tamb8 tamb tamb tamb |
    tamb4. tamb8 tamb tamb |
    % the trick with the scaled duration and the shorter rest
    % is neccessary for the correct ending of the trill-span!
    tamb2.*5/6 \startTrillSpan s8 \stopTrillSpan |
  }
}
```



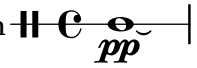
Noten für Tam-Tam (notiert mit `tt`):

```
#(define mydrums '((tamtam default #f 0)))

tamtamstaff = {
  \override Staff.StaffSymbol.line-positions = #'( 0 )
  \override Staff.BarLine.bar-extent = #'(-1.5 . 1.5)
  \set DrumStaff.instrumentName = "Tamtam"
}

\new DrumStaff {
  \tamtamstaff
  \set DrumStaff.drumStyleTable = #(alist->hash-table mydrums)
```

```
  \drummode {
    tt 1 \pp \laissezVibrer
  }
}
```

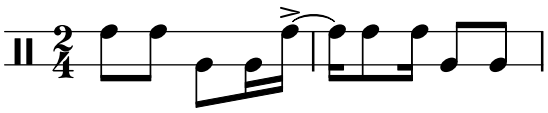
Tamtam 

Zwei Glocken, notiert mit cb (Kuhglocke) und rb (Reiterglocke)

```
#(define mydrums '((ridebell default #f  3)
                  (cowbell  default #f -2)))

bellstaff = {
  \override DrumStaff.StaffSymbol.line-positions = #'(-2 3)
  \set DrumStaff.drumStyleTable = #(alist->hash-table mydrums)
  \override Staff.BarLine.bar-extent = #'(-1.5 . 1.5)
  \set DrumStaff.instrumentName = "Different Bells"
}

\new DrumStaff {
  \bellstaff
  \drummode {
    \time 2/4
    rb8 rb cb cb16 rb-> ~ |
    16 rb8 rb16 cb8 cb |
  }
}
```

Different Bells 

Hier ein kurzes Beispiel von Stravinsky (aus „L'Histoire du soldat“):

```
#(define mydrums '((bassdrum   default #f  4)
                  (snare      default #f -4)
                  (tambourine default #f  0)))

global = {
  \time 3/8 s4.
  \time 2/4 s2*2
  \time 3/8 s4.
  \time 2/4 s2
}

drumsA = {
  \context DrumVoice <<
    { \global }
    { \drummode {
        \autoBeamOff
        \stemDown sn8 \stemUp tamb s8 |
        sn4 \stemDown sn4 |
        \stemUp tamb8 \stemDown sn8 \stemUp sn16 \stemDown sn \stemUp sn8 |
```
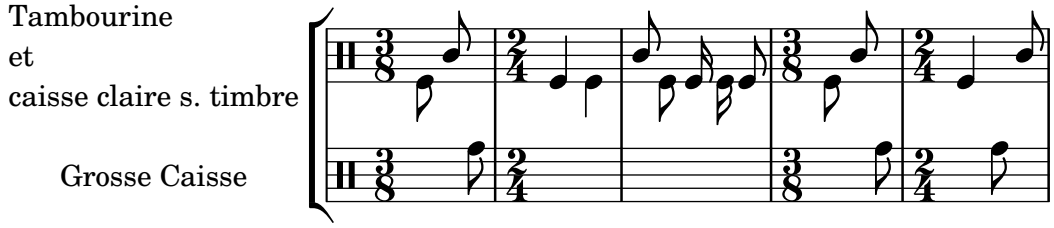
```
        \stemDown sn8 \stemUp tamb s8 |
        \stemUp sn4 s8 \stemUp tamb
      }
    }
  >>
}

drumsB = {
  \drummode {
    s4 bd8 s2*2 s4 bd8 s4 bd8 s8
  }
}

\layout {
  indent = #40
}

\score {
  \new StaffGroup <<
    \new DrumStaff {
      \set DrumStaff.instrumentName = \markup {
        \column {
          "Tambourine"
          "et"
          "caisse claire s. timbre"
        }
      }
      \set DrumStaff.drumStyleTable = #(alist->hash-table mydrums)
      \drumsA
    }

   \new DrumStaff {
     \set DrumStaff.instrumentName = "Grosse Caisse"
     \set DrumStaff.drumStyleTable = #(alist->hash-table mydrums)
     \drumsB }
  >>
}
```



Siehe auch

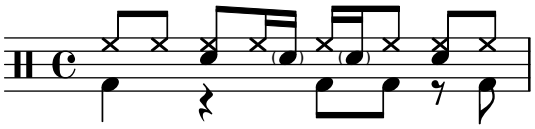
Schnipsel: Abschnitt “Percussion” in *Schnipsel*.

Referenz der Interna: Abschnitt “DrumStaff” in *Referenz der Interna*, Abschnitt “DrumVoice” in *Referenz der Interna*.

13.1.7 Geisternoten

Geisternoten für Schlagzeug und Perkussion können mit dem Klammer- (\parenthesize)-Befehl, beschrieben in Abschnitt 7.1.5 [Klammern], Seite 219, erstellt werden. Im Standard-\drummode-Modus ist aber das Parenthesis_engraver-Plugin nicht automatisch enthalten.

```
\new DrumStaff \with {
  \consists Parenthesis_engraver
}
<<
  \context DrumVoice  = "1" { s1 }
  \context DrumVoice  = "2" { s1 }
  \drummode {
    <<
      {
        hh8[ hh] <hh sn> hh16
        < \parenthesize sn > hh
        < \parenthesize sn > hh8 <hh sn> hh
      } \\
      {
        bd4 r4 bd8 bd r8 bd
      }
    >>
  }
>>
```

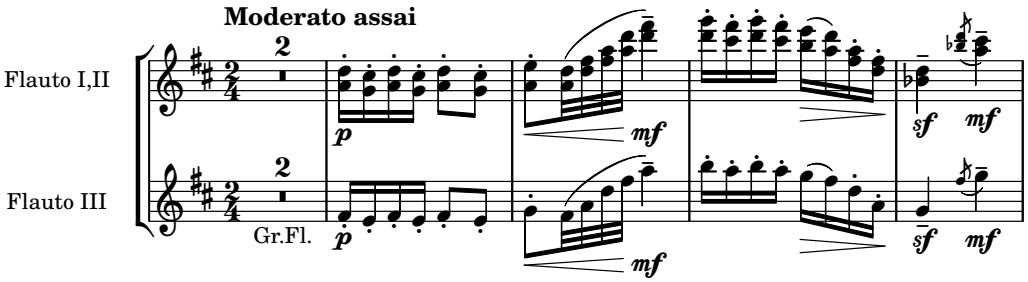


Um jede Klammer-Definition (\parenthesize) müssen zusätzlich die spitzen Klammern für Akkorde (< >) gesetzt werden.

Siehe auch

Schnipsel: Abschnitt “Percussion” in *Schnipsel*.

14 Blasinstrumente



Dieser Abschnitt beinhaltet einige Notationselemente, die bei der Notation von Blasinstrumenten auftreten.

14.1 Übliche Notation für Bläser

Dieser Abschnitt erklärt Notation, die für die meisten Blasinstrumente gültig sind.

14.1.1 Referenz für Blasinstrumente

Viele Besonderheiten der Blasinstrumentenotation haben mit Atmung und Spielart zu tun:

- Atmung kann durch Pausen oder mit Atemzeichen angezeigt werden,, siehe Abschnitt 3.2.3 [Atemzeichen], Seite 134.
- Legato kann durch Legatobögen angezeigt werden, siehe Abschnitt 3.2.1 [Legatobögen], Seite 130.
- Unterschiedliche Artikulationen, Legato, Portato, Staccato, werden normalerweise mit Artikulationszeichen angemerkt, teilweise auch in Verbindung mit Legatobögen, siehe Abschnitt 3.1.1 [Artikulationszeichen und Verzierungen], Seite 120, und Abschnitt A.13 [Liste der Artikulationszeichen], Seite 760.
- Flatterzunge wird angezeigt, indem ein Tremolozeichen und eine Anmerkung für die entsprechende Note gesetzt wird. Siehe Abschnitt 4.2.2 [Tremolo-Wiederholung], Seite 158.

Es gibt auch noch weitere Aspekte der Notation, die für Blasinstrumente relevant sein können:

- Viele Instrumente sind transponierend, siehe Abschnitt 1.3.4 [Transposition von Instrumenten], Seite 25.
- Das Zug-Glissando ist charakteristisch für die Posaune, aber auch andere Instrumente können Glissandos ausführen. Siehe Abschnitt 3.3.1 [Glissando], Seite 136.
- Obertonreihenglissandi, die auf allen Blechblasinstrumenten möglich, aber besonders üblich für das Waldhorn sind, werden üblicherweise mit Verzierungsnoten geschrieben. Siehe Abschnitt 2.6.1 [Verzierungen], Seite 112.
- Tonhöhenschwankungen am Ende eines Tons werden gezeigt in Abschnitt 3.2.4 [Glissando zu unbestimmter Tonhöhe], Seite 136.
- Ventil- oder Klappenschläge werden oft als Kreuznoten dargestellt, siehe Abschnitt 1.4.1 [Besondere Notenköpfe], Seite 36.
- Holzbläser können tiefe Noten überblasen. Derartige Noten werden als `flageolet`-Artikulation notiert. Siehe Abschnitt A.13 [Liste der Artikulationszeichen], Seite 760.
- Die Benutzung von Dämpfern für Blechblasinstrumente wird meistens durch Text gefordert, aber bei schnellem Wechsel bietet es sich an, die Artikulationszeichen `stopped` und `open` zu benutzen. Siehe Abschnitt 3.1.1 [Artikulationszeichen und Verzierungen], Seite 120, und Abschnitt A.13 [Liste der Artikulationszeichen], Seite 760.
- Gestopfte Hörner werden mit dem `stopped`-Artikulationszeichen notiert. Siehe Abschnitt 3.1.1 [Artikulationszeichen und Verzierungen], Seite 120.

Siehe auch

Notationsreferenz: Abschnitt 3.2.3 [Atemzeichen], Seite 134, Abschnitt 3.2.1 [Legatobögen], Seite 130, Abschnitt 3.1.1 [Artikulationszeichen und Verzierungen], Seite 120, Abschnitt A.13 [Liste der Artikulationszeichen], Seite 760, Abschnitt 4.2.2 [Tremolo-Wiederholung], Seite 158, Abschnitt 1.3.4 [Transposition von Instrumenten], Seite 25, Abschnitt 3.3.1 [Glissando], Seite 136, Abschnitt 2.6.1 [Verzierungen], Seite 112, Abschnitt 3.2.4 [Glissando zu unbestimmter Tonhöhe], Seite 136, Abschnitt 1.4.1 [Besondere Notenköpfe], Seite 36,

Schnipsel: Abschnitt “Wind instruments” in *Schnipsel*.

14.1.2 Fingersatz

Alle Blasinstrumente außer der Posaune benötigen mehrere Finger, um verschiedene Tonhöhen zu produzieren. Einige Fingersatzbeispiele zeigen die Schnipsel unten.

Diagramme für Holzbläser können erstellt werden nach den Anweisungen in Abschnitt 14.3.1 [Holzbläserdiagramme], Seite 389.

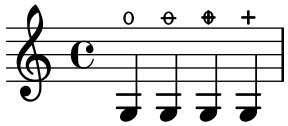
Ausgewählte Schnipsel

Fingering symbols for wind instruments

Special symbols can be achieved by combining existing glyphs, which is useful for wind instruments.

```
lineup =
  \tweak outside-staff-padding #0
  \tweak staff-padding #0
  \tweak padding #0.2
  \tweak parent-alignment-X #CENTER
  \tweak self-alignment-X #CENTER
  \etc

\relative c' {
  g\open
  g\lineup ^\markup \combine
              \musicglyph "scripts.open"
              \musicglyph "scripts.tenuto"
  g\lineup ^\markup \combine
              \musicglyph "scripts.open"
              \musicglyph "scripts.stopped"
  g\stopped
}
```



Recorder fingering chart

The following example demonstrates how fingering charts for wind instruments can be realized.

```
% range chart for paetzold contrabass recorder

centermarkup = {
  \once \override TextScript.self-alignment-X = #CENTER
  \once \override TextScript.X-offset = #(lambda (g)
    (+ (ly:self-alignment-interface::centered-on-x-parent g)
       (ly:self-alignment-interface::x-aligned-on-self g)))
}

\new Staff \with {
  \remove "Time_signature_engraver"
  \omit Stem
  \omit Flag
  \consists "Horizontal_bracket_engraver"
} {
  \clef bass
  \set Score.timing = ##f

  f,1*1/4 \glissando

  \clef violin
  gis'1*1/4

  a'4^\markup "1)"

  \centermarkup
    \once \override TextScript.padding = 2
    bes'1*1/4_\markup \override #'(baseline-skip . 1.7) \column {
      \fontsize #-5
      \slashed-digit #0 \finger 1 \finger 2
      \finger 3 \finger 4 \finger 5 \finger 6 \finger 7 }

  b'1*1/4

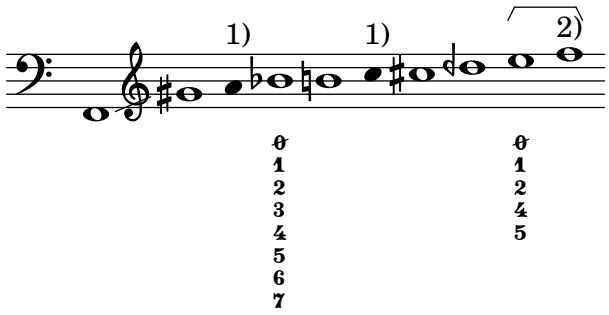
  c''4^\markup "1)"

  cis''1*1/4

  deh''1*1/4

  \centermarkup
    \once \override TextScript.padding = 2
    \once \override Staff.HorizontalBracket.direction = #UP
    e''1*1/4_\markup \override #'(baseline-skip . 1.7) \column {
      \fontsize #-5
      \slashed-digit #0 \finger 1 \finger 2
      \finger 4 \finger 5 } \startGroup

  f''1*1/4^\markup "2)" \stopGroup
}
```



Siehe auch

Notationsreferenz: Abschnitt 14.3.1 [Holzbläserdiagramme], Seite 389.

Snippets: Abschnitt “Wind instruments” in *Schnipsel*.

14.2 Dudelsack

Dieser Abschnitt beinhaltet die Notation von Dudelsackmusik.

14.2.1 Dudelsack-Definitionen

LilyPond besitzt spezielle Definitionen, mit der die Musik des schottischen Hochland-Dudelsacks notiert wird. Um sie zu benutzen, muss

```
\include "bagpipe.ly"
```
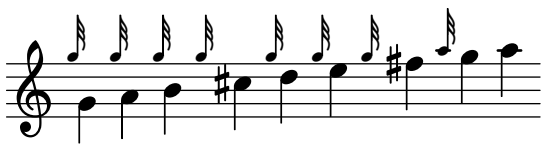
am Anfang der LilyPond-Quelldatei eingefügt werden. Hierdurch können dann bestimmte Verzierungsnoten, die für die Dudelsackmusik üblich sind, mit kurzen Befehlen eingefügt werden. So reicht etwa der Befehl `\taor`, anstatt

```
\grace { \small G32[ d G e] }
```

zu schreiben.

`bagpipe.ly` enthält außerdem Definitionen für Tonhöhen von Dudelsacknoten in bestimmten Oktaven, so dass man sich nicht mehr um `\relative` oder `\transpose` kümmern muss.

```
\include "bagpipe.ly"
{ \grg G4 \grg a \grg b \grg c \grg d \grg e \grg f \grA g A }
```



Musik für den Dudelsack wird in D-Dur geschrieben (auch wenn das eigentlich nicht stimmt). Weil das aber die einzige Tonart ist, die benutzt werden kann, werden die Vorzeichen meistens nicht geschrieben. Damit das funktioniert, müssen die Noten immer mit `\hideKeySignature` beginnen. Wenn die Vorzeichen hingegen angezeigt werden sollen, kann das mithilfe des Befehls `\showKeySignature` vorgenommen werden.

Einige moderne Dudelsacknoten benutzen halbe Finger auf c und f, um diese Noten zu erniedrigen. Das kann angezeigt werden mit `c-flat` bzw. `f-flat`. Gleicherweise kann das piobaireachd hohe g als `g-flat` geschrieben werden, wenn es in leichter Musik vorkommt.

Siehe auch

Schnipsel: Abschnitt “Wind instruments” in *Schnipsel*.
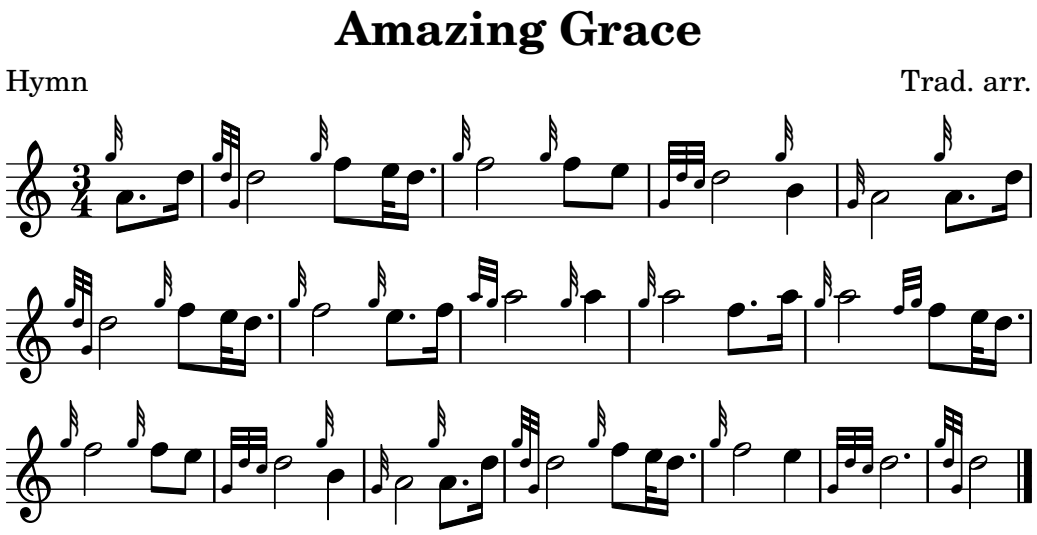
14.2.2 Dudelsack-Beispiele

So sieht die bekannte Melodie Amazing Grace aus, wenn man sie für Dudelsack notiert.

```
\include "bagpipe.ly"
\layout {
  indent = 0.0\cm
  \context { \Score \remove Bar_number_engraver }
}

\header {
  title = "Amazing Grace"
  meter = "Hymn"
  arranger = "Trad. arr."
}

{
  \hideKeySignature
  \time 3/4
  \grg \partial 4 a8. d16
  \slurd d2 \grg f8[ e32 d16.]
  \grg f2 \grg f8 e
  \thrwd d2 \grg b4
  \grG a2 \grg a8. d16
  \slurd d2 \grg f8[ e32 d16.]
  \grg f2 \grg e8. f16
  \dblA A2 \grg A4
  \grg A2 f8. A16
  \grg A2 \hdblf f8[ e32 d16.]
  \grg f2 \grg f8 e
  \thrwd d2 \grg b4
  \grG a2 \grg a8. d16
  \slurd d2 \grg f8[ e32 d16.]
  \grg f2 e4
  \thrwd d2.
  \slurd d2
  \bar "|."
}
```



Siehe auch

Schnipsel: Abschnitt “Wind instruments” in *Schnipsel*.

14.3 Holzbläser

Dieser Abschnitt zeigt Notation, die spezifisch für Holzbläser ist.

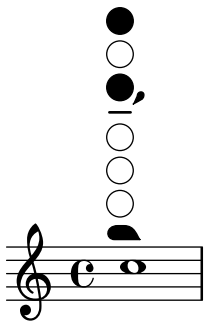
14.3.1 Holzbläserdiagramme

Holzbläserdiagramme können benutzt werden, um die Griffe für eine bestimmte Note darzustellen. Diagramme gibt es für folgende Instrumente:

- Piccolo
- Flöte
- Oboe
- Clarinette
- BassClarinette
- Saxophon
- Fagott
- Kontrafagott

Holzbläserdiagramme werden als Beschriftung erstellt:

```
c''1^\markup {
  \woodwind-diagram #'piccolo #'((lh . (gis))
                                 (cc . (one three))
                                 (rh . (ees)))
}
```



Löcher können offen, halboffen, Ring oder geschlossen sein:

```
\textLengthOn
c''1^\markup {
  \center-column {
    "Ein Viertel"
    \woodwind-diagram #'flute #'((cc . (one1q))
                                 (lh . ())
                                 (rh . ()))
  }
}

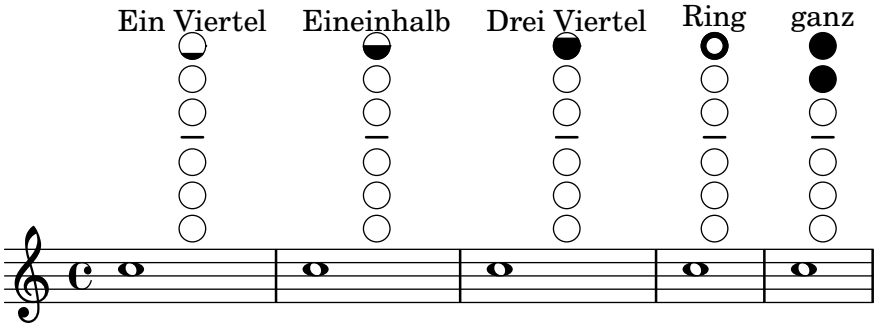
c''1^\markup {
  \center-column {
    "Eineinhalb"
    \woodwind-diagram #'flute #'((cc . (one1h))
```

```
                                  (lh . ())
                                  (rh . ()))
  }
}

c''1^\markup {
  \center-column {
    "Drei Viertel"
    \woodwind-diagram #'flute #'((cc . (one3q))
                                  (lh . ())
                                  (rh . ()))
  }
}

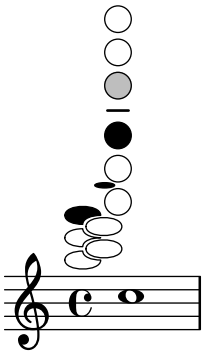
c''1^\markup {
  \center-column {
    "Ring"
    \woodwind-diagram #'flute #'((cc . (oneR))
                                  (lh . ())
                                  (rh . ()))
  }
}

c''1^\markup {
  \center-column {
    "ganz"
    \woodwind-diagram #'flute #'((cc . (oneF two))
                                  (lh . ())
                                  (rh . ()))
  }
}
```



Triller werden als schattierte Löcher in den Diagrammen angezeigt:

```
c''1^\markup {
  \woodwind-diagram #'bass-clarinet
                    #'((cc . (threeT four))
                       (lh . ())
                       (rh . (b fis)))
}
```



Eine Vielzahl von Trillern ist möglich:

```
\textLengthOn
c''1^\markup {
  \center-column {
    "ein Viertel zu Ring"
    \woodwind-diagram #'flute #'((cc . (one1qTR))
                                 (lh . ())
                                 (rh . ()))
  }
}

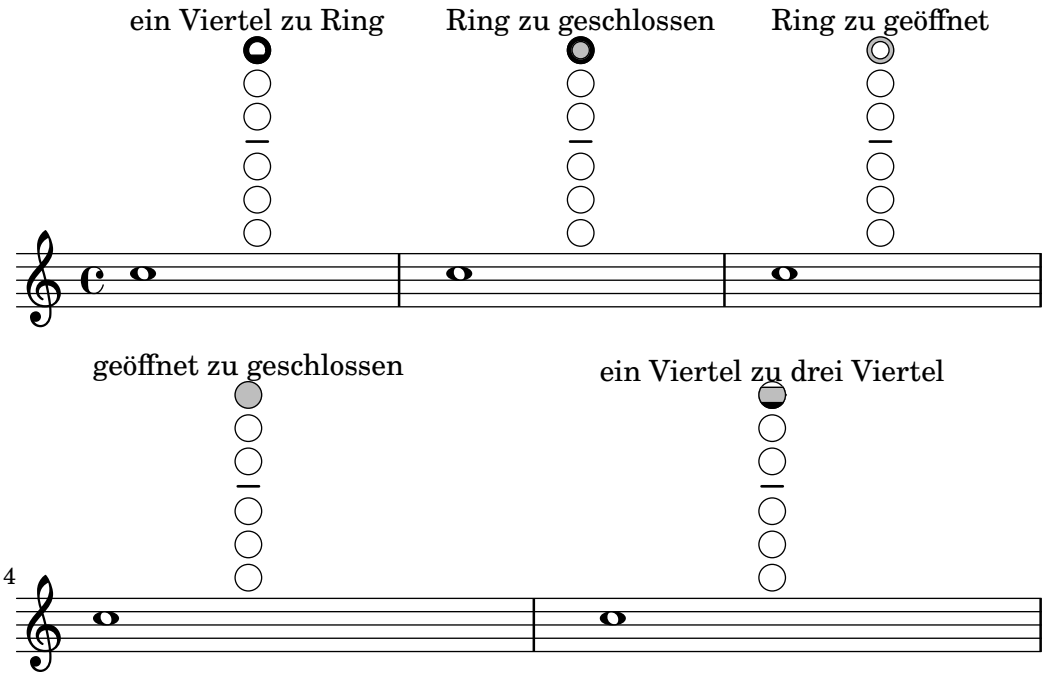
c''1^\markup {
  \center-column {
    "Ring zu geschlossen"
    \woodwind-diagram #'flute #'((cc . (oneTR))
                                 (lh . ())
                                 (rh . ()))
  }
}

c''1^\markup {
  \center-column {
    "Ring zu geöffnet"
    \woodwind-diagram #'flute #'((cc . (oneRT))
                                 (lh . ())
                                 (rh . ()))
  }
}

c''1^\markup {
  \center-column {
    "geöffnet zu geschlossen"
    \woodwind-diagram #'flute #'((cc . (oneT))
                                 (lh . ())
```

```
                                (rh . ()))
  }
}

c''1^\markup {
  \center-column {
    "ein Viertel zu drei Viertel"
    \woodwind-diagram #'flute #'((cc . (one1qT3q))
                                 (lh . ())
                                 (rh . ()))
  }
}
```



Die Liste aller möglichen Löcher und Einstellungen eines bestimmten Instruments kann auf der Kommandozeile mit dem Befehl `#(print-keys-verbose 'flute)` oder in einer Log-Datei mit `#(print-keys-verbose 'flute (current-error-port))` angezeigt werden; durch diese Befehle wird der Notensatz nicht verändert.

Neue Diagramme können erstellt werden, hierzu benötigt man jedoch Scheme-Kenntnisse. Die Muster für die Diagramme befinden sich in den Dateien `scm/define-woodwind-diagrams.scm` und `scm/display-woodwind-diagrams.scm`.

Ausgewählte Schnipsel

Liste der Holzbläserdiagramme

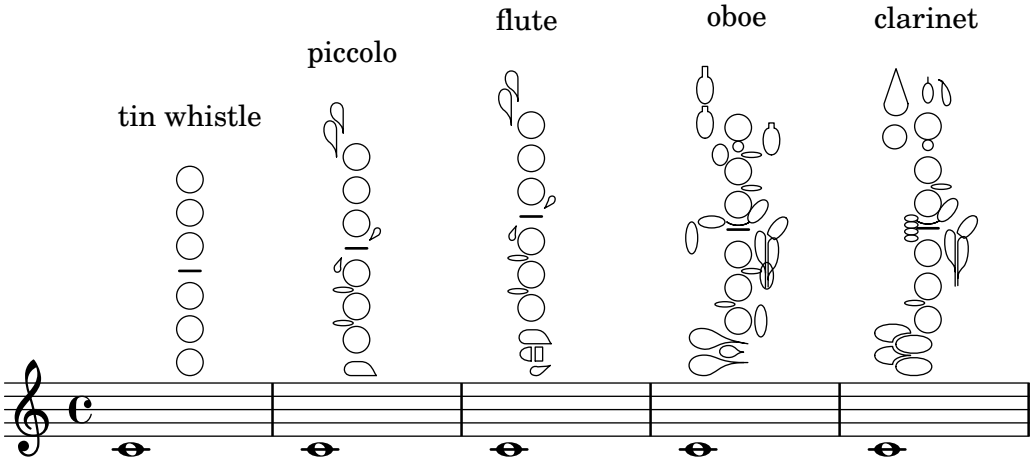
Folgende Noten zeige alle Holzbläserdiagramme, die für LilyPond definiert sind.

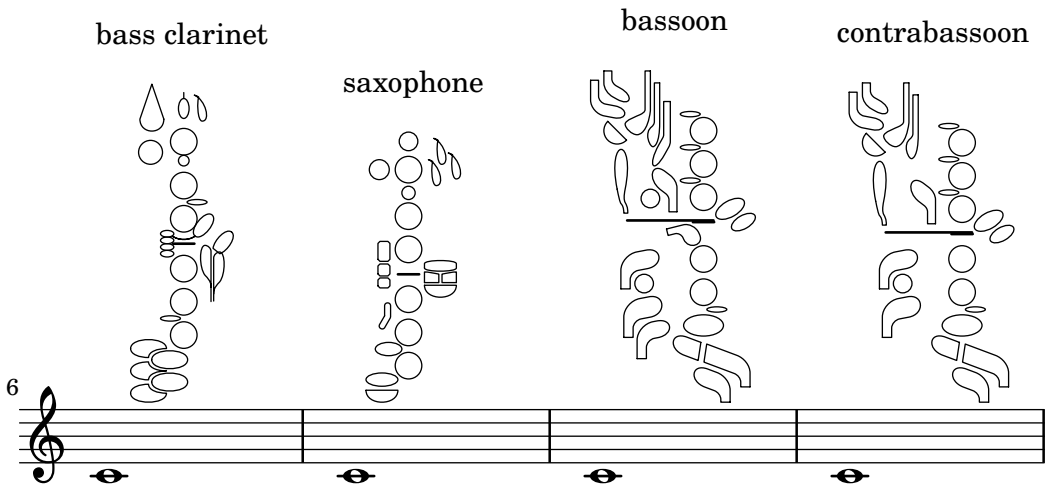
```
\relative c' {
  \textLengthOn
  c1^\markup \center-column { "tin whistle"
                              " "
                              \woodwind-diagram #'tin-whistle #'() }
  c1^\markup \center-column { "piccolo"
                              " "
```

```
                                \woodwind-diagram #'piccolo #'() }
    c1^\markup \center-column { "flute"
                                " "
                                \woodwind-diagram #'flute #'() }
    c1^\markup \center-column { "oboe"
                                " "
                                \woodwind-diagram #'oboe #'() }
    c1^\markup \center-column { "clarinet"
                                " "
                                \woodwind-diagram #'clarinet #'() }
    \break

    c1^\markup \center-column { "bass clarinet"
                                " "
                                \woodwind-diagram #'bass-clarinet #'() }
    c1^\markup \center-column { "saxophone"
                                " "
                                \woodwind-diagram #'saxophone #'() }
    c1^\markup \center-column { "bassoon"
                                " "
                                \woodwind-diagram #'bassoon #'() }
    c1^\markup \center-column { "contrabassoon"
                                " "
                                \woodwind-diagram #'contrabassoon #'() }
}

\paper {
  system-system-spacing.padding = 5
}
```



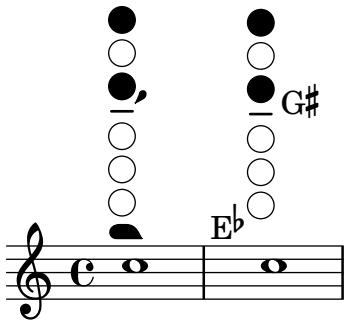


Graphische und Text-Holzbläserdiagramme

In vielen Fällen können die nicht in der mittleren Reihe befindlichen Löcher dargestellt werden, indem man die Lochbezeichnung oder graphische Zeichen benutzt.

```
\relative c'' {
  \textLengthOn
  c1^\markup
    \woodwind-diagram #'piccolo
                      #'((cc . (one three))
                         (lh . (gis))
                         (rh . (ees)))

  c^\markup
    \override #'(graphical . #f)
      \woodwind-diagram #'piccolo
                        #'((cc . (one three))
                           (lh . (gis))
                           (rh . (ees)))
}
```



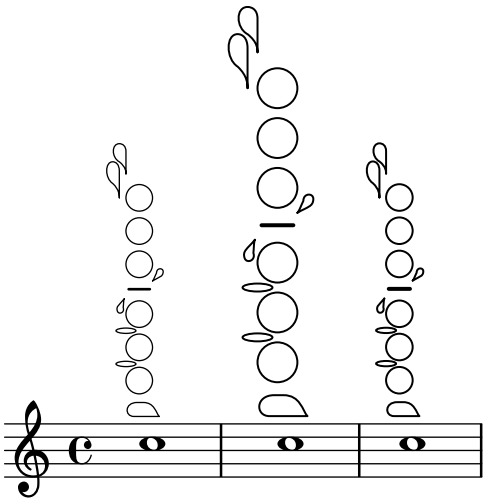
Größe von Holzbläserdiagrammen ändern

Die Größe und Dicke der Holzbläserdiagramme kann geändert werden.

```
\relative c'' {
  \textLengthOn
  c1^\markup
       \woodwind-diagram #'piccolo #'()

  c^\markup \override #'(size . 1.5)
```

```
        \woodwind-diagram #'piccolo #'()
    c^\markup \override #'(thickness . 0.15)
        \woodwind-diagram #'piccolo #'()
}
```
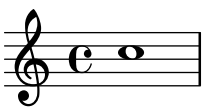


Liste der Löcher für Holzbläserdiagramme

Dieses Schnipsel erzeugt eine Liste aller möglichen Löcher und Locheinstellungen für Holzbläserdiagramme, wie sie in der Datei `scm/define-woodwind-diagrams.scm` definiert sind. Die Liste wird in der Log-Datei angezeigt, nicht in den Noten. Wenn Ausgabe auf der Kommandozeile gewünscht ist, muss `(current-error-port)` von den Befehlen weggelassen werden.

```
#(print-keys-verbose 'piccolo (current-error-port))
#(print-keys-verbose 'flute (current-error-port))
#(print-keys-verbose 'flute-b-extension (current-error-port))
#(print-keys-verbose 'tin-whistle (current-error-port))
#(print-keys-verbose 'oboe (current-error-port))
#(print-keys-verbose 'clarinet (current-error-port))
#(print-keys-verbose 'bass-clarinet (current-error-port))
#(print-keys-verbose 'low-bass-clarinet (current-error-port))
#(print-keys-verbose 'saxophone (current-error-port))
#(print-keys-verbose 'soprano-saxophone (current-error-port))
#(print-keys-verbose 'alto-saxophone (current-error-port))
#(print-keys-verbose 'tenor-saxophone (current-error-port))
#(print-keys-verbose 'baritone-saxophone (current-error-port))
#(print-keys-verbose 'bassoon (current-error-port))
#(print-keys-verbose 'contrabassoon (current-error-port))

\score {c''1}
```



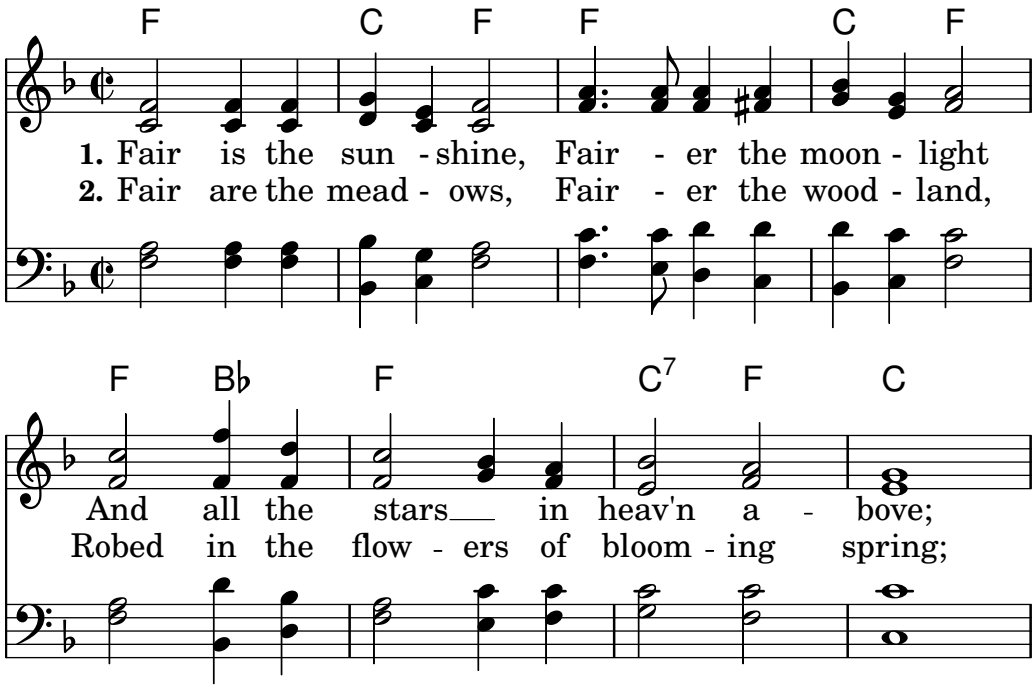
Siehe auch

Installierte Dateien: `scm/define-woodwind-diagrams.scm`, `scm/display-woodwind-diagrams.scm`.

Schnipsel: Abschnitt “Wind instruments” in *Schnipsel*.

Referenz der Interna: Abschnitt “TextScript” in *Referenz der Interna*, Abschnitt “instrument-specific-markup-interface” in *Referenz der Interna*.

15 Notation von Akkorden



Akkorde können entweder als normale Noten oder im Akkordmodus notiert werden; bei letztere Eingabemethode können unterschiedliche europäische Akkordbezeichnungsstile eingesetzt werden. Akkordbezeichnungen und Generalbass können auch angezeigt werden.

15.1 Akkord-Modus
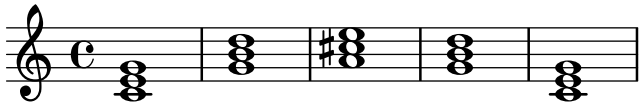
Im Akkordmodus (engl. „chord“) werden Akkorde anhand von einem Symbol der erwünschten Akkordstruktur notiert, anstatt dass die einzelnen Tonhöhen ausgeschrieben werden.

15.1.1 Überblick über den Akkord-Modus

Akkorde können als simultane Noten eingegeben werden, wie gezeigt in Abschnitt 5.1.1 [Noten mit Akkorden], Seite 160.

Akkorde können aber auch im Akkordmodus notiert werden. Das ist ein Eingabemodus, der sich an Akkordstrukturen traditioneller europäischer Musik und nicht an bestimmten einzelnen Tonhöhen orientiert. Er bietet sich an, wenn man es gewohnt ist, Akkordsymbole zur Beschreibung von Akkorden zu benutzen. Mehr Information zu unterschiedlichen Eingabemethoden findet sich in Abschnitt 35.1 [Eingabe-Modi], Seite 610.

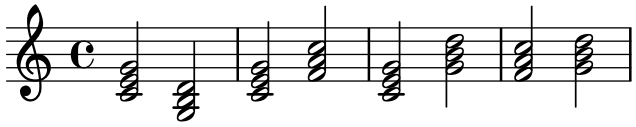
```
\chordmode { c1 g a g c }
```



Akkorde, die im Akkordmodus eingegeben werden, sind musikalische Elemente und können genauso wie Akkorde im Notenmodus transponiert werden. `\chordmode` ist absolut, und deshalb hat `\relative` keine Auswirkung auf die `\chordmode`-Abschnitte. Im Akkord-Modus ist jedoch die absolute Tonhöhe eine Oktave höher als im Notationsmodus.

Akkordmodus und Notenmodus können gemischt verwendet werden:

```
\relative {
  <c' e g>2 <g b d>
  \chordmode { c2 f }
  <c e g>2 <g' b d>
  \chordmode { f2 g }
}
```



Siehe auch

Glossar: Abschnitt “chord” in *Glossar*.

Notationsreferenz: Abschnitt 5.1.1 [Noten mit Akkorden], Seite 160, Abschnitt 35.1 [Eingabe-Modi], Seite 610.
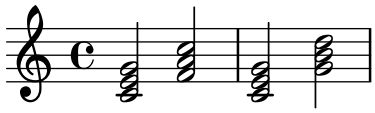
Schnipsel: Abschnitt “Chord notation” in *Schnipsel*.

Bekannte Probleme und Warnungen

Vordefinierte Abkürzung für Artikulationen und Ornamente können mit Noten im Akkordmodus nicht benutzt werden, siehe auch Abschnitt 3.1.1 [Artikulationszeichen und Verzierungen], Seite 120.
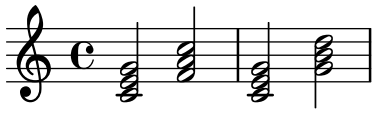
Wenn Akkord- und Notenmodus in linearer Musik abwechseln eingesetzt werden und der Akkordmodus am Anfang steht, erstellt der Notenmodus ein neues Notensystem:

```
\chordmode { c2 f }
<c e g>2 <g' b d>
```



Um dieses Verhalten zu verhindert, muss der `Staff`-Kontext explizit aufgerufen werden:

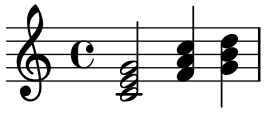
```
\new Staff {
  \chordmode { c2 f }
  <c e g>2  <g' b d>
}
```



15.1.2 Übliche Akkorde
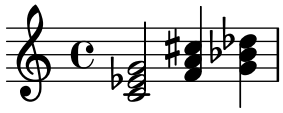
Ein Dreiklang wird mit seinem Grundton mit einer möglichen Dauer dahinter notiert:

```
\chordmode { c2 f4 g }
```



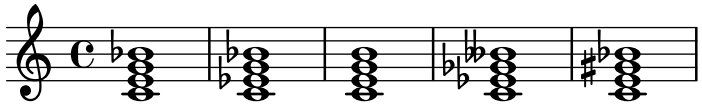
Moll- übermäßige und verminderte Dreiklänge werden notiert, indem : und ein Modifikator hinter der Dauer angegeben wird:

```
\chordmode { c2:m f4:aug g:dim }
```

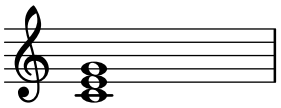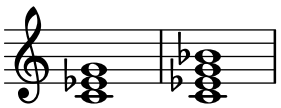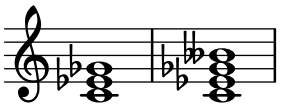
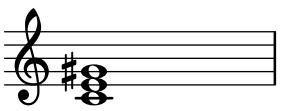
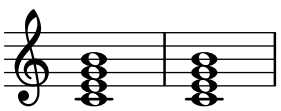


Septakkorde können erstellt werden:

```
\chordmode { c1:7 c:m7 c:maj7 c:dim7 c:aug7 }
```



Diese Tabelle zeigt die Funktion der Modifikatoren von Dreiklängen und Septakkorden. Die siebte Stufe wird standardmäßig als kleine Septime realisiert, sodass der Dominantseptakkord die Grundform des Septakkordes darstellt. Alle Alterationen sind relativ zur Dominantsept. Eine vollständigere Tabelle findet sich in Abschnitt A.2 [Übliche Akkord-Variablen], Seite 643.

Modifikator	**Funktion**	**Beispiel**
Kein	Standard: erzeugt einen Durdreiklang.	
m, m7	Mollakkord: Dieser Modifikator erniedrigt die dritte Stufe.	
dim, dim7	Verminderter Akkord: Dieser Modifikator erniedrigt die dritte, fünfte und (wenn vorhanden) die siebte Stufe.	
aug	Übermäßiger Akkord: Dieser Modifikator erhöht die fünfte Stufe.	
maj, maj7	Großer Septakkord: Dieser Modifikator fügt eine erhöhte siebe Stufe hinzu. `7` nach dem `maj` ist optional. NICHT benutzen, um einen Durdreiklang zu notieren.	

Siehe auch

Notationsreferenz: Abschnitt A.2 [Übliche Akkord-Variablen], Seite 643, Abschnitt 15.1.3 [Erweiterte und modifizierte Akkorde], Seite 400.

Schnipsel: Abschnitt “Chord notation” in *Schnipsel*.

Bekannte Probleme und Warnungen

Nur ein Qualitätsmodifikator sollte pro Akkord benutzt werden, meistens für die höchste Stufe des Akkordes. Akkorde mit weiteren Qualitätsmodifikatoren werden ohne Warnung oder Fehlermeldung gelesen, aber das Ergebnis ist nicht vorhersagbar. Akkorde, die nicht mit einem einzigen

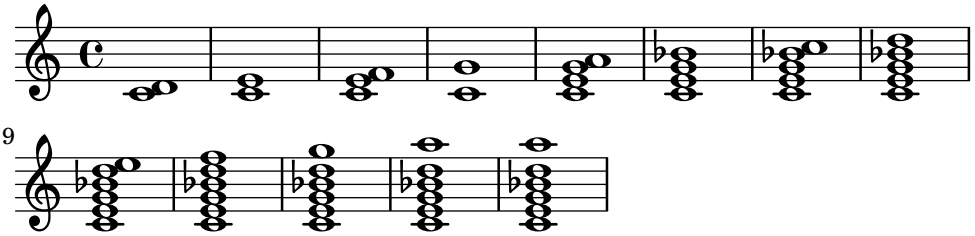
Qualitätsmodifikator erreicht werden können, sollten mit einzelnen Tonhöhen alteriert werden, wie beschrieben in Abschnitt 15.1.3 [Erweiterte und modifizierte Akkorde], Seite 400.

15.1.3 Erweiterte und modifizierte Akkorde

Akkordstrukturen können im Akkordmodus beliebig komplex konstruiert werden. Die Modifikatoren können benutzt werden, um den Akkord zu erweitern, bestimmte Stufen hinzuzufügen oder zu entfernen, Stufen zu erhöhen oder zu erniedrigen und Bassnoten hinzuzufügen bzw. Umkehrungen zu erzeugen.

Die erste Zahl, die auf den Doppelpunkt folgt, wird als „Bereich“ des Akkordes interpretiert: Terzen werden auf dem Grundton gestapelt, bis die angegebene Zahl (=Tonstufe) erreicht ist. Die siebte Stufe, die zu einem Akkord hinzugefügt wird, ist die kleine Septime, nicht die große. Wenn der Bereich keine Terz ist (also etwa 6), dann werden Terzen bis zur höchst möglichen Terz unter dem Bereich gestapelt, und der Endton des Bereichs wird hinzugefügt. Der größtmögliche Wert ist 13. Jeder größere Werte wird als 13 interpretiert.

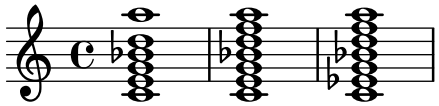
```
\chordmode {
  c1:2 c:3 c:4 c:5
  c1:6 c:7 c:8 c:9
  c1:10 c:11 c:12 c:13
  c1:14
}
```



Sowohl `c:5` als auch `c` erzeugen einen D-Dur-Dreiklang.
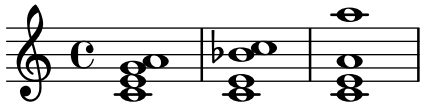
Da eine unveränderte 11 nicht gut klingt, wenn sie mit einer unveränderten 13 zusammenklingt, wird die 11 von einem `:13`-Akkord entfernt (es sei denn sie wird explizit verlangt).

```
\chordmode {
  c1:13 c:13.11 c:m13
}
```



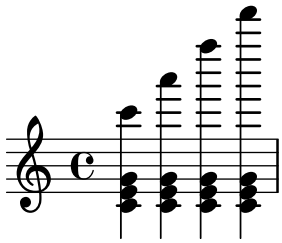
Kompliziertere Akkorde können auch konstruiert werden, indem einzelne Intervalle zu dem Grundton addiert werden. Diese Additionen werden nach dem Bereich notiert und mit Punkten voneinander getrennt. Die normale Septime, die zu einem Akkord hinzugefügt wird, ist die kleine Septime, nicht die große.

```
\chordmode {
  c1:3.5.6 c:3.7.8 c:3.6.13
}
```
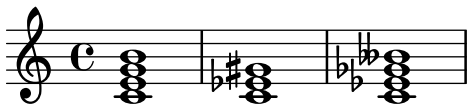


Hinzugefügte Stufen können beliebig groß sein:

```
\chordmode {
  c4:3.5.15 c:3.5.20 c:3.5.25 c:3.5.30
}
```



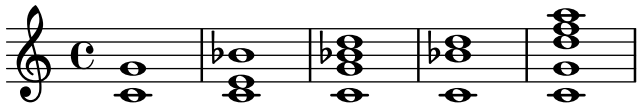
Einzelne Stufen können mit - oder + vergrößert oder verkleinert werden. Um eine Stufe zu verändert, die automatisch in den Akkord aufgenommen wurde, kann sie in veränderter Form nach dem Bereich hinzugefügt werden.

```
\chordmode {
  c1:7+ c:5+.3- c:3-.5-.7-
}
```



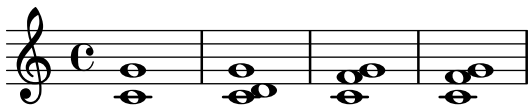
Zu entfernende Töne werden mit der gleichen Methode notiert, allerdings mit einem Dach (^) vor der Sequenz, die nicht erscheinen soll. Sie müssen nach den zu addierenden Tönen notiert werden. Die einzelnen zu entfernenden Töne werden mit Punkten getrennt.

```
\chordmode {
  c1^3 c:7^5 c:9^3 c:9^3.5 c:13.11^3.7
}
```



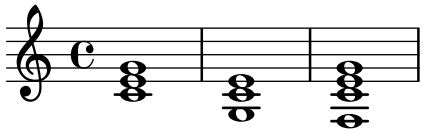
Sekund- und Quartakkorde können mit dem Modifikator sus notiert werden. Hiermit wird die dritte Stufe aus dem Akkord entfernt. Mit einer anschließenden 2 wird die zweite, mit einer 4 die vierte Stufe hinzugefügt. sus entspricht ^3 und sus4 ist gleich .4^3.

```
\chordmode {
  c1:5 c:sus2 c:sus4 c:5.4
}
```



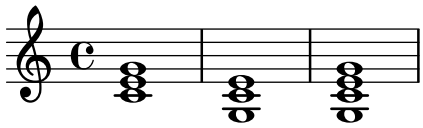
Eine Umkehrung (ein Ton des Akkordes wird unter den Grundton gesetzt) sowie auch zusätzliche Bassnoten können mit dem Schrägstrich (/) markiert werden:

```
\chordmode {
  c1 c/g c/f
}
```



Eine Bassnote, die zum Akkord hinzugehört, kann hinzugefügt werden, anstatt dass sie aus dem Akkord entnommen wird, indem noch ein Plus zwischen den Schrägstrich und die Tonhöhe gesetzt wird:

```
\chordmode {
  c1 c/g c/+g
}
```



Akkordmodifikatoren, die benutzt werden können, um eine große Anzahl an Standardakkorden zu erzeugen, werden gezeigt in Abschnitt A.2 [Übliche Akkord-Variablen], Seite 643.

Siehe auch
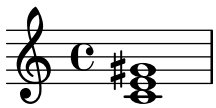
Notationsreferenz: Abschnitt A.2 [Übliche Akkord-Variablen], Seite 643.

Schnipsel: Abschnitt “Chord notation” in *Schnipsel*.

Bekannte Probleme und Warnungen

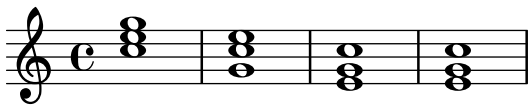
Jede Stufe kann nur einmal in einem Akkord vorkommen. Im folgenden Beispiel wird ein erweiterter Akkord erstellt, weil 5+ zuletzt gelesen wird.

```
\chordmode { c1:3.5.5-.5+ }
```



Nur die zweite Umkehrung kann erstellt werden, indem eine Bassnote hinzugefügt wird. Die erste Umkehrung erfordert, dass der Grundton des Akkordes geändert wird.

```
\chordmode {
  c'1: c':/g e:6-3-^5 e:m6-^5
}
```



15.2 Akkorde anzeigen

Akkorde können zusätzlich zur üblichen Notation als Töne auf einem Notensystem auch mit einem Akkordsymbol gesetzt werden.

15.2.1 Akkordbezeichnungen drucken

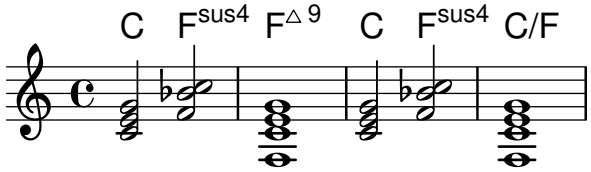
Akkordsymbole anstelle der Noten werde im ChordNames-Kontext notiert.

```
\new ChordNames {
  \chordmode {
    c2 f4. g8
  }
}
```



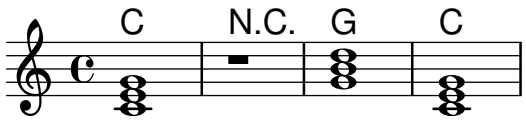
Die Akkorde können entweder als simultane Noten oder unter Einsatz des Akkordmodus (chordmode) notiert werden. Der angezeigte Akkord ist der gleiche, es sei denn, Umkehrungen oder zusätzliche Basstöne werden notiert:

```
<<
  \new ChordNames {
    <c e g>2  <f bes c>
    <f c' e g>1
    \chordmode {
      c2 f:sus4 c1:/f
    }
  }
  {
    <c e g>2  <f bes c>
    <f, c' e g>1
    \chordmode {
      c2 f:sus4 c1:/f
    }
  }
>>
```



Pausen, die in einem ChordNames-Kontext notiert werden, werden mit der noChordSymbol-Beschriftung darstellt.

```
<<
  \new ChordNames \chordmode {
    c1
    r1
    g1
    c1
  }
  \chordmode {
    c1
    r1
    g1
    c1
  }
>>
```



`\chords { ... }` ist eine Kurznotation für die Bezeichnung `\new ChordNames { \chordmode { ... } }`.

```
\chords {
  c2 f4.:m g8:maj7
}
```

C Fm G△

```
\new ChordNames {
  \chordmode {
    c2 f4.:m g8:maj7
  }
}
```

C Fm G△

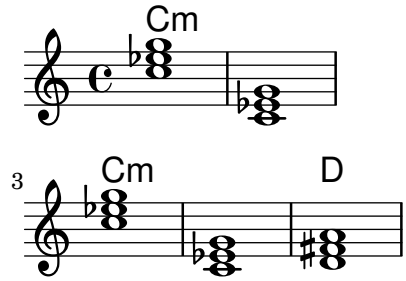
Ausgewählte Schnipsel

Akkordsymbole bei Wechsel anzeigen

Akkordsymbole können so eingestellt werden, dass sie nur zu Beginn der Zeile und bei Akkordwechseln angezeigt werden.

```
harmonies = \chordmode {
  c'1:m c:m \break
  c'1:m c:m d
}

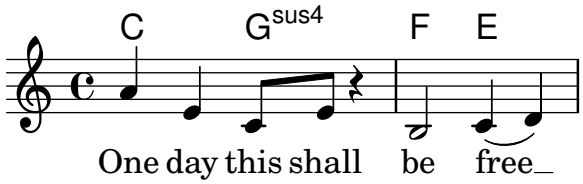
<<
  \new ChordNames {
    \set chordChanges = ##t
    \harmonies
  }
  \new Staff {
    \harmonies
  }
>>
```



Ein einfaches Liedblatt

Ein Liedblatt besteht aus Akkordbezeichnungen, einer Melodie und dem Liedtext:

```
<<
  \chords { c2 g:sus4 f e }
  \new Staff \relative c'' {
    a4 e c8 e r4
    b2 c4( d)
  }
  \addlyrics { One day this shall be free __ }
>>
```



Siehe auch

Glossar: Abschnitt “chord” in *Glossar*.

Notationsreferenz: Abschnitt 5.2.5 [Musik parallel notieren], Seite 179.

Schnipsel: Abschnitt “Chord notation” in *Schnipsel*.

Referenz der Interna: Abschnitt “ChordNames” in *Referenz der Interna*, Abschnitt “ChordName” in *Referenz der Interna*, Abschnitt “Chord_name_engraver” in *Referenz der Interna*, Abschnitt “Volta_engraver” in *Referenz der Interna*, Abschnitt “Bar_engraver” in *Referenz der Interna*.

Bekannte Probleme und Warnungen
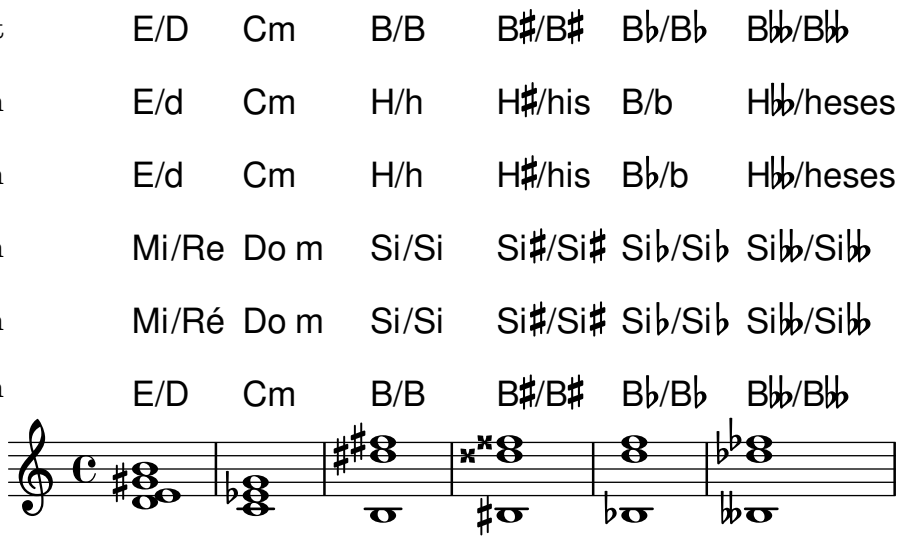
Akkorde, die Umkehrungen oder zusätzliche Basstöne beinhalten, werden nicht richtig bezeichnet, wenn sie im Notenmodus notiert werden.

15.2.2 Akkordbezeichnungen anpassen

Es gibt kein allein gültiges System zur Benennung von Akkorden. Unterschiedliche Musiktraditionen benutzen unterschiedliche Bezeichnungen für die gleichen Akkorde. Es gibt zusätzlich auch unterschiedliche Symbole, die für den gleichen Akkord angezeigt werden können. Die Bezeichnungen und dargestellten Symbole können angepasst werden.

Die Standardeinstellungen für die Symbole entsprechen den Konventionen im Jazz, wie sie von Klaus Ignatzek (siehe Abschnitt “Literatur” in *Aufsatz*). vorgeschlagen wurden. Das Benennungssystem für die Akkorde kann verändert werden, wie weiter unten gezeigt wird. Ein alternatives Notationssystem für Jazzakkorde ist auch erhältlich. Die Ignatzek und die alternative Jazznotation finden sich in der Tabelle in Abschnitt A.1 [Liste der Akkordbezeichnungen], Seite 643.

Zusätzlich zu den unterschiedlichen Bezeichnungssystemen werden unterschiedliche Notenbezeichnungen für die Grundtöne. Die vordefinierten Befehle `\germanChords`, `\semiGermanChords`, `\italianChords` und `\frenchChords` setzen diese Variablen. Die Auswirkungen werden im nächsten Beispiel gezeigt.



Deutsche Liederbücher zeigen Mollakkorde oft durch die Verwendung von Kleinbuchstaben an, ohne die Endung *m*. Dieses Verhalten kann erreicht werden, indem man die `chordNameLowercaseMinor`-Eigenschaft setzt:

```
\chords {
  \set chordNameLowercaseMinor = ##t
  c2 d:m e:m f
}
```


Wenn keine der definierten Einstellungen zum gewünschten Ergebnis führt, kann die Anzeige des Akkordsymbols durch die folgenden Eigenschaften verändert werden:

`chordRootNamer`
: Das Akkordsymbol wird normalerweise als Buchstabe des Grundtons mit optionaler Alteration dargestellt. Die Interpretation einer Tonhöhe als Buchstabe wird von der `chordRootNamer`-Funktion übernommen. Besondere Bezeichnungen, wie etwa im Deutschen H für einen H-Dur-Akkord (und nicht „B“ wie im Englischen), können durch Hinzufügen einer neuen Funktion zu dieser Eigenschaft erstellt werden.

`majorSevenSymbol`
: Mit dieser Eigenschaft wird das Aussehen der Notation für die große Septime (7) bestimmt. Vordefiniert sind die Optionen `whiteTriangleMarkup` und `blackTriangleMarkup`.

`additionalPitchPrefix`
: Wenn die Akkordbezeichnung zusätzliche Tonhöhen enthält, können sie optional mit einer Textvorsilbe versehen werden. Als Standard ist keine Vorsilbe eingestellt, sodass die visuelle Einfachheit gewahrt bleibt, aber für eine kleine Anzahl zusätzlicher Töne kann diese Option effektiv sein.

```
\new ChordNames {
  <c e g d'>    % add9
  \set additionalPitchPrefix = "add"
  <c e g d'>    % add9
}
```

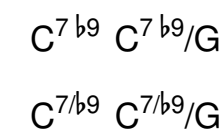
`chordNoteNamer`

Wenn das Akkordsymbol zusätzliche Tonhöhen enthält, die nicht den Grundton darstellen (etwa eine zusätzliche Bassnote), wird diese Funktion eingesetzt, um die zusätzliche Tonhöhe auszugeben. In den Standardeinstellungen wird die Tonhöhe mit der `chordRootNamer`-Funktion gesetzt. Die `chordNoteNamer`-Eigenschaft hingegen kann dieses Verhalten verändern und etwa den Basston etwa als Kleinbuchstaben darstellen.

`chordNameSeparator`

Verschiedene Teile eines Akkordsymboles werden normalerweise durch eine kleinen Freiraum angezeigt. Indem `chordNameSeparator` ein Wert zugewiesen wird, kann ein beliebiges Zeichen für den Trenner benutzt werden. Das hat keine Auswirkung auf den Trenner zwischen einem Akkord und seiner Bassnote. Um diesen einzustellen muss `slashChordSeparator` benutzt werden.

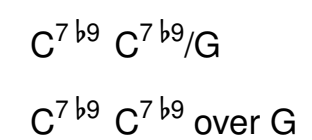
```
\chords {
  c4:7.9- c:7.9-/g
  \set chordNameSeparator = \markup { "/" }
  \break
  c4:7.9- c:7.9-/g
}
```



`slashChordSeparator`

Akkorde können auch aufbauend auf einer anderen Bassnote als dem üblichen Grundton gespielt werden. Sie werden Umkehrungen genannt, sind aber auch als Slash-Akkorde bekannt, weil ihre übliche Notation aus dem Akkordsymbol, einem Schrägstrich und dem Basston besteht. Deshalb ist der Standardwert von `slashChordSeparator` ein Schrägstrich, aber jedes andere Beschriftungszeichen kann auch eingestellt werden.

```
\chords {
  c4:7.9- c:7.9-/g
  \set slashChordSeparator = \markup { " over " }
  \break
  c4:7.9- c:7.9-/g
}
```



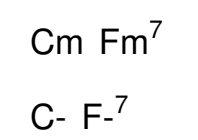
`chordNameExceptions`

Diese Funktion ist eine Liste mit Paaren. Das erste Objekt eines Paares ist eine Anzahl von Tonhöhen, die die Stufen eines Akkordes definieren. Das zweite Objekt ist eine Beschriftung, die nach `chordRootNamer` ausgegeben wird, um das Akkordsymbol zu erstellen.

`minorChordModifier`

Moll-Akkorde werden oft durch ein nachgestelltes „m“ rechts des Akkordgrundtons angezeigt. Manche bevorzugen aber andere Zeichen, wie etwa ein Minus-Zeichen.

```
\chords {
  c4:min f:min7
  \set minorChordModifier = \markup { "-" }
  \break
  c4:min f:min7
}
```



chordPrefixSpacer
: Das Zeichen für Moll-Akkorde, durch `minorChordModifier` erstellt, wird normalerweise direkt hinter dem Akkordbuchstaben gesetzt. Mit der Eigenschaft `chordPrefixSpacer` kann ein Abstand(halter) zwischen den Buchstaben und das Zeichen gesetzt werden. Der Abstandhalter wird nicht verwendet, wenn der Grundton erhöht oder erniedrigt ist.

Vordefinierte Befehle

`\whiteTriangleMarkup`, `\blackTriangleMarkup`, `\germanChords`, `\semiGermanChords`, `\italianChords`, `\frenchChords`.

Ausgewählte Schnipsel

Akkordsymbolausnahmen

Die Eigenschaft `chordNameExceptions` kann benutzt werden, um eine Liste an besonderen Notationen für bestimmte Akkorde zu speichern.

```
% Step 1: Define music with chords and markup for maj9 and 6(add9).
chExceptionMusic = {
  <c e g b d'>-\markup { \super "maj9" }
  <c e g a d'>-\markup { \super "6(add9)" }
}

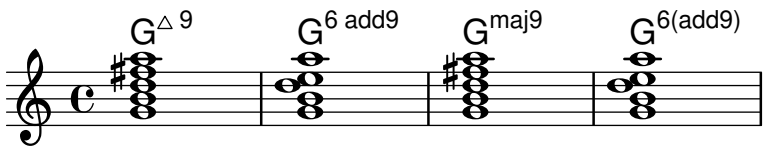
% Step 2: Create extended exception list.
chExceptions =
#(append (sequential-music-to-chord-exceptions chExceptionMusic #t)
         ignatzekExceptions)

theMusic = \chordmode {
  g1:maj9 g1:6.9
  % Step 3: Register extended exception list.
  \set chordNameExceptions = #chExceptions
  g1:maj9 g1:6.9
}

<<
  \new ChordNames \theMusic
  \new Voice \theMusic
>>

\layout {
  line-width = 10\cm
```

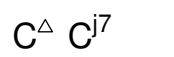
```
  ragged-right = ##f
}
```



Akkordbezeichnung maj7

Das Aussehen des großen Septakkords kann mit `majorSevenSymbol` verändert werden.

```
\chords {
  c:7+
  \set majorSevenSymbol = \markup { j7 }
  c:7+
}
```

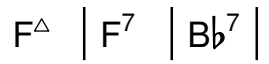


Tatkstriche in einen ChordNames-Kontext hinzufügen

Um Taktstriche in einem `ChordNames`-Kontext anzeigen zu lassen, muss der `Bar_engraver` hinzugefügt werden.

```
\new ChordNames \with {
  \override BarLine.bar-extent = #'(-1 . 3)
  \consists "Bar_engraver"
}

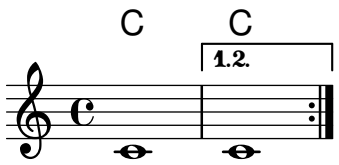
\chordmode {
  f1:maj7 f:7 bes:7
}
```



Wiederholungs-(Volta-)Klammern unterhalb der Akkordsymbole

Indem man den `Volta_engraver` zu dem entsprechenden Notensystem hinzufügt, können Wiederholungsklammern unterhalb der Akkorde gesetzt werden.

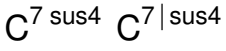
```
\score {
  <<
    \chords { c1 c1 }
    \new Staff \with { \consists "Volta_engraver" }
    {
      \repeat volta 2 { c'1 \alternative { c' } }
    }
  >>
  \layout {
    \context {
      \Score
      \remove "Volta_engraver"
    }
  }
}
```



Akkordsymboltrenner verändern

Der Trenner zwischen unterschiedlichen Teilen eines Akkordsymbols kann beliebiger Text sein.

```
\chords {
  c:7sus4
  \set chordNameSeparator = \markup { \typewriter | }
  c:7sus4
}
```



Siehe auch

Notationsreferenz: Abschnitt A.1 [Liste der Akkordbezeichnungen], Seite 643, Abschnitt A.2 [Übliche Akkord-Variablen], Seite 643.

Aufsatz über den automatischen Musiksatz: Abschnitt “Literatur” in *Aufsatz*.

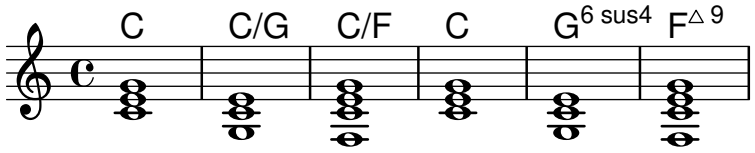
Installierte Dateien: `scm/chords-ignatzek-names.scm`, `scm/chord-entry.scm`, `ly/chord-modifiers-init.ly`.

Schnipsel: Abschnitt “Chord notation” in *Schnipsel*.
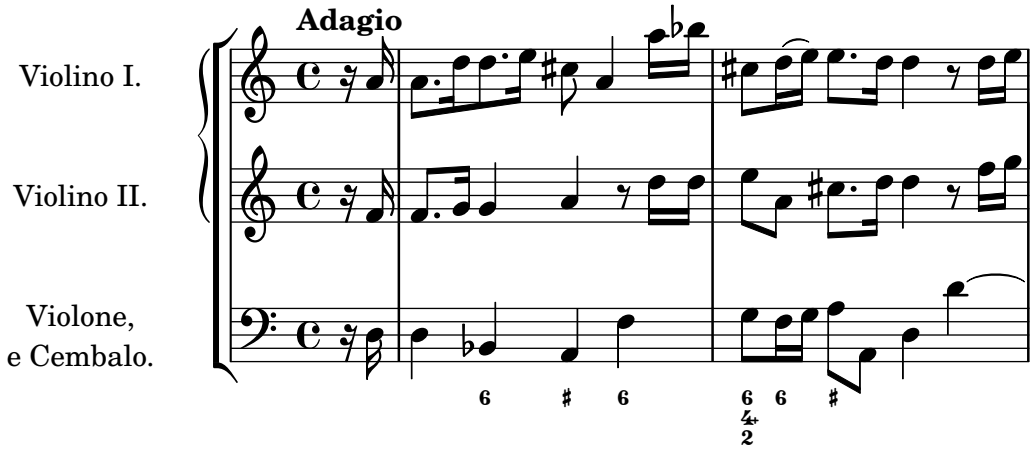
Bekannte Probleme und Warnungen

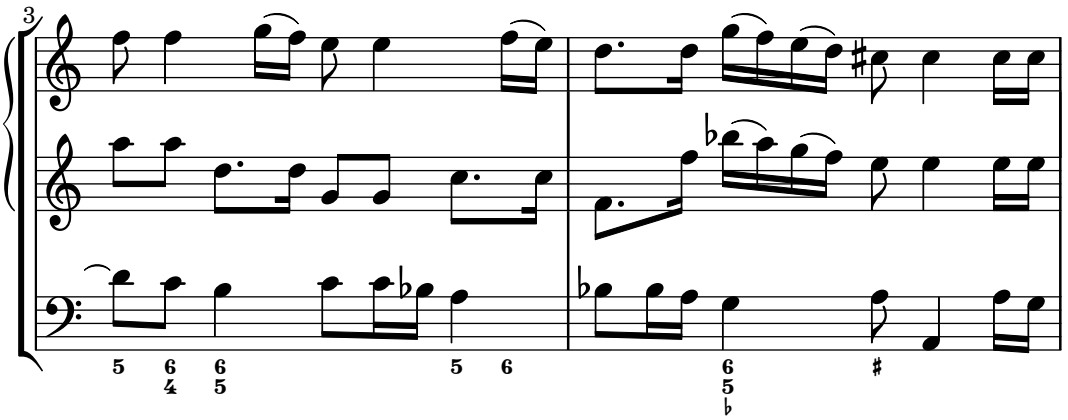
Akkordsymbole werden von den Tonhöhenbezeichnungen innerhalb des Akkordes und der Information über die Akkordstruktur, die innerhalb von `\chordmode` notiert wurde, bestimmt. Wenn der direkte Notenmodus benutzt wird, stammen unerwünschte Bezeichnungen daher, dass Umkehrungen und zusätzliche Bassnoten nicht richtig interpretiert werden.

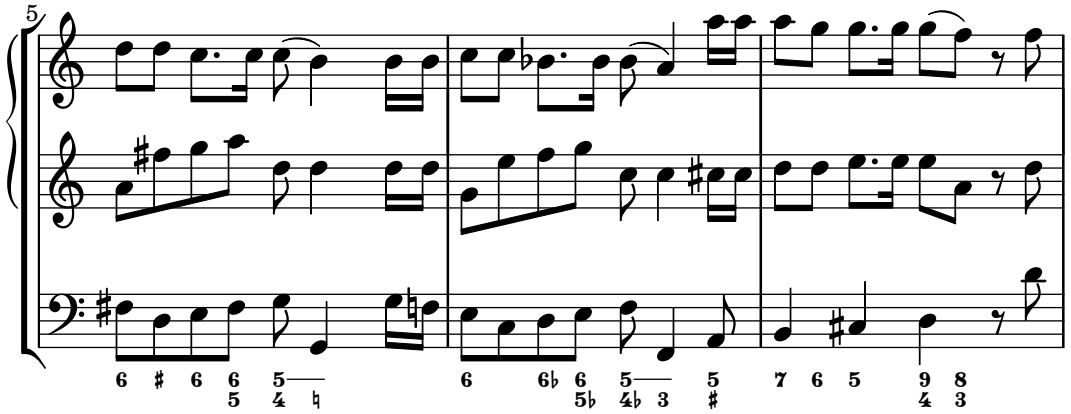
```
myChords = \relative c' {
  \chordmode { c1 c/g c/f }
  <c e g>1 <g c e> <f c' e g>
}
<<
  \new ChordNames { \myChords }
  \new Staff { \myChords }
>>
```



15.3 Generalbass





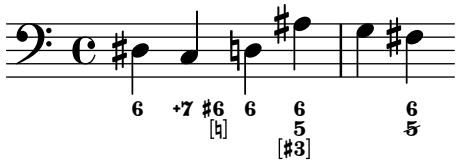


Generalbassnotation kann dargestellt werden.

15.3.1 Grundlagen des Bezifferten Basses

LilyPond stellt Unterstützung für Generalbassnotation, auch als Basso Continuo bezeichnet, zur Verfügung.

```
<<
  \new Voice { \clef bass dis4 c d ais g fis}
  \new FiguredBass {
    \figuremode {
      < 6 >4 < 7\+ >8 < 6+ [_!] >
      < 6 >4 <6 5 [3+] >
      < _ >4 < 6 5/>4
    }
  }
>>
```



Die Unterstützung für Generalbass besteht aus zwei Teilen: Es gibt einen Eingabe-Modus, aktiviert durch den Befehl `\figuremode`, in dem Ziffern für den Bass als Nummern eingegeben werden können, und einen Kontext `FiguredBass`, der dafür sorgt, dass die entsprechenden `BassFigure`-Objekte auch erstellt werden. Generalbass kann auch in einem `Staff`-Kontext dargestellt werden.

`\figures{ ... }` ist eine Kurznotation für `\new FiguredBass { \figuremode { ... } }`.

Auch wenn die Unterstützung für Generalbass auf den ersten Blick wie die Akkordunterstützung ausschauen mag, ist sie sehr viel einfacher. `\figuremode` speichert einfach die Zahlen und der `FiguredBass`-Kontext setzt sie in der Form, wie sie notiert wurden. Sie werden nicht in Tonhöhen umgewandelt.

Siehe auch

Glossar: Abschnitt “figured bass” in *Glossar*.

Schnipsel: Abschnitt “Chord notation” in *Schnipsel*.

15.3.2 Eingabe des Generalbass'

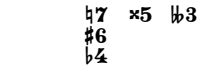
`\figuremode` (Zahlenmodus) wird benutzt, um den Eingabemodus auf den Zahlenmodus umzustellen. Mehr Information zu unterschiedlichen Eingabemodi findet sich in Abschnitt 35.1 [Eingabe-Modi], Seite 610.

Im Zahlenmodus wird eine Gruppe von Bassziffern mit den Zeichen < und > begrenzt. Die Dauer wird nach dem >-Zeichen eingegeben.

```
\new FiguredBass {
  \figuremode {
    <6 4>2
  }
}
```


Versetzungszeichen (inklusive Auflösungszeichen) können hinzugefügt werden:

```
\figures {
  <7! 6+ 4-> <5++> <3-->
}
```

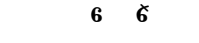


Übermäßige und verminderte Stufen können dargestellt werden:

```
\figures {
  <6\+ 5/> <7/>
}
```


Ein Schrägstrich von links nach rechts (üblicherweise für erhöhte Sexten benutzt) kann erstellt werden:

```
\figures {
  <6> <6\\>
}
```



Vertikaler Platz und Klammern können zu den Zahlen hinzugefügt werden:

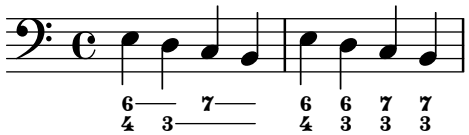
```
\figures {
  <[12 _!] 8 [6  4]>
}
```


Beliebiger Text kann als Zahl notiert werden:

```
\figures {
  <\markup { \tiny \number 6 \super (1) } 5>
}
```
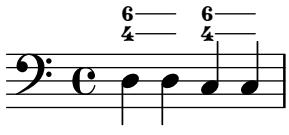

Es ist auch möglich, Fortsetzungslinien für wiederholte Ziffern zu benutzen.

```
<<
  {
    \clef bass
    e4 d c b,
    e4 d c b,
  }
  \figures {
    \bassFigureExtendersOn
    <6 4>4 <6 3> <7 3> <7 3>
    \bassFigureExtendersOff
    <6 4>4 <6 3> <7 3> <7 3>
  }
>>
```

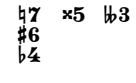


In diesem Fall werden wiederholte Ziffern immer durch eine Linie ersetzt, es sei denn, die Linie wird explizit beendet.

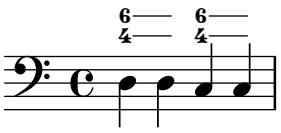
```
<<
  \figures {
    \bassFigureExtendersOn
    <6 4>4 <6 4> <6\! 4\!> <6 4>
  }
  {
    \clef bass
    d4 d c c
  }
>>
```



Die folgende Tabelle zeigt die vorhandenen Zahlenmodifikatoren:

Modifier	**Purpose**	**Example**
+, -, !	Accidentals	
\+, /	Augmented and diminished steps	
\\	Raised sixth step	

\! End of continuation line



Vordefinierte Befehle

`\bassFigureExtendersOn`, `\bassFigureExtendersOff`.

Ausgewählte Schnipsel

Positionen von Generalbass-Alterationszeichen verändern

Versetzungszeichen und Pluszeichen können vor oder nach den Ziffern erscheinen, je nach den Einstellungen der `figuredBassAlterationDirection` und `figuredBassPlusDirection`-Eigenschaften.

```
#(set-global-staff-size 26)

\figures {
  <5\+> <5+ 4\+> <6 4- 2\+> r
  \set figuredBassAlterationDirection = #RIGHT
  <5\+> <5+ 4\+> <6 4- 2\+> r
  \set figuredBassPlusDirection = #RIGHT
  <5\+> <5+ 4\+> <6 4- 2\+> r
  \set figuredBassAlterationDirection = #LEFT
  <5\+> <5+ 4\+> <6 4- 2\+> r
}
```



Siehe auch

Schnipsel: Abschnitt “Chord notation” in *Schnipsel*.

Referenz der Interna: Abschnitt “BassFigure” in *Referenz der Interna*, Abschnitt “BassFigureAlignment” in *Referenz der Interna*, Abschnitt “BassFigureLine” in *Referenz der Interna*, Abschnitt “BassFigureBracket” in *Referenz der Interna*, Abschnitt “BassFigureContinuation” in *Referenz der Interna*, Abschnitt “FiguredBass” in *Referenz der Interna*.
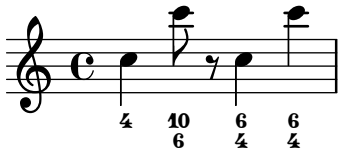
15.3.3 Generalbass anzeigen

Generalbass kann mit dem `FiguredBass`-Kontext, aber auch in den meisten anderen `Staff`-Kontexten dargestellt werden.

Wenn die Ziffern im `FiguredBass`-Kontext dargestellt werden, ist die vertikale Position der Ziffern unabhängig von den Noten des parallelen Systems.

```
<<
  \relative {
    c''4 c'8 r8 c,4 c'
  }
  \new FiguredBass {
    \figuremode {
      <4>4 <10 6>8 s8
      <6 4>4 <6 4>
```

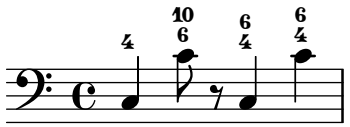
```
    }
  }
>>
```



In diesem Beispiel muss der FiguredBass-Kontext explizit erstellt werden, damit kein zusätzliches (leeres) Notensystem erstellt wird.

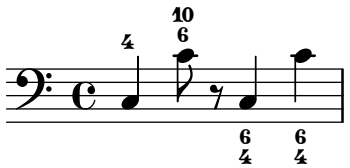
Bassziffern können auch direkt einem Notensystemkontext (Staff) hinzugefügt werden. In diesem Fall wird ihre vertikale Position automatisch bestimmt.

```
<<
  \new Staff = myStaff
  \figuremode {
    <4>4 <10 6>8 s8
    <6 4>4 <6 4>
  }
  %% Put notes on same Staff as figures
  \context Staff = myStaff
  {
    \clef bass
    c4 c'8 r8 c4 c'
  }
>>
```



Wenn Generalbass zu einem vorhandenen System hinzugefügt wird, ist es möglich, die Ziffern über oder unter dem System anzuzeigen:

```
<<
  \new Staff = myStaff
  \figuremode {
    <4>4 <10 6>8 s8
    \bassFigureStaffAlignmentDown
    <6 4>4 <6 4>
  }
  %% Put notes on same Staff as figures
  \context Staff = myStaff
  {
    \clef bass
    c4 c'8 r8 c4 c'
  }
>>
```



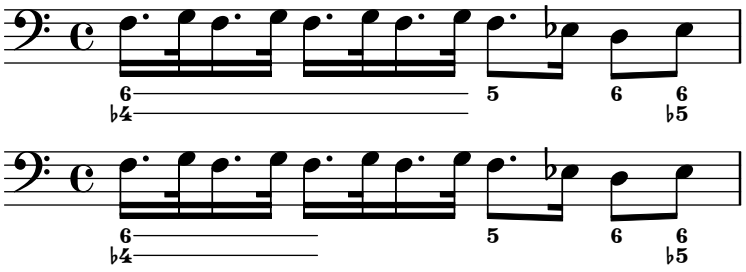
Schnipsel: Abschnitt “Chord notation” in *Schnipsel*.

Referenz der Interna: Abschnitt “BassFigure” in *Referenz der Interna*, Abschnitt “BassFigureAlignment” in *Referenz der Interna*, Abschnitt “BassFigureLine” in *Referenz der Interna*, Abschnitt “BassFigureBracket” in *Referenz der Interna*, Abschnitt “BassFigureContinuation” in *Referenz der Interna*, Abschnitt “FiguredBass” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Um sicherzugehen, dass die Fortsetzungslinien funktionieren, sollte der gleiche Rhythmus für die Bassfiguren und die eigentlichen Noten der Bassstimme benutzt werden.

```
<<
  {
    \clef bass
    \repeat unfold 4 { f16. g32 } f8. es16 d8 es
  }
  \figures {
    \bassFigureExtendersOn
    % The extenders are correct here, with the same rhythm as the bass
    \repeat unfold 4 { <6 4->16. <6 4->32 }
    <5>8. r16 <6>8 <6\! 5->
  }
>>
<<
  {
    \clef bass
    \repeat unfold 4 { f16. g32 } f8. es16 d8 es
  }
  \figures {
    \bassFigureExtendersOn
    % The extenders are incorrect here, even though the timing is the same
    <6 4->4 <6 4->4
    <5>8. r16 <6>8 <6\! 5->
  }
>>
```



16 Zeitgenössische Musik

Seit Anfang des 20. Jahrhunderts wurden die kompositorischen Stile und Kompositionstechniken sehr stark erweitert. Neue harmonische und rhythmische Entwicklungen, eine Erweiterung der verwendeten Tonhöhen und die Entwicklung eines großen Spektrums neuer instrumentaler Techniken wurden von einer parallelen Evolution der Notationstechnik begleitet. Die Absicht dieses Abschnittes ist es, Informationen und Hintergrundwissen zu bieten, der zur Notation zeitgenössischer Musik benötigt wird.

16.1 Tonhöhe und Harmonie in zeitgenössischer Musik

Dieser Abschnitt zeigt Lösungen zur Notation von zeitgenössischen Tonhöhen und Harmonien.

16.1.1 Verweise zu Tonhöhe und Harmonie in zeitgenössischer Musik

- Normale Vierteltonmusik wird behandelt in Abschnitt 1.1.4 [Notenbezeichnungen in anderen Sprachen], Seite 9.
- Nicht-Standardvorzeichen werden behandelt in Abschnitt 1.3.2 [Tonartbezeichnung], Seite 22.
- Contemporary practises in displaying accidentals are addressed in Abschnitt 1.3.5 [Automatische Versetzungszeichen], Seite 26.

16.1.2 Mikrotonale Notation

16.1.3 Zeitgenössische Tonartvorzeichnung und Harmonie

16.2 Zeitgenössische Notation von Rhythmen

Dieser Abschnitt erklärt Besonderheiten, die wichtig für die Notation von Rhythmus in zeitgenössischer Musik sind.

16.2.1 Verweise für zeitgenössische Benutzung von Rhythmus

- Zusammengesetzte Taktarten werden erklärt in #Abschnitt 2.3.1 [Taktangabe], Seite 65.
- Grundlegende polymetrische Notation ist erklärt in Abschnitt 2.3.5 [Polymetrische Notation], Seite 75.
- Gespreizte Balken sind erklärt in Abschnitt 2.4.4 [Gespreizte Balken], Seite 96.
- Mensurstrich-Taktstriche (zwischen den Systemen) finden sich erklärt in Abschnitt 6.1.2 [Systeme gruppieren], Seite 185.

16.2.2 N-tolen in zeitgenössischer Musik

16.2.3 Zeitgenössische Taktarten

16.2.4 Erweiterte polymetrische Notation

16.2.5 Balken in zeitgenössischer Musik

16.2.6 Taktstriche in zeitgenössischer Musik

16.3 Graphische Notation

16.4 Zeitgenössische Partiturtechniken

16.5 Neue Instrumententechniken

16.6 Leseliste und interessante Referenzpartituren

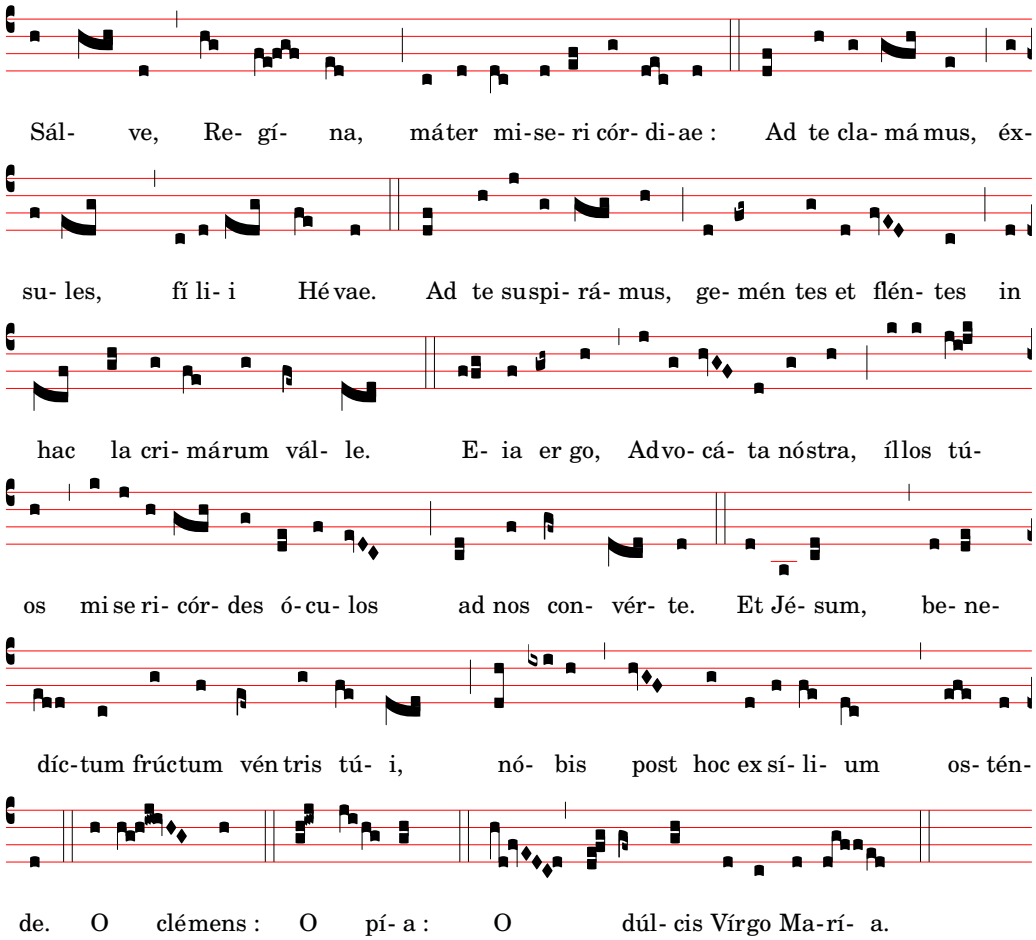
Dieser Abschnitt weist auf einige Bücher, Musikbeispiele und andere Resourcen hin, die relevant für die Notation zeitgenössischer Musik sind.

16.6.1 Bücher und Artikel über zeitgenössische Notation

- *Music Notation in the Twentieth Century: A Practical Guidebook* von Kurt Stone [W. W. Norton, 1980]
- *Music Notation: A Manual of Modern Practice* von Gardner Read [Taplinger, 1979]
- *Instrumentation and Orchestration* von Alfred Blatter [Schirmer, 2nd ed. 1997]

16.6.2 Partituren und Musikbeispiele

17 Notation von alter Musik



Unterstützung für Notation der Alten Musik enthält einige Eigenheiten der Mensuralnotation, der Notation des gregorianischen Chorals und der Kiever Quadratnotation. Diese Eigenheiten können eingestellt werden, indem man Stileigenschaften von graphischen Objekten wie Notenköpfen und Pausen verändert, oder indem man vordefinierte fertige Kontexte für mensurale oder Choralnotation einsetzt.

Viele graphische Objekte, wie Notenköpfe, Fähnchen, Versetzungszeichen, Taktarten und Pausen haben eine `style`-Eigenschaft, die verändert werden kann, um verschiedene Stile Alter Notation nachzuahmen. Siehe auch:

- Abschnitt 17.3.4 [Mensurale Notenköpfe], Seite 426,
- Abschnitt 17.3.7 [Mensurale Versetzungszeichen und Tonartbezeichnung], Seite 428,
- Abschnitt 17.3.6 [Mensurale Pausen], Seite 428,
- Abschnitt 17.3.2 [Mensurale Schlüssel], Seite 424,
- Abschnitt 17.4.2 [Gregorianische Schlüssel], Seite 432,
- Abschnitt 17.3.5 [Mensurale Fähnchen], Seite 427,
- Abschnitt 17.3.3 [Mensurale Taktartenbezeichnungen], Seite 425.

Ein paar notationelle Konzepte sind inbesondere für die Notation Alter Musik eingeführt worden:

- Abschnitt 17.2.3 [Custodes], Seite 423,
- Abschnitt 17.4.4 [Divisiones], Seite 433,
- Abschnitt 17.2.2 [Ligaturen], Seite 422.

Siehe auch

Glossar: Abschnitt “custos” in *Glossar*, Abschnitt “ligature” in *Glossar*, Abschnitt “mensural notation” in *Glossar*.

Notationsreferenz: Abschnitt 17.3.4 [Mensurale Notenköpfe], Seite 426, Abschnitt 17.3.7 [Mensurale Versetzungszeichen und Tonartbezeichnung], Seite 428, Abschnitt 17.3.6 [Mensurale Pausen], Seite 428, Abschnitt 17.3.2 [Mensurale Schlüssel], Seite 424, Abschnitt 17.3.5 [Mensurale Fähnchen], Seite 427, Abschnitt 17.3.3 [Mensurale Taktartenbezeichnungen], Seite 425, Abschnitt 17.4.2 [Gregorianische Schlüssel], Seite 432, Abschnitt 17.2.3 [Custodes], Seite 423, Abschnitt 17.4.4 [Divisiones], Seite 433, Abschnitt 17.2.2 [Ligaturen], Seite 422.

17.1 Überblick über die unterstützten Stile

Drei Stile sind vorhanden, um den gregorianischen Choral zu setzen:

- *Editio Vaticana* ist ein vollständiger Stil für den gregorianischen Choral, der stilistisch den Choralausgaben von Solsemes folgt. Hierbei handelt es sich um die offizielle Choralausgabe des Vatikans seit 1904. LilyPond unterstützt alle Notationszeichen, die in diesem Stil benutzt werden, inklusive Ligaturen, custodes und besondere Zeichen wie die Quilisma und den Oriscus.
- Der *Editio Medicaea*-Stil stellt bestimmte Eigenschaften zur Verfügung, die in den Medicaea (oder Ratisbona)-Editionen benutzt wurden. Dieser Stil war vor den Solesmes-Editionen in Benutzung. Der größte Unterschied von dem *Vaticana*-Stil sind die Schlüssel, die nach unten gerichtete Striche haben, und die Notenköpfe, die hier quadratisch und ebenmäßig geformt sind.
- Der *Hufnagel-* oder *gotische* Stil ahmt den Stil der Schreiber bestimmter Manuskripte aus dem Deutschland und Mitteleuropa des Mittelalters nach. Er ist nach der Form der wichtigsten Note (der *Virga*) benannt, die wie ein kleiner Nagel aussieht.

Drei Stile ahmen die Erscheinung von Renaissancehandschriften und -drucken der Mensuralmusik nach:

- Der *Mensural*-Stil versucht, den Stil von Handschriften nachzuahmen und hat recht kleine, rhombenförmige Notenköpfe und wie handgeschriebene Pausenzeichen.
- Der *Neomensural*-Stil ist eine modernisierte und stilisierte Version des erstens: Die Notenköpfe sind etwas breiter und die Pausen bestehen aus graden Linien. Dieser Stil ist besonders gut geeignet, um moderne Editionen der Mensuralmusik mit einem Incipit zu versehen.
- Der *Petrucci*-Stil ist nach Ottaviano Petrucci (1466-1539) benannt, dem ersten Drucker, der bewegliche Stempel benutzt hat, um musikalische Notation zu drucken (in seinem Buch *Harmonice musices odhecaton*, 1501). Dieser Stil setzt größere Notenköpfe ein als die anderen mensuralen Stile.

Baroque (Barockstil) und *Classical* (klassischer Stil) sind keine vollständigen Stile, sondern unterscheiden sich vom Standard nur in einigen Details: der Barockstil verändert bestimmte Notenköpfe, der klassische Stil die Form der Viertelpause.

Nur der Mensuralstil hat für alle Aspekte der Notation eine alternative Form. Die anderen Stile sind nur teilweise ausgeführt: die gregorianischen Stile haben keine Pausen oder Fähnchen, weil diese Zeichen im Choral nicht vorkommen, und der Petrucci-Stil hat keine eigenen Fähnchen und Versetzungszeichen.

Jedes Notationselement kann unabhängig von den anderen verändert werden, sodass man gut mensurale Fähnchen, Petrucci-Notenköpfe, klassische Pausen und Vatikana-Schlüssel nebeneinander benutzen kann, wenn das gewünscht ist.

Siehe auch

Glossary: Abschnitt “mensural notation” in *Glossar*, Abschnitt “flag” in *Glossar*.

17.2 Alte Notation – Allgemeines

17.2.1 Vordefinierte Umgebungen

Für den gregorianischen Choral und die Mensuralnotation gibt es vordefinierte Stimm- und Systemkontexte, die all die Notationszeichen auf Werte setzen, die diesem Stil angemessen sind. Wenn man mit den Werten zufrieden ist, kann man sofort mit der Notation beginnen, ohne sich um die Einzelheiten von tiefergreifenden Kontextanpassungen kümmern zu müssen. Die definierten Kontexte sind: `VaticanaVoice`, `VaticanaStaff`, `MensuralVoice` und `MensuralStaff`.

Siehe auch

- Abschnitt 17.4.1 [Gregorianische Gesangs-Kontexte], Seite 431,
- Abschnitt 17.3.1 [Mensural-Kontexte], Seite 424.

Siehe auch

Glossar: Abschnitt “mensural notation” in *Glossar*.
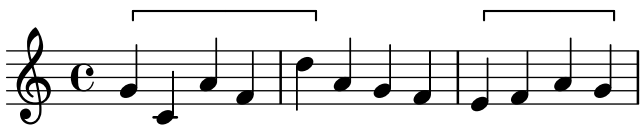
Notationsreferenz: Abschnitt 17.4.1 [Gregorianische Gesangs-Kontexte], Seite 431, Abschnitt 17.3.1 [Mensural-Kontexte], Seite 424.

17.2.2 Ligaturen

Eine Ligatur ist ein graphisches Symbol das wenigstens zwei unterschiedliche Noten darstellt. Ligaturen treten ursprünglich in Manuskripten des Gregorianischen Chorals auf, um auf- oder absteigende Notensequenzen zu notieren.

Ligaturen werden in LilyPond notiert, indem die dazugehörigen Noten zwischen `\[` und `\]` eingeschlossen werden. Einige Ligaturstile benötigen zusätzliche Syntax für eine bestimmte Ligatur. In der Standardeinstellung setzt der `LigatureBracket`-Engraver ganz einfach eckige Klammern über die Noten der Ligatur.

```
\transpose c c' {
  \[ g c a f d' \]
  a g f
  \[ e f a g \]
}
```



Es gibt zwei weitere Ligaturstile: Vaticana für den gregorianischen Choral und mensural für Mensuralnotation (wobei hier nur weiße Ligaturen unterstützt sind, und auch sie nur beschränkt). Um einen gestimmten Ligaturstil auszuwählen, muss der `Ligature_bracket_engraver` mit einem entsprechenden Ligatur-Engraver im Stimmenkontext ausgetauscht werden, wie erklärt in Abschnitt 17.3.9 [Weiße Mensuralligaturen], Seite 429, und Abschnitt 17.4.7 [Ligaturen der gregorianischen Quadratnotation], Seite 435.

Siehe auch

Glossar: Abschnitt “ligature” in *Glossar*.

Notationsreferenz: Abschnitt 17.3.9 [Weiße Mensuralligaturen], Seite 429, Abschnitt 17.4.7 [Ligaturen der gregorianischen Quadratnotation], Seite 435.

Bekannte Probleme und Warnungen

Ligaturen benötigen eine Platzaufteilung, die sich von der klassischen Notation deutlich unterscheidet. Das ist bisher sehr schlecht verwirklicht, sodass fast immer zu viel Platz zwischen Ligaturen ist und Zeilenumbrüche unbefriedigend ausfallen. Text lässt sich auch nicht richtig an Ligaturen ausrichten.

Versetzungszeichen dürfen nicht innerhalb von einer Ligatur gedruckt werden, sondern müssen gesammelt und vor der Ligatur ausgegeben werden.

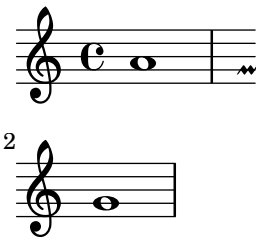
Die Syntax verwendet immer noch den verworfenen Infix-Stil (`\[ musik. Ausdr. \]`). Aus Gründen der Konsistenz soll dies geändert werden in den Postfix-Stil (`Note\[ ... Note\]`).

17.2.3 Custodes

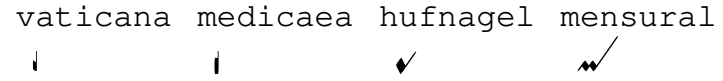
Ein *Custos* (Plural: *Custodes*; Lateinisch: „Weiser“) ist ein Symbol, das am Ende jedes Notensystems erscheint. Es nimmt die Tonhöhe der ersten Note der nächsten Zeile vorweg und hilft damit dem Vortragenden, die Zeilenwechsel während der Vorführung zu bewältigen.

Custodes wurden bis zum 17. Jahrhundert sehr häufig in der Musiknotation eingesetzt. Heute finden sie sich nur noch in einigen bestimmten Notationsformen, etwa modernen Editionen des Gregorianischen Chorals wie der *editio vaticana*. LilyPond stellt unterschiedliche Custos-Symbole für die unterschiedlichen Notationsstile zur Verfügung.

Damit Custodes angezeigt werden, muss ein `Custos_engraver` im `Staff`-Kontext gefordert werden. Der Aufruf folgt im Rahmen des Layout-Kontextes, wie das folgende Beispiel zeigt. Der Stil des Custos wird mit dem `override`-Befehl eingestellt, wie in dem folgenden Beispiel gezeigt:



Das Custos-Zeichen wird mit der `style`-Eigenschaft ausgewählt. Die unterstützten Stile sind: `vaticana`, `medicaea`, `hufnagel` und `mensural`. Sie werden im folgenden Fragment demonstriert.



Siehe auch

Music Glossary: Abschnitt “custos” in *Glossar*.

Schnipsel: Abschnitt “Ancient notation” in *Schnipsel*.

Referenz der Interna: Abschnitt “Custos” in *Referenz der Interna*.

17.2.4 Unterstützung für Generalbass

Es gibt beschränkte Unterstützung für Generalbassziffern aus der Barockzeit. Siehe hierzu Abschnitt 15.3 [Generalbass], Seite 411.

Siehe auch

Glossar: Abschnitt “figured bass” in *Glossar*.

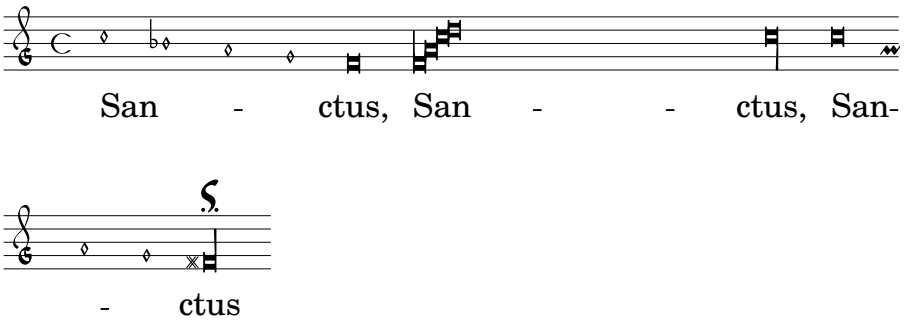
Notationsreferenz: Abschnitt 15.3 [Generalbass], Seite 411.

17.3 Mensurale Musik setzen

17.3.1 Mensural-Kontexte

Die vordefinierten Kontexte `MensuralVoice` und `MensuralStaff` können eingesetzt werden, um ein Stück in Mensuralnotation zu schreiben. Die Kontexte initialisieren alle relevanten Eigenschaften und graphischen Objekte, so dass unmittelbar mit der Notation begonnen werden kann. Siehe das folgende Beispiel:

```
\score {
  <<
    \new MensuralVoice = "discantus" \transpose c c' {
      \hide Score.BarNumber {
        c'1\melisma bes a g\melismaEnd
        f\breve
        \[ f1\melisma a c'\breve d'\melismaEnd \]
        c'\longa
        c'\breve\melisma a1 g1\melismaEnd
        fis\longa^\signumcongruentiae
      }
    }
    \new Lyrics \lyricsto "discantus" {
      San -- ctus, San -- ctus, San -- ctus
    }
  >>
}
```



Siehe auch
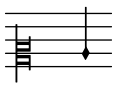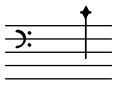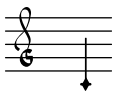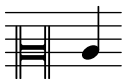
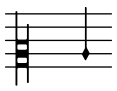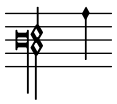
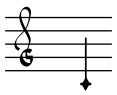
Glossar: Abschnitt “mensural notation” in *Glossar*.

17.3.2 Mensurale Schlüssel

In der Tabelle unten werden alle Mensuralschlüssel gezeigt, die mit dem `\clef`-Befehl erreicht werden. Manche Schlüssel benutzen dasselbe Zeichen, unterscheiden sich aber in der Notenlinie, auf der der Schlüssel notiert wird. In diesem Fällen ist eine Nummer im Schlüsselnamen eingefügt, nummeriert von unten nach oben. Man kann aber trotzdem eine beliebige Nummer erzwingen, wie es im Abschnitt Abschnitt 1.3.1 [Notenschlüssel], Seite 19, beschrieben wird. Die Note, die rechts von jedem Schlüssel gesetzt ist, zeigt das `c'` in Bezug zu dem jeweiligen Schlüssel.

Petrucci hat C-Schlüssel benutzt, die unterschiedlich ausbalancierte vertikale Balken auf der linken Seite hatten, je nachdem, auf welcher Notenlinie er sich befand.

Beschreibung	**Unterstützte Schlüssel**	**Beispiel**

Mensuraler C-Schlüssel im historischen Stil	`mensural-c1`, `mensural-c2`, `mensural-c3`, `mensural-c4`	
Mensuraler F-Schlüssel im historischen Stil	`mensural-f`	
Mensuraler G-Schlüssel im historischen Stil	`mensural-g`	
Mensuraler C-Schlüssel im modernen Stil	`neomensural-c1`, `neomensural-c2`, `neomensural-c3`, `neomensural-c4`	
Mensuraler C-Schlüssel im Petrucci-Stil, zur Benutzung auf verschiedenen Notenlinien (im Beispiel den Schlüssel auf der zweiten Linie)	`petrucci-c1`, `petrucci-c2`, `petrucci-c3`, `petrucci-c4`, `petrucci-c5`	
Mensuraler F-Schlüssel im Petrucci-Stil, kann auf verschiedenen Notenlinien benutzt werden (im Beispiel auf der dritten Linie)	`petrucci-f3`, `petrucci-f4`, `petrucci-f5`	
Mensuraler G-Schlüssel im Petrucci-Stil	`petrucci-g`	

Siehe auch

Glossar: Abschnitt “mensural notation” in *Glossar*, Abschnitt “clef” in *Glossar*.

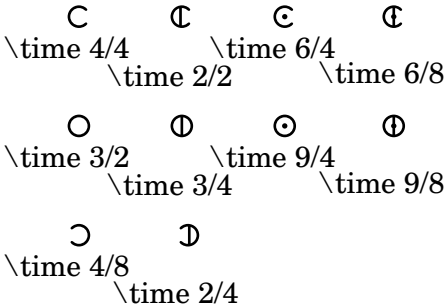
Notationsreferenz: Abschnitt 1.3.1 [Notenschlüssel], Seite 19.

Bekannte Probleme und Warnungen

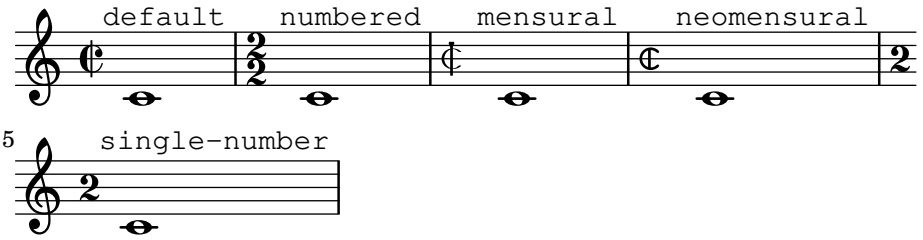
Der mensurale G-Schlüssel ist als Petrucci-G-Schlüssel deklariert.

17.3.3 Mensurale Taktartenbezeichnungen

LilyPond besitzt beschränkte Unterstützung für Mensurzeichen (die den heutigen Taktarten ähneln, aber doch einige Eigenheiten haben). Die Symbole sind starr verknüpft mit bestimmten Brüchen. Darum müssen die Werte `n` und `m` der folgenden Tabelle in den Befehl `\time n/m` eingesetzt werden, um die entsprechenden Symbole zu erhalten.



Mit der style-Eigenschaft des Objektes TimeSignature können die Taktarten angewählt werden. Unterstützte Stile sind: neomensural und mensural. In der Tabelle oben wurde der neomensurale Stil verwendet. Im folgenden Beispiel sind die unterschiedlichen Stile dargestellt.



Siehe auch

Glossary: Abschnitt “mensural notation” in *Glossar*.

Notationsreferenz: Abschnitt 2.3.1 [Taktangabe], Seite 65.

Bekannte Probleme und Warnungen

Die Verhältnisse der Notenwerte können nicht bei Mensurwechsel geändert werden, weil sie nicht konstant sind. Zum Beispiel kann das Verhältnis 1 brevis = 3 semibrevis (tempus perfectum) manuell erstellt werden, indem folgende Variable erstellt wird:

```
breveTP = #(ly:make-duration -1 0 3/2)
...
{ c\breveTP f1 }
```

Hiermit wird die Variable breveTP auf den Wert „3/2 mal 2 = 3 mal eine Ganze“ gesetzt.

Die Symbole mensural68alt und neomensural68alt (alternative Symbole für 6/8) können nicht mit dem \time-Befehl. Anstelle dess muss \markup {\musicglyph "timesig.mensural68alt" } benutzt werden.

17.3.4 Mensurale Notenköpfe

Für die Mensuralnotation kann ein Notenkopfstil ausgewählt werden, der sich vom Standard (default) unterscheidet. Dies wird erreicht, indem die style-Eigenschaft des Notenkopf-(NoteHead)-Objekts auf einen der Werte baroque, neomensural, mensural, petrucci, blackpetrucci oder semipetrucci gesetzt wird.

Der barocke (baroque) Stil unterscheidet sich vom Standard (default) folgendermaßen:

- Er stellt einen maxima-Notenkopf zur Verfügung und
- setzt eine eckige Form für die Brevis (\breve) ein.

Die Stile neomensural, mensural und petrucci unterscheiden sich vom barocken Stil folgendermaßen:
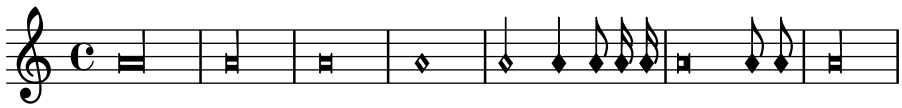
- Für Semibrevis und kleinere Notenwerte werden rhombenförmige Notenköpfe eingesetzt und
- die Hälse werden über den Kopf zentriert.

Der blackpetrucci-Stil erstellt Notenköpfe zur Benutzung für die schwarze Mensuralnotation oder *coloratio*-Abschnitten in der weißen Menusralnotation. Weil der Notenkopfstil nicht die Anzahl der Fähnchen beeinflusst, muss eine Semiminia in diesem Stil als a8*2 notiert werden, nicht als a4, weil sie sonst wie eine Minima aussehen würde. Der Faktor, mit dem der Notenwert multipliziert wird, kann sich ändern, wenn *coloratio* etwa zur Notation von Triolen eingesetzt wurde.

Mit dem semipetrucci-Stil können halb-schwarze Notenköpfe notiert werden (Brevis, Longa und Maxima).

Das folgende Beispiel zeigt den Petrucci-Stil:

```
\set Score.skipBars = ##t
\autoBeamOff
\override NoteHead.style = #'petrucci
a'\maxima a'\longa a'\breve a'1 a'2 a'4 a'8 a'16 a'
\override NoteHead.style = #'semipetrucci
a'\breve*5/6
\override NoteHead.style = #'blackpetrucci
a'8*4/3 a'
\override NoteHead.style = #'petrucci
a'\longa
```



Siehe auch

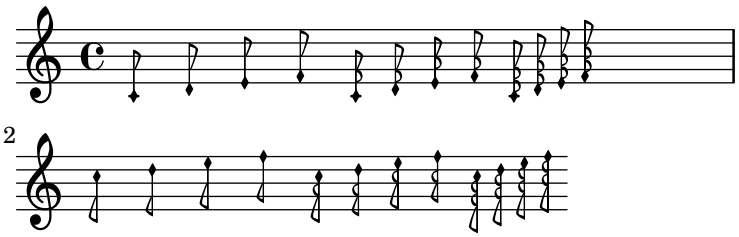
Glossar: Abschnitt “mensural notation” in *Glossar*, Abschnitt “note head” in *Glossar*.

Notationsreferenz: Abschnitt A.9 [Notenkopfstile], Seite 678.

17.3.5 Mensurale Fähnchen

Mit der Fähnchen-(`flag-style`)-Eigenschaft der graphischen Objekte „Hals“ (`Stem`) können auch Mensuralfähnchen gesetzt werden. Neben dem Standardstil (`default`) ist nur (`mensural`) unterstützt.

```
\override Flag.style = #'mensural
\override Stem.thickness = #1.0
\override NoteHead.style = #'mensural
\autoBeamOff
c'8 d'8 e'8 f'8 c'16 d'16 e'16 f'16 c'32 d'32 e'32 f'32 s8
c''8 d''8 e''8 f''8 c''16 d''16 e''16 f''16 c''32 d''32 e''32 f''32
```



Dabei ist die innerste Fahne immer vertikal auf eine Notenlinie ausgerichtet.

Es gibt keinen eigenen Stil für den neomensuralen oder Petrucci-Stil. Für die Notation des Gregorianischen Chorals gibt es keine Fähnchen.

Siehe auch

Glossar: Abschnitt “mensural notation” in *Glossar*, Abschnitt “flag” in *Glossar*.

Bekannte Probleme und Warnungen

Die Positionierung der Fähnchen an den Hälsen ist leicht verschoben.

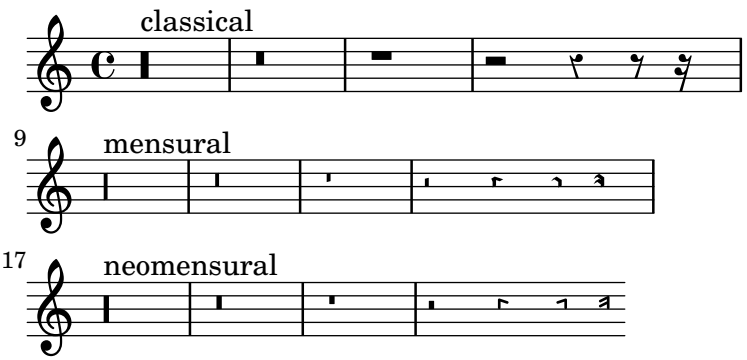
Vertikale Ausrichtung der Fähnchen an einer Notenlinie geht von der Annahme aus, dass der Hals entweder genau auf einer Notenlinie oder genau zwischen zwei Notenlinien endet. Das ist aber nicht unbedingt immer der Fall, weil LilyPond komplizierte Methoden zur Ermittlung des besten Layouts verwendet. Diese Methoden sollten aber eigentlich nicht zur Notation von mensuraler Musik eingesetzt werden.

17.3.6 Mensurale Pausen

Besondere Pausensymbole für die Notation der Alten Musik können mit der `style`-Eigenschaft des graphischen Objektes (grob) „Pause“ (`Rest`) angewählt werden. Unterstützte Stile sind klassisch (`classical`), `neomensural` und `mensural`. Der klassische (`classical`) Stil unterscheidet sich vom Standardstil (`default`) nur darin, dass die Viertelpause wie eine gespiegelte Achtelpause aussieht. Der mensurale und neomensurale Stil ahmt die Form von Pausen nach, wie man sie in Drucken bis zum 16. Jahrhundert finden kann.

Das folgende Beispiel demonstriert den mensuralen und den neomensuralen Stil:

```
\set Score.skipBars = ##t
\override Rest.style = #'classical
r\longa^"classical" r\breve r1 r2 r4 r8 r16 s \break
\override Rest.style = #'mensural
r\longa^"mensural" r\breve r1 r2 r4 r8 r16 s \break
\override Rest.style = #'neomensural
r\longa^"neomensural" r\breve r1 r2 r4 r8 r16
```



Es gibt keine 32-stel- und 64-stel-Pausen für den mensuralen oder neomensuralen Stil. Anstatt dessen werden die Pausenformen des Standardstiles verwendet.

Eine Liste aller Pausen findet sich in Abschnitt “Ancient notation” in *Schnipsel*.

Siehe auch

Notationsreferenz: Abschnitt 2.2.1 [Pausen], Seite 57.

Schnipsel: Abschnitt “Ancient notation” in *Schnipsel*.

Bekannte Probleme und Warnungen

Das Zeichen für die Maxima-Pause im mensuralen Stil ist eigentlich eine perfekte Longa-Pause: zwei (oder drei) Longa-Pausen müssen benutzt werden, um eine Maxima-Pause zu setzen. Longa-Pausen werden nicht automatisch gruppiert, sodass man das manuell vornehmen muss, indem man Pausen mit Tonhöhe einsetzt.

17.3.7 Mensurale Versetzungszeichen und Tonartbezeichnung

Der `mensural`-Stil stellt ein Kreuz und ein B zur Verfügung, die sich vom Standardstil unterscheiden. Wenn das Auflösungszeichen notiert wird, wird es aus dem `vaticana`-Stil gesetzt.



Der Stil für Versetzungszeichen und Vorzeichen wird durch die `alteration-glyph-name-alist`-Eigenschaft der Grobs `Accidental` und `KeySignature` bestimmt, also etwa folgendermaßen:

```
\override Staff.Accidental.alteration-glyph-name-alist =
```

```
#alteration-mensural-glyph-name-alist
```

Siehe auch

Glossar: Abschnitt “mensural notation” in *Glossar*, Abschnitt “Pitch names” in *Glossar*, Abschnitt “accidental” in *Glossar*, Abschnitt “key signature” in *Glossar*.
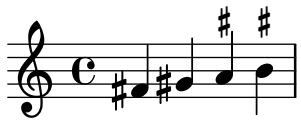
Notationsreferenz: Kapitel 1 [Tonhöhen], Seite 3, Abschnitt 1.1.3 [Versetzungszeichen], Seite 7, Abschnitt 1.3.5 [Automatische Versetzungszeichen], Seite 26, Abschnitt 1.3.2 [Tonartbezeichnung], Seite 22.

Referenz der Interna: Abschnitt “KeySignature” in *Referenz der Interna*.

17.3.8 Vorgeschlagene Versetzungszeichen (*musica ficta*)

In der europäischen Notation bis etwa 1600 wurde von Sängern erwartet, dass sie eigenständig Noten nach bestimmten Regeln chromatisch veränderten. Das wird als *musica ficta* bezeichnet. In modernen Transkriptionen werden diese Versetzungszeichen üblicherweise über die Note notiert.
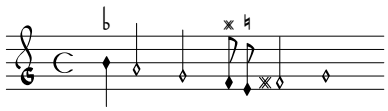
Es ist möglich, derartige Versetzungszeichen zu notieren, und die Anzeige kann zwischen normaler Satzweise und musica ficta hin- und hergewechselt werden. Hierzu muss `suggestAccidentals` auf wahr gesetzt werden:



Damit wird *jedes* folgende Versetzungszeichen über dem System gesetzt werden, bis die Eigenschaft mit `\set suggestAccidentals = ##f` wieder zum Standardverhalten verändert wurde. Eine praktischere Lösung ist es, `\once \set suggestAccidentals = ##t` zu benutzen, was man als Variable definieren kann:

```
ficta = { \once \set suggestAccidentals = ##t }
\score { \relative
  \new MensuralVoice  {

\once \set suggestAccidentals = ##t
  bes'4 a2 g2 \ficta fis8 \ficta e! fis2 g1
  }
}
```



Siehe auch

Referenz der Interna: Abschnitt “Accidental_engraver” in *Referenz der Interna*, Abschnitt “AccidentalSuggestion” in *Referenz der Interna*.

17.3.9 Weiße Mensuralligaturen

Begrenzte Unterstützung für Ligaturen der weißen Mensuralnotation ist vorhanden.

Um weiße Mensuralligaturen zu benutzen, muss innerhalb des Layout-Blocks im `Voice`-Kontext der `Mensural_ligature_engraver` aktiviert werden und gleichzeitig der `Ligature_bracket_engraver` (der die Klammern über den Noten setzt) entfernt werden, wie im Beispiel.

```
\layout {
```

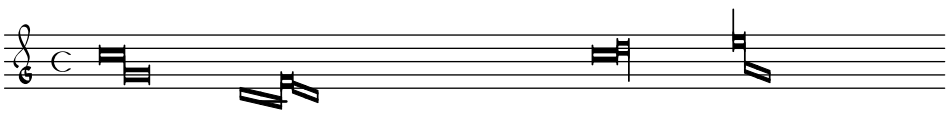
```
    \context {
      \Voice
      \remove Ligature_bracket_engraver
      \consists Mensural_ligature_engraver
    }
  }
```

Zusätzlich zu diesen Einstellungen gibt es keine eigenen Befehle, die die Form einer Ligatur bestimmen. Die Form wird vielmehr aus Tonhöhen und Tondauern der in Klammern gesetzten Noten geschlossen. Diese Herangehensweise erfordert einige Eingewöhnung, hat aber den großen Vorteil, dass der musikalische Inhalt der Ligatur dem Programm bekannt ist. Das ist nicht nur notwendig für korrekte MIDI-Ausgabe, sondern erlaubt es auch, automatische Transkriptionen von Ligaturen anzufertigen.
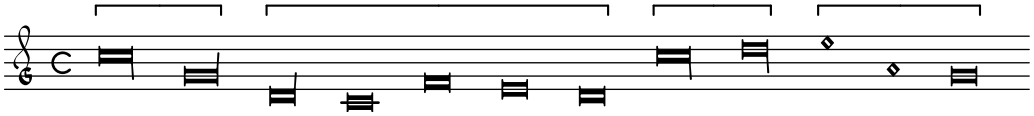
An bestimmten Stellen können zwei aufeinanderfolgende Noten entweder als zwei Quadrate oder eine Obliqua (Flexa) dargestellt werden. In derartigen Fällen ist die Quadratform der Standard, aber die Obliqua kann verlangt werden, indem man die `ligature-flexa`-Eigenschaft des *zweiten* Notenkopfes setzt. Die Länge der Obliqua kann durch die Notenkopfeigenschaft `flexa-width` definiert werden.

Eine Datei kann zum Beispiel so aussehen:

```
\score {
  \transpose c c' {
    \set Score.timing = ##f
    \set Score.measureBarType = ""
    \override NoteHead.style = #'petrucci
    \override Staff.TimeSignature.style = #'mensural
    \clef "petrucci-g"
    \[ c'\maxima g \]
    \[ d\longa
       \override NoteHead.ligature-flexa = ##t
       \once \override NoteHead.flexa-width = #3.2
       c\breve f e d \]
    \[ c'\maxima d'\longa \]
    \[ e'1 a g\breve \]
  }
  \layout {
    \context {
      \Voice
      \remove Ligature_bracket_engraver
      \consists Mensural_ligature_engraver
    }
  }
}
```



Wenn der `Ligature_bracket_engraver` nicht durch den `Mensural_ligature_engraver` ersetzt wird, werden die Noten wie folgt ausgegeben:



Siehe auch

Glossar: Abschnitt “ligature” in *Glossar*.

Notationreferenz: Abschnitt 17.4.7 [Ligaturen der gregorianischen Quadratnotation], Seite 435, Abschnitt 17.2.2 [Ligaturen], Seite 422.

Bekannte Probleme und Warnungen

Die horizontale Positionierung ist sehr schlecht.

Versetzungszeichen können mit vorhergehenden Noten kollidieren.

17.4 Gregorianischen Choral setzen

Wenn ein gregorianischer Choral notiert wird, wählt der `Vaticana_ligature_engraver` automatisch die richtigen Notenköpfe aus, so dass man den Notenkopfstil nicht explizit setzen muss. Der Stil kann dennoch gesetzt werden, etwa auf `vaticana.punctum` um punctum-Neumen zu erzeugen. Ähnlich funktioniert auch der `Mensural_ligature_engraver`, der Mensuralligaturen setzt.

Siehe auch

Glossar: Abschnitt “ligature” in *Glossar*.

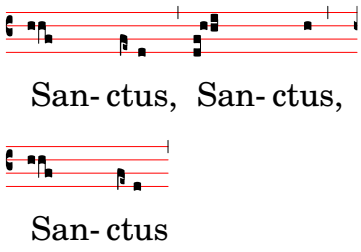
Notationreferenz: Abschnitt 17.3.9 [Weiße Mensuralligaturen], Seite 429, Abschnitt 17.2.2 [Ligaturen], Seite 422.

17.4.1 Gregorianische Gesangs-Kontexte

Die vordefinierten Kontexte `VaticanaVoice` (für eine gregorianische Stimme) und `VaticanaStaff` (für ein gregorianisches Notensystem) können eingesetzt werden, um Gregorianischen Choral im Stil der Editio Vaticana zu setzen. Diese Kontexte initialisieren alle relevanten Eigenschaften für das Notensystem und die graphischen Objekte, so dass unmittelbar mit der Notation begonnen werden kann. Siehe das folgende Beispiel:

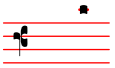
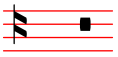
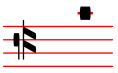
```
\new VaticanaScore {
  <<
    \new VaticanaVoice = "cantus" {
      \[ c'\melisma c' \flexa a \]
      \[ a \flexa \deminutum g\melismaEnd \]
      f \divisioMinima
      \[ f\melisma \pes a c' c' \pes d'\melismaEnd \]
      c' \divisioMinima \break
      \[ c'\melisma c' \flexa a \]
      \[ a \flexa \deminutum g\melismaEnd \] f \divisioMinima
    }
    \new Lyrics \lyricsto "cantus" {
      San- ctus, San- ctus, San- ctus
    }
  >>
}

\layout {
  indent = 0
  ragged-last = ##t
}
```



17.4.2 Gregorianische Schlüssel

Die folgende Tabelle zeigt alle Schlüssel für den gregorianischen Choral, die mit dem `\clef`-Befehl unterstützt sind. Einige Schlüssel benutzen das selbe Zeichen, unterscheiden sich aber in der Notenlinie, auf der der Schlüssel gesetzt wird. In diesem Fall wird eine Nummer benutzt, die die Notenlinie von unten nach oben kennzeichnet. Man kann die Schlüssel aber auch manuell auf eine bestimmte Notenlinie zwingen, wie gezeigt in Abschnitt 1.3.1 [Notenschlüssel], Seite 19. Die Note, die rechts von den Schlüsseln im Beispiel gezeigt wird, ist ein `c'` in Bezug auf den aktuellen Schlüssel.

Beschreibung	**unterstützter Schlüssel**	**Beispiel**
Do-Schlüssel der Editio Vaticana	`vaticana-do1`, `vaticana-do2`, `vaticana-do3`	
Fa-Schlüssel der Editio Vaticana	`vaticana-fa1`, `vaticana-fa2`	
Do-Schlüssel der Editio Medicaea	`medicaea-do1`, `medicaea-do2`, `medicaea-do3`	
Fa-Schlüssel der Editio Medicaea	`medicaea-fa1`, `medicaea-fa2`	
Hufnagel Do-Schlüssel für den historischen Stil	`hufnagel-do1`, `hufnagel-do2`, `hufnagel-do3`	
Hufnagel Fa-Schlüssel für den historischen Stil	`hufnagel-fa1`, `hufnagel-fa2`	
Kombinierter Do/Fa-Hufnagelschlüssel für den historischen Stil	`hufnagel-do-fa`	

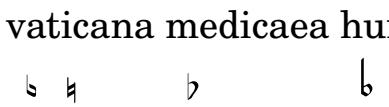
Siehe auch

Glossar: Abschnitt “clef” in *Glossar*.

Notationsreferenz: Abschnitt 1.3.1 [Notenschlüssel], Seite 19.

17.4.3 Gregorianische Versetzungszeichen und Tonartbezeichnung

Es gibt Versetzungszeichen in drei unterschiedlichen Stilen für die Notation des gregorianischen Chorals:



Wie zu sehen ist, werden nicht alle Versetzungszeichen von jedem Stil unterstützt. Wenn versucht wird, ein Versetzungszeichen zu notieren, das von einem bestimmten Stil nicht unterstützt wird, wechselt LilyPond zu einem anderen Stil.

Der Stil für Versetzungs- und Vorzeichen wird von der `alteration-glyph-name-alist`-Eigenschaft der Grobs `Accidental` und `KeySignature` kontrolliert, beispielsweise:

```
\override Staff.Accidental.alteration-glyph-name-alist =
  #alteration-mensural-glyph-name-alist
```

Siehe auch

Glossar: Abschnitt “accidental” in *Glossar*, Abschnitt “key signature” in *Glossar*.

Notationsreferenz: Kapitel 1 [Tonhöhen], Seite 3, Abschnitt 1.1.3 [Versetzungszeichen], Seite 7, Abschnitt 1.3.5 [Automatische Versetzungszeichen], Seite 26, Abschnitt 1.3.2 [Tonartbezeichnung], Seite 22.

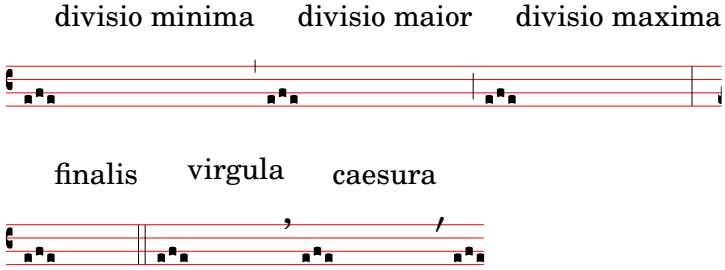
Referenz der Interna: Abschnitt “KeySignature” in *Referenz der Interna*.

17.4.4 Divisiones

Die Notation des gregorianischen Chorals benutzt keine Pausen, anstatt dessen werden *Divisiones* eingesetzt.

Eine *divisio* (Plural: *divisiones*; Latein: „Teilung“) ist ein Symbol des Notensystemkontextes, das benutzt wird, um Phrasierung und Abschnitte im Gregorianischen Choral anzuzeigen. Die musikalische Bedeutung von *divisio minima*, *divisio maior* und *divisio maxima* kann beschrieben werden als kurze, mittlere und lange Pause, ungefähr wie die Atemzeichen aus dem Abschnitt Abschnitt 3.2.3 [Atemzeichen], Seite 134. Das *finalis*-Zeichen bezeichnet nicht nur das Ende eines Chorals, sondern wird auch oft innerhalb eines Antiphons/Responsoriums benutzt, um das Ende eines Abschnitts anzuzeigen.

Einige Editionen verwenden eine *virgula* oder *caesura* anstelle der divisio minima.



Vordefinierte Befehle

`\virgula`, `\caesura`, `\divisioMinima`, `\divisioMaior`, `\divisioMaxima`, `\finalis`.

Siehe auch

Glossary: Abschnitt “caesura” in *Glossar*, Abschnitt “divisio” in *Glossar*.

Notationsreferenz: Abschnitt 3.2.3 [Atemzeichen], Seite 134.

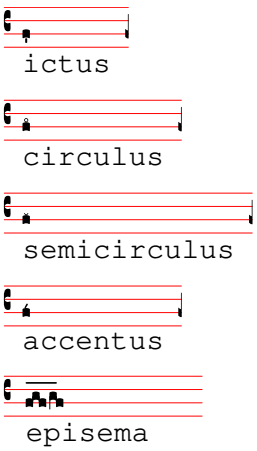
Referenz der Interna: Abschnitt “BreathingSign” in *Referenz der Interna*.

17.4.5 Artikulationszeichen des Gregorianischen Chorals

Zusätzlich zu den Standardartikulationszeichen, wie sie im Abschnitt Abschnitt 3.1.1 [Artikulationszeichen und Verzierungen], Seite 120, beschrieben werden, werden auch Artikulationszeichen für die Notation des Editio Vaticana-Stils zur Verfügung gestellt.

```
\new VaticanaScore {
  \new VaticanaVoice {
    \override TextScript.font-family = #'typewriter
    \override TextScript.font-shape = #'upright
    \override Script.padding = #-0.1
    a\ictus_"ictus " \bar "" \break
    a\circulus_"circulus " \bar "" \break
    a\semicirculus_"semicirculus " \bar "" \break
    a\accentus_"accentus " \bar "" \break
    \[ a_"episema" \episemInitium \pes b \flexa a b \episemFinis \flexa a \]
  }
}

\layout {
  indent = 0
  ragged-last = ##t
}
```



Siehe auch

Notationreferenz: Abschnitt 3.1.1 [Artikulationszeichen und Verzierungen], Seite 120.

Schnipsel: Abschnitt “Ancient notation” in *Schnipsel*.

Referenz der Interna: Abschnitt “Episema” in *Referenz der Interna*, Abschnitt “EpisemaEvent” in *Referenz der Interna*, Abschnitt “Episema_engraver” in *Referenz der Interna*, Abschnitt “Script” in *Referenz der Interna*, Abschnitt “ScriptEvent” in *Referenz der Interna*, Abschnitt “Script_engraver” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Einige Artikulationszeichen sind vertikal zu dich an den entsprechenden Notenköpfen gesetzt.

17.4.6 Augmentationspunkte (*morae*)

Verlängerungspunkte, auch als *morae* bezeichnet, werden mit der Musikfunktion `\augmentum` hinzugefügt. Es handelt sich um eine eigenständige Funktion und nicht um einen Präfix, der zu einer Note gehört. Die Funktion wirkt sich nur auf den direkt vorhergehenden musik. Ausdruck

aus. Das heißt, dass `\augmentum \virga c` keine sichtbare Wirkung hat. Anstelle dessen sollte geschrieben werden: `\virga \augmentum c` oder `\augmentum {\virga c}`. Man kann `\augmentum {a g}` als Kurznotation für `\augmentum a \augmentum g` schreiben.

```
\new VaticanaScore {
  \new VaticanaVoice {
    \[ \augmentum a \flexa \augmentum g \]
    \augmentum g
  }
}
```



Siehe auch

Notationsreferenz: Abschnitt 3.2.3 [Atemzeichen], Seite 134.

Referenz der Interna: Abschnitt “BreathingSign” in *Referenz der Interna*.

Schnipsel: Abschnitt “Ancient notation” in *Schnipsel*.

17.4.7 Ligaturen der gregorianischen Quadratnotation

Beschränkte Unterstützung für gregorianische Quadratneumen-Ligaturen (nach dem Stil der Editio Vaticana) ist vorhanden. Die wichtigsten Ligaturen können schon gesetzt werden, aber wichtige Eigenschaften anspruchsvoller Typographie wie horizontale Ausrichtung von mehreren Ligaturen, korrekte Silbenpositionierung und richtiger Umgang mit Versetzungszeichen fehlen noch.

Notenköpfe können verändert und/bzw. verbunden werden.

- Die Form des Notenkopf kann verändert werden, indem man *vor* die Noten folgende Befehle schreibt: `\virga`, `\stropha`, `\inclinatum`, `\auctum`, `\ascendens`, `\descendens`, `\oriscus`, `\quilisma`, `\deminutum`, `\cavum`, `\linea`.
- Eigentliche Ligaturen (also Noten, die miteinander verbunden sind), werden erstellt, indem man einen der verbindenden Befehle, `\pes` oder `\flexa` für Aufwärts- bzw. Abwärtsbewegung, zwischen die zu verbindenden Noten setzt.

Eine Notenbezeichnung ohne jeglichen Modifikator produziert ein *punctum*. Alle anderen Neumen, auch einzelne Noten-Neumen mit einer anderen Form als der *Virga* werden generell als Ligaturen betrachtet und deshalb von den Zeichen `\[...\]` eingeklammert werden.

Einzelne Noten-Neumen:

- Das *punctum* ist die grundlegende Notenform (im *Vaticana*-Stil: ein Quadrat mit gebogenen Ober- und Unterkanten). Zusätzlich gibt es auch noch das oblique *punctum inclinatum*, das mit dem Präfix `\inclinatum` erstellt wird. Das normale *punctum* kann durch `\cavum` verändert werden, wodurch eine hohle Note erstellt wird, und durch `\linea`, wodurch vertikale Linien zu den Seiten der Note gezogen werden.
- Die *virga* hat einen absteigenden Hals auf der rechten Seite. Sie wird durch den Modifikator `\virga` erstellt.

Ligaturen

Anders als in anderen Neumennotationssystemen, wird das typographische Aussehen einer Ligatur nicht durch Eingabebefehle direkt vorgegeben, sondern richtet sich nach bestimmten Darstellungsregeln, die durch die musikalische Bedeutung bestimmt werden. Eine Ligatur mit drei Noten beispielsweise, mit der Form tief-hoch-tief, wie etwa `\[ a \pes b \flexa g \]`, ergibt einen Torculus, der aus drei Punctum-Köpfen besteht, während die Form hoch-tief-hoch, wie etwa `\[ a \flexa g \pes b \]`, einen Porrectus mit einer gebogenen Flexa und nur einem

Punctum-Kopf ergibt. Es gibt keinen Befehl, mit dem explizit eine gebogene Flexa gesetzt werden können; die Entscheidung, wann eine derartige Form im Notenbild vorkommen soll, wird durch die musikalische Bedeutung der Noten vorgegeben. Die Idee hinter dieser Art der Eingabe ist es, dass der musikalische Inhalt von der graphischen Ausgabe getrennt wird. Dadurch wird es möglich, die gleiche Quelldatei zu benutzen, um beispielsweise die Noten in einem anderen Stil darzustellen.

Liquescente Neumen

Eine weitere Hauptkategorie der Notation von gregorianischem Choral sind die sogenannten liquescenten Neumen. Sie werden unter bestimmten Umständen am Ende einer Silbe eingesetzt, die auf einen „liquescenten“ Buchstaben endet (das sind die Konsonanten, die eine Tonhöhe haben können, also die Nasale, l, r, v, j und ihre diphtongalen Entsprechungen). Liquescente Neumen werden also nie alleine eingesetzt (auch wenn sie isoliert produziert werden können) und treten immer am Ende einer Silbe auf.

Liquescente Neumen werden graphisch auf zwei Arten dargestellt: mit einer kleineren Note oder indem die Hauptnote nach oben bzw. unten „gedreht“ wird. Die erste Darstellungsweise erreicht man, indem einen normalen `pes` oder `flexa` schreibt und dann die Form der zweiten Note verändert: `\[ a \pes \deminutum b \]` . Die zweite Darstellungsweise erreicht man, indem die Form einer einzelnen Neume mit `\auctum` und einem der Richtungsanzeiger `\descendens` bzw. `\ascendens` versieht: `\[ \auctum \descendens a \]` .

Spezielle Zeichen

Eine dritte Kategorie besteht aus einer kleinen Anzahl an Zeichen mit einer besonderen Bedeutung: die *quilisma*, der *oriscus* und der *strophicus*. Sie werden notiert, indem man vor die entsprechende Note den Modifikator `\quilisma`, `\oriscus` oder `\stropha` schreibt.

Im Grunde kann innerhalb der Ligaturbegrenzer `\[` und `\]` eine beliebige Anzahl ans Notenköpfen eingefügt werden und Präfixe wie `\pes`, `\flexa`, `\virga`, `\inclinatum` usw. können beliebig untereinander kombiniert werden. Der Einsatz der Regeln, mit denen die Ligaturen konstruiert werden, wird entsprechend angepasst. Auf diese Art kann eine unendliche Anzahl an Ligaturen erstellt werden.

Die Benutzung der Notationszeichen folgt allerdings bestimmten Regeln, die nicht von LilyPond überprüft werden. Die *quilisma* beispielsweise findet sich immer als mittlere Note einer aufsteigenden Ligatur und fällt üblicherweise auf einen Halbtonschritt, aber es ist durchaus möglich, wenn auch nicht *richtig*, eine Quilisma bestehend aus einer Note zu notieren.

Neben den Notenformen definiert LilyPond auch die Befehle `\versus`, `\responsum`, `\ij`, `\iij`, `\IJ` und `\IIJ`, mit denen die entsprechenden Zeichen, etwa für den Text oder als Abschnittsmarkierung erstellt werden können. Diese Befehl benutzen bestimmte Unicode-Zeichen und funktionieren nur, wenn eine Schriftart vorhanden ist, die diese Zeichen unterstützt.

In der folgenden Tabelle wird eine begrenzte, aber dennoch repräsentative Anzahl an Ligaturen der Neumennotation dargestellt, denen Fragmente beigefügt sind, die die Notation in LilyPond zeigen. Die Tabelle basiert auf der erweiterten Neumentabelle des zweiten Bands des Antiphonale Romanum (*Liber Hymnarius*), 1983 von den Mönchen von Solsemes herausgegeben. Die erste Spalte zeigt die Bezeichnungen der Ligaturen, fett für die Normalform, kursiv für die liquescente Form. Die dritte Spalte zeigt Code-Schnipsel, mit denen die Ligatur notiert werden kann, wobei die Noten `g`, `a` und `b` als Tonhöhen eingesetzt werden.

Neumen aus einzelnen Noten

Grundform und *liquescente Form*	**Ausgabe**	**LilyPond-Code**

Punctum


```
\[ b \]
```



```
\[ \cavum b \]
```



```
\[ \linea b \]
```

Punctum Auctum Ascendens



```
\[ \auctum \ascendens b \]
```

Punctum Auctum Descendens



```
\[ \auctum \descendens b \]
```

Punctum inclinatum



```
\[ \inclinatum b \]
```

Punctum Inclinatum Auctum



```
\[ \inclinatum \auctum b \]
```

Punctum Inclinatum Parvum


```
\[ \inclinatum \deminutum b \]
```

Virga



Ligaturen aus zwei Noten

Clivis vel Flexa



```
\[ b \flexa g \]
```

Clivis Aucta Descendens



```
\[ b \flexa \auctum \descendens
g \]
```

Clivis Aucta Ascendens



```
\[ b \flexa \auctum \ascendens
g \]
```

Cephalicus



```
\[ b \flexa \deminutum g \]
```

Podatus/Pes



```
\[ g \pes b \]
```

Pes Auctus Descendens



```
\[ g \pes \auctum \descendens b \]
```

Pes Auctus Ascendens



```
\[ g \pes \auctum \ascendens b \]
```

Epiphonus



```
\[ g \pes \deminutum b \]
```

Pes Initio Debilis



```
\[ \deminutum g \pes b \]
```

Pes Auctus Descendens Initio Debilis



```
\[ \deminutum g \pes \auctum \descendens b \]
```

Ligaturen mit mehr als zwei Noten

Torculus



```
\[ a \pes b \flexa g \]
```

Torculus Auctus Descendens



```
\[ a \pes b \flexa \auctum \descendens g \]
```

Torculus Deminutus



```
\[ a \pes b \flexa \deminutum g \]
```

Torculus Initio Debilis



```
\[ \deminutum a \pes b \flexa g \]
```

Torculus Auctus Descendens Initio Debilis



```
\[ \deminutum a \pes b \flexa \auctum \descendens g \]
```

Torculus Deminutus Initio Debilis



```
\[ \deminutum a \pes b \flexa
\deminutum g \]
```

Porrectus



```
\[ a \flexa g \pes b \]
```

Porrectus Auctus Descendens



```
\[ a \flexa g \pes \auctum
\descendens b \]
```

Porrectus Deminutus



```
\[ a \flexa g \pes \deminutum b
\]
```

Climacus



```
\[ \virga b \inclinatum a
\inclinatum g \]
```

Climacus Auctus



```
\[ \virga b \inclinatum a
\inclinatum \auctum g \]
```

Climacus Deminutus



```
\[ \virga b \inclinatum a
\inclinatum \deminutum g \]
```

Scandicus



```
\[ g \pes a \virga b \]
```

Scandicus Auctus Descendens



```
\[ g \pes a \pes \auctum
\descendens b \]
```

Scandicus Deminutus



```
\[ g \pes a \pes \deminutum b \]
```

Special Signs

Quilisma



```
\[ g \pes \quilisma a \pes b \]
```

Quilisma Pes Auctus Descendens



```
\[ \quilisma g \pes \auctum
\descendens b \]
```

Oriscus


```
\[ \oriscus b \]
```

Pes Quassus



```
\[ \oriscus g \pes \virga b \]
```

Pes Quassus Auctus Descendens



```
\[ \oriscus g \pes \auctum
\descendens b \]
```

Salicus



```
\[ g \oriscus a \pes \virga b \]
```

Salicus Auctus Descendens



```
\[ g \oriscus a \pes \auctum
\descendens b \]
```

(Apo)stropha


```
\[ \stropha b \]
```

Stropha Aucta


```
\[ \stropha \auctum b \]
```

Bistropha



```
\[ \stropha b \stropha b \]
```

Tristropha



```
\[ \stropha b \stropha b
\stropha b \]
```

Trigonus



```
\[ \stropha b \stropha b
\stropha a \]
```

Vordefinierte Befehle

Folgende Notenpräfixe sind unterstützt: `\virga`, `\stropha`, `\inclinatum`, `\auctum`, `\descendens`, `\ascendens`, `\oriscus`, `\quilisma`, `\deminutum`, `\cavum`, `\linea`.

Präfixe können kombiniert werden, wenn es hier auch Begrenzungen gibt. Zum Beispiel können die Präfixe `\descendens` oder `\ascendens` vor einer Note geschrieben werden, aber nicht beide für die selbe Note.

Zwei benachbarte Noten können mit den `\pes` und `\flexa`-Infixen verbunden werden, um eine steigende bwz. fallende Melodielinie zu notieren.

Die musikalische Funktion `\augmentum` muss benutzt werden, um augmentum-Punkte hinzuzufügen.

Siehe auch

Glossar: Abschnitt “ligature” in *Glossar*.

Notationreferenz: Abschnitt 17.4.7 [Ligaturen der gregorianischen Quadratnotation], Seite 435, Abschnitt 17.3.9 [Weiße Mensuralligaturen], Seite 429, Abschnitt 17.2.2 [Ligaturen], Seite 422.

Bekannte Probleme und Warnungen

Wenn ein `\augmentum`-Punkt am Ende des letzten Systems innerhalb einer Ligatur gesetzt wird, ist er vertikal etwas falsch positioniert. Als Abhilfe kann eine unsichtbare Note (z. B. `s8`) als letzte Note im System eingegeben werden.

`\augmentum` sollte als Präfix implementiert sein, nicht als eigene musikalische Funktion, so dass `\augmentum` mit den anderen Präfixen in arbiträrer Reihenfolge notiert werden kann.

17.5 Kiever Quadratnotation setzen

17.5.1 Kiever Kontexte
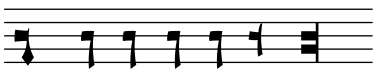
Wie auch für die Mensural- und Gregorianische Notation können die Kontexte `KievanVoice` and `KievanStaff` eingesetzt werdne, um Noten der Kiever Quadratnotation zu setzen. Diese Kontexte initialisieren die benötigten Kontexteigenschaften und Grob-Eigenschaften mit den richtigen Werten, sodass man sofort den Choral notieren kann:

```
% Font settings for Cyrillic
\paper {
  property-defaults.fonts.serif = "Linux Libertine O,serif"
}

\score {
  <<
    \new KievanVoice = "melody" \transpose c c' {
      \cadenzaOn

c4 c c c c2 b,\longa

\bar "kievan"
    }
    \new Lyrics \lyricsto "melody" {
      Го -- спо -- ди по -- ми -- луй.
    }
  >>
}
```



Господи помилуй.

Siehe auch

Glossar: Abschnitt “Kievan notation” in *Glossar*.

Bekannte Probleme und Warnungen
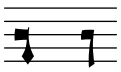
LilyPond unterstützt Kiever Notation des Synodischen Stils, welcher im Korpus der Gesangsbücher eingesetzt wurde, die durch Russische Heilige Synode 1910 gedruckt wurden und

neuerdings durch das Moskauer Patriarchat Verlagshaus neu heruasgegeben wurden. LilyPond kann nicht die älteren (selteren) Formen der Kiever Notation setzen, mit denen in Galizien rusinischer Choral notiert worden ist.

17.5.2 Kiever Schlüssel

Es gibt nur einen Schlüssel in der Kiever Notation (der Tse-fa-ut-Schlüssel). Er bezeichnet die Position von `c`:

```
\clef "kievan-do"
\kievanOn
c'
```



Siehe auch

Glossar: Abschnitt “Kievan notation” in *Glossar*, Abschnitt “clef” in *Glossar*.
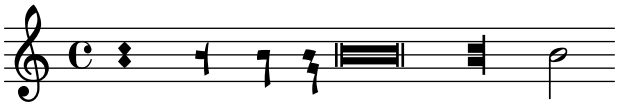
Notationsreferenz: Abschnitt 1.3.1 [Notenschlüssel], Seite 19.

17.5.3 Kiever Notenköpfe

Für die Kiever Notation muss der richtige Notenkopfstil gewählt werden. Die erreicht man, indem man die `style`-Eigenschaft des `NoteHead`-Objekts auf `kievan` setzt.

Die Kiever Schlussnote, welche am Ende eines Stückes gesetzt wird, kann gewählt werden, indem man die Notendauer `\longa` einsetzt. Das Kiever Rezitativzeichen, welches die Rezitation auf einer Tonhöhe anzeigt, kann gesetzt werden, indem die Notendauer `\breve` notiert wird. Folgendes Beispiel demonstriert die unterschiedlichen Notenköpfe:

```
\autoBeamOff
\cadenzaOn
\kievanOn
b'1 b'2 b'4 b'8 b'\breve b'\longa
\kievanOff
b'2
```



Siehe auch

Glossar: Abschnitt “Kievan notation” in *Glossar*, Abschnitt “note head” in *Glossar*.

Notationsreferenz: Abschnitt A.9 [Notenkopfstile], Seite 678.
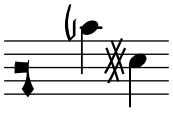
Bekannte Probleme und Warnungen

LilyPond bestimmt automatisch die Richtung eines Halses. Für Gesang in der Quadratnotation zeigen die Hälse jedoch alle in die gleiche Richtung innerhalb eines Melismas. Das kann man manuell erreichen, indem man die `direction`-Eigenschaft des `Stem`-Objekts setzt.

17.5.4 Kiever Versetzungszeichen

Der Versetzungszeichenstil `kievan` wird durch die Eigenschaft `alteration-glyph-name-alist` des `Accidental`-Grobs ausgewählt. Dieser Stil stellt ein Kreuz und ein B-Zeichen zur Verfügung, die sich von den Standardzeichen unterscheiden. Es gibt kein Auflösungszeichen in der Kiever Notation. Das Kreuz wird in der Synodalen Musik nicht eingesetzt, kann aber in früheren Manuskripten auftreten. Es wurde vor allem der Vollständigkeit halber eingesetzt.

```
\clef "kievan-do"
\override Accidental.alteration-glyph-name-alist =
 #alteration-kievan-glyph-name-alist
bes' dis'
```



Siehe auch
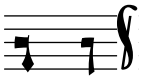
Glossar: Abschnitt “Kievan notation” in *Glossar*, Abschnitt “accidental” in *Glossar*.

Notationsreferenz: Abschnitt 1.1.3 [Versetzungszeichen], Seite 7, Abschnitt 1.3.5 [Automatische Versetzungszeichen], Seite 26, Abschnitt A.8 [Die Emmentaler-Schriftart], Seite 658.

17.5.5 Kiever Taktstriche

Eine dekorative Figur wird üblicherweise am Ende von eines Musikstückes der Kiever Notation gesetzt, was man als Kiever Schlussstrich bezeichnen kann. Es wird gesetzt mit `\bar "k"`.

```
\clef "kievan-do"
\kievanOn
c \bar "k"
```



Siehe auch

Abschnitt 2.5 [Takte], Seite 97, Abschnitt A.8 [Die Emmentaler-Schriftart], Seite 658,

17.6 Musiksatz Alter Musik in der Praxis – Szenarien und Lösungen

Wenn man mit Alter Notation zu tun hat, fallen oft Aufgaben an, die in der modernen Notation nicht vorkommen, für welche LilyPond geschaffen wurde. In diesem Abschnitt sollen darum einige praktische Problemstellungen und Lösungsvorschläge dargestellt werden. Dabei handelt es sich um:

- wie man Incipite in modernen Editionen von Mensuralnotation notieren kann (d.h. ein kleiner Abschnitt vor der eigentlichen Partitur, der die Originalnotenformen darstellt),
- wie man *Mensurstriche* einstellt, mit denen oft moderne Transkriptionen polyphoner Musik notiert werden,
- wie man den gregorianischen Choral mit moderner Notation darstellt und
- wie man sowohl ein Mensuralnotationsbild als auch eine moderne Edition aus der selben Quelle erstellt.

17.6.1 Incipite

In Arbeit.

17.6.2 Mensurstriche

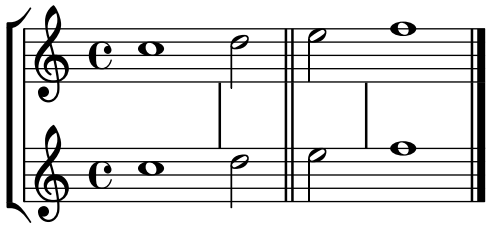
Als *Mensurstriche* wird ein Notenlayout bezeichnet, in dem die Taktlinien nicht auf den Systemen, sondern nur zwischen Systemen gezogen werden. Damit soll signalisiert werden, dass das Original keine Takteinteilung besessen hat und etwa Synkopen nicht über Taktlinien hinweg aufgeteilt werden müssen, während man sich dennoch an den Taktlinien rhythmisch orientieren kann.

Das Mensurstiche-Layout, in welchem die Taktlinien nicht auf den Systemen, sondern zwischen den Systemen gesetzt werden, kann mit einer `StaffGroup` anstelle von `ChoirStaff` erreicht werden. Die Taktlinien auf den Systemen werden mit der `transparent`-Eigenschaft ausgelöscht.

```
\layout {
  \context {
    \Staff
    measureBarType = "-span|"
  }
}

music = \fixed c'' {
  c1
  d2 \section e2
  f1 \fine
}

\new StaffGroup <<
  \new Staff \music
  \new Staff \music
>>
```



17.6.3 Gregorianischen Choral transkribieren

Gregorianischer Choral kann mit einigen einfachen Einstellungen in moderner Notation notiert werden.

Hälse. Hälse können meistens weggelassen werden, was geschieht, indem man den `Stem_engraver` aus dem Stimmenkontext entfernt:

```
\layout {
  ...
  \context {
    \Voice
      \remove Stem_engraver
  }
}
```

In einigen Transkriptionsstilen werden jedoch teilweise Hälse eingesetzt, um etwa den Übergang von einem Einton-Rezitativ zu einer melodischen Geste anzuzeigen. In diesem Fall können Hälse entweder mit `\hide Stem` unsichtbar gemacht werden oder mit `\override Stem.length = #0` auf die Länge von 0 reduziert werden. Die Hälse müssen dann wieder an den entsprechenden Stellen mit `\once \override Stem.transparent = ##f` sichtbar gemacht werden (siehe auch Beispiel unten). Wenn Hälse eingesetzt werden, die Fähnchen haben, muss zusätzlich auch noch `\hide Flag` eingestellt werden.

Takt. Für Gesang ohne Metrum gibt es einige Alternativen.

Der `Time_signature_engraver` kann aus dem `Staff`-Kontext entfernt werden, ohne dass es negative Seiteneffekte gäbe. Alternativ kann er durchsichtig gemacht werden, dabei entsteht

aber ein leerer Platz zu Beginn der Noten an der Stelle, wo normalerweise die Taktangabe stehen würde.

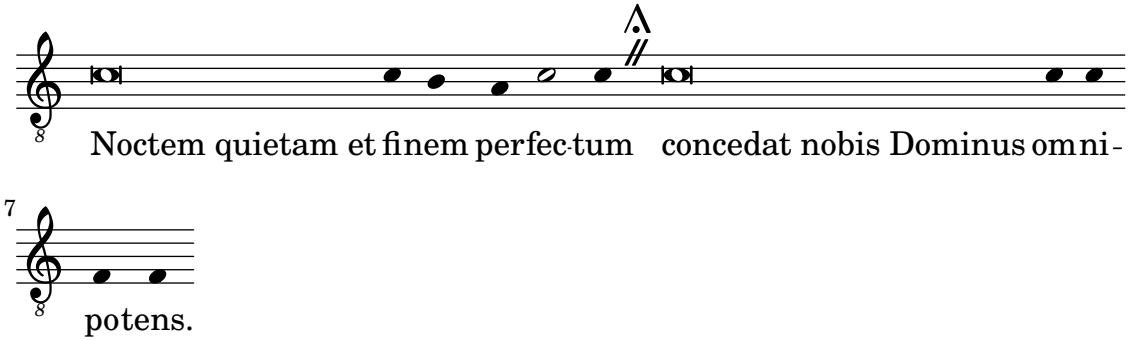
In vielen Fällen ergibt `\set Score.timing = ##f` gute Ergebnisse. Eine andere Möglichkeit ist es, `\cadenzaOn` und `\cadenzaOff` zu benutzen.

Um Taktstriche zu entfernen, kann man radikal den `Bar_engraver` aus dem `Staff`-Kontext entfernen. Wenn man ab und zu einen Taktstrich braucht, sollten die Striche nur mit `\hide BarLine` unsichtbar gemacht werden.

Oft werden Rezitativtöne mit einer Brevis angezeigt. Der Text für die Rezitativnote kann auf zwei Arten notiert werden: entweder als einzelne, links ausgerichtete Silbe:

```
chant = \relative {
  \clef "G_8"
  c'\breve c4 b4 a c2 c4  \divisioMaior
  c\breve c4 c f, f \finalis
}

verba = \lyricmode {
  \once \override LyricText.self-alignment-X = #-1
  "Noctem quietam et" fi -- nem per -- fec -- tum
  \once \override LyricText.self-alignment-X = #-1
  "concedat nobis Dominus" om -- ni -- po -- tens.
}
\score {
  \new Staff <<
  \new Voice = "melody" \chant
  \new Lyrics = "one" \lyricsto melody \verba
  >>
  \layout {
    \context {
      \Staff
      \remove Time_signature_engraver
      \remove Bar_engraver
      \hide Stem
    }
  }
}
```
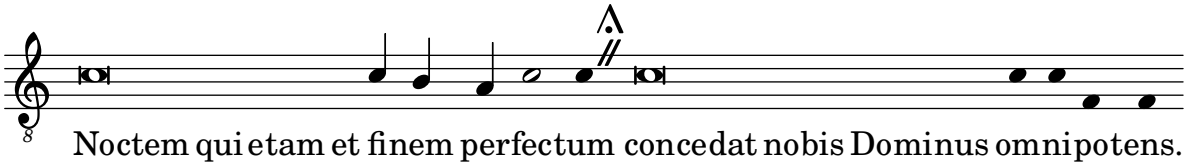


Das funktioniert gut, solange der Text nicht über einen Zeilenumbruch reicht. In diesem Fall kann man etwa die Noten der Silben verstecken (hier werden auch die Hälse unsichtbar gemacht):

```
chant = \relative {
  \clef "G_8"
  \set Score.timing = ##f
  c'\breve \hide NoteHead  c c c c c
  \undo \hide NoteHead
  \override Stem.transparent = ##f \stemUp c4 b4 a
  \hide Stem c2 c4  \divisioMaior
  c\breve \hide NoteHead c c c c c c c
  \undo \hide NoteHead c4 c f, f \finalis
}

verba = \lyricmode {
  No -- ctem qui -- e -- tam et fi -- nem per -- fec -- tum
  con -- ce -- dat no -- bis Do -- mi -- nus om -- ni -- po -- tens.
}

\score {
  \new Staff <<
    \new Voice = "melody" \chant
    \new Lyrics \lyricsto "melody" \verba
  >>
  \layout {
    \context {
      \Staff
      \remove Time_signature_engraver
      \hide BarLine
      \hide Stem
    }
  }
}
```

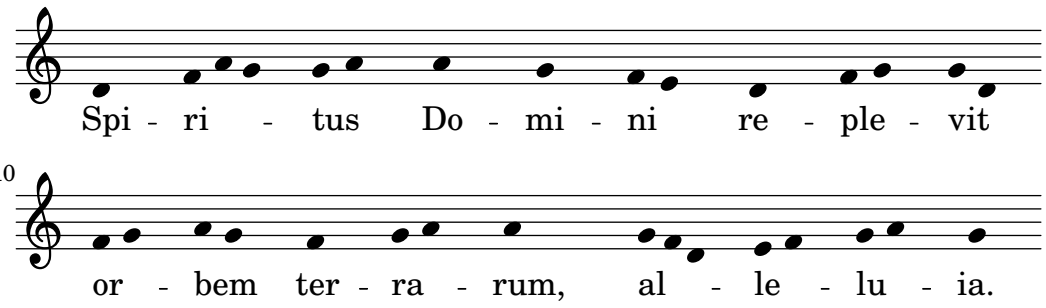


Eine andere übliche Situation ist die Transkription von neumatischem oder melismatischem Gesang, d.h. Gesang, der eine unterschiedliche Anzahl von Noten pro Silbe hat. In diesem Fall sollen die Silbengruppen üblicherweise deutlich voneinander getrennt gesetzt werden, oft auch die Untergruppen eines längeren Melismas. Eine Möglichkeit, das zu erreichen, ist es, eine feste Taktart, etwa 1/4, zu benutzen und dann jeder Silbe oder Notengruppe einen ganzen Takt zuzuweisen, u.U. mit Hilfe von Triolen und kleinen Notenwerten. Wenn die Taktstriche und alle anderen rhythmischen Anweisungen unsichtbar gemacht werden, und der Platz um die Taktstriche vergrößert wird, ergibt sich eine recht gute Repräsentation der Originalnotation.

Damit Silben mit unterschiedlicher Länge (etwa „-ri“ und „-rum“) die Silbengruppen nicht ungleichmäßig aufweiten, kann die `'X-extent`-Eigenschaft des `LyricText`-Objekts auf einen festen Wert gesetzt werden. Eine andere Möglichkeit wäre es, die Silben als Textbeschriftung einzufügen. Wenn weitere horizontale Anpassungen nötig sind, können sie mit unsichtbaren (`s`)-Noten vorgenommen werden.

```
spiritus = \relative {
  \time 1/4
  \override Lyrics.LyricText.X-extent  = #'(0 . 3)
  d'4 \tuplet 3/2 { f8 a g } g a a4 g f8 e
  d4 f8 g g8 d f g a g f4 g8 a a4  s
  \tuplet 3/2 { g8 f d } e f g a g4
}

spirLyr = \lyricmode {
  Spi -- ri -- _ _ tus  _ Do -- mi -- ni  _ re -- ple -- _ vit _
  or -- _ bem _  ter -- ra -- _ rum, al -- _ _ le -- _ lu
  -- _ ia.
}
\score {
  \new Staff <<
    \new Voice = "chant" \spiritus
    \new Lyrics = "one" \lyricsto "chant" \spirLyr
  >>
  \layout {
    \context {
      \Staff
      \remove Time_signature_engraver
      \override BarLine.X-extent = #'(-1 . 1)
      \hide Stem
      \hide Beam
      \hide BarLine
      \hide TupletNumber
    }
  }
}
```



17.6.4 Alte und moderne Edition aus einer Quelldatei

In Arbeit.

18 Weltmusik

Dieser Abschnitt soll Besonderheiten der Notation aufzeigen, die insbesondere relevant sind, um Musik nicht-westlicher Tradition zu notieren.

18.1 Übliche Notation für nichteuropäische Musik

Dieser Abschnitt zeigt, wie man Partituren erstellt, die nicht der europäischen klassischen Musiktradition angehören.

18.1.1 Erweiterung von Notation und Stimmungssystemen

Die klassische Standardnotation wird üblicherweise zur Notation verschiedenster Musikarten benutzt und ist nicht auf die „klassische Musik“ beschränkt. Diese Notation wird behandelt in Abschnitt 1.1 [Tonhöhen setzen], Seite 3, und die unterschiedlichen Notenbezeichnungen, die eingesetzt werden können, finden sich in Abschnitt 1.1.4 [Notenbezeichnungen in anderen Sprachen], Seite 9.

Viele nicht-europäische Musik (und auch manche europäische Volksmusik) benutzt jedoch alternative oder erweiterte Skalen (Tonleitern), die man nicht mit der normalen westlichen Notation notieren kann.

In einigen Fällen wird die klassische Notation dennoch benutzt, wobei man die Tonhöhenunterschiede implizit mitliest. Beispielsweise arabische Musik wird mit normalen Halb- und Vierteltonversetzungszeichen notiert und die exakte Tonhöhe (die etwas von der notierten abeweichen kann) dann aus dem Kontext erschlossen. Italienische Notenbezeichnungen werden normalerweise benutzt, und die Init-Datei `arabic.ly` stellt eine Anzahl an Makros zur Verfügung, die die Standardnoation erweitern. Siehe auch Abschnitt 18.2 [Arabische Musik], Seite 449.

Andere Musik brauchen erweiterte oder ganz einzigartige Notation. Die klassische Musik der Türkei, oder ottomanische Musik, benutzt melodische Formen, die als *makamlar* bekannt sind und deren Intervalle auf 1/9-Bruchteilen des Ganztones beruhen. Die normale europäische Notation wird trotzdem auf dem System mit normalen Noten benutzt mit speziellen türkischen Versetzungszeichen. Diese Versetzungszeichen sind in der Datei `makam.ly` definiert. Zu weiterer Information über die klassische türkische Musik und Makamlar, siehe Abschnitt 18.3 [Türkische klassische Musik], Seite 454.

Um Dateien wie `arabic.ly` oder `makam.ly` zu finden, siehe Abschnitt “Mehr Information” in *Handbuch zum Lernen*.
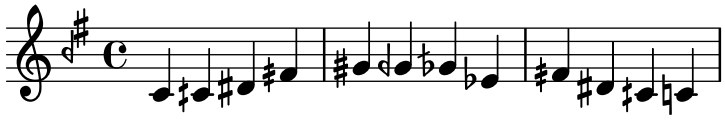
Ausgewählte Schnipsel

Makam-Beispiel

Makam ist eine türkische Melodie, in der 1/9-Tonabstände eingesetzt werden. Sehen Sie sich die Initialisierungsdatei `makam.ly` für weiter Information zu Tonhöhenbezeichnungen und Alterationen an (siehe Handbuch zum Lernen 2.27.4, 4.6.3 Weitere Information zu Hinweisen, wo diese Datei gespeichert ist).

```
\include "makam.ly"

\relative c' {
  \set Staff.keyAlterations = #`((6 . ,(- KOMA)) (3 . ,BAKIYE))
  c4 cc db fk
  gbm4 gfc gfb efk
  fk4 db cc c
}
```



Siehe auch

Glossar: Abschnitt “Common Practice Period” in *Glossar*, Abschnitt “makamlar” in *Glossar*.

Handbuch zum Lernen: Abschnitt “Mehr Information” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 1.1 [Tonhöhen setzen], Seite 3, Abschnitt 1.1.4 [Notenbezeichnungen in anderen Sprachen], Seite 9, Abschnitt 18.2 [Arabische Musik], Seite 449, Abschnitt 18.3 [Türkische klassische Musik], Seite 454.

18.2 Arabische Musik

Dieser Abschnitt zeigt Möglichkeiten, wie arabische Musik notiert werden kann.

18.2.1 Referenz für arabische Musik

Arabische Musik wurde bisher vor allem mündlich tradiert. Wenn Musik transkribiert wird, handelt es sich meistens um ein Gerüst, auf dem der Musiker eigene Improvisationen ausführt. Mehr und mehr wird die westliche Notation mit einigen Veränderungen benutzt, um die arabische Musiktradition weiterzugeben und zu konservieren.

Einige Elemente der westlichen Notation wie etwa die Transkription von Akkorden oder eigenständige Stimmen werden für die traditionelleren arabischen Noten nicht benötigt. Es gibt allerdings einige andere Probleme, wie etwa die Notwendigkeit, Zwischenintervalle zu notieren, die sich irgendwo zwischen einem Halbton und einem Ganzton befinden. Daneben werden auch die westlichen Halb- und Ganztöne eingesetzt. Es muss auch möglich sein, eine große Anzahl an maqam (Modi) der arabischen Musik zu bezeichnen und zu gruppieren.

Üblicherweise müssen Mikrotöne in der arabischen Musik nicht präzise notiert werden.

Einige Bereiche, die für die arabische Notation wichtig sind, sind an anderer Stelle behandelt:

- Notenbezeichnungen und Versetzungszeichen (inklusive Vierteltöne) können angepasst werden, wie behandelt in Abschnitt 18.1 [Übliche Notation für nichteuropäische Musik], Seite 448.
- Zusätzliche Taktarten können erstellt werden, siehe Abschnitt 1.3.2 [Tonartbezeichnung], Seite 22.
- Komplexe Taktarten erfordern evtl., dass Noten manuell gruppiert werden, wie gezeigt in Abschnitt 2.4.3 [Manuelle Balken], Seite 94.
- *Takasim*, rhythmisch freie Improvisationen, können ohne Taktlinien notiert werden, siehe hierzu Abschnitt 2.3.4 [Musik ohne Metrum], Seite 73.

Siehe auch

Notationsreferenz: Abschnitt 18.1 [Übliche Notation für nichteuropäische Musik], Seite 448, Abschnitt 1.3.2 [Tonartbezeichnung], Seite 22, Abschnitt 2.4.3 [Manuelle Balken], Seite 94.

Schnipsel: Abschnitt “World music” in *Schnipsel*.
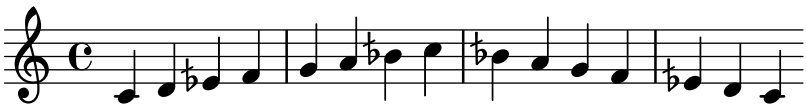
18.2.2 Arabische Notenbezeichnungen

An der arabischen Tradition orientierte Notenbezeichnungen können sehr land sein und eignen sich daher nicht gut für die Notation von Musik. Sie werden nicht benutzt. Englische Notenbezeichnungen hingegen sind in der arabischen Musikerziehung recht unbekannt, weshalb italienische Notenbezeichnungen (`do`, `re`, `mi`, `fa`, `sol`, `la`, `si`) eingesetzt werden. Modifikatoren (Versetzungszeichen) können auch benutzt werden. Italienische Notenbezeichnungen finden sich

erklärt in Abschnitt 1.1.4 [Notenbezeichnungen in anderen Sprachen], Seite 9, die Benutzung der normalen europäischen Notation für nichteuropäische Musik findet sich erklärt in Abschnitt 18.1 [Übliche Notation für nichteuropäische Musik], Seite 448.
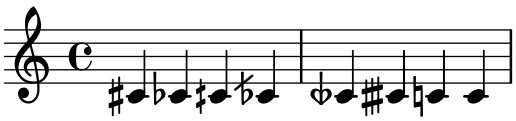
Hier ein Beispiel der arabischen *rast*-Tonleiter:

```
\include "arabic.ly"
\relative {
  do' re misb fa sol la sisb do sisb la sol fa misb re do
}
```



Das Symbol für das Halb-B sieht anders aus als das Symbol, was üblicherweise in arabischer Notation benutzt wird. Das `\dwn`-Symbol, das in der Datei `arabic.ly` definiert ist, kann als ein Workaround eingesetzt werden, wenn es notwendig ist, das arabische Symbol zu benutzen. Das Aussehen des Halb-Bs in den Vorzeichen kann mit dieser Methode nicht verändert werden.

```
\include "arabic.ly"
\relative {
  \set Staff.extraNatural = ##f
  dod' dob dosd \dwn dob dobsb dodsd do do
}
```



Siehe auch

Notationsreferenz: Abschnitt 1.1.4 [Notenbezeichnungen in anderen Sprachen], Seite 9, Abschnitt 18.1 [Übliche Notation für nichteuropäische Musik], Seite 448.

Schnipsel: Abschnitt “World music” in *Schnipsel*.

18.2.3 Arabische Tonarten

Neben den westlichen Dur- und Moll-Tonarten sind folgende Tonarten in `arabic.ly` definiert: *bayati*, *rast*, *sikah*, *iraq* und *kurd*. Diese Tonarten definieren eine kleine Gruppe von Maqams, die weitverbreitet sind.

Ein Maqam kann die Tonart der Gruppe benutzen, zu der er gehört, oder die einer benachbarten Gruppe. Zusätzlich können verschiedene Versetzungszeichen in den Noten markiert werden.

Um also etwa die Tonart des Maqams „muhayer“ folgendermaßen notiert:

```
\key re \bayati
```

re ist die Tonhöhe für den „muhayer“-Maqam und *bayati* ist die Bezeichnung des Basismaqams der Gruppe.

Während die Vorzeichen eine Gruppe anzeigen, wird meistens der eigentliche Maqam im Titel definiert. In diesem Beispiel müsste also der „muhayer“-Maqam im Titel erscheinen.

Andere Maqams derselben Bayati-Gruppe, wie in der Tabelle unten gezeigt ((bayati, hussaini, saba und ushaq) können auf die gleiche Weise notiert werden. Sie sind alle Variationen des Grundmaqams Bayati. Sie unterscheiden sich üblicherweise vom grundlegenden Maqam in ihrem oberen Tetrachord oder in bestimmten Einzelheiten, die aber nicht ihre eigentliche Qualität verändern.

Der andere Maqam der gleichen Gruppe (Nawa) ist mit bayati durch eine Modulation verwandt, deren Grundton in der Tabelle angezeigt wird, wenn es sich um einen Maqam handelt, der eine Modulation eines anderen Maqams darstellt. Nawa kann folgenderweise notiert werden:

```
\key sol \bayati
```

In der arabischen Musik ist ein Begriff wie bayati, der eine Maqam-Gruppe bezeichnet, gleichzeitig auch selber ein Maqam, meistens der häufigste dieser Gruppe.

Hier ist eine Möglichkeit, Maqams zu gruppieren, womit die häufigsten Maqams bestimmten Vorzeichen zugeordnet werden:

Maqam-Gruppe	**Vorzeichen (\key)**	**Finalis**	**Andere Maqams der Gruppe (Finalis)**
ajam	major	sib	jaharka (fa)
bayati	bayati	re	hussaini, muhayer, saba, ushaq, nawa (sol)
hijaz	kurd	re	shahnaz, shad arban (sol), hijazkar (do)
iraq	iraq	sisb	-
kurd	kurd	re	hijazkar kurd (do)
nahawand	minor	do	busalik (re), farah faza (sol)
nakriz	minor	do	nawa athar, hisar (re)
rast	rast	do	mahur, yakah (sol)
sikah	sikah	misb	huzam

Ausgewählte Schnipsel

Untypische Tonarten

Der üblicherweise benutzte \key-Befehl setzt die keySignature-Eigenschaft im Staff-Kontext.
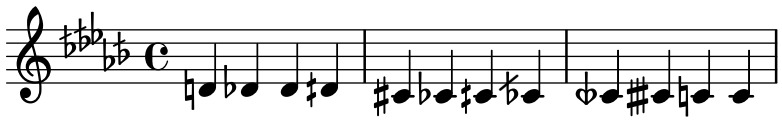
Um untypische Tonartenvorzeichen zu erstellen, muss man diese Eigenschaft direkt setzen. Das Format für den Befehl ist eine Liste: `\set Staff.keySignature = #`(((Oktave . Schritt) . Alteration) ((Oktave . Schritt) . Alteration) ...)` wobei für jedes Element in der Liste `Oktave` die Oktave angibt (0 ist die Oktave vom eingestrichenen C bis zum eingestrichenen H), `Schritt` gibt die Note innerhalb der Oktave an (0 heißt C und 6 heißt H), und `Alteration` ist `,SHARP` `,FLAT` `,DOUBLE-SHARP` usw. (Beachte das beginnende Komma.)

Alternativ kann auch jedes Element der Liste mit dem allgemeineren Format `(Schritt . Alteration)` gesetzt werden, wobei dann die Einstellungen für alle Oktaven gelten.

Hier ein Beispiel einer möglichen Tonart für eine Ganztonleiter:

```
\include "arabic.ly"

\relative do' {
  \set Staff.keyAlterations = #`((0 . ,SEMI-FLAT)
                                 (1 . ,SEMI-FLAT)
                                 (2 . ,FLAT)
                                 (5 . ,FLAT)
                                 (6 . ,SEMI-FLAT))
  % \set Staff.extraNatural = ##f
  re reb \dwn reb resd
  dod dob dosd \dwn dob |
  dobsb dodsd do do |
}
```



Siehe auch

Glossar: Abschnitt “maqam” in *Glossar*, Abschnitt “bayati” in *Glossar*, Abschnitt “rast” in *Glossar*, Abschnitt “sikah” in *Glossar*, Abschnitt “iraq” in *Glossar*, Abschnitt “kurd” in *Glossar*.

Notationsreferenz: Abschnitt 1.3.2 [Tonartbezeichnung], Seite 22.

Handbuch zum Lernen: Abschnitt “Tonhöhen und Tonartbezeichnungen (Vorzeichen)” in *Handbuch zum Lernen*.

Referenz der Interna: Abschnitt “KeySignature” in *Referenz der Interna*.

Schnipsel: Abschnitt “World music” in *Schnipsel*, Abschnitt “Pitches” in *Schnipsel*.

18.2.4 Arabische Taktarten

Einige klassische Formen der arabischen und türkischen Musik wie etwa *Semai* haben ungewöhnliche Taktarten wie etwa 10/8. Das kann dazu führen, dass die automatische Bebalkung der Noten nicht zu dem Ergebnis kommt, welches in der üblichen Notation dieser Musik eingesetzt wird. Die Noten werden nicht anhand einer Taktzeit, sondern anhand von Kriterien gruppiert, die man schwer mit einer automatischen Balkenfunktion erfassen kann. Das kann umgangen werden, indem die automatische Bebalkung ausgeschaltet wird und die Balken explizit gesetzt werden. Auch wenn es nicht darauf ankommen sollte, eine schon notierte Musik nachzuahmen, ist es in vielen Fällen dennoch erforderlich, die Bebalkung anzupassen und/oder zusammengesetzte Taktarten zu benutzen.

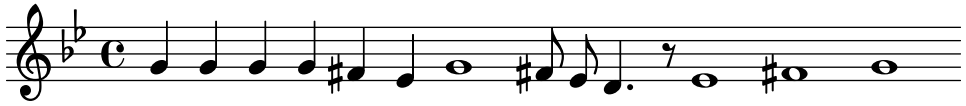
Ausgewählte Schnipsel

Arabische Improvisation

Bei Improvisation oder *taqasim*, die zeitlich frei gespielt werden, kann die Taktart ausgelassen werden und `\cadenzaOn` kann eingesetzt werden. Es kann nötig sein, den Versetzungszeichenstil anzupassen, weil sonst die Versetzungszeichen nur einmal ausgegeben werden, da keine Taktlinien gesetzt sind. Hier ein Beispiel, wie der Beginn einer *hijaz*-Improvisation aussehen könnte:

```
\include "arabic.ly"

\relative sol' {
  \key re \kurd
  \accidentalStyle forget
  \cadenzaOn
  sol4 sol sol sol fad mib sol1 fad8 mib re4. r8 mib1 fad sol
}
```



Siehe auch

Glossar: Abschnitt “semai” in *Glossar*, Abschnitt “taqasim” in *Glossar*.

Notationsreferenz: Abschnitt 2.4.3 [Manuelle Balken], Seite 94, Abschnitt 2.4.1 [Automatische Balken], Seite 81, Abschnitt 2.3.4 [Musik ohne Metrum], Seite 73, Abschnitt 1.3.5 [Automatische Versetzungszeichen], Seite 26, Abschnitt 2.4.2 [Einstellung von automatischen Balken], Seite 84, Abschnitt 2.3.1 [Taktangabe], Seite 65.
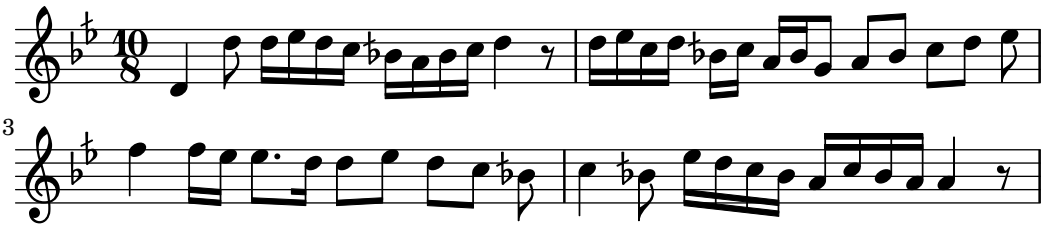
Schnipsel: Abschnitt “World music” in *Schnipsel*.

18.2.5 Arabische Notenbeispiele

Hier eine Vorlage, welche den Beginn eines türkischen *Semai* benutzt, der in der arabischen Musikerziehung oft herangezogen wird, um Besonderheiten der arabischen Musiknotation, wie etwa Zwischenintervalle und ungewöhnliche Modi, zu illustrieren.

```
\include "arabic.ly"
\score {
  \header {
    title = "Semai Muhayer"
    composer = "Jamil Bek"
  }
  \relative {
    \set Staff.extraNatural = ##f
    \set Staff.autoBeaming = ##f
    \key re \bayati
    \time 10/8

    re'4 re'8 re16 [misb re do] sisb [la sisb do] re4 r8
    re16 [misb do re] sisb [do] la [sisb sol8] la [sisb] do [re] misb
    fa4 fa16 [misb] misb8. [re16] re8 [misb] re  [do] sisb
    do4 sisb8 misb16 [re do sisb] la [do sisb la] la4 r8
  }
}
```



Siehe auch

Schnipsel: Abschnitt “World music” in *Schnipsel*.

18.2.6 Weitere Literatur zur arabischen Musik

1. *The Music of the Arabs* von Habib Hassan Touma (Amadeus Press, 1996) enthält eine Beschreibung von Maqams und Methoden zu ihrer Gruppierung.

   Es gibt auch einige Internetseiten, die Maqams erklären und teilweise auch Klangdateien zur Verfügung stellen:

   - `https://www.maqamworld.com/`
   - `https://www.turath.org/`

   Die Maqam-Gruppierungen unterscheiden sich in einigen Details, auch wenn die allgemeinen Kriterien weithin anerkannt sind: gemeinsame untere Tetrachorde sowie Modulation.
2. Es gibt keine Übereinstimmung darüber, wie die Vorzeichen für bestimmte Maqams angegeben werden sollen. Oft wird eine Vorzeichenart für eine ganze Maqam-Gruppe verwendet, anstatt dass jeder Maqam eigene Vorzeichen hätte.

   Lehrbücher für *Oud*, die arabische Laute, folgender Autoren enthalten Beispiele vor allem türkischer und arabischer Kompositionen:

   - Charbel Rouhana
   - George Farah
   - Ibrahim Ali Darwish Al-masri

18.3 Türkische klassische Musik

Dieser Abschnitt zeigt Probleme, die bei der Notation von klassischer türkischer Musik auftreten können.

18.3.1 Verweise für türkische klassische Musik

Türkische klassische Musik wurde im Osmanischen Reich während einer Periode entwickelt, die ungefähr zur gleichen Zeit der westlichen klassischen Musik stattfand. Diese lebendige und starke Tradition wird bis heute mit ihren eigenen kompositorischen Formen, Musiktheorie und Aufführungsstilen weitergeführt. Unter den Eigenheiten dieser Tradition befinden sich die Benutzung von Mikrointervallen basierend auf „Kommas“ von 1/9-Tönen, aus denen melodische Formen konstruiert werden, welche man als *makam* (Pl. *makamlar*) bezeichnet.

Einige Probleme der Notation türkischer klassischer Musik sind woanders behandelt:

- Notenbezeichnungen und Versetzungszeichen finden sich in Abschnitt 18.1 [Übliche Notation für nichteuropäische Musik], Seite 448.

18.3.2 Türkische Notenbezeichnungen

Tonhöhen der türkischen klassischen Musik haben traditionell einmalige Bezeichnungen, und weil die Tonhöhen auf 1/9-Tönen basieren, unterscheiden sich die Intervalle von makamlar deutlich von den Intervallen westlicher klassischer Musik: *koma* (1/9 eines Ganztons), *eksik bakiye* (3/9), *bakiye* (4/9), *kücük mücenneb* (5/9), *büyük mücenneb* (8/9), *tanîni* (ein Ganzton) und *artık ikili* (12/9 oder 13/9 eines Ganztons).

Es bietet sich an, die normalen westlichen Noten auf dem Notensystem zu benutzen (also c, d, e . . .) angrereichert mit besonderen Versetzungszeichen, die die Noten um 1/9, 4/9, 5/9 und 8/9 eines Ganztons erhöhen oder erniedrigen. Diese Versetzungszeichen sind definiert in der Datei `makam.ly`.

Die folgende Tabelle zeigt

- die Bezeichnung dieser besonderen Versetzungszeichen
- die Endung, die an die Note gehängt werden muss und
- die Tonhöhenveränderung als Bruch eines Ganztones.

Versetzungszeichen	**Endung**	**Tonhöhenveränderung**
büyük mücenneb (Kreuz)	-bm	+8/9
kücük mücenneb (Kreuz)	-k	+5/9
bakiye (Kreuz)	-b	+4/9
koma (Kreuz)	-c	+1/9
koma (B)	-fc	-1/9
bakiye (B)	-fb	-4/9
kücük mücenneb (B)	-fk	-5/9
büyük mücenneb (B)	-fbm	-8/9

Eine weitergehende Erklärung der Notation nichteuropäischer Musik findet sich in Abschnitt 18.1 [Übliche Notation für nichteuropäische Musik], Seite 448.

Siehe auch

Glossar: Abschnitt “makam” in *Glossar*, Abschnitt “makamlar” in *Glossar*.

Notationsreferenz: Abschnitt 18.1 [Übliche Notation für nichteuropäische Musik], Seite 448.

Allgemeine Eingabe und Ausgabe

19 Eingabestruktur

Das hauptsächliche Eingabeformat von LilyPond sind Textdateien. Üblicherweise werden diese Dateien mit der Endung `.ly` versehen.

19.1 Struktur einer Partitur

Eine `\score`-Umgebung muss einen einzelnen musikalischen Ausdruck beinhalten, der durch geschweifte Klammern begrenzt wird:
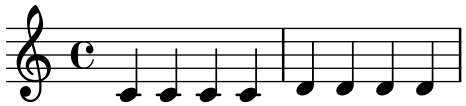
```
\score {
...
}
```

> **Achtung:** Es darf **nur ein** äußerer musikalischer Ausdruck in der `\score`-Umgebung geschrieben werden, und er **muss** von geschweiften Klammern umgeben sein.

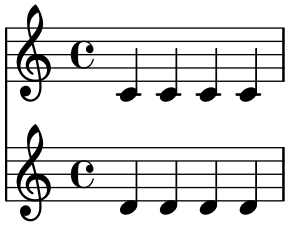
Dieser einzelne musikalische Ausdruck kann beliebige Größe annehmen und andere musikalische Ausdrücke von beliebiger Komplexität beinhalten. Alle diese Beispiele sind musikalische Ausdrücke:

```
{ c'4 c' c' c' }
```

```
{
  { c'4 c' c' c' }
  { d'4 d' d' d' }
}
```



```
<<
  \new Staff { c'4 c' c' c' }
  \new Staff { d'4 d' d' d' }
>>
```



```
{
  \new GrandStaff <<
    \new StaffGroup <<
      \new Staff { \Flöte }
      \new Staff { \Oboe }
    >>
    \new StaffGroup <<
      \new Staff { \GeigeI }
      \new Staff { \GeigeII }
    >>
```

```
  >>
}
```

Kommentare bilden eine Ausnahme dieser Regel. (Andere Ausnahmen siehe Abschnitt 19.5 [Die Dateistruktur], Seite 461.) Sowohl einzeilige als auch Blockkommentare (eingegrenzt durch %{ .. %}) können an beliebiger Stelle einer Eingabedatei geschrieben werden. Sie können innerhalb oder außerhalb der \score-Umgebung vorkommen, und innerhalb oder außerhalb des einzelnen musikalischen Ausdrucks innerhalb der \score-Umgebung.

Denken Sie daran, dass auch eine Datei, die nur eine \score-Umgebung enhält, implizit in eine \book-Umgebung eingeschlossen wird. Eine \book-Umgebung in einer Eingabdatei produziert wenigstens eine Ausgabedatei, und standardmäßig wird der Name der Ausagabedatei aus dem Namen der Eingabedatei abgeleitet. `fandangoforelephants.ly` produziert also `fandangoforelephants.pdf`.

Zu weiteren Einzelheiten zu \book-Umgebungen siehe Abschnitt 19.2 [Mehrere Partituren in einem Buch], Seite 458, Abschnitt 19.3 [Mehrere Ausgabedateien aus einer Eingabedatei], Seite 459, und Abschnitt 19.5 [Die Dateistruktur], Seite 461.

Siehe auch

Handbuch zum Lernen: Abschnitt “Arbeiten an Eingabe-Dateien” in *Handbuch zum Lernen*, Abschnitt “Musikalische Ausdrücke erklärt” in *Handbuch zum Lernen*, Abschnitt “Score ist ein (einziger) zusammengesetzter musikalischer Ausdruck” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 19.2 [Mehrere Partituren in einem Buch], Seite 458, Abschnitt 19.3 [Mehrere Ausgabedateien aus einer Eingabedatei], Seite 459, und Abschnitt 19.5 [Die Dateistruktur], Seite 461.

19.2 Mehrere Partituren in einem Buch

Eine Partitur kann mehrere musikalische Stücke und verschiedene Texte beinhalten. Beispiele hierzu sind etwa eine Etüdensammlung oder ein Orchesterstück mit mehreren Sätzen. Jeder Satz wird in einer eigenen \score-Umgebung notiert:

```
\score {
  ..Noten..
}
```

und Texte werden mit einer \markup-Umgebung geschrieben:

```
\markup {
  ..Text..
}
```

Alle Sätze und Texte, die in derselben `.ly`-Datei vorkommen, werden normalerweise in eine einzige Ausgabedatei gesetzt.

```
\score {
  ..
}
\markup {
  ..
}
\score {
  ..
}
```

Eine wichtige Ausnahme stellen Dokumente dar, die mit lilypond-book erstellt werden, für die Sie explizit \book-Umgebungen notieren müssen, weil sonst nur die erste \score- bzw. \markup-Umgebung angezeigt wird.

Der Kopfbereich für jedes Musikstück kann innerhalb der \score-Umgebung definiert werden. Die piece-(Stück)-Bezeichnung aus dieser \header-Umgebung wird vor jedem Satz ausgegeben. Die Überschrift für ein ganzes Buch kann innerhalb von \book notiert werden, aber wenn diese Umgebung fehlt, wird die \header-Umgebung genommen, die auf erster Ebene der Datei notiert ist.

```
\header {
  title = "Acht Miniaturen"
  composer = "Igor Stravinsky"
}
\score {
  \header { piece = "Romanze" }
  ...
}
\markup {
   ..Text der zweiten Strophe..
}
\markup {
   ..Text der dritten Strophe..
}
\score {
  \header { piece = "Menuetto" }
  ...
}
```

Stücke können innerhalb eines Buches mit \bookpart gruppiert werden. Derartige Buchabschnitte werden durch einen Seitenumbruch voneinander getrennt und können wie auch das ganze Buch selber mit einem Titel innerhalb einer \header-Umgebung beginnen.

```
\bookpart {
  \header {
    title = "Buchtitel"
    subtitle = "Erster Teil"
  }
  \score { ... }
  ...
}
\bookpart {
  \header {
    subtitle = "Zweiter Teil"
  }
  \score { ... }
  ...
}
```

19.3 Mehrere Ausgabedateien aus einer Eingabedatei

Wenn Sie mehrere Ausgabedateien aus derselben .ly-Datei haben wollen, können Sie mehrere \book-Umgebungen hinzufügen, wobei jede Umgebung eine neue Ausgabedatei produziert. Wenn Sie keine \book-Umgebung in der Eingabedatei angeben, wird die Datei von LilyPond implizit als eine große \book-Umgebung behandelt, siehe auch Abschnitt 19.5 [Die Dateistruktur], Seite 461.

Wenn man mehrere Dateien aus einer einzigen Eingabedatei erstellt, stellt LilyPond sicher, dass keine der Ausgabedateien der vorhandenen \book-Umgebungen eine andere Ausgabedatei, etwa von der vorherigen \book-Umgebung, überschreibt.

Dies geschieht, indem ein Suffix an den Ausgabenamen für jede \book-Umgebung gehängt wird, die den Dateinamen der Eingabdatei als Grundlage nimmt.

Das Standardverhalten ist es, einen Zahlen-Suffix für die Namen hinzuzufügen, die in Konflikt stehen. Der Code

```
\book {
  \score { ... }
  \layout { ... }
}
\book {
  \score { ... }
  \layout { ... }
}
\book {
  \score { ... }
  \layout { ... }
}
```

produziert also

- eightminiatures.pdf,
- eightminiatures-1.pdf and
- eightminiatures-2.pdf.

19.4 Dateinamen der Ausgabedateien

LilyPond stellt die Möglichkeit zur Verfügung zu kontrollieren, welche Dateinamen für welche Back-ends benutzt werden sollen, wenn die Ausgabedateien erstellt werden.

Im vorhergehenden Abschnitt wurde gezeigt, wir LilyPond gleichnamige Ausgabedateien verhindert, wenn mehrere Ausgabedateien aus derselben Eingabedatei erstellt werden. Es gibt auch die Möglichkeit, eigene Suffixe für jeden \book-Abschnitt zu definieren, sodass man etwa Dateinamen wie eightminiatures-Romanze.pdf, eightminiatures-Menuetto.pdf und eightminiatures-Nocturne.pdf produzieren kann, indem man eine \bookOutputSuffix-Angabe in jede \book-Umgebung einfügt.

```
\book {
  \bookOutputSuffix "Romanze"
  \score { ... }
  \layout { ... }
}
\book {
  \bookOutputSuffix "Menuetto"
  \score { ... }
  \layout { ... }
}
\book {
  \bookOutputSuffix "Nocturne"
  \score { ... }
  \layout { ... }
}
```

Man kann auch einen anderen Dateinamen für die Ausgabedatei einer \book-Umgebung erstellen, indem man \bookOutputName-Angabe macht:

```
\book {
  \bookOutputName "Romanze"
```

```
  \score { ... }
  \layout { ... }
}
\book {
  \bookOutputName "Menuetto"
  \score { ... }
  \layout { ... }
}
\book {
  \bookOutputName "Nocturne"
  \score { ... }
  \layout { ... }
}
```

Die obige Datei produziert folgene Ausgabedateien:

- `Romanze.pdf`,
- `Menuetto.pdf` and
- `Nocturne.pdf`.

19.5 Die Dateistruktur

Eine `.ly`-Datei kann eine beliebige Anzahl an Ausdrücken auf der obersten Ebene beinhalten, wobei ein Ausdruck der obersten Ebene einer der folgenden sein kann:

- Eine Ausgabedefinition, wie `\paper`, `\midi` und `\layout`. Derartige Definitionen auf oberster Ebene verändern die globalen Einstellungen für das ganze „Buch“. Wenn mehr als eine derartige Definition desselben Typs auf oberster Ebene angegeben wird, hat die spätere Vorrang. Für Einzelheiten, wie dadurch die `\layout`-Umgebung beeinflusst wird, siehe Abschnitt 26.1 [Die `\layout`-Umgebung], Seite 532.
- Ein direkter Scheme-Ausdruck, wie etwa `#(set-default-paper-size "a7" 'landscape)` oder `#(ly:set-option 'point-and-click #f)`.
- Eine `\header`-Umgebung. Damit wird die globale Titelei eingestellt. Das ist die Umgebung, in der sich Definition für das ganze Buch befinden, wie Komponist, Titel usw.
- Eine `\score`-Umgebung. Die in ihr enthaltene Partitur wird zusammen mit anderen vorkommenden `\score`-Umgebungen gesammelt und in ein `\book` zusammengefasst. Dieses Verhalten kann verändert werden, indem die Variable `toplevel-score-handler` auf höchster Ebene gesetzt wird. Die Definition des Standards findet sich in der Datei `../scm/lily.scm`.
- Eine `\book`-Umgebung fasst mehrere Sätze (d. h. mehrere `\score`-Umgebungen) logisch in ein Dokument zusammen. Wenn mehrere `\score`-Partituren vorkommen, wird für jede `\book`-Umgebung eine eigene Ausgabedatei erstellt, in der alle in der Umgebung enthaltenen Partituren zusammengefasst sind. Der einzige Grund, explizit eine `\book`-Umgebung zu setzen, ist, wenn mehrere Ausgabedateien aus einer einzigen Quelldatei erstellt werden sollen. Eine Ausnahme sind lilypond-book-Dokumente, in denen eine `\book`-Umgebung explizit hinzugefügt werden muss, wenn mehr als eine `\score`- oder `\markup`-Umgebung im gleichen Beispiel angezeigt werden soll. Dieses Verhalten kann verändert werden, indem die Variable `toplevel-book-handler` auf höchster Ebene gesetzt wird. Die Definition des Standards findet sich in der Datei `../scm/lily.scm`.
- Eine `\bookpart`-Umgebung. Ein Buch (`\book`) kann in mehrere Teile untergliedert sein, indem `\bookpart`-Umgebungen eingesetzt werden. Jeder Buchabschnitt beginnt auf einer neuen Seite und kann eigene Papierdefinitionen in einer `\paper`-Umgebung haben.
- Ein zusammengesetzter musikalischer Ausdruck wie etwa

  ```
  { c'4 d' e'2 }
  ```

Dieses Beispiel wird von LilyPond automatisch in einer \score-Umgebung in einem Buch interpretiert und mit anderen \score-Umgebungen und musikalischen Ausdrücken auf der höchsten Ebene zusammen ausgegeben. Anders gesagt: eine Datei, die nur das obige Beispiel beinhaltet, wird übersetzt zu

```
\book {
  \score {
    \new Staff {
      \new Voice {
        { c'4 d' e'2 }
      }
    }
    \layout { }
  }
  \paper { }
  \header { }
}
```

Dieses Verhalten kann verändert werden, indem die Variable toplevel-music-handler auf der obersten Ebene gesetzt wird. Die Definition des Standards findet sich in der Datei ../scm/lily.scm.

- Eine Textbeschriftung, eine Strophe etwa:

```
\markup {
   2.  Die erste Zeile der zweiten Strophe.
}
```

Textbeschriftungen werden über, zwischen oder unter musikalischen Ausdrücken gesetzt, so wie sie notiert werde.

- Eine Variable, wie

```
foo = { c4 d e d }
```

Sie kann dann später in der Datei eingesetzt werden, indem \foo geschrieben wird. Die Bezeichnung der Variable darf nur aus alphabetischen Zeichen bestehen, keine Zahlen, Unter- oder Bindestriche.

Das folgende Beispiel zeigt drei Dinge, die auf der obersten Ebene notiert werden können:

```
\layout {
  % Zeilen rechtsbündig setzen
  ragged-right = ##t
}

\header {
   title = "Do-re-mi"
}

{ c'4 d' e2 }
```

An einer beliebigen Stelle der Datei kann jede der folgenden lexikalen Anweisungen notiert werden:

- \version
- \include
- \sourcefilename
- \sourcefileline
- Ein einzeiliger Kommentar, beginnend mit %.

- Ein mehrzeiliger Kommentar, umgeben von %{ .. %}.

Leerzeichen zwischen Einheiten in der Eingabe werden generell ignoriert und können nach Belieben weggelassen werden oder hinzugefügt werden, um die Lesbarkeit des Codes zu verbessern. Mindestens ein Leerzeichen sollte jedoch unter folgenden Umständen immer eingesetzt werden, um Fehler zu vermeiden:

- Vor und hinter jeder schließenden oder öffnenden Klammer,
- nach jedem Befehl oder jeder Variable, also jeder Einheit, die mit \ beginnt,
- nach jeder Einheit, die als Scheme-Ausdruck interpretiert werden, also alle Einheiten, die mit # beginnen.
- Alle Einheiten von Scheme-Ausdrücken müssen mit Leerzeichen getrennt werden,
- in Gesangstextabschnitten (`lyricmode`) müssen Leerzeichen zwischen alle Ausdrücke in `\override`- und `\set`-Befehlen gesetzt werden. Insbesondere müssen um Punkte und Gleichzeitszeichen in Befehlen wie `\override Score.LyricText.font-size = #5)` und vor dem gesamten Befehl geschrieben werden.

Siehe auch

Handbuch zum Lernen: Abschnitt “Wie eine LilyPond-Eingabe-Datei funktioniert” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 26.1 [Die `\layout`-Umgebung], Seite 532.

20 Titel

Fast alle gedruckten Noten beinhalten einen Titel und den Namen des Komponisten, teilweise wird auch noch sehr viel mehr Information zur Verfügung gestellt.

20.1 Titel, Kopf- und Fußzeilen erstellen

20.1.1 Wie funktioniert die Titel-Umgebung?

Es gibt zwei Arten von Titelumgebungen: die Hauptumgebung, die über der ersten \score-Umgebung innerhalb eines „book“ notiert wird, und individuelle Titelumgebungen, die innerhalb von \score auftreten können. Textfelder für beide Typen werden in der \header-Umgebung eingegeben.
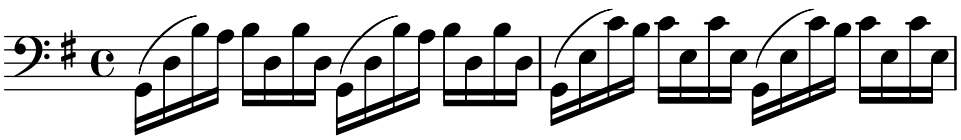
Wenn in dem „book“ nur eine einzelne Partitur vorkommt, kann die \header-Umgebung innheralb oder außerhalb der \score-Umgebung geschrieben werden.

```
\header {
  title = "SUITE I."
  composer = "J. S. Bach."
}
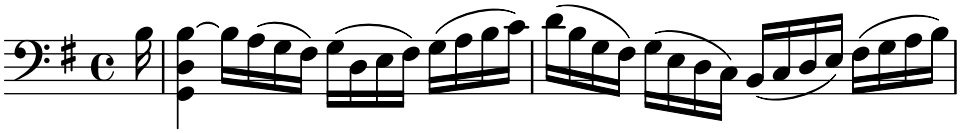
\score {
  \header {
    piece = "Prélude."
  }
  \new Staff \relative {
    \clef bass
    \key g \major
    \repeat unfold 2 { g,16( d' b') a b d, b' d, } |
    \repeat unfold 2 { g,16( e' c') b c e, c' e, } |
  }
}

\score {
  \header {
    piece = "Allemande."
  }
  \new Staff \relative {
    \clef bass
    \key g \major
    \partial 16 b16 |
    <g, d' b'~>4 b'16 a( g fis) g( d e fis) g( a b c) |
    d16( b g fis) g( e d c) b(c d e) fis( g a b) |
  }
}
```





Allemande.



Textfelder der Haupttitelumgebung können in allen \score-Umgebungen gesetzt oder manuell unterdrückt werden:

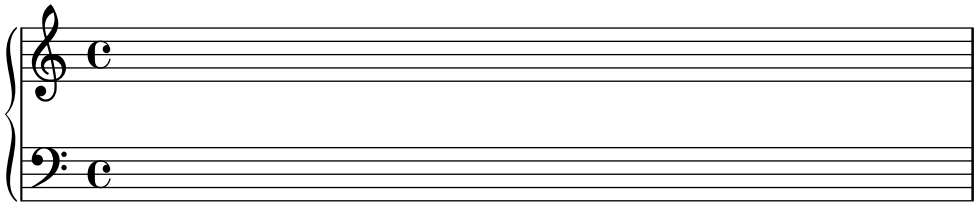
```
\book {
  \paper {
    print-all-headers = ##t
  }
  \header {
    title = "DAS WOHLTEMPERIRTE CLAVIER"
    subtitle = "TEIL I"
    % Do not display the tagline for this book
    tagline = ##f
  }
  \markup { \vspace #1 }
  \score {
    \header {
      title = "PRAELUDIUM I"
      opus = "BWV 846"
      % Do not display the subtitle for this score
      subtitle = ##f
    }
    \new PianoStaff <<
      \new Staff { s1 }
      \new Staff { \clef "bass" s1 }
    >>
  }
  \score {
    \header {
      title = "FUGA I"
      subsubtitle = "A 4 VOCI"
      opus = "BWV 846"
      % Do not display the subtitle for this score
      subtitle = ##f
    }
    \new PianoStaff <<
      \new Staff { s1 }
      \new Staff { \clef "bass" s1 }
    >>
  }
}
```

DAS WOHLTEMPERIRTE CLAVIER

TEIL I

PRAELUDIUM I

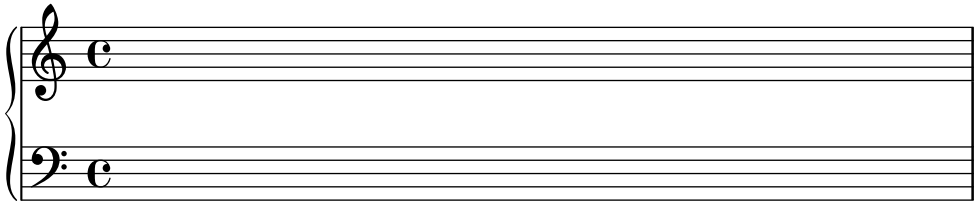
BWV 846



FUGA I

A 4 VOCI

BWV 846



Siehe auch

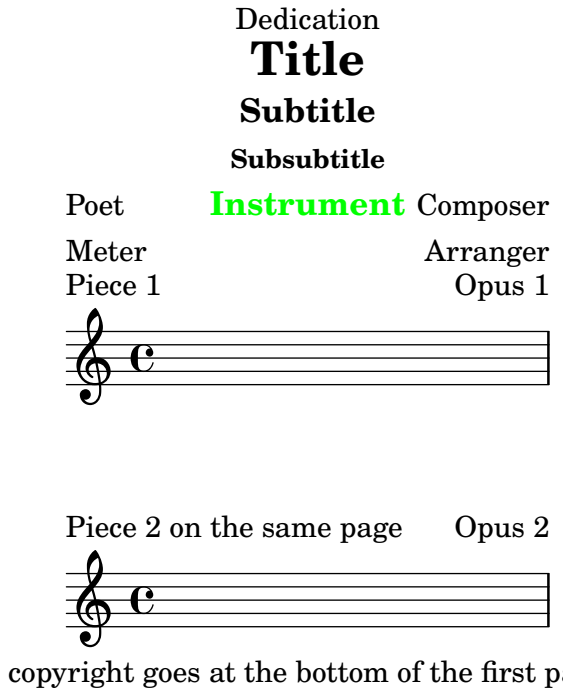
Notationsreferenz: Abschnitt 19.5 [Die Dateistruktur], Seite 461, Abschnitt 20.2.2 [Angepasstes Layout für Titelumgebungen], Seite 471.

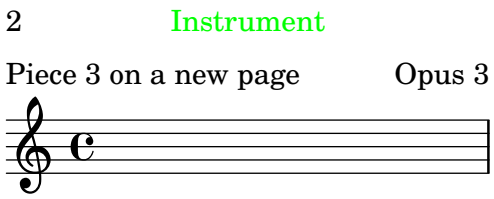
20.1.2 Standardlayout von book- und Partitur-Titelumgebungen

Dieses Beispiel zeigt alle \header-(Überschrifen)-Variablen:

```
\book {
  \header {
      % The following fields are centered
    dedication = "Dedication"
    title = "Title"
    subtitle = "Subtitle"
    subsubtitle = "Subsubtitle"
      % The following fields are evenly spread on one line
      % the field "instrument" also appears on following pages
    instrument = \markup \with-color #green "Instrument"
    poet = "Poet"
    composer = "Composer"
      % The following fields are placed at opposite ends of the same line
    meter = "Meter"
    arranger = "Arranger"
      % The following fields are centered at the bottom
    tagline = "tagline goes at the bottom of the last page"
    copyright = "copyright goes at the bottom of the first page"
```

```
  }
  \score {
    \header {
        % The following fields are placed at opposite ends of the same line
      piece = "Piece 1"
      opus = "Opus 1"
    }
    { s1 }
  }
  \score {
    \header {
        % The following fields are placed at opposite ends of the same line
      piece = "Piece 2 on the same page"
      opus = "Opus 2"
    }
    { s1 }
  }
  \pageBreak
  \score {
    \header {
        % The following fields are placed at opposite ends of the same line
      piece = "Piece 3 on a new page"
      opus = "Opus 3"
    }
    { s1 }
  }
}
```





tagline goes at the bottom of the last page

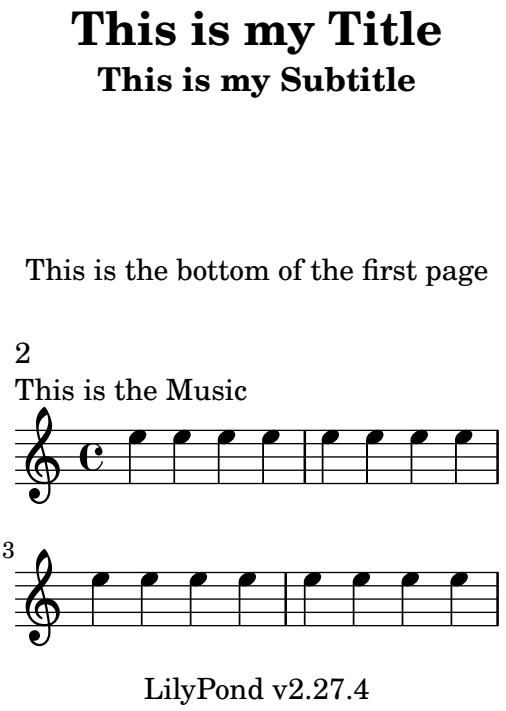
Beachten Sie:

- Die Instrumentenbezeichnung wird auf jeder Seite wiederholt.
- Nur `piece` (Stück)uand `opus` werden für eine Partitur (`\score`) gesetzt, wenn die `\paper`-Variable `print-all-headers` auf `##f` gesetzt ist (Standardeinstellung).
- Textfelder, die in einer `\header`-Umgebung nicht benutzt werden, werden durch `\null`-Textbeschriftung ersetzt, sodass sie keinen leeren Platz belegen.
- Die Standardeinstellungen von `scoreTitleMarkup` platzieren die Felder `piece` (Stück) und `opus` zu den gegenüberliegenden Seiten der selben Zeile.

Um die Standardeinstellungen des Layouts zu ändern, siehe Abschnitt 20.2.2 [Angepasstes Layout für Titelumgebungen], Seite 471.

Mit der Variable `breakbefore` innerhalb einer `\header`-Umgebung, die für sich auch eine `\score`-Umgebung darstellt, kann man die Hauptüberschriften auf der ersten Seite allein ausgeben, sodass die Noten (in der `score`-Umgebung definiert) erst auf der folgenden Seite beginnen.

```
\book {
  \header {
    title = "This is my Title"
    subtitle = "This is my Subtitle"
    copyright = "This is the bottom of the first page"
  }
  \score {
    \header {
      piece = "This is the Music"
      breakbefore = ##t
    }
    \repeat unfold 4 { e'' e'' e'' e'' }
  }
}
```



Siehe auch

Handbuch zum Lernen: Abschnitt “Wie eine LilyPond-Eingabe-Datei funktioniert” in *Handbuch zum Lernen*,

Notationsreferenz: Abschnitt 19.5 [Die Dateistruktur], Seite 461, Abschnitt 20.2.2 [Angepasstes Layout für Titelumgebungen], Seite 471.

Installierte Dateien: `ly/titling-init.ly`.
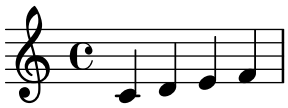
20.1.3 Standardlayout von Kopf- und Fußzeilen

Kopf- und *Fußzeilen* sind Textzeilen, die ganz oben und ganz unten auf der Seite stehen, unabhängig vom Textbereich eines Buches. Sie können mit folgenden \paper-Variablen kontrolliert werden:

- `oddHeaderMarkup`
- `evenHeaderMarkup`
- `oddFooterMarkup`
- `evenFooterMarkup`

Diese Beschriftungsvariablen können nur auf Textfelder einer Haupttitelumgebung (eine \header-Umgebung auf höchster Ebene, die sich auf alle \score-Umgebungen einer Datei bezieht) zugreifen und sind definiert in der Datei `ly/titling-init.ly`. In den Standardeinstellungen
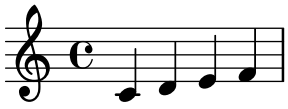
- werden Seitenzahlen automatisch ganz oben links (wenn gerade) oder ganz oben rechts (wenn ungerade) gesetzt, beginnend mit der zweiten Seite.
- wird das `instrument`-Textfeld auf jeder Seite zentriert, beginnend mit der zweiten Seite.
- wird der `copyright`-Text unten auf der ersten Zeite zentriert.
- wird der Inhalt von `tagline` unten auf der letzten Seite zentriert und unterhalb des `copyright`-Texts, wenn es sich nur um eine Seite handelt.



LilyPond v2.27.4

Die Standardeinstellung von `tagline` kann verändert werden, indem man ein `tagline`-Feld in die `\header`-Umgebung auf höchster Ebene schreibt.

```
\book {
  \header {
    tagline = "... music notation for Everyone"
  }
  \score {
    \relative {
      c'4 d e f
    }
  }
}
```



... music notation for Everyone

Um die *tagline* ganz zu entfernen, wird ihr Inhalb als `##f` (falsch) definiert.

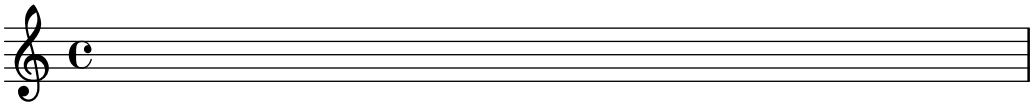
20.2 Eigene Kopf- und Fußzeilen sowie Titel

20.2.1 Angepasste Textformatierung für Titelumgebungen

Standard-`\markup`-Befehle können eingesetzt werden, um jeglichen Text in Titeln, Kopf- und Fußzeilen innerhalb der `\header`-Umgebung zu verändern.

```
\score {
  \header {
    piece = \markup { \fontsize #4 \bold "PRAELUDIUM I" }
    subtitle = \markup { \italic "(Excerpt)" }
  }
  { s1 }
}
```

PRAELUDIUM I



Siehe auch

Notationsreferenz: Abschnitt 8.2 [Text formatieren], Seite 233.

20.2.2 Angepasstes Layout für Titelumgebungen

`\markup`-Befehle in der `\header`-Umgebung sind sinnvoll für einfaches Formatieren von Text, aber sie gewähren keine genaue Kontrolle über die Positionierung von Titeln. Um die Positionierung von Titeltextfeldern vorzunehmen, eignen sich beide oder eine von folgenden Variablen (die man in der `\paper`-Umgebung einsetzen muss):

- `bookTitleMarkup`
- `scoreTitleMarkup`

Die Positionierung von Titeln, wenn die Standardeinstellung dieser Variablen benützt werden, wird anhand einem Beispiel gezeigt in Abschnitt 20.1.2 [Standardlayout von `book`- und Partitur-Titelumgebungen], Seite 466.

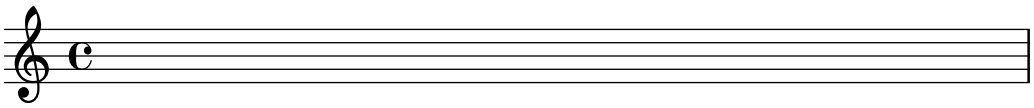
Die Standardeinstellungen von `scoreTitleMarkup`, wie in der Datei `ly/titling-init.ly` definiert, sind:

```
scoreTitleMarkup = \markup { \column {
  \if \should-print-all-headers { \bookTitleMarkup \hspace #1 }
  \fill-line {
    \fromproperty #'header:piece
    \fromproperty #'header:opus
  }
}
}
```

Dadurch werden die Textfelder `piece` (Stück) und `opus` an den gegenüberliegenden Enden der gleichen Zeile platziert:

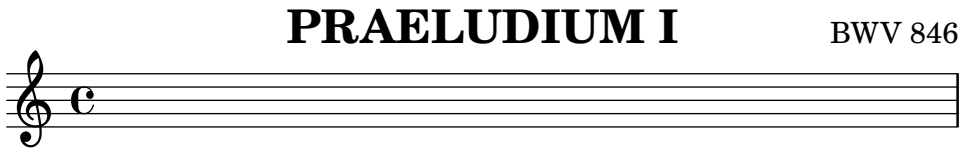
```
\score {
  \header {
    piece = "PRAELUDIUM I"
    opus = "BWV 846"
  }
  { s1 }
}
```

PRAELUDIUM I BWV 846



Das folgende Beispiel verändert die Einstellungen von `scoreTitleMarkup`, sodass das Textfeld `piece` zentriert wird und in fetter, größerer Schrift erscheint.

```
\book {
  \paper {
    indent = 0\mm
    scoreTitleMarkup = \markup {
      \fill-line {
        \null
        \fontsize #4 \bold \fromproperty #'header:piece
        \fromproperty #'header:opus
      }
    }
  }
  \header { tagline = ##f }
  \score {
    \header {
      piece = "PRAELUDIUM I"
      opus = "BWV 846"
    }
    { s1 }
  }
}
```

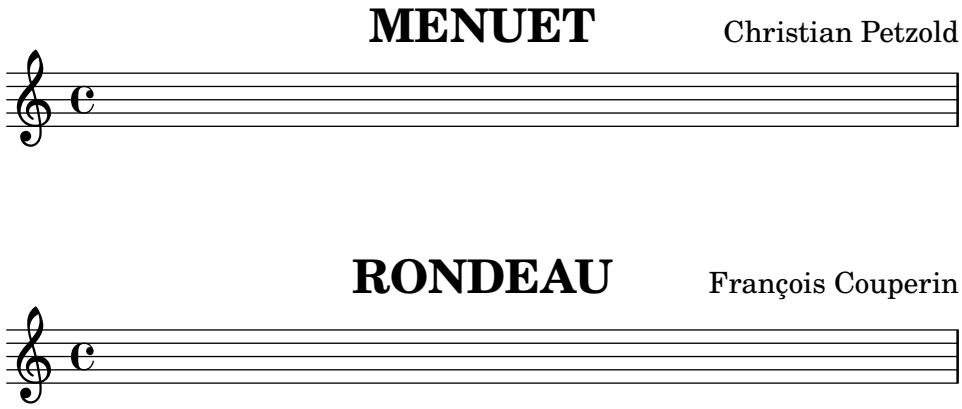


Textfelder, die normalerweise nur im Haupttitel ausgegeben werden, können auch in die Titel einzelner Partituren aufgenommen werden, indem man `print-all-headers` in die \paper-Umgebung einfügt. Ein Nachteil dieser Methode ist, dass Textfelder, die tatsächlich nur im Haupttitel erscheinen sollen, manuell für jede Partitur unterdrückt werden müssen. Siehe auch Abschnitt 20.1.1 [Wie funktioniert die Titel-Umgebung?], Seite 464.

Um das zu vermeiden, kann das gewünschte Feld zur Definition von `scoreTitleMarkup` hinzugefügt werden. Im folgenden Beispiel wird das Komponistenfeld (`composer`) (normalerweise mit `bookTitleMarkup` assoziiert) zu `scoreTitleMarkup` hinzugefügt, sodass jede Partitur einen eigenen Komponisten haben kann.
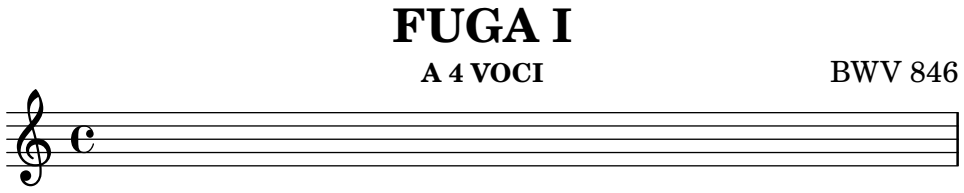
```
\book {
  \paper {
    indent = 0\mm
    scoreTitleMarkup = \markup {
      \fill-line {
        \null
        \fontsize #4 \bold \fromproperty #'header:piece
        \fromproperty #'header:composer
      }
    }
  }
  \header { tagline = ##f }
  \score {
    \header {
      piece = "MENUET"
      composer = "Christian Petzold"
    }
```

```
    { s1 }
  }
  \score {
    \header {
      piece = "RONDEAU"
      composer = "François Couperin"
    }
    { s1 }
  }
}
```



Es ist auch möglich, eigene Textfelder zu erstellen und dann auf sie in der Beschriftungsdefinition zu verweisen:

```
\book {
  \paper {
    indent = 0\mm
    scoreTitleMarkup = \markup {
      \fill-line {
        \null
        \override #`(direction . ,UP)
        \dir-column {
          \center-align \fontsize #-1 \bold
            \fromproperty #'header:mycustomtext %% User-defined field
          \center-align \fontsize #4 \bold
            \fromproperty #'header:piece
        }
        \fromproperty #'header:opus
      }
    }
  }
  \header { tagline = ##f }
  \score {
    \header {
      piece = "FUGA I"
      mycustomtext = "A 4 VOCI" %% User-defined field
      opus = "BWV 846"
    }
    { s1 }
  }
}
```



Siehe auch

Notationsreferenz: Abschnitt 20.1.1 [Wie funktioniert die Titel-Umgebung?], Seite 464.

20.2.3 Angepasstes Layout für Kopf- und Fußzeilen

\markup-Befehle in der \header-Umgebung sind nützlich um einfachen Text zu formatieren, erlauben aber keine Kontrolle über die Positionierung von Kopf- und Fußzeilen. Um die Positionierung der Textfelder zu ändern, eignen sich beide oder eine von folgenden Variablen (die man in der \paper-Umgebung einsetzen muss):

- `oddHeaderMarkup`
- `evenHeaderMarkup`
- `oddFooterMarkup`
- `evenFooterMarkup`

Der \markup-Befehl `\on-the-fly` kann eingesetzt werden, um Beschriftung anhand einer Bedingung zu Kopf- und Fußzeilentext innerhalb der \paper-Umgebung hinzuzufügen. Hierzu wird folgende Syntax eingesetzt:

```
Variable = \markup {
  ...
  \on-the-fly  #Prozedur  Beschriftung
  ...
}
```

Die *Prozedur* wird jedes mal aufgerufen, wenn dar \markup-Befehl, auf den sie sich auswirkt, ausgewertet wird. Die *Prozedur* sollte nach einer bestimmten Bedingung fragen und das *Beschriftung*-Argument ausführen (also setzen), wenn diese Bedinung wahr ist.

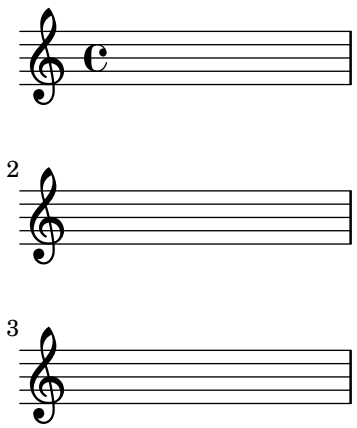
Eine Anzahl von fertigen Prozeduren, um verschiedene Bedinungen zu testen, werden bereitgestellt:

Prozedurbezeichnung	**Getestete Bedinung**
print-page-number-check-first	sollte diese Seitenzahl gesetzt werden?
create-page-number-stencil	'print-page-numbers wahr?
print-all-headers	'print-all-headers wahr?
first-page	erste Seite im Buch?
(on-page nmbr)	Seitenzahl = nmbr?
last-page	letzte Seite im Buch?
not-first-page	nicht erste Seite im Buch?
part-first-page	erste Seite im book part?
part-last-page	letzte Seite im book part?
not-single-page	Seiten im book part > 1?

Das folgende Beispiel zentriert die Seitenzahlen unten auf jeder Seite. Zuerst werden die Standardeinstellungen von `oddHeaderMarkup` und `evenHeaderMarkup` entfernt, indem sie als

„Null“-Beschriftung definiert werden. Dann wird `oddFooterMarkup` mit zentrierter Seitennummer neu definiert. Schließlich erhält `evenFooterMarkup` die selbe Layoutdefinition wie `\oddFooterMarkup`:

```
\book {
  \paper {
    print-page-number = ##t
    print-first-page-number = ##t
    oddHeaderMarkup = \markup \null
    evenHeaderMarkup = \markup \null
    oddFooterMarkup = \markup {
      \fill-line {
        \if \should-print-page-number
        \fromproperty #'page:page-number-string
      }
    }
    evenFooterMarkup = \oddFooterMarkup
  }
  \score {
    \new Staff { s1 \break s1 \break s1 }
  }
}
```



1

Verschiedene `\on-the-fly`-Bedingungen können mit dem Operator „UND“ verknüpft werden, beispielsweise bestimmt

```
\if \on-first-page
\if \on-last-page
{ \markup ... \fromproperty #'header: ... }
```

ob es sich um eine einzelne Seite bei der Ausgabe handelt.

Siehe auch

Notationsreferenz: Abschnitt 20.1.1 [Wie funktioniert die Titel-Umgebung?], Seite 464, Abschnitt 20.1.2 [Standardlayout von `book`- und Partitur-Titelumgebungen], Seite 466.

Installierte Dateien: `../ly/titling-init.ly`.

20.3 Fußnoten erstellen

Zwei Arten an Fußnoten können erstellt werden: automatische und manuelle Fußnoten.

20.3.1 Übersicht über Fußnoten

Automatische Fußnoten erstellen aufsteigende Zahlenverweise, während mit manuellen Fußnoten eigene angepasste Verweise erstellt werden können. Fußnoten werden normalerweise wie ein `\tweak`-Befehl an Noten gehängt und können deshalb auch direkt mit den Grobs (graphischen Objekten) verknüpft werden, die von den meisten musikalischen Elementen und Postelementen erstellt werden. In Fällen, wo das nicht funktioniert (etwa bei Taktstrichen und Taktartänderungen, wo die Grobs als Folge einer Eigenschaftsänderung erstellt werden), können Fußnoten als einzelnstehendes musikalisches Ereigenis ersetllt werden, das sich auf alle Grobs eines bestimmten Typs zu einer bestimmten Zeit bezieht.

Der vollständige Befehl ist:

```
\footnote Zeichen Verschiebung Grob-Bezeichnung Fußnote
Noten
```

Die Elemente sind folgende:

Zeichen ist eine Beschriftung oder Zeichenkette, die das Fußnotenzeichen angbit, welches für den Referenzpunkt als auch für die Fußnote unten auf der Seite benützt wird. Es kann ausgelassen werden (oder durch `\default` ersetzt werden); in diesem Fall wird eine aufsteigende Zahlenfolge erstellt.

Verschiebung (offset)
ist ein Zahlenpaar wie etwa '`#'(2 . 1)`', das die X- und Y-Verschiebung vom Referenzpunkt aus angibt, wo das Zeichen gesetzt werden soll.

Grob-Bezeichnung
gibt die Grob-Art an, der ein Fußnotenzeichen hinzugefügt werden soll (wie etwa '`#'Flag`'). Wenn sie angegeben wird, wird der entsprechende Grob als Referenzpunkt eingesetzt, auch wenn das referenzierte Element nicht die *Noten* selber sind, sondern ein Grob, der durch sie erstellt wird. Das Element kann ausgelassen werden (oder durch `\default` ersetzt werden); dann wird nur ein direkt erstellter Grob mit Fußnote versehen.

Fußnote Diese Beschriftung oder Zeichenkette bezeichnet den Fußnotentext, der am unteren Seitenrand gesetzt werden soll.

Noten Das ist das Element, ein musikalisches Ereignis oder eine Akkordkonstruktion oder ein Post-Ereignis, das die Fußnote erhält. Man kann es nicht auslassen, aber man *kann* es durch `\default` ersetzten. In diesem Fall wird die Fußnote aber nicht an einen bestimmten musikalischen Ausdruck angehänt, sondern an einen zeitlichen Moment. In diesem Fall muss man zwingend die *Grob-Bezeichnung* angeben, um den Grob-Typ auszuwählen, auf den sich die Fußnote bezieht (etwa '`#'TimeSignature`').
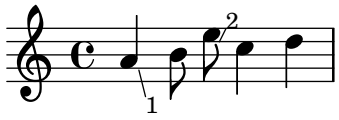
Wie auch mit `\tweak` muss dem `\footnote`-Befehl - vorangestellt werde,n wenn er an ein Post-Ereignis oder eine Artikulation angehängt werden soll, damit der Parser das Ergebnis der vorherigen Note oder Pause zuordnen kann.

20.3.2 Automatische Fußnoten

Automatische Fußnoten haben vier Argumente: die *(x . y)*-Position des Indikators, die optionale *Grob-Bezeichnung*, die die Anmerkung erhalten soll, die *Fußnote*-Beschriftung, die den Inhalt der Fußnote enthält, und natürlich die *Noten*, welche mit einer Fußnote versehen werden sollen.

Der Befehl `\footnote` muss *vor* dem Grob geschrieben werden, auf den sich die Fußnote bezieht.

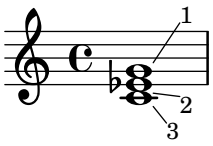
```
\book {
  \header { tagline = ##f }
  \relative c' {
    \footnote #'(0.5 . -2)
      \markup { Die erste Note }
    a'4 b8
    \footnote #'(0.5 . 1)
      \markup { Die dritte Note } Flag
    e\noBeam c4 d4
  }
}
```



[1]Die erste Note
[2]Die dritte Note

Noten in Akkorden stellen keine Schwierigkeit dar:

```
\book {
  \header { tagline = ##f }
  \relative c' {
    <
    \footnote #'(1 . -1.25) "Hier ein C" c
    \footnote #'(2 . -0.25) \markup { \italic "Ein Es" } es
    \footnote #'(2 . 3) \markup { \bold "Das ist ein G" } g
    >1
  }
}
```

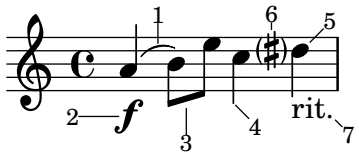


[1]**Das ist ein G**
[2]*Ein Es*
[3]Hier ein C

Achtung: Wenn Fußnoten die selbe vertikale Position haben, werden sie von oben nach unten nummeriert.

Hier noch einige Beispiele von Grobs, die mit Fußnoten versehen sind, wobei auch die Position des Fußnotentextes relativ zu Tagline und Copyright gezeigt wird.

```
\book {
  \header { copyright = \markup { "Copyright 1970" } }
  \relative {
    a'4-\footnote #'(-3 . 0) \markup { \bold Forte } \f
    -\footnote #'(0 . 1.5) \markup { Ein Bogen } (
    b8)-\footnote #'(0 . -2) \markup { Balken } [ e]
    \single\footnote #'(1 . -1)
      \markup  { \teeny { Das ist ein Hals } } Stem
    c4
    \single\footnote #'(0 . 0.5)
      \markup \italic { Ein warnendes Versetzungszeichen } AccidentalCautionary
    \footnote #'(1 . 1) "Die Note selber"
    dis?4-\footnote #'(0.5 . -0.5) \markup \italic { langsamer werden }
         _"rit."
  }
}
```



[1]Ein Bogen
[2]**Forte**
[3]Balken
[4]Das ist ein Hals
[5]Die Note selber
[6]*Ein warnendes Versetzungszeichen*
[7]*langsamer werden*
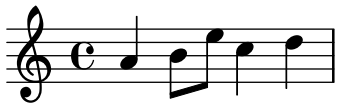
Copyright 1970

LilyPond v2.27.4

Für eine \markup-Beschriftung auf oberster Ebene braucht man den Befehl \footnote:

```
\book {
  \header { tagline = ##f }
  \markup { \footnote "Eine einfache Melodie" \italic "Von mir" }
  \relative {
    a'4 b8 e c4 d
  }
}
```

Eine einfache Melodie



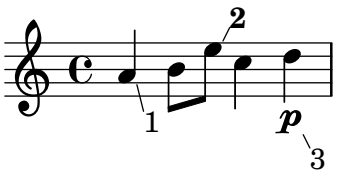
Von mir

20.3.3 Manuelle Fußnoten

Manuell erstellte Fußnoten haben ein zusätzliches erstes Argument *Zeichen*, das das Fußnotenzeichen erstellt. Im Gegensatz zu automatisch erstellten Fußnotenzeichen erscheinen sie nicht unbedingt vor dem Fußnotentext unten auf der Seite – das Herstellen eines visualen Zusammenhanges ist dem Setzer überlassen. LilyPond stellt nur ischer, dass der zugehörige Text unten auf der selben Seite erscheint.

Davon abgesehen ist die Benutzung identisch mit automatischen Fußnoten.

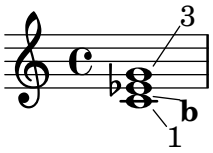
```
\book {
  \header { tagline = ##f }
  \relative c' {
    \footnote
          "1" #'(0.5 . -2)
          \markup { \italic "1. Die erste Note" }
    a'4
    b8
    \footnote
          \markup { \bold "2" } #'(0.5 . 1)
          "2. Die zweite Note"
    e
    c4
    d-\footnote "3" #'(0.5 . -1) "3. Piano" \p
  }
}
```



1. *Die erste Note*
2. Die zweite Note
3. Piano

Fußnoten für Akkordnoten werden wie folgt notiert:

```
\book {
  \header { tagline = ##f }
  \relative c' {
    <
    \footnote "1" #'(1 . -1.25) "1. C" c
    \footnote
       \markup { \bold "b" } #'(2 . -0.25) "b. E-flat" es
    \footnote "3" #'(2 . 3) \markup { \italic "iii. G" } g
    >1
  }
}
```



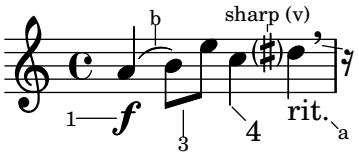
iii. G
b. E-flat
1. C

Achtung: Wenn Fußnoten die selbe vertikale Position haben, werden sie von oben nach unten nummeriert.

Hier einige Beispiele manueller Fußnoten, die auch die relative Position der Fußnotentexte zu Tagline und Copyright anzeigen:

```
\book {
  \header { tagline = ##f }
  \relative {
    a'4-\footnote
      \markup { \teeny 1 } #'(-3 . 0)
      \markup { 1. \bold Forte } \f
    -\footnote
      \markup { \teeny b } #'(0 . 1.5)
      \markup { b. Ein Bogen } (
    b8)-\footnote
      \markup { \teeny 3 } #'(0 . -2)
      \markup { 3. Balken } [
    e]
    \single\footnote
      \markup { 4 } #'(1 . -1)
      \markup  { \bold 4. { Das ist ein Hals } } Stem
    c4
    \single\footnote
      \markup \concat \teeny { "sharp (v)" }
```

```
          #'(0 . 0.5)
        \markup \italic { v. Ein warndendes Versetzungszeichen } AccidentalCautionary
      dis?4-\footnote
        \markup \concat \teeny { "a" } #'(0.5 . -0.5)
        \markup \italic { a. Langsamer werden } _"rit."
      \footnote
        \markup { \teeny \musicglyph "rests.4" }
          #'(1.5 . -0.25)
        \markup { \null } \breathe
    }
  }
```
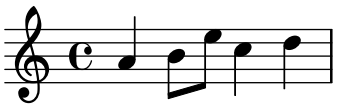


b. Ein Bogen
1. **Forte**
3. Balken
4. Das ist ein Hals
v. Ein warndendes Versetzungszeichen
a. Langsamer werden

Eine \markup-Beschriftung der höchsten Ebene wird wie folgt annotiert:

```
\book {
  \header { tagline = ##f }
  \markup { "Eine einfache Melodie" \footnote "*" \italic "* Von mir" }
  \relative {
    a'4 b8 e c4 d4
  }
}
```

Eine einfache Melodie *



* *Von mir*

Siehe auch

Handbuch zum Lernen: Abschnitt “Objekte und Schnittstellen” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 7.2.1 [Erklärungen in Ballonform], Seite 221, Kapitel 25 [Seitenlayout], Seite 521, Abschnitt 8.1.3 [Textartige Zeichen], Seite 229, Abschnitt 8.1.1 [Textarten], Seite 226, Kapitel 20 [Titel], Seite 464.

Referenz der Interna: Abschnitt “FootnoteEvent” in *Referenz der Interna*, Abschnitt “Footnote” in *Referenz der Interna*, Abschnitt “Footnote_engraver” in *Referenz der Interna*.

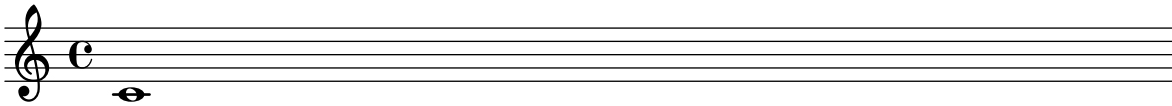
Bekannte Probleme und Warnungen

Mehrere Fußnoten auf einer Seite können ausschließlich übereinander gedruckt werden und nicht auf der gleiche Zeile gesetzt werden. Fußnoten können nicht an `MultiMeasureRests` (mehrtaktige Pause) angehängt werden und können mit `Staff`-(Notensystem)-, `\markup`-(Beschriftungs)-Objekten und anderen Fußnoten zusammenstoßen. Wenn man den manuellen Fußnotenbefehl einsetzt, braucht man in der `\paper`-Umgebung den Eintrag `footnote-auto-number = ##f`.

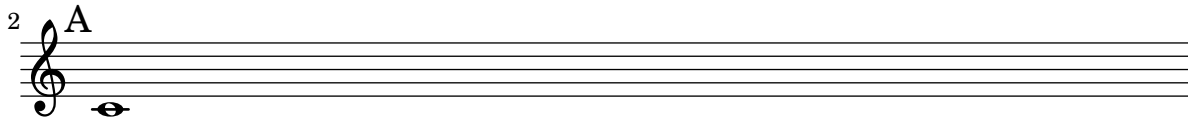
20.4 Verweis auf die Seitenzahlen

Eine bestimmte Stelle der Partitur kann mit einem `\label`-Befehl markiert werden, sowohl auf oberster Ebene als auch innerhalb eines musikalischen Ausdrucks. Auf diese Marke kann dann verwiesen werden, um die Seitenzahl zu erhalten, auf der die Marke vorkommt. Der Verweis wird mit dem Befehl `\page-ref` gefordert (innerhalb von `\markup`).

```
\header { tagline = ##f }
\book {
  \label #'ErstePartitur
  \score {
    {
      c'1
      \pageBreak \mark A \label #'ZeichenA
      c'1
    }
  }
  \markup { Die erste Partitur fängt auf
            Seite \page-ref #'ErstePartitur "0" "?" an.}
  \markup { Zeichen A befindet sich auf Seite
            \concat { \page-ref #'ZeichenA "0" "?" . } }
}
```



2



Die erste Partitur fängt auf Seite 1 an.
Zeichen A befindet sich auf Seite 2.

Der `\page-ref`-Textbeschriftungsbefehl braucht drei Argumente:

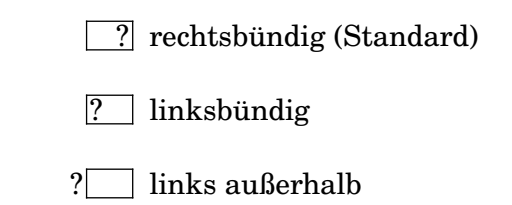
1. die Marke, ein Scheme-Symbol, etwa `#'ErstePartitur`,
2. eine Beschriftung, die als Platzhalter benutzt wird, um die Breite des Verweisen zu schätzen,
3. eine Beschriftung, die anstelle der Seitenzahl gesetzt wird, wenn die Marke unbekannt ist.

Der Grund, warum ein Platzhalter benötigt wird, ist dass zu dem Zeitpunkt, an dem die Textbeschriftungen ausgewertet werden, noch keine Seitenumbrüche vorgenommen wurden und die Seitenzahlen deshalb noch nicht bekannt sind. Um hier ein Problem zu vermeiden, wird die

eigentliche Auswertung der Textbeschriftung erst später ausgeführt, die Größe des Textes muss aber schon vorher bekannt sein. Die Größe wird mithilfe des Platzhalters bestimmt. Wenn eine Partitur zwischen 10 und 99 Seiten hat, kann man "00" schreiben, also eine zweistellige Zahl.

Falls die Größe des finalen Textes sich von der des Platzhalters unterscheidet, kann es sinnvoll sein, die horizontale Ausrichtung gegenüber dem Platzhalter mit der Eigenschaft `x-align` festzulegen. Standardmäßig erfolgt die Ausrichtung rechtsbündig mit dem Platzhalter.

```
\markup {
  \box
    \page-ref #'foo "???" "?" " rechtsbündig (Standard)"
}
\markup {
  \box
    \override #`(x-align . ,LEFT)
    \page-ref #'foo "???" "?" " linksbündig"
}
\markup {
  \box
    \override #'(x-align . -2.5)
    \page-ref #'foo "???" "?" " links außerhalb"
}
```



Im Beispiel ist der Platzhalter '`???`' größer als der eingesetzte Text '`?`', der genommen wird, weil die Marke `#'foo` nicht existiert. `x-align` kann mit beliebigen Zahlen eingestellt werden oder die bekannten Symbole `LEFT`, `CENTER` oder `RIGHT` annehmen, welche die Ausrichtung linksbündig, zentriert oder rechtsbündig gegenüber dem Platzhalter vornehmen.

Vordefinierte Befehle

`\label`, `\page-ref`.

20.5 Inhaltsverzeichnis

Ein Inhaltsverzeichnis kann eingefügt werden mit dem Befehl `\markuplist \table-of-contents`. Die Elemente, die im Inhaltsverzeichnis aufgelistet werden sollen, werden mit dem `\tocItem`-Befehl markiert, welches sowohl auf höchster Ebene als auch in einem musikalischen Ausdruck verwendet werden kann.

```
\markuplist \table-of-contents
\pageBreak

\tocItem \markup "Erste Partitur"
\score {
  {
    c'4  % ...
    \tocItem \markup "Ein bestimmter Punkt innerhalb der ersten Partitur"
    d'4  % ...
  }
}
```

```
\tocItem \markup "zweite Partitur"
\score {
  {
    e'4 % ...
  }
}
```

Die Beschriftungen, die benutzt werden um das Inhaltsverzeichnis zu formatieren, sind in der \paper-Umgebung definiert. Die Standardformatierungselemente sind `tocTitleMarkup` um die Überschrift zu formatieren und `tocItemMarkup` um die einzelnen Inhaltselemente zu formatieren, bestehend aus dem Titelelement und einer Seitenzahl. Die Variablen können durch den Benutzer geändert werden:

```
\paper {
  %% Übersetzung der Inhaltsverzeichnisüberschrift nach französisch:
  tocTitleMarkup = \markup \huge \column {
    \fill-line { \null "Table des matières" \null }
    \hspace #1
  }
  %% hier größere Schriftarten
  tocItemMarkup = \markup \large \fill-line {
    \fromproperty #'toc:text \fromproperty #'toc:page
  }
}
```

Die Inhaltsverzeichniselemente Text und Seitenzahl werden in der Definition von `tocItemMarkup` aufgerufen mit `#'toc:text` und `#'toc:page`.

Neue Befehle und Beschriftungen können auch definiert werden, um eigene Inhaltsverzeichnisse zu gestalten:

- zuerst muss eine neue Beschriftungsvariable in der \paper-Umgebung definiert werden
- dann muss die musikalische Funktion definiert werden, die ein Element zum Inhaltsverzeichnis hinzufügt, indem die neue Variable benutzt wird.

Das folgende Beispiel definiert einen neuen Stil um Akt-Bezeichnungen einer Oper in das Inhaltsverzeichnis aufzunehmen:

```
\paper {
  tocActMarkup = \markup \large \column {
    \hspace #1
    \fill-line { \null \italic \fromproperty #'toc:text \null }
    \hspace #1
  }
}

tocAct =
#(define-music-function (text) (markup?)
   (add-toc-item! 'tocActMarkup text))
```

Table of Contents

Atto Primo

Coro. Viva il nostro Alcide **1**
Cesare. Presti omai l'Egizzia terra **1**

Atto Secondo

Sinfonia **1**
Cleopatra. V'adoro, pupille, saette d'Amore **1**

Die Zeile zwischen dem Inhalt und der Seitenzahl kann mit einer gepunkteten Linie gefüllt werden:

```
\header { tagline = ##f }
\paper {
  tocItemMarkup = \tocItemWithDotsMarkup
}

\book {
  \markuplist \table-of-contents
  \tocItem \markup { Allegro }
  \tocItem \markup { Largo }
  \markup \null
}
```

Table of Contents

Allegro 1
Largo 1

Siehe auch

Installierte Dateien: `ly/toc-init.ly`.

Vordefinierte Befehle

`\table-of-contents`, `\tocItem`.

21 Arbeiten an Eingabe-Dateien

21.1 LilyPond-Dateien einfügen

Ein größeres Projekt kann in einzelne Dateien aufgeteilt werden. Um eine andere Datei einzubinden, kann der Befehl

```
\include "andereDatei.ly"
```

benutzt werden.

Die Zeile `\include "andereDatei.ly"` benimmt sich genauso, also ob der Inhalt der Datei `andereDatei.ly` komplett in die Hauptdatei eingefügt werden würde. So kann man für ein größeres Projekt die einzelnen Stimmen der Instrumente getrennt notieren und sie dann in einer Partitur-Datei benutzen. Meistens werden in den eingefügten Dateien einige Variablen definiert, die dann auch in der Hauptdatei eingesetzt werden können. Mit Marken (Tags) gekennzeichnete Abschnitte können eingesetzt werden, um die entsprechenden Noten etc. an verschiedenen Stellen in der Datei zur Verfügung zu stellen. Siehe auch Abschnitt 21.2 [Verschiedene Editionen aus einer Quelldatei], Seite 489.

Auf Dateien im aktuellen Verzeichnis kann einfach mit dem Dateinamen nach dem `\include`-Befehl verwiesen werden. Dateien an anderen Stellen können eingebunden werden, indem entweder ein vollständiger Pfad oder ein relativer Pfad zu der Datei angegeben wird. Hierbei sollten die für UNIX typischen Schrägstriche (/) und nicht die rückwärtsgeneigten von Windows (\) verwendet werden, um die Verzeichnisse zu trennen. Wenn etwa die Datei `kram.ly` ein Verzeichnis höher liegt als das aktuelle Verzeichnis, sollte der Befehl so aussehen:

```
\include "../kram.ly"
```

Wenn die Orchesterstimmen andererseits in einem Unterordner mit der Bezeichnung `stimmen` liegen, sieht er folgendermaßen aus:

```
\include "stimmen/VI.ly"
\include "stimmen/VII.ly"
... etc
```

Dateien, die eingebunden werden sollen, können selber auch wiederum ein `\include` enthalten. Diese Einbindung zweiter Ebene werden erst interpretiert, wenn sie sich in der Hauptdatei befinden, sodass die Pfadangaben hier nicht relativ zur eingebundenen Datei, sondern relativ zur Hauptdatei gesetzt werden müssen. Dieses Verhalten kann jedoch auch verändert werden, indem man lilypond die Option `-drelative-includes` auf der Kommandozeile zuweist (oder indem man den Befehl `#(ly:set-option 'relative-includes #t)` an den Beginn der Quelldatei schreibt). Mit `relative-includes` wird der Pfad jedes `\include`-Befehls als relativ zu der Datei angenommen, in der sich der Befehl befindet. Dieses Verhalten wird empfohlen und wird in zukünftigen Versionen von LilyPond den Standard darstellen.

Dateien können auch aus einem Verzeichnis eingebunden werden, dass im Suchpfad von LilyPond liegt. Hierzu muss auf der Kommandozeile das entsprechende Verzeichnis angegeben werden und die Dateien, die eingebunden werden, müssen nur mit ihrem Namen notiert sein. Wenn etwa die Datei `Haupt.ly` kompiliert werden soll, die Dateien aus dem Unterverzeichnis `stimmen` einbindet, müssen sie sich im Verzeichnis von `Haupt.ly` befinden und dann LilyPond folgendermaßen aufrufen:

```
lilypond --include=stimmen Haupt.ly
```

In `Haupt.ly` steht:

```
\include "VI.ly"
\include "VII.ly"
... usw.
```

Dateien, die in vielen Partituren verwendet werden sollen, können im LilyPond-Verzeichnis `../ly` gespeichert werden. (Die Stelle, an der dieses Verzeichnis sich befindet, hängt vom Betriebssystem ab, siehe hierzu Abschnitt "Mehr Information" in *Handbuch zum Lernen*). Dateien in diesem Verzeichnis können einfach mit ihrem Namen eingefügt werden. So werden auch die Sprachdateien wie etwa `deutsch.ly` eingefügt.

LilyPond lädt eine Anzahl an Dateien, wenn das Programm aufgerufen wird. Diese Dateien sind für den Benutzer nicht ersichtlich, aber die Dateien können identifiziert werden, indem LilyPond auf der Kommandozeile mit Option aufgerufen wird: `lilypond --verbose`. Hiermit wird neben anderer Information auch eine Liste von Pfaden und Dateien aufgeführt, die LilyPond benutzt. Die wichtigeren Dateien werden im Abschnitt Abschnitt "Mehr Information" in *Handbuch zum Lernen* besprochen. Diese Dateien können verändert werden, aber Änderungen gehen verloren, wenn eine neue LilyPond-Version installiert wird.

Einige einfache Beispiele, die die Benutzung von `\include` demonstrieren, sind dargestellt in Abschnitt "Partituren und Stimmen" in *Handbuch zum Lernen*.

Siehe auch

Handbuch zum Lernen: Abschnitt "Mehr Information" in *Handbuch zum Lernen*, Abschnitt "Partituren und Stimmen" in *Handbuch zum Lernen*.

Bekannte Probleme und Warnungen

Wenn eine Datei eingebunden wird, deren Name einer Datei aus dem Installationsverzeichnis von LilyPond entspricht, wird die installierte Datei anstelle der eigenen verwendet.

21.2 Verschiedene Editionen aus einer Quelldatei

Es gibt verschiedene Funktionen, die es möglich machen, unterschiedliche Versionen einer Partitur aus der gleichen Quelldatei zu produzieren. Variablen werden am besten eingesetzt, wenn es darum geht, längere Notenpassagen und/oder Anmerkungen/Textmarken miteinander auf verschiedene Weise zu kombinieren. Tag-Marken dagegen werden am besten eingesetzt, wenn eine von mehreren kurzen alternativen Notenabschnitten ausgewählt werden soll und können auch eingesetzt werden, um Musikabschnitte an unterschiedlichen Stellen zusammenzufügen.

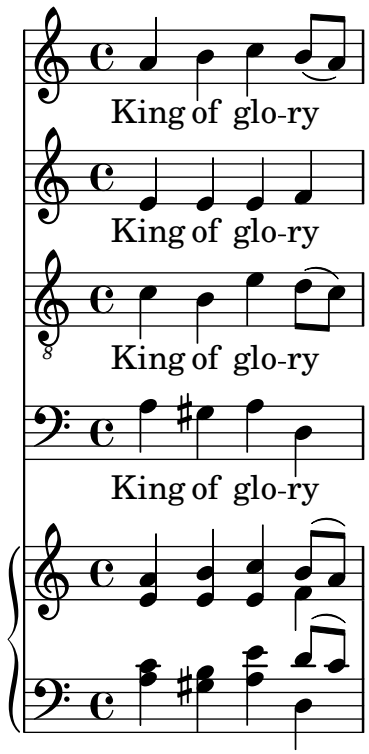
Egal welche Methode am Ende eingesetzt wird: Es erleichtert die Arbeit in jedem Fall, wenn die eigentlichen Noten und die Struktur der Partitur voneinander getrennt notiert werden – so kann die Struktur geändert werden, ohne dass man Änderungen an den Noten vornehmen muss.

21.2.1 Variablen benutzen

Wenn Notenabschnitt in Variablen definiert werden, können sie an unterschiedlichen Stellen in der Partitur eingesetzt werden, siehe auch Abschnitt "Stücke durch Variablen organisieren" in *Handbuch zum Lernen*. Zum Beispiel enthält eine Vokalpartitur für ein *a cappella* Stück oft einen Klavierauszug, der das Einüben einfacher macht. Der Klavierauszug enthält die gleichen Noten, sodass man sie nur einmal notieren muss. Noten aus zwei Variablen können auf einem System kombiniert werden, siehe Abschnitt 5.2.4 [Automatische Kombination von Stimmen], Seite 175. Hier ein Beispiel:

```
sopranoMusic = \relative { a'4 b c b8( a) }
altoMusic = \relative { e'4 e e f }
tenorMusic = \relative { c'4 b e d8( c) }
bassMusic = \relative { a4 gis a d, }
allLyrics = \lyricmode {King of glo -- ry }
<<
  \new Staff = "Soprano" \sopranoMusic
  \new Lyrics \allLyrics
```

```
    \new Staff = "Alto" \altoMusic
    \new Lyrics \allLyrics
    \new Staff = "Tenor" {
      \clef "treble_8"
      \tenorMusic
    }
    \new Lyrics \allLyrics
    \new Staff = "Bass" {
      \clef "bass"
      \bassMusic
    }
    \new Lyrics \allLyrics
    \new PianoStaff <<
      \new Staff = "RH" {
        \set Staff.printPartCombineTexts = ##f
        \partCombine
        \sopranoMusic
        \altoMusic
      }
      \new Staff = "LH" {
        \set Staff.printPartCombineTexts = ##f
        \clef "bass"
        \partCombine
        \tenorMusic
        \bassMusic
      }
    >>
  >>
```



Unterschiedliche Partituren, die entweder nur den Chor oder das Klavier zeigen, können produziert werden, indem die Struktur verändert wird; die Noten müssen dazu nicht verändert werden.

Für längere Partituren können Variablen in eigene Dateien notiert werden, die dann eingebunden werden, siehe Abschnitt 21.1 [LilyPond-Dateien einfügen], Seite 488.

21.2.2 Marken benutzen

Der `\tag #'`*TeilA*-Befehl markiert einen musikalischen Ausdruck mit der Bezeichnung *TeilA*. Ausdrücke, die auf diese Weise markiert werden, können mit ihrer Bezeichnung später ausgewählt bzw. ausgefiltert werden. Das geschieht mit den Befehlen `\keepWithTag #'`*Bezeichnung* bzw. `\removeWithTag #'`*Bezeichnung*. Die Wirkung dieser Filter auf die markierten Notenabschnitte ist wie folgt:

Filter	**Resultat**
Markierte Noten mit vorgesetztem `\keepWithTag #'`*Bezeichnung* oder `\keepWithTag #'(`*Bezeichnung1 Bezeichnung2 ...*`)`	Unmarkierte Noten und Noten, die mit einer der angegebenen Bezeichnung markiert sind, werden gesetzt, Noten mit einer anderen Markierung werden nicht angezeigt.
Markierte Noten mit vorgesetztem `\removeWithTag #'`*Bezeichnung* oder `\removeWithTag #'(`*Bezeichnung1 Bezeichnung2 ...*`)`	Unmarkierte Noten und Noten, die mit keiner der angegebenen Bezeichnungen markiert sind, werden gesetzt, Noten, die mit einer der Bezeichnungen markiert sind, werden nicht angezeigt.
Markierte Noten, weder mit vorgesetztem `\keepWithTag` noch `\removeWithTag`	Alle markierten und unmarkierten Noten werden angezeigt.

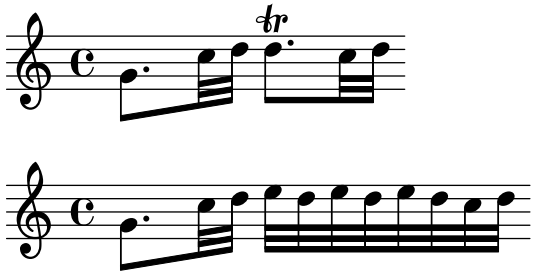
Die Argumente der Befehle `\tag`, `\keepWithTag` und `\removeWithTag` sollten ein Symbol oder eine Liste von Symbolen sein (wie etwa `#'score` oder `#'(violineI violineII)`), gefolgt von einem musikalischen Ausdruck. Nur wenn die Symbole gültige LilyPond-Bezeichner sind (nur Buchstaben, keine Zahlen, Unter- oder Bindestriche), die nicht mit Noten verwechselt werden können, kann `#'` als Kurzschreibweise weggelassen werden. Eine Liste von Symbolen kann ein Komma als Trenner nutzen, d.h. `\tag #'(violineI violineII)` kann auch als `\tag violineI,violineII` geschrieben werden. Das gleiche gilt für `\keepWithTag` und `\removeWithTag`. Die Tag-Befehle sind musikalische Funktionen und können damit nicht zum Filtern von Objekten benutzt werden, die kein musikalischer Ausdruck sind, also sowas wie `\book`- oder `\score`-Blöcke. Es gibt jedoch auch Befehle für Tags in `\markup`-Umgebungen.

Im folgenden Beispiel erscheinen zwei Versionen der Noten, eine zeigt Triller in normaler Notation, die andere zeigt sie ausgeschrieben:

```
music = \relative {
  g'8. c32 d
  \tag #'trills { d8.\trill }
  \tag #'expand { \repeat unfold 3 { e32 d } }
  c32 d
 }

\score {
  \keepWithTag #'trills \music
```

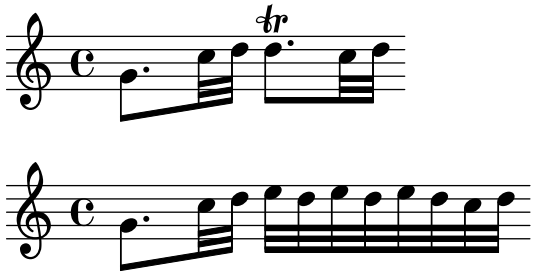
```
}
\score {
  \keepWithTag #'expand \music
}
```



Entsprechend können auch Abschnitte ausgeschlossen werden; das erfordert manchmal weniger Schreibarbeit:

```
music = \relative {
  g'8. c32 d
  \tag #'trills { d8.\trill }
  \tag #'expand {\repeat unfold 3 { e32 d } }
  c32 d
 }

\score {
  \removeWithTag #'expand
  \music
}
\score {
  \removeWithTag #'trills
  \music
}
```



Marken können auch auf Artikulationen, Text usw angewendet werden, indem man ihnen

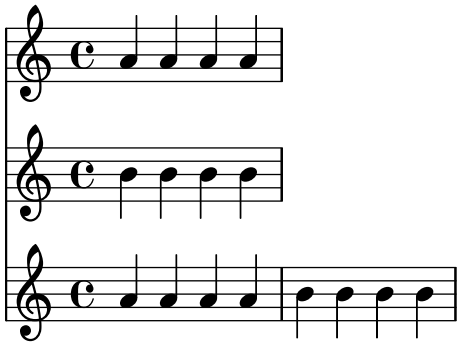
```
-\tag #'your-tag
```

voranstellt (jedoch nach der Note, an die sie gebunden sind). Mit diesem Code etwa könnte man entweder Fingersatz oder aber einen Text ausgeben:

```
c1-\tag #'finger ^4
c1-\tag #'warn ^"Achtung!"
```
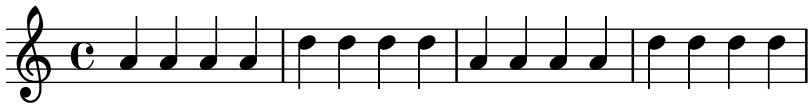
Mehrfache Marken können mithilfe von mehreren \tag-Befehlen notiert werden:

```
music = \relative c'' {
  \tag #'a \tag #'both { a4 a a a }
  \tag #'b \tag #'both { b4 b b b }
}
<<
\keepWithTag #'a \music
\keepWithTag #'b \music
\keepWithTag #'both \music
>>
```



Mehrfache \removeWithTag-Filter können auf einen musikalischen Ausdruck angewendet werden, um mehrere unterschiedliche markierte Abschnitte aus dem Druckbild zu entfernen. Alternativ kann auch ein einziger \removeWithTag-Befehl mit einer Liste von Marken benutzt werden.

```
music = \relative c'' {
  \tag #'A { a4 a a a }
  \tag #'B { b4 b b b }
  \tag #'C { c4 c c c }
  \tag #'D { d4 d d d }
}
\new Voice {
  \removeWithTag #'B
  \removeWithTag #'C
  \music
  \removeWithTag #'(B C)
  \music
}
```
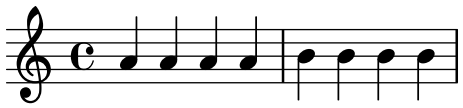


Zwei oder mehr \keepWithTag-Filter in einem musikalischen Ausdruck bewirken, dass *alle* markierten Abschnitte entfernt werden, weil der erste Befehl alle markierten Abschnitt außer dem im Befehl genannten wegfiltert und der zweite Befehl dann auch diesen eben genannten zusätzlich entfernt. Ein einzelner \keepWithTag-Befehl mit mehreren Marken entfernt nur diejenigen markierten Abschnitte, die nur andere Marken als die angegebenen enthalten.

In folgendem Beispiel

```
music = \relative c'' {
  \tag #'violinI { a4 a a a }
  \tag #'violinII { b4 b b b }
  \tag #'viola { c4 c c c }
  \tag #'cello { d4 d d d }
}

\new Staff {
  \keepWithTag #'(violinI violinII)
  \music
}
```



werden nur die mit `\tag` markierten Abchnitte ausgegeben, die die Bezeichner *violinI* und *violinII*, aber nicht *viola* und *cello* benutzen.

Während `\keepWithTag` gut geeignet ist, um mit *einer* Art von Alternativen umzugehen, führt das Entfernen von *andersartigen* Marken zu Problemen, bei Nutzung für verschiedene Zwecke. In diesem Fall können Marken gruppiert werden:

```
\tagGroup #'(violinI violinII viola cello)
```

Verschiedene Marken können somit zu einer Gruppe zusammengefasst werden. Allerdings kann eine einzelne Marke nur in einer Gruppe sein.
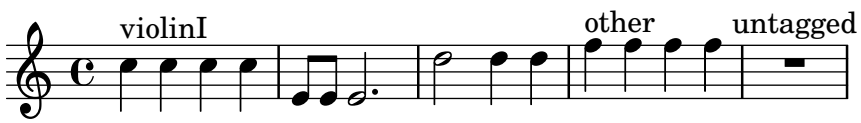
Die Nutzung von

```
\keepWithTag #'violinI ...
```

entfernt nur markierte Abschnitte mit Marken der gleichen Gruppe, in der auch `violinI` ist, sofern es nicht `violinI` selber ist. Marken anderer Gruppen bleiben unangetastet.

```
music = \relative {
  \tagGroup #'(violinI violinII)
  \tagGroup #'(viola cello)
  \tag #'violinI { c''4^"violinI" c c c }
  \tag #'violinII { a2 a }
  \tag #'viola { e8 e e2. }
  \tag #'cello { d'2 d4 d }
  \tag #'other { f^"other" f f f }
  R1^"untagged"
}

\new Voice {
  \keepWithTag #'violinI
  \music
}
```



Im Beispiel entfernt der `\keepWithTag`-Befehl lediglich `violinII`, da diese Marke in der selben Gruppe wie `violinI` ist. `viola` und `cello` bleiben erhalten, weil sie in einer anderen Gruppe sind und `other` bleibt ebenfalls erhalten, weil diese Marke gar keiner Gruppe zugeordnet ist.

Es ist möglich, bereits erstellte Taggruppen zu modifizieren. Um diese referenzieren zu können, existiert eine alternative Definitionsmöglichkeit mit `\tagGroupRef`. Der Aufruf

```
tg = \tagGroupRef violinI,violinII
```

ist im Wesentlichen das gleiche wie

```
tg = #'(violinI violinII)
\tagGroup \tg
```

Bereits erstellte Taggruppen können auf mehrere Arten verändert werden.

Um die Auswirkungen von allen `\tagGroup`- und `\tagGroupRef`-Befehlen rückgängig zu machen, lässt sich `\resetTagGroups` verwenden.

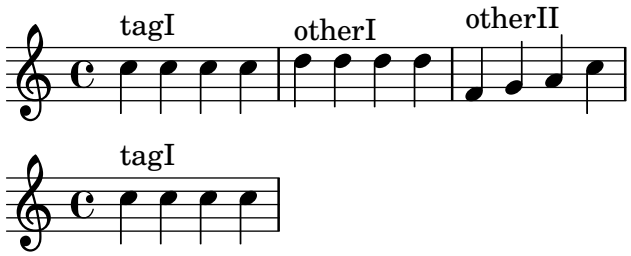
```
\tagGroup tagI,tagII

music = \relative {
  \tag tagI { c''4^"tagI" c c c }
  \tag tagII { a2^"tagII" a }
  \tag otherI { d4^"otherI" d d d }
  \tag otherII { f,4^"otherII" g a c }
}

\new Voice {
  \keepWithTag tagI \music
}

\resetTagGroups

\new Voice {
  \keepWithTag tagI \music
}
```



Soll hingegen nur eine einzelne Taggruppe aufgelöst werden, so ist `\resetTagGroup` mit einer Taggruppe zu nutzen. An dieser Stelle ist die Nutzung von `\tagGroupRef` zur Erstellung der Taggruppe vorteilhaft, um eine Referenz in Form einer Variablen zu erhalten.

```
tgI = \tagGroupRef tagI,tagII
tgII = \tagGroupRef otherI,otherII

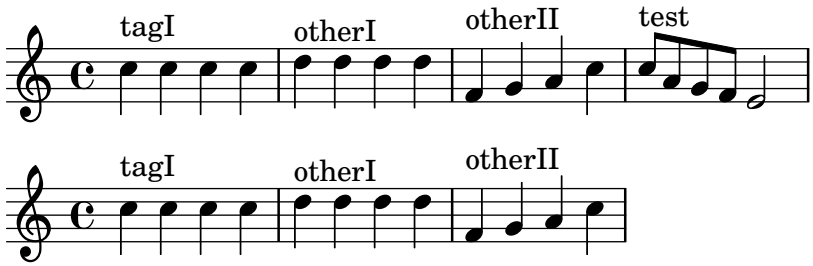
music = \relative {
  \tag tagI { c''4^"tagI" c c c }
  \tag tagII { a2^"tagII" a }
  \tag otherI { d4^"otherI" d d d }
  \tag otherII { f,4^"otherII" g a c }
  \tag test { c8^"test" a g f e2 }
}

% 'otherI', 'otherII' und 'test' sind nicht in der selben
```

```
% Taggruppe wie 'tagI' und werden daher nicht rausgefiltert.
\new Voice {
  \keepWithTag tagI \music
}

\resetTagGroup \tgI

% Jetzt sind nur noch 'otherI' und 'otherII' in einer Taggruppe,
% daher filtert '\keepWithTag tagI' jetzt 'tagII' und 'test' heraus.
\new Voice {
  \keepWithTag tagI \music
}
```



Eine weitere Modifikationsmöglichkeit bieten die Befehle \addToTagGroup und \removeFromTagGroup, die neue Tags zu bestehenden Taggruppen hinzufügen oder bereits vorhandene wieder entfernen können.

```
tgI = \tagGroupRef tagI,tagII
tgII = \tagGroupRef otherI,otherII

music = \relative {
  \tag tagI { c''4^"tagI" c c c }
  \tag tagII { a2^"tagII" a }
  \tag otherI { d4^"otherI" d d d }
  \tag otherII { f,4^"otherII" g a c }
  \tag test { c8^"test" a g f e2 }
}

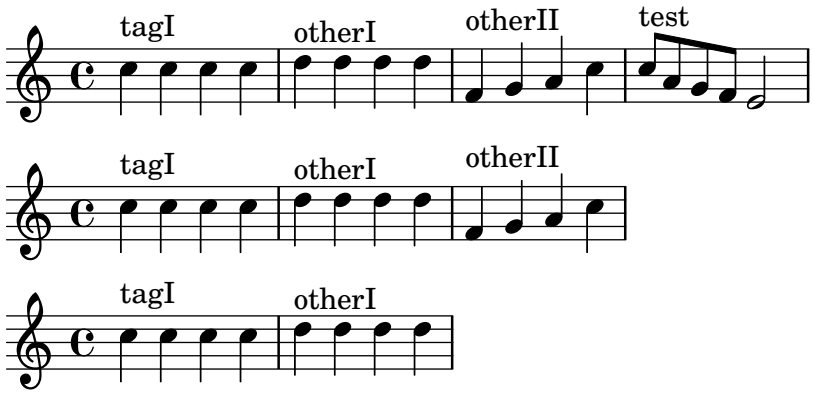
% 'otherI', 'otherII' und 'test' sind nicht in der gleichen
% Taggruppe wie 'tagI' und bleiben somit ungefiltert.
\new Voice {
  \keepWithTag tagI \music
}

\addToTagGroup \tgI test

% Jetzt sind 'tagI', 'tagII' und 'test' in der gleichen Taggruppe.
% Somit werden 'tagII' und 'test' herausgefiltert.
\new Voice {
  \keepWithTag tagI \music
}

\removeFromTagGroup \tgII otherII
\addToTagGroup \tgI otherII
```
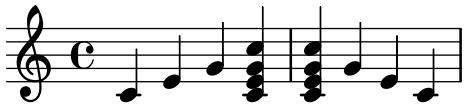
```
% Jetzt wird 'otherII' von der Taggruppe 'tgII' entfernt
% und der Taggruppe 'tgI' hinzugefügt.
% Damit ist nur noch 'otherI' in seiner eigenen Gruppe
% und wird damit nicht gefiltert.
\new Voice {
  \keepWithTag tagI \music
}
```



Manchmal will man Noten an einem bestimmen Platz in existierenden Noten einfügen. Dafür kann entweder \pushToTag oder \appendToTag benutzt werden, um Material hinter bzw. vor den Elementen (elements) der existierenden Noten einzufügen. Nicht alle musikalischen Konstruktionen haben Elemente, aber sequentielle und simultane Noten sind ziemlich gute Kandidaten:

```
test = { \tag #'here { \tag #'here <<c''>> } }

{
  \pushToTag #'here c'
  \pushToTag #'here e'
  \pushToTag #'here g' \test
  \appendToTag #'here c'
  \appendToTag #'here e'
  \appendToTag #'here g' \test
}
```
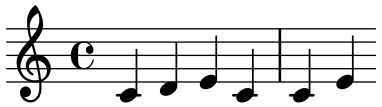


Beide Befehle erhalten einen Tag, das Material, das bei jedem Auftreten des Tags eingefügt werden soll und den Ausdruck, der mit dem Tag versehen ist. Der Befehl stellt sicher, dass alle Änderungen kopiert werden, sodass das ursprüngliche \test seine Bedeutung behält.

Es ist außerdem möglich, ganze Abschnitte auszutauschen.

```
music = { c' \tag #'here { d' } e' }
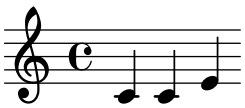
{
  \music
  \replaceWithTag #'here c' \music
}
```



Es ist zu beachten, dass dabei der Tag an sich mit ersetzt wird. Das bedeutet, man kann keine weiteren Ersetzungen oder Filterungen, die sich auf diesen Tag beziehen, mehr vornehmen.

```
music = { c' \tag #'here { d' } e' }

{
  \removeWithTag #'here % tut nichts, da der Tag ersetzt wurde
  \replaceWithTag #'here c'
  \music
}
```



Der \tag-Befehl kann auch in \markup-Umgebungen eingesetzt werden. Auch die Befehle \keep-with-tag, \remove-with-tag, \push-to-tag, \append-to-tag und \replace-with-tag können genutzt werden und verhalten sich wie ihre Pendants in musikalischen Umgebungen.

```
test = \markup {
  \tag #'a a
  \tag #'b b
  \tag #'c c
}

\markup { \keep-with-tag #'b \test }
\markup { \remove-with-tag #'b \test }
\markup { \push-to-tag #'c pre \test }
\markup { \append-to-tag #'c post \test }
\markup { \replace-with-tag #'c sub \test }
```

b

a c

a b pre c
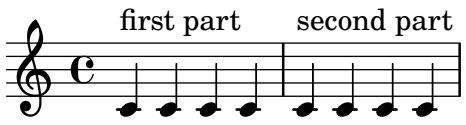
a b c post

a b sub

Musikalische \keepWithTag- und \removeWithTag-Befehle filtern auch Marken in \markup-Abschnitten innerhalb des betroffenen Musikabschnitts.

```
music = \relative {
  c'4^\markup { \tag #'one first \tag #'two second part } c c c
}

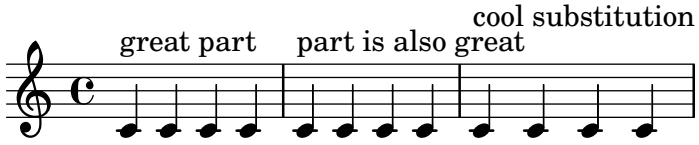
{
  \keepWithTag #'one \music
  \removeWithTag #'one \music
}
```



Soll an die markierten Stellen in \markup-Abschnitten von musikalischen Objekten etwas hinzugefügt werden, kann man \pushToTag, \appendToTag oder \removeWithTag nicht verwenden, denn diese fügen nur musikalische Ausdrücke hinzu. Hierfür wird \pushToTagMarkup, \appendToTagMarkup oder \removeWithTagMarkup benötigt.

```
music = \relative {
  c'4^\markup { \tag #'part part } c c c
}
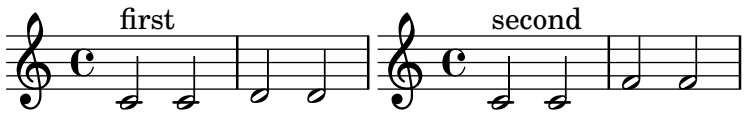
{
  \pushToTagMarkup #'part "great" \music
  \appendToTagMarkup #'part \markup { is also great } \music
  \replaceWithTagMarkup #'part \markup { cool substitution } \music
}
```



Die Filterung von Marken wirkt sich umgekehrt auch auf Musik innerhalb eines \score-Befehls in einem \markup-Abschnitt aus.

```
music = \relative {
  c'2^\markup { \tag #'first first \tag #'second second } c
  \tag #'first { d d }
  \tag #'second { f f }
}

\markup {
  \keep-with-tag #'first \score { \music }
  \remove-with-tag #'first \score { \music }
}
```



Bei der Verwendung von Marken in Markups ist allerdings beim Markieren von Listen besondere Vorsicht geboten. Während die Filter

```
\markup {
  \remove-with-tag #'test { a \tag #'test { b c } d }
}
```

a d

gut funktionieren, gibt es bei den einfügenden Befehlen \push-to-tag, \append-to-tag und \replace-with-tag Probleme.

```
\markup {
  \remove-with-tag #'test { a \tag #'test { b c } d }
}
\markup {
  \push-to-tag #'test "twice" { a \tag #'test { b c } d }
}
\markup {
  \replace-with-tag #'test "twice" { a \tag #'test { b c } d }
}
```

a d

a twice b twice c d

a twice twice d

Das liegt in diesem Falle daran, dass LilyPond das Konstrukt

```
\tag #'test { b c }
```

intern auflöst zu

```
\tag #'test b
\tag #'test c
```

und daher den übergebenen Text zweimal einfügt. Geht es wirklich darum, vor oder nach einer Liste an sich etwas einzufügen, muss auch die ganze Liste entsprechend markiert werden. Das geht mit `\tag-list`.

```
\markup {
  \push-to-tag #'test "once" { a \tag-list #'test { b c } d }
}
```

a once b c d

Siehe auch

Handbuch zum Lernen: Abschnitt “Stücke durch Variablen organisieren” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 5.2.4 [Automatische Kombination von Stimmen], Seite 175, Abschnitt 21.1 [LilyPond-Dateien einfügen], Seite 488.

Bekannte Probleme und Warnungen

Die Nutzung von `\relative` auf musikalische Abschnitte, die mittels `\keepWithTag` oder `\removeWithTag` gefiltert wurden, kann die Oktavlage verändern, da nur die Töne betrachtet werden, die nach der Filterung noch übrig sind. Wenn `\relative` zuerst angewendet wird, bevor `\keepWithTag` oder `\removeWithTag` filtern, kann das Problem vermieden werden, da `\relative` dann auf alle Töne angewendet wird und die Filterung danach erfolgt.

21.2.3 Globale Einstellungen benutzen

Man kann globale Einstellungen aus einer externen Datei einfügen:

```
lilypond -dinclude-settings=MY_SETTINGS.ly MY_SCORE.ly
```

Einstellungsgruppen, wie etwa Seitengröße, Schriftart oder Schriftschnitt, können in eigenen Dateien gespeichert werden. Das ermöglicht es, aus der gleichen Partitur unterschiedliche Editionen zu erstellen bzw. Standardeinstellungen für eine ganze Anzahl von Partituren wiederzuverwenden, indem man einfach die entsprechende Einstellungsdatei angibt.

Diese Technik funktioniert auch gut für Formatvorlagen, wie in Abschnitt “Formatvorlagen” in *Handbuch zum Lernen* behandelt.

Siehe auch

Handbuch zum Lernen: Abschnitt “Stücke durch Variablen organisieren” in *Handbuch zum Lernen*, Abschnitt “Formatvorlagen” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 21.1 [LilyPond-Dateien einfügen], Seite 488.

21.3 Sonderzeichen

21.3.1 Zeichenkodierung

LilyPond benutzt alle Zeichen, die durch das Unicode-Konsortium und ISO/IEC 10646 definiert sind. Hiermit wird den Zeichen fast aller Schriftsysteme der Welt ein eindeutiger Name und ein Code-Punkt zugewiesen, mit dem sie identifizierbar sind. Unicode kann mit mehreren Zeichenkodierungen verwirklicht werden. LilyPond benutzt die UTF-8-Kodierung (UTF = Unicode Transformation Format), in der die normalen Zeichen des lateinischen Alphabets mit einem Byte dargestellt werden, während alle anderen Zeichen zwischen zwei und vier Byte Länge haben.
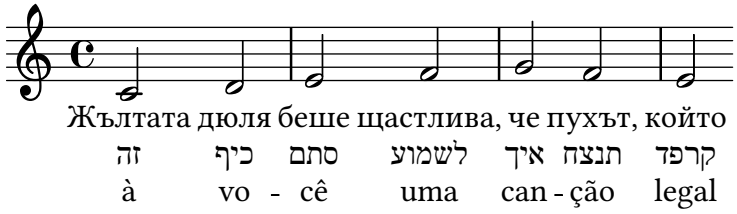
Das Aussehen des Zeichens wird bestimmt durch die gerade benutzte Schriftart (engl. font). In einer Schriftartdatei werden die Nummern der Unicode-Zeichen einem bestimmten Glyphen zugeordnet. LilyPond verwendet die Pango-Bibliothek um mehrsprachige Texte und komplexe Skripte korrekt zu setzen.

LilyPond verändert die Kodierung der Eingabedatei nicht. Das heißt, dass jeder Text – Überschriften, Gesangstext, Spielanweisungen etc. – der nicht nur aus ASCII-Zeichen besteht, in UTF-8 kodiert sein musst. Am einfachsten geht das, indem man einen Texteditor einsetzt, der mit Unicode-Zeichen umgehen kann. Die meisten modernen weit verbreiteten Editoren besitzen heute UTF-8-Unterstützung, wie etwa vim, Emacs, jEdit oder GEdit. Alle MS Windows-Systeme nach NT benutzen Unicode intern, sodass sogar Notepad Dateien in UTF-8 lesen und speichern kann. Ein Editor mit mehr Funktionen unter Windows ist BabelPad oder Notepad++.

Wenn eine LilyPond-Eingabedatei nicht-ASCII-Zeichen enthält und nicht in UTF-8 gespeichert ist, gibt es folgende Fehlermeldung:

```
FT_Get_Glyph_Name () error: invalid argument
```

Heir ein Beispiel mit Kyrilliza, hebräischem und portugiesischem Text:



21.3.2 Unicode

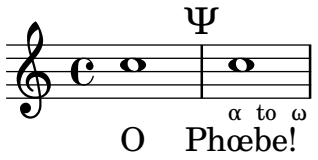
Um einen einzelnen Buchstaben zu notieren, für den die Unicode-Ziffernfolge bekannt ist, der aber nicht auf der Tastatur zu finden ist, kann der Befehl `\char ##xhhhh` oder `\char #dddd` innerhalb einer `\markup`-Umgebung benutzt werden. Hierbei bedeutet `hhhh` die hexadezimale Zahl und `ddd` die entsprechende dezimale Zahl für das erforderliche Zeichen. Nullen zu Beginn können ausgelassen werden, aber normalerweise werden alle vier Zeichen der hexadezimalen Notation notiert. (Achten Sie darauf, dass Sie *nicht* UTF-8-Codepunkte einsetzen, weil UTF-8 zusätzliche Bits enthält, die die Nummer der Oktets bezeichnet.) Unicode-Tabellen und ein Verzeichnis der Zeichenbezeichnungen mit einer hexadezimalen Verweiszahl finden sich auf der Internetseite des Unicode Consortiums: `https://www.unicode.org/`.

Mit `\char ##x03BE` und `\char #958` wird beispielsweise das Unicode-Zeichen U+03BE notiert, welches die Unicode-Bezeichnung „Greek Small Letter Xi“ hat.

Alle existierenden Unicode-Zeichen können auf diese Weise notiert werden, und wenn für alle Zeichen dieses Format angewandt wird, muss die Datei nicht im utf-8-Format gespeichert werden. Es muss natürlich auch noch eine Schriftart auf dem System installiert sein, die die notierten Zeichen darstellen kann.

Das nächste Beispiel zeigt, wie Unicode-Zeichen an vier Stellen mit dem hexadezimalen Code notiert werden: in einem Übungszeichen, als Artikulationszeichen, im Gesangstext und als normaler Text außerhalb der Partitur.

```
\score {
  \relative {
    c''1 \mark \markup { \char ##x03A8 }
    c1_\markup { \tiny { \char ##x03B1 " to " \char ##x03C9 } }
  }
  \addlyrics { O \markup { \concat { Ph \char ##x0153 be! } } }
}
\markup { "Copyright 2008--2026" \char ##x00A9 }
```



Copyright 2008--2026 ©

Um das Copyright-Zeichen zu notieren, kann folgender Code eingesetzt werden:

```
\header {
  copyright = \markup { \char ##x00A9 "2008" }
}
```

21.3.3 ASCII-Aliase

Eine Liste von ASCII-Befehlen für Sonderzeichen kann eingefügt werden:

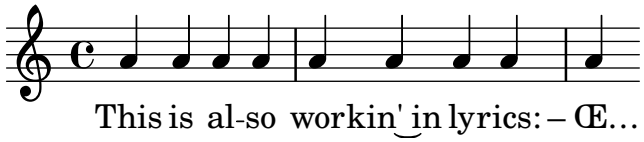
```
\paper {
  #(include-special-characters)
}

\markup "&flqq; &ndash; &OE;uvre incomplète&hellip; &frqq;"

\score {
  \new Staff { \repeat unfold 9 a'4 }
  \addlyrics {
    This is al -- so wor -- kin'~in ly -- rics: &ndash;_&OE;&hellip;
  }
}

\markup \column {
  "The replacement can be disabled:"
  "&ndash; &OE; &hellip;"
  \override #'(replacement-alist . ()) "&ndash; &OE; &hellip;"
}
```

« – Œuvre incomplète… »



The replacement can be disabled:
– Œ …
– &OE; …

Man kann auch eigen Aliase erstellen, entweder global:

```
\paper {
  #(add-text-replacements!
    '(("100" . "hundred")
      ("dpi" . "dots per inch")))
}
\markup "A 100 dpi."
```

A hundred dots per inch.

oder lokal:

```
\markup \replace #'(("100" . "hundred")
                    ("dpi" . "dots per inch")) "A 100 dpi."
```

A hundred dots per inch.

Siehe auch

Notationsreferenz: Abschnitt A.12 [Liste der Sonderzeichen], Seite 759.

Installierte Dateien: `ly/text-replacements.ly`.

22 Ausgabe kontrollieren

22.1 Notationsfragmente extrahieren

Es ist möglich, kleine Abschnitte einer großen Partitur direkt aus der Quelldatei zu erzeugen. Das kann damit verglichen werden, dass man mit der Schere bestimmte Regionen ausschneidet.

Es wird erreicht, indem man die Takte, die ausgeschnitten werden sollen (engl. to clip = ausschneiden), extra definiert. Mit folgender Definition beispielsweise

```
\layout {
  clip-regions
  = #(list
      (cons
       (make-rhythmic-location 5 1 2)
       (make-rhythmic-location 7 3 4)))
}
```

wird ein Fragment ausgeschnitten, dass auf der Mitte des fünften Taktes beginnt und im siebten Takt endet. Die Bedeutung von `5 1 2` ist: nach einer Halben in Takt fünf, `7 3 4` heißt: nach drei Vierteln in Takt 7.

Weitere Bereiche, die ausgeschnitten werden sollen, können definiert werden, indem mehrere derartige Paare definiert werden.

Um diese Funktion auch nutzen zu können, muss LilyPond mit dem Parameter `-dclip-systems` aufgerufen werden. Die Schnipsel werden als EPS ausgegeben und dann zu PDF und PNG konvertiert, wenn diese Formate auch als Parameter angegeben werden.

Zu mehr Information über Ausgabeformate siehe Abschnitt “lilypond aufrufen” in *Anwendungsbenutzung*.

22.2 Korrigierte Musik überspringen

Wenn man Noten eingibt oder kopiert, sind meistens nur die Noten nahe dem Ende (wo gerade neue Noten notiert wurden) wichtig für Kontrolle und Korrektur. Um die Korrektur zu beschleunigen, kann eingestellt werden, dass nur die letzten paar Takte angezeigt werden. Das erreicht man mit dem Befehl

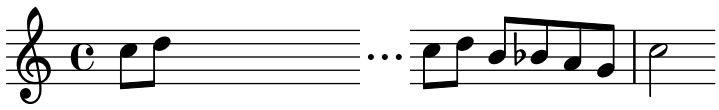
```
showLastLength = R1*5
\score { ... }
```

in der Quelldatei. Damit werden nur die letzten fünf Takte (in einem 4/4-Takt) eines jeden `\score`-Abschnitts übersetzt. Besonders bei längeren Stücken ist es meistens sehr viel schneller, nur einen kleinen Teil des Stückes zu setzen als die gesamte Länge. Wenn man am Anfang eines Stückes arbeitet (weil etwa ein neuer Teil hinzugefügt werden soll), kann auch die `showFirstLength`-Eigenschaft nützlich sein.

Nur bestimmte Teile einer Partitur zu überspringen, kann mit der Eigenschaft `Score.skipTypesetting` sehr genau kontrolliert werden. Für den Bereich, für den sie auf „wahr“ gesetzt wird, wird kein Notensatz ausgegeben.

Diese Eigenschaft kann auch benutzt werden, um die MIDI-Ausgabe zu kontrollieren. Hiermit werden alle Ereignisse, auch Tempo- und Instrumentenwechsel ausgelassen. Man muss also sehr genau darauf achten, dass nichts unerwartetes geschieht.

```
\relative {
  c''8 d
  \set Score.skipTypesetting = ##t
  e8 e e e e e e
  \set Score.skipTypesetting = ##f
  c8 d b bes a g c2
}
```



In polyphoner Notation wirkt sich `Score.skipTypesetting` auf alle Stimmen und Systeme aus, sodass noch mehr Zeit bei der Übersetzung der Datei gespart wird.

22.3 Alternative Ausgabeformate
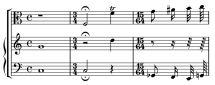
Das Standardausgabeformat für gedruckte Partituren ist PDF (Portable Document Forma) und PS (PostScript). SVG (Scalable Vector Graphics), EPS (Encapsulated PostScript) und PNG (Portable Network Graphics) gibt es auch als Ausgabeformate über die Kommandozeile. Siehe Abschnitt "Grundlegende Optionen auf der Kommandozeile für LilyPond" in *Anwendungsbenutzung*.

22.4 Die Notationsschriftart verändern
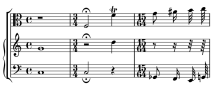
Gonville ist eine Alternative zu der Emmentaler-Schriftart, die in LilyPond eingesetzt wird und kann unter der Adresse

`http://www.chiark.greenend.org.uk/~sgtatham/gonville/` (`http://www.chiark.greenend.org.uk/~sgtatham/gonville/`)

heruntergeladen werden. Hier einige Takte Noten mit der Gonville-Schriftart:



Und hier einige Beispieltakte in der Feta-Glyphe:



Installationsanweisungen für MacOS

Laden Sie die Datei herunter und entpacken Sie die ZIP-Datei. Kopieren Sie das `lilyfonts`-Verzeichnis nach *SHARE_DIR*`/lilypond/current`; für mehr Information siehe Abschnitt "Mehr Information" in *Handbuch zum Lernen*. Benennen Sie das existierende `fonts`-Verzeichnis in `fonts_orig` um und benennen Sie das Verzeichnis `lilyfonts` in `fonts`. Das alte Verzeichnis `fonts_orig` können Sie einfach in `fonts` zurückbenennen, um wieder nach Feta zu wechseln.

Siehe auch

Handbuch zum Lernen: Abschnitt "Mehr Information" in *Handbuch zum Lernen*.

Bekannte Probleme und Warnungen

Gonville kann nicht verwendet werden, um Alte Notation zu setzen und es ist wahrscheinlich, dass neuere Glyphen in späteren Versionen von LilyPond nicht in Gonville enthalten sein werden. Bitte lesen Sie die Webseite des Autoren zu mehr Information hierzu und zu anderen Einzelheiten, wie auch der Lizenz von Gonville.

23 MIDI-Ausgabe

MIDI (Musical Instrument Digital Interface) ist ein Standard zur Kontrolle und Interaktion mit digitalen Instrumenten. Eine MIDI-Datei ist eine Anzahl von Noten auf einer Anzahl von Bändern/Stimmen. Es ist keine eigentliche Klangdatei, denn man benötigt spezielle Programme die die Notenereignisse in Klang umwandeln können.

Der Notensatz von LilyPond kann in MIDI umgewandelt werden, so dass man sich anhören kann, was man notiert hat. Das hilft oft sehr gut bei der Überprüfung: falsche Oktaven oder falsche Versetzungszeichen lassen sich meist sehr gut hören.

Normale MIDI-Ausgabe ist etwas roh; optional kann eine verbesserte und realistischere MIDI-Ausgabe mit einem Abschnitt 23.7 [Artikuliere-Skript], Seite 515, erzeugt werden.

Die MIDI-Ausgabe benötigt einen Kanal für jedes System und reserviert Kanal 10 für Schlagzeug. Es gibt nur 16 MIDI-Kanäle pro Gerät, sodass MIDI-Kanäle mehrfach benutzt werden, wenn eine Partitur mehr als 15 Notensysteme hat.

23.1 MIDI-Dateien erstellen

Um eine MIDI-Datei aus einer LilyPond-Quelldatei zu erstellen, muss eine `\midi`-Umgebung zu der `\score`-Umgebung hinzugefügt werden, etwa so:

```
\score {
  ...Noten...
  \midi { }
}
```

Wenn in einer `\score`-Umgebung nur eine `\midi`-Umgebung, aber keine `\layout`-Umgebung vorkommt, wird nur MIDI produziert. Wenn auch die Notation gewünscht ist, muss zusätzlich die `\layout`-Umgebung vorhanden sein:

```
\score {
  ...music...
  \midi { }
  \layout { }
}
```

Tonhöhen, Rhythmen, Überbindungen, Dynamik und Tempoänderungen werden korrekt in das MIDI-Format übersetzt. Dynamikzeichen, Crescendo und Decrescendo werden in den MIDI-Lautstärkekanal übertragen. Dynamikzeichen werden in einen bestimmten Lautstärkenwert übersetzt, Crescendo und Decrescendo erreichen einen Übergang zwischen Lautstärkewerten. Die Wirkung von Dynamikzeichen kann auch aus der MIDI-Datei entfernt werden. Siehe hierzu Abschnitt 23.2 [Der MIDI-Block], Seite 508.

Das Anfangstempo und spätere Tempoänderungen können mit dem `\tempo`-Befehl innerhalb der Notation notiert werden. Er bewirkt Tempoänderungen auch in der MIDI-Datei. Der Befehl setzt gleichzeitig auch eine Tempobezeichnung in die Noten, welches aber auch unterdrückt werden kann, siehe Abschnitt 2.3.2 [Metronomangabe], Seite 69. Eine andere Möglichkeit, ein eigenes MIDI-Tempo anzugeben, wird weiter unten gezeigt, siehe Abschnitt 23.2 [Der MIDI-Block], Seite 508.

Aufgrund einiger Einschränkungen auf Windows ist auf Windows-Systemen die Standarddateierweiterung von MIDI-Dateien `.mid`. Andere Betriebssysteme verwenden weiterhin `.midi`. Wenn eine andere Endung erwünscht ist, kann man die folgende Zeile auf oberster Ebene der Quelldatei, vor Beginn eines `\book`, `\bookpart` oder `\score`-Blocks einfügen:

```
#(ly:set-option 'midi-extension "midi")
```

Diese Codezeile setzt die Dateiendung auf `.midi`.

Als Alternative kann man diese Option auch als Kommandozeilenparameter übergeben:

```
lilypond ... -dmidi-extension=midi lilyDatei.ly
```

Instrumentenbezeichnungen

Das MIDI-Instrument, mit dem ein bestimmtes System wiedergegeben werden soll, wird durch die `Staff.midiInstrument`-Eigenschaft bestimmt, die auf eine Instrumentenbezeichnung gesetzt werden muss. Die Bezeichnungen sind aufgelistet in Abschnitt A.6 [MIDI-Instrumente], Seite 655, und müssen in der dort definierten Schreibweise notiert werden.

```
\new Staff {
  \set Staff.midiInstrument = "glockenspiel"
  ...Noten...
}
\new Staff \with {midiInstrument = "cello"} {
  ...Noten...
}
```

Wenn die Schreibweise nicht genau einem definierten Instrument aus der Liste entspricht, wird ein Piano-Klang benutzt (`"acoustic grand"`).

Ausgewählte Schnipsel
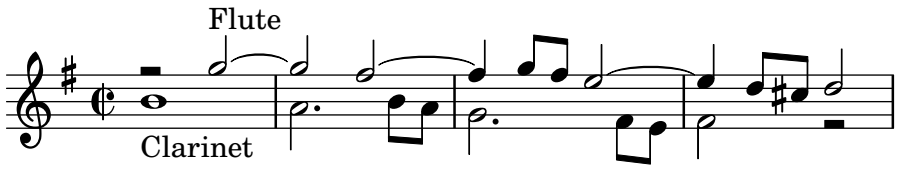
Changing MIDI output to one channel per voice

When outputting MIDI, the default behavior is for each staff to represent one MIDI channel, with all the voices on a staff amalgamated. This minimizes the risk of running out of MIDI channels, since there are only 16 available per track.

However, by moving the `Staff_performer` to the `Voice` context, each voice on a staff can have its own MIDI channel, as is demonstrated by the following example: despite being on the same staff, two MIDI channels are created, each with a different `midiInstrument`.

```
\score {
  \new Staff <<
    \new Voice \relative c''' {
      \set midiInstrument = "flute"
      \voiceOne
      \key g \major
      \time 2/2
      r2 g-"Flute" ~
      g fis ~
      fis4 g8 fis e2 ~
      e4 d8 cis d2
    }
    \new Voice \relative c'' {
      \set midiInstrument = "clarinet"
      \voiceTwo
      b1-"Clarinet"
      a2. b8 a
      g2. fis8 e
      fis2 r
    }
  >>
  \layout { }
  \midi {
```

```
    \context {
      \Staff
      \remove "Staff_performer"
    }
    \context {
      \Voice
      \consists "Staff_performer"
    }
    \tempo 2 = 72
  }
}
```



Bekannte Probleme und Warnungen

Veränderungen der MIDI-Lautstärke sind nur effektiv, wenn sie zu Beginn einer Note angefordert werden, sodass die Lautstärke während einer Notendauer nicht geändert werden kann.

Nicht alle MIDI-Spieler können Tempoänderungen richtig wiedergeben. Spieler, die hierzu in der Lage sind, sind unter Anderen MS Windows Media Player und timidity (`http://timidity.sourceforge.net/`).

23.2 Der MIDI-Block

Eine `\midi`-Umgebung muss innerhalb von einer `\score`-Umgebung vorkommen, wenn MIDI-Ausgabe gewünscht ist. Sie entspricht der `\layout`-Umgebung, aber ist etwas einfacher aufgebaut. Oft wird die MIDI-Umgebung einfach leer gelassen, aber hier können auch Kontexte umgeändert werden, neue Kontexte definiert werden oder neue Werte definiert werden. Das folgende Beispiel etwa definiert das MIDI-Tempo, ohne dass in der Partitur eine Metronombezeichnung gesetzt wird:

```
\score {
  ...Noten...
  \midi {
    \tempo 4 = 72
  }
}
```

Hier wird das Tempo auf 72 Viertelnoten pro Minute definiert. `\tempo` ist eigentlich ein Musikbefehl, der die Eigenschaften während der Interpretation der Musik einstellt: im Kontext von Ausgabedefinitionen wie etwa einem `\midi`-Kontext werden sie neu interpretiert, als ob es sich um Kontextmodifikatoren handelte.

Kontextdefinitionen des `\midi`-Kontextes entsprechen der Syntax, wie sie in der `\layout`-Umgebung benutzt wird. Klangübersetzungsmodule werden `performer` genannt. Die Kontexte für die MIDI-Ausgabe sind in der Datei `../ly/performer-init.ly` definiert, siehe Abschnitt "Mehr Information" in *Handbuch zum Lernen*. Um beispielsweise die Auswirkung von Dynamikzeichen aus der MIDI-Ausgabe zu entfernen, müssen folgende Zeilen eingefügt werden:

```
\midi {
  ...
  \context {
    \Voice
```

```
    \remove Dynamic_performer
  }
}
```

Eine MIDI-Ausgabe wird nur erstellt, wenn die `\midi`-Umgebung in eine Partiturumgebung eingefügt wird, die mit dem Befehl `\score` beginnt.

```
\score {
  { ...Noten... }
  \midi { }
}
```

23.3 Was geht in die MIDI-Ausgabe

In MIDI unterstützt

Die folgenden Notationselemente werden in die MIDI-Ausgabe aufgenommen:

- Tonhöhen
- Mikrotöne (siehe Abschnitt 1.1.3 [Versetzungszeichen], Seite 7. Für die Ausgabe wird ein Spieler benötigt, der Tonhöhen verändern kann.)
- Akkorde, die als Symbole notiert werden
- Rhythmen, die als Dauern notiert sind, inklusive N-tolen
- Tremolo, das ohne ‚:[*Zahl*]‘ notiert ist
- Überbindungen
- Dynamikzeichen
- Crescendi, decrescendi zu mehreren Noten
- Tempoänderungen, die mit einer Tempo-Bezeichnung eingegeben werden
- Gesangstext

Durch Einsatz vom Abschnitt 23.7 [Artikuliere-Skript], Seite 515, können noch einige Elemente zu der Liste hinzugefügt werden:

- Artikulationen (Bögen, Staccato usw.)
- Triller usw.
- Rallentando und accelerando

In MIDI nicht unterstützt
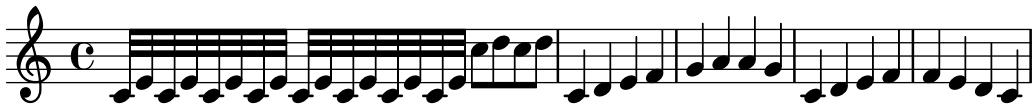
Folgende Notationselemente werden nicht in die MIDI-Ausgabe einbezogen, außer am setzt das Abschnitt 23.7 [Artikuliere-Skript], Seite 515, ein:

- Rhythmus, der als Anmerkung notiert wird, bspw. Swing
- Tempoveränderungen, die als Anmerkung ohne Tempobezeichnung notiert werden
- Staccato und andere Artikulationen und Ornamente
- Legato- und Phrasierungsbögen
- Crescendi, decrescendi zu einer einzelnen Note
- Tremolo, notiert mit ‚:[*number*]‘
- Bezifferter Bass
- Akkorde mit Mikrotönen

23.4 Wiederholungen im MIDI

Mit einigen Veränderungen im Notentext können alle Wiederholungstypen auch in der MIDI-Ausgabe wiedergegeben werden. Das wird erreicht, indem die `\unfoldRepeats`-Funktion eingesetzt wird. Diese Funktion verändert alle Wiederholungen in ausgeschriebene Noten.

```
\unfoldRepeats {
  \repeat tremolo 8 { c'32 e' }
  \repeat percent 2 { c''8 d'' }
  \repeat volta 2 { c'4 d' e' f' }
  \alternative {
    { g' a' a' g' }
    { f' e' d' c' }
  }
}
\bar "|."
```



In Parituren mit mehreren Stimmen funktioniert das Ausschreiben der Wiederholungen im MIDI nur richtig, wenn *jede* Stimme vollständig notierte Wiederholungsanweisungen enthält.

Wenn eine Partitur mit diesem `\unfoldRepeats`-Befehl erstellt wird, ist er notwendig, zwei `\score`-Umgebungen einzurichten: in der einen werden die Wiederholungen ausgeschrieben und nur eine MIDI-Ausgabe produziert, in der anderen werden die Wiederholungen notiert und als Partitur gesetzt. Das Beispiel gibt einen Hinweis, wie eine derartige Datei aussehen kann:

```
\score {
  ..music..
  \layout { .. }
}
\score {
  \unfoldRepeats ..music..
  \midi { .. }
}
```

23.5 MIDI-Lautstärke kontrollieren

Dynamik in der MIDI-Ausgabe wird durch den Dynamic_performer erstellt, welcher sich in einem `Voice`-Kontext befindet. Es ist möglich, sowohl die generelle Lautstärke einer MIDI-Datei als auch relative Lautstärken von Dynamikanweisungen und auch relative Lautstärke von einzelnen Instrumenten einzustellen.

Dynamik-Zeichen
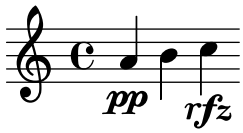
Dynamikanweisungen werden als ein bestimmter Bruch der insgesamt zur Verfügung stehenden MIDI-Lautstärke notiert. Die Standardbrüche reichen von 0,25 für *ppppp* bis hin zu 0,95 für *fffff*. Diese Anweisung befinden sich in der Datei `../scm/midi.scm`, siehe auch Abschnitt “Mehr Information” in *Handbuch zum Lernen*. Diese Brüche können nach Belieben geändert oder erweitert werden, indem eine Funktion erstellt wird, die ein Dynamikzeichen als Argument nimmt und den erforderlichen Bruch ausgibt; schließlich muss noch `Score.dynamicAbsoluteVolumeFunction` auf diese Funktion gesetzt werden.

Beispielhaft soll gezeigt werden, wie man eine *Rinforzando*-Dynamik, `\rfz`, auch in die MIDI-Ausgabe übernehmen kann. Gleiches gilt für neue, selbstdefinierte Dynamikzeichen, die in den

Standarddefinitionen nicht enthalten sind. Die Scheme-Funktion, die hier definiert wird, setzt den Bruch von 0,9 für eine rfz-Anweisung und ruft andernfalls die Standardanweisungen auf:

```
#(define (myDynamics dynamic)
    (if (equal? dynamic "rfz")
      0.9
      (default-dynamic-absolute-volume dynamic)))

\score {
  \new Staff {
    \set Staff.midiInstrument = "cello"
    \set Score.dynamicAbsoluteVolumeFunction = #myDynamics
    \new Voice {
      \relative {
        a'4\pp b c-\rfz
      }
    }
  }
  \layout {}
  \midi {}
}
```



Alternativ, insbesondere wenn die gesamte Tabelle der MIDI-Lautstärken umdefiniert werden soll, ist es besser, die *default-dynamic-absolute-volume*-Prozedur in der Datei `../scm/midi.scm` und die hiermit verknüpfte Tabelle als Modell zu benutzen. Das letzte Beispiel dieses Abschnittes zeigt, wie das gemacht werden kann.

MIDI-Lautstärke
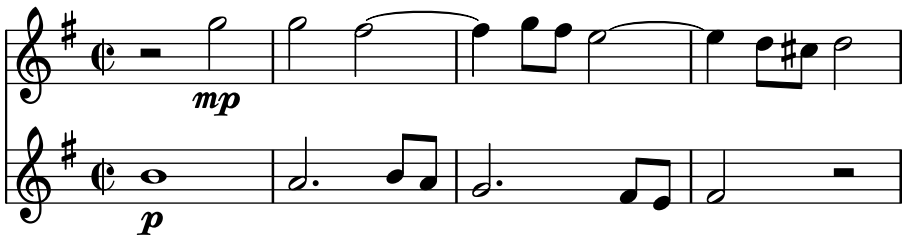
Die generellen Mindest- und Höchstwerte für die Lautstärke der MIDI-Datei wird kontrolliert, indem die Eigenschaften `midiMinimumVolume` und `midiMaximumVolume` auf der `Score`-Ebene gesetzt werden. Diese Eigenschaften haben nur Einfluss auf Dynamikzeichen, sodass ein Dynamikzeichen direkt an den Anfang der Partitur gestellt werden muss, wenn diese Einstellung von Anfang an Wirkung zeigen soll. Der Bruch, der dann den einzelnen Dynamikzeichen entspricht, wird mit der Formel

```
midiMinimumVolume + (midiMaximumVolume - midiMinimumVolume) * Bruch
```

errechnet. Im folgenden Beispiel wird die generelle MIDI-Lautstärke auf den Bereich zwischen 0.2 und 0.5 eingeschränkt.

```
\score {
  <<
    \new Staff {
      \key g \major
      \time 2/2
      \set Staff.midiInstrument = "flute"
      \new Voice \relative {
        r2 g''\mp g fis~
        4 g8 fis e2~
        4 d8 cis d2
      }
```

```
    }
    \new Staff {
      \key g \major
      \set Staff.midiInstrument = "clarinet"
      \new Voice \relative {
        b'1\p a2. b8 a
        g2. fis8 e
        fis2 r
      }
    }
  >>
  \layout {}
  \midi {
    \tempo 2 = 72
    \context {
      \Score
      midiMinimumVolume = #0.2
      midiMaximumVolume = #0.5
    }
  }
}
```
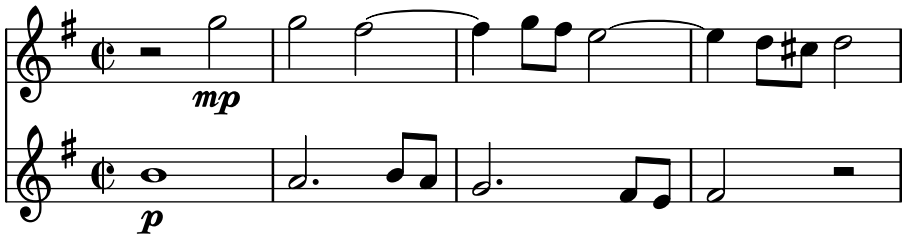


Verschiedene Instrumente angleichen (i)

Wenn die Mindest- und Höchstwerte für die MIDI-Lautstärke innerhalb eines `Staff`-Kontextes gesetzt werden, kann damit die relative Lautstärke einzelner Instrumente kontrolliert werden. Damit kann man die Qualität der MIDI-Datei merklich verbessern.

In diesem Beispiel wird die Lautstärke der Klarinette relativ zur Lautstärke der Flöte verringert. In jeder Stimme muss eine Dynamikanweisung für die erste Note gesetzt werden, damit diese Einstellung korrekt funktioniert.

```
\score {
  <<
    \new Staff {
      \key g \major
      \time 2/2
      \set Staff.midiInstrument = "flute"
      \set Staff.midiMinimumVolume = #0.7
      \set Staff.midiMaximumVolume = #0.9
      \new Voice \relative {
        r2 g''\mp g fis~
        4 g8 fis e2~
        4 d8 cis d2
      }
    }
    \new Staff {
```

```
      \key g \major
      \set Staff.midiInstrument = "clarinet"
      \set Staff.midiMinimumVolume = #0.3
      \set Staff.midiMaximumVolume = #0.6
      \new Voice \relative {
        b'1\p a2. b8 a
        g2. fis8 e
        fis2 r
      }
    }
  >>
  \layout {}
  \midi {
    \tempo 2 = 72
  }
}
```



Verschiedene Instrumente angleichen (ii)

Wenn Mindest- und Höchstwerte für die Lautstärke der MIDI-Datei nicht vorgegeben werden, nimmt LilyPond standardmäßig einige Anpassungen für die Lautstärken bestimmter Instrumente vor. Diese Instrumente und ihre entsprechende Veränderung lassen sich aus der Tabelle *instrument-equalizer-alist* in der Datei `../scm/midi.scm` entnehmen.

Dieser grundlegende Equalizer kann ersetzt werden, indem die Funktion `instrumentEqualizer` im `Score`-Kontext auf eine neue Scheme-Funktion gesetzt wird, die MIDI-Instrumentbezeichnungen als einziges Argument akzeptiert und ein Zahlenpaar ausgibt, das den Höchst- und Mindestwert für die Lautstärke des entsprechenden Instruments darstellt. Die Ersetzung der Standardfunktion wird auf gleiche Weise vorgenommen, wie es schon für die `dynamicAbsoluteVolumeFunction` zu Beginn dieses Abschnittes gezeigt wurde. Der Standard-Equalizer, *default-instrument-equalizer* in der Datei `../scm/midi.scm` zeigt, wie solche eine Funktion erstellt werden kann.

Das folgende Beispiel definiert für die Flöte und Klarinette relative Lautstärkewerte, die denen des vorigen Beispiels entsprechen.
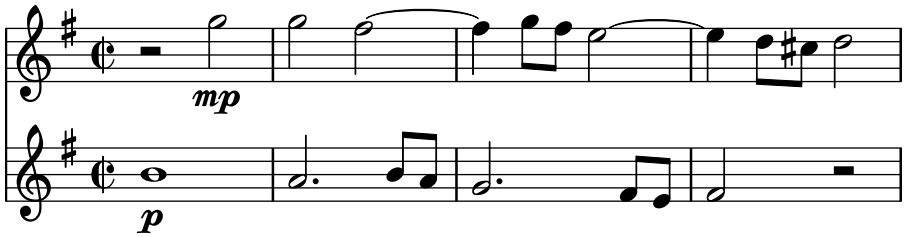
```
#(define my-instrument-equalizer-alist '())

#(set! my-instrument-equalizer-alist
  (append
    '(
      ("flute" . (0.7 . 0.9))
      ("clarinet" . (0.3 . 0.6)))
    my-instrument-equalizer-alist))

#(define (my-instrument-equalizer s)
  (let ((entry (assoc s my-instrument-equalizer-alist)))
    (if entry
```

```
      (cdr entry))))

\score {
  <<
    \new Staff {
      \key g \major
      \time 2/2
      \set Score.instrumentEqualizer = #my-instrument-equalizer
      \set Staff.midiInstrument = "flute"
      \new Voice \relative {
        r2 g''\mp g fis~
        4 g8 fis e2~
        4 d8 cis d2
      }
    }
    \new Staff {
      \key g \major
      \set Staff.midiInstrument = "clarinet"
      \new Voice \relative {
        b'1\p a2. b8 a
        g2. fis8 e
        fis2 r
      }
    }
  >>
  \layout { }
  \midi {
    \tempo 2 = 72
  }
}
```



23.6 Schlagzeug in MIDI

Schlagzeuginstrumente werden üblicherweise in einem `DrumStaff`-Kontext notiert. Aus diese Weise werden sie korrekt in den MIDI-Kanal 10 ausgegeben. Eine Schlagzeuge mit diskreten Tonhöhen, wie Xylophon, Marima, Vibraphone, Pauken usw. werden wie „normale“ Instrumente in einem `Staff`-Kontext notiert. Nur so lässt sich auch hier eine richtige MIDI-Ausgabe erreichen.

Einige Instrumente, die keine diskreten Tonhöhen haben, können nicht über den MIDI-Kanal 10 erreicht werden und müssen deshalb in einem normalen `Staff`-Kontext mit passenden normalen Tonhöhen notiert werden. Es handelt sich um `melodic tom`, `taiko drum`, `synth drum` u. A.

Viele Schlagzeuginstrumente sind nicht in den MIDI-Standard aufgenommen, z. B. Kastagnetten. Die einfachste Methode, derartige Instrumente zu ersetzen, ist, einen Klang auszuwählen, der ihnen halbwegs ähnlich kommt.

Bekannte Probleme und Warnungen

Weil der MIDI-Standard keine Peitschenschläge kennt, wird ein Schlagstock (sidestick) für diesen Zweck eingesetzt.

23.7 Artikuliere-Skript

Eine realistischere MIDI-Ausgabe ist möglich, wenn man das Artikuliere-Skript einsetzt. Es versucht, Artikulationen (Bögen, Staccato) mit einzubeziehen, indem Noten mit sequentieller Musik von passender Verlängerung oder Verkürzung mit entprechenden Skips ersetzt werden. Es versucht auch, Triller usw. klingen zu lassen und rallantando und accelerando wahrzunehmen.

Um das Artikuliere-Skript einzusetzen, muss oben in der Eingabedatei

```
\include "articulate.ly"
```

eingefügt werden. Im `\score`-Abschnitt schreibt man

```
\unfoldRepeats \articulate <<
der Rest der Partitur...
>>
```

Wenn die Eingabedatei auf diese Art verändert wurd, wird die Notenausgabe stark verändert, aber die MIDI-Ausgabe produziert ein besseres Ergebnis.

Wenn auch nicht unbedingt notwendig, damit das Artikuliere-Skript funktioniert, bietet es sich an, `\unfoldRepeats` wie im Beispiel oben zu verwenden, weil dadurch Abkürzungen wie etwa Triller ausgeführt werden.

Bekannte Probleme und Warnungen

Articulate verkürzt Akkorde und manche Musik (besonders Orgelmusik) kann schlechter klingen.

24 Musikalische Information extrahieren

Neben graphischer Ausgabe und MIDI kann LilyPond auch die musikalische Information als Text anzeigen:

24.1 LilyPond-Notation anzeigen

Mit der musikalischen Funktion `\displayLilyMusic` kann man einen musikalischen Ausdruck anzeigen. Um die Ausgabe zu sehen, wird LilyPond üblicherweise auf der Kommandozeile aufgerufen. Beispielsweise

```
{
  \displayLilyMusic \transpose c a, { c4 e g a bes }
}
```

zeigt an:

```
{ a,4 cis e fis g }
```

Standardmäßig gibt LilyPond diese Nachrichten auf die Kommandozeile aus, zusammen mit all den anderen LilyPond-Nachrichten über die Kompilation. Um die Nachrichten zu speichern, kann man die Ausgabe in eine Datei umleiten:

```
lilypond file.ly >display.txt
```

LilyPond zeit nicht nur die musikalischen Ausdrücke an, sondern interpretiert sie auch (weil `\displayLilyMusic` sie an das Programm zurückgibt, zusätzlich zur Anzeige). Das ist sehr praktisch, denn man kann einfach `\displayLilyMusic` in vorhandene Noten einfügen, um Informationen darüber zu erhalten. Wenn Sie nicht wollen, dass LilyPond die angezeigten musikalischen Ausdrücke auch interpretiert, muss `\void` eingesetzt werden, damit der Ausdruck für die Interpretation ignoriert wird:

```
{
  \void \displayLilyMusic \transpose c a, { c4 e g a bes }
}
```

24.2 Musikalische Scheme-Ausdrücke anzeigen

Siehe Abschnitt “Musikalische Funktionen darstellen” in *Extending*.

24.3 Musikalische Ereignisse in einer Datei speichern

Musikereignisse können in einer Datei Notensystem für Notensystem gespeichert werden, indem eine Datei in die Paritur eingefügt wird:

```
\include "event-listener.ly"
```

Das erstellt Dateien mit der Bezeichnung `DATEINAME-SYSTEMNAME.notes` oder `DATEINAME-unnamed-staff.notes` für jedes Notensystem. Wenn mehrere Systeme ohne Bezeichnung auftreten, werden die Ereignisse aller Notensysteme zusammen in der selben Datei gemischt. Die Ausgabe sieht folgendermaßen aus:

```
0.000   note     57       4   p-c 2 12
0.000   dynamic  f
0.250   note     62       4   p-c 7 12
0.500   note     66       8   p-c 9 12
0.625   note     69       8   p-c 14 12
0.750   rest     4
0.750   breathe
```

Die Syntax ist eine durch Tabulatoren getrennte Zeile mit zwei festen Zellen, gefolgt von optionalen Parametern.

```
time  type  ...params...
```

Diese Information kann einfach ein ein anderes Programm wie etwa ein Python-Skript eingelsen werden und kann nützlich für Forscher sein, die musikalische Analyse- oder Wiedergabeexperimente mit LilyPond machen wollen.

Bekannte Probleme und Warnungen

Nicht alle musikalischen Ereignisse werden von `event-listener.ly` unterstützt. Es handelt sich eher um ein gut gemachtes „proof of concept“. Wenn Ereignisse, die Sie brauchen, nicht in enthalten sind, können Sie `event-listener.ly` in Ihr LilyPond-Verzeichnis kopieren und die Datei verändern, sodass sie die benötigte Information ausgibt.

Layout-Kontrolle

25 Seitenlayout

Das finale Layout der Seite wird von drei Faktoren bestimmt: dem Layout der Seite, den Zeilenumbrüchen und der Platzverteilung. Jeder Faktor beeinflusst auch die anderen mit. Die Wahl der Platzverteilung entscheidet, wie eng die Notensysteme gesetzt werden. Das wiederum hat Einfluss auf die gewählten Zeilenumbrüche und letztendlich also auch darauf, wieviele Seiten ein Stück beansprucht.

Die Verteilung der Musik auf der Seite geschieht grob gesagt in vier Schritten. Zuerst werden flexible Entfernungen („springs“) gewählt, die auf den Notendauern basieren. Alle möglichen Zeilenumbrüche werden getestet und ein „Schlechtigkeitsscore“ für die Umbrüche erstellt. Danach wird die mögliche Höhe eines Systems ermittelt und schließlich wird eine bestimmte Kombination aus Seiten- und Zeilenumbruch ausgewählt, sodass weder die horizontale noch die vertikale Platzverteilung zu eng oder zu weit gesetzt wird.

Einstellungen, die das Layout beeinflussen, können in zwei Umgebungen gesetzt werden: in der `\paper {...}`- und der `\layout {...}`-Umgebung. Die `\paper`-Umgebung enthält Einstellungen des Seitenlayouts, die für alle Partituren innerhalb eines `\book` die gleichen sein sollen, wie etwa Papierhöhe oder ob Seitenzahlen ausgegeben werden sollen. Siehe Kapitel 25 [Seitenlayout], Seite 521. Die `\layout`-Umbegung enthält Layouteinstellungen der Partitur selber, wie etwa die Zahl der Systeme oder den Platz zwischen Systemgruppen usw. Siehe Kapitel 26 [Partiturlayout], Seite 532.

25.1 Die \paper-Umgebung

Die `\paper`-Umgebung kann innerhalb einer `\book`-, nicht aber innerhalb einer `\score`-Umgebung vorkommen. Einstellungen in `\paper` wirken sich auf das gesamte Buch aus, welches viele einzelne Partituren beinhalten kann. Einstellungen, die in der `\paper`-Umgebung vorkommen können, beinhalten:

- die `set-paper-size`-Scheme-Funktion,
- `\paper`-Variablen, die zum Verändern des Seitenlayouts eingesetzt werden und
- Beschriftungsdefinitionen, mit denen das Layout von Kopf- und Fußleisten sowie Titeln beeinflusst wird.

Die `set-paper-size`-Funktion wird im nächsten Abschnitt behandelt: Abschnitt 25.2 [Papierformat und automatische Skalierung], Seite 522. Die `\paper`-Variablen, die das Seitenlayout beeinflussen, werden in späteren Abschnitten behandelt. Die Beschriftungsdefinitionen für Kopf- und Fußzeilen sowie Titeln werden behandelt in Abschnitt 20.2 [Eigene Kopf- und Fußzeilen sowie Titel], Seite 470.

Die meisten `\paper`-Variablen funktionieren nur innerhalb der `\paper`-Umgebung. Die wenigen, die auch in der `\layout`-Umgebung funktionieren, finden sich in Abschnitt 26.1 [Die \layout-Umgebung], Seite 532.

Außer wenn anders angegeben, werden alle `\paper`-Variablen, die Abständen auf der Seite antsprechen, in Millimetern gemessen, es sei denn, eine andere Maßeinheit ist definiert. Beispielsweise wird mit folgender Definition der obere Rand (top-margin) 10 mm breit definiert:

```
\paper {
  top-margin = 10
}
```

Damit etwa `0.5` Zoll benutzt werden, muss `\in` dem Maß nachgestellt werden:

```
\paper {
  top-margin = 0.5\in
}
```

Mögliche Maßeinheiten sind \mm, \cm, \in und \pt. Diese Maßeinheiten sind einfach Werte, um von Millimetern zu Konvertieren, sie sind in ly/paper-defaults-init.ly definiert. Um Missverständnisse zu vermeiden, wird normalerweise \mm geschrieben, auch wenn es eigentlich nicht notwendig wäre.

Man kann die \paper-Werte auch mit Scheme definieren. Die Scheme-Entsprechung der obigen Definition ist:

```
\paper {
  #(define top-margin (* 0.5 in))
}
```

Siehe auch

Notationsreferenz Abschnitt 25.2 [Papierformat und automatische Skalierung], Seite 522, Abschnitt 20.2 [Eigene Kopf- und Fußzeilen sowie Titel], Seite 470, Abschnitt 26.1 [Die \layout-Umgebung], Seite 532.

Installierte Dateien: ly/paper-defaults-init.ly.

25.2 Papierformat und automatische Skalierung

25.2.1 Das Papierformat einstellen

‚A4‘ ist der Standardwert, wenn keine ausdrückliches Papierformat eingestellt ist. Es gibt jedoch zwei Funktionen, mit denen man das Papierformat ändern kann: set-default-paper-size

```
#(set-default-paper-size "quarto")
```

welcher immer auf oberster Ebene der Datei geschrieben werden muss, und set-paper-size

```
\paper {
  #(set-paper-size "tabloid")
}
```

welcher in eine \paper-Umgebung geschrieben werden muss.

Wenn die set-default-paper-size-Funktion auf oberster Ebene der Datei eingesetzt wird, muss sie vor allen \paper-Umgebungen kommen. set-default-paper-size definiert das Papierformat für alle Seiten, während set-paper-size nur das Format der Seiten bestimmt, auf die sich die \paper-Umgebung bezieht. Wenn beispielsweise die \paper-Umgebung oben in der Datei steht, dann bezieht sie sich auf alle Seiten in der Datei. Wenn die \paper-Umgebung innerhalb einer \book-Umgebung steht, dann bezieht sie sich nur auf das eine Buch.

Wenn die set-paper-size-Funktion eingesetzt wird, muss sie *vor* allen anderen Funktionen stehen, die in derselben \paper-Umgebung benützt werden. Siehe Abschnitt 25.2.2 [Automatische Skalierung auf ein Papierformat], Seite 523.

Die Papierformate finden sich in der Datei scm/paper.scm definiert. Hierhin kann man eigene Formate definieren, sie werden jedoch bei einer Aktualisierung von LilyPond überschrieben. Die vorhandenen Papierformate finden sich in Abschnitt A.5 [Vordefinierte Papierformate], Seite 652.

Der folgende Befehl kann benützt werden, um ein eigenes Papierformat hinzuzufügen, welches dann mit set-default-paper-size oder set-paper-size benützt werden kann:

```
#(set! paper-alist (cons '("mein Format" . (cons (* 15 in) (* 3 in))) paper-alist))

\paper {
  #(set-paper-size "mein Format")
}
```

Die Einheiten in (Fuß), cm (Centimeter) and mm (Millimeter) können benützt werden.

Wenn das Symbol `'landscape` an die Funktion `set-default-paper-size` gehängt wird, werden die Seiten um 90° gedreht und die Notensysteme entsprechend breiter gesetzt.

```
#(set-default-paper-size "a6" 'landscape)
```

Die Notenausgabe wird *nicht* gedreht, nur das Papierformat.

Siehe auch

Notationsreferenz: Abschnitt 25.2.2 [Automatische Skalierung auf ein Papierformat], Seite 523, Abschnitt A.5 [Vordefinierte Papierformate], Seite 652.

Installierte Dateien: `scm/paper.scm`.

25.2.2 Automatische Skalierung auf ein Papierformat

Wenn das Papierformat mit einer der Scheme-Funktionen (`set-default-paper-size` oder `set-paper-size`) geändert wird, werden die Werte einiger \paper-Variablen automatisch an die neue Größe angepasst. Um die automatische Skalierung für eine bestimmte Variable zu umgehen, kann die Variable definiert werden, nachdem man das Papierformat angegeben hat. Es sollte beachtet werden, dass die automatische Anpassung nicht ausgelöst wird, wenn man nur die `paper-height` oder `paper-width`-Variablen verändert, obwohl `paper-width` andere Werte beeinflussen kann (das muss von der automatischen Skalierung unterschieden werden und wird unten behandelt). Die Funktionen `set-default-paper-size` und `set-paper-size` werden behandelt in Abschnitt 25.2.1 [Das Papierformat einstellen], Seite 522.

Die vertikalen Dimensionen, die durch die automatische Skalierung verändert werden sind: `top-margin` und `bottom-margin` (siehe Abschnitt 25.3 [Vertikale \paper-Variablen mit festen Abständen], Seite 523). Die horizontalen Dimensionen, die durch die automatische Skalierung verändert werden, sind `left-margin`, `right-margin`, `inner-margin`, `outer-margin`, `binding-offset`, `indent` und `short-indent` (siehe Abschnitt 25.5 [\paper-Variablen für horizontale Abstände], Seite 526).

Die Standardwerte für diese Dimensionen sind in `ly/paper-defaults-init.ly` definiert, wobei interne Variablen mit den Bezeichnungen `top-margin-default`, `bottom-margin-default`, usw. benutzt werden. Das sind die Werte für die Standardpapiergröße `a4`. Zum Vergleich: `a4` hat Werte von `297\mm` für `paper-height` und `210\mm` für `paper-width`.

Siehe auch

Notationsreferenz: Abschnitt 25.3 [Vertikale \paper-Variablen mit festen Abständen], Seite 523, Abschnitt 25.5 [\paper-Variablen für horizontale Abstände], Seite 526.

Installierte Dateien: `ly/paper-defaults-init.ly`, `scm/paper.scm`.

25.3 Vertikale \paper-Variablen mit festen Abständen

> **Achtung:** Einige \paper-Dimensionen werden automatisch nach Papierformat skaliert, was zu ungewolltem Verhalten führen kann. Siehe Abschnitt 25.2.2 [Automatische Skalierung auf ein Papierformat], Seite 523.

Standardwerte (vor der Skalierung) sind definiert in `ly/paper-defaults-init.ly`.

`paper-height`
: Die Höhe der Seite, standardmäßig nicht definiert. Die automatische Skalierung einiger vertikalen Dimensionen wird hiervon nicht betroffen.

`top-margin`
: Der Rand zwischen dem oberen Ende der Seite und dem oberen Ende des bedruckbaren Bereichs. Wenn das Papierformat verändert wurde, wird der Standardwert dieser Dimension entsprechend skaliert.

`bottom-margin`
: Der Rand zwischen dem unteren Ende der Seite und dem unteren Ende des bedruckbaren Bereichs. Wenn das Papierformat verändert wurde, wird der Standardwert dieser Dimension entsprechend skaliert.

`ragged-bottom`
: Wenn auf wahr gesetzt, werden die Systeme nicht vertikal bis zum unteren Seitenrand verteilt. Sollte auf wahr gesetzt sein für Stücke, die nur ein bis zwei Notensystemgruppen pro Seite haben, etwa Orchesterpartituren.

`ragged-last-bottom`
: Wenn auf falsch gesetzt, werden die Systeme vertikal auf der letzten Seite verteilt. Bei Stücken, die grob zwei oder mehr Seiten füllen, sollten es auf falsch (`false`) gesetzt werden. Hiermit wird auch die letzte Seite von Teilen eines `\book`, die mit `\bookpart` erstellt sind, beeinflusst.

Siehe auch

Notationsreferenz: Abschnitt 25.2.2 [Automatische Skalierung auf ein Papierformat], Seite 523.

Installierte Dateien: `ly/paper-defaults-init.ly`.

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

Bekannte Probleme und Warnungen

Die Titel (aus der `\header`-Umgebung) werden als Systemgruppe behandelt, sodass `ragged-bottom` und `ragged-last-bottom` auch zusätzlichen Abstand zwischen den Titel und dem ersten System einer Paritur einfügt.

Explizit definierte Papierformate überschreiben alle vom Benutzer erstellte Randeinstellung für die Ränder oben und unten.

25.4 Vertikale \paper-Variablen mit flexiblen Abständen

In den meisten Fällen bietet es sich an, dass die vertikalen Abstände zwischen bestimmten Objekten (wie Ränder, Titel, Notensystemgruppen und einzelne Partituren) flexibel gehalten werden, sodass sie je nach Situation gedehnt oder komprimiert werden können. Es gibt eine Anzahl von Variablen für die `\paper`-Umgebung, mit denen man das Dehnungsverhalten dieser Dimensionen beeinflussen kann. Sie finden sich unten aufgelistet.

Dabei sollte beachtet werden, dass die Variablen, die in diesem Abschnitt behandelt werden, nicht die Platzierung und das Dehnungsverhalten von Notensystemen innerhalb der einzelnen Systemgruppen behandelt. Die Dehnung zwischen Notensystemen wird mit Grob-Eigenschaften kontrolliert, deren Einstellungen normalerweise innerhalb der `\score`-Umgebung vorgenommen werden, und nicht innerhalb der `\paper`-Umgebung. Siehe auch Abschnitt 28.1 [Flexible vertikale Abstände in Systemgruppen], Seite 545.

25.4.1 Struktur der Alisten für flexible vertikale Abstände

Jede der flexiblen vertikalen Abstandsvariablen ist eine Aliste (eine assoziative Liste), die vier *Schlüssel* (engl. key) enthält:

- `basic-distance` (Grund-Abstand) – der vertikale Abstand, gemessen in Systemzwischenräumen, zwischen den *Referenzpunkten* zweier Elemente, wenn keine Zusammenstöße

vorkommen würden und keine Dehnung oder Kompression stattfindet. Der Referenzpunkt einer (Titel-)Beschriftung (auf höchster Ebene) ist sein höchster Punkt und der Referenzpunkt einer Systemgruppe ist der vertikale Mittelpunkt des nächsten `StaffSymbol` – sogar wenn eine Nicht-Notensystemzeile (wie etwa ein `Lyrics`-Kontext) dazwischen kommt. Werte für `basic-distance`, die weniger als entweder `padding` oder `minimum-distance` sind, haben keine Bedeutung, weil der sich daraus ergebende Abstand niemals weniger als entweder `padding` oder `minimum-distance` ergibt.

- `minimum-distance` (minimaler Abstand) – der kleinste erlaubte vertikale Abstand, gemessen in Systemzwischenräumen, zwischen den Referenzpunkten der zwei Elemente, wenn Kompression stattfindet. Werte für `minimum-distance`, die geringer als `padding` sind, haben keine Bedeutung, weil der sich daraus ergebende Abstand niemals weniger als `padding` ergibt.
- `padding` (Verschiebung) – der minimal benötigte vertikale blanke Freiraum zwischen den Bounding-Boxen (oder Skyline) der zwei Objekten, gemessen in Notenlinienabständen.
- `stretchability` (Dehnbarkeit) – ein einheitsloses Maß der Leichtigkeit, mit der sich die Dimension dehnen lässt (ohne dass Zusammenstöße auftreten). Wenn es null ist, wird der Abstand nicht gedehnt (außer ein Zusammenstoß würde auftreten), wenn es positiv ist, hängt die Wichtigkeit der Dehnbarkeit eines bestimmten Objekts nur noch von seiner Beziehung zu dem Wert des anderen Objekts ab. Beispielsweise wennn eine Dimension die doppelte Dehnbarkeit als die andere hat, wird sie auch zweimal so einfach gedehnt. Werte sollten nicht-negativ und reale Zahlen sein. Der Wert `+inf.0` ruft einen `programming_error` hervor und wird ignoriert, aber `1.0e7` kann für einen so gut wie unendlich dehnbaren Abstand eingesetzt werden. Wenn der Wert nicht gesetzt wird, ist der Standardwert der von `basic-distance`. Die Wahrscheinlichkeit einer Dimension, sich zu verkleinern, kann man nicht direkt beeinflussen, sondern sie ergibt sich aus (`space` − `minimum-distance`).

Wenn eine Seite einen nicht ausgeglichenen unteren Rand hat, ist der resultierende Abstand der größte von:

- `basic-distance`,
- `minimum-distance` und
- `padding` plus der kleinste nötige Abstand, um Zusammenstöße zu vermeiden.

Bei Partituren über mehrere Seiten mit nicht ausgeglichenem unteren Rand greift die letzte Seite auf die gleiche Positionierung zurück wie die vorhergehende Seite, vorausgesetzt, dafür ist genügend Platz vorhanden.

Spezifische Methoden, um Alisten zu verändern, werden behandelt in Abschnitt 34.6 [Alisten verändern], Seite 607. Das folgende Beispiel demonstriert beide Arten, wie diese Alisten verändert werden können. Der erste Aufruf verändert nur einen Schlüsselwert einzeln, während der zweite die Variable vollständig neu definiert:

```
\paper {
  system-system-spacing.basic-distance = #8
  score-system-spacing =
    #'((basic-distance . 12)
       (minimum-distance . 6)
       (padding . 1)
       (stretchability . 12))
}
```

25.4.2 Liste der flexiblen vertikalen Abstandsvariablen in \paper

Die Bezeichnungen dieser Variablen entsprechen dem Format *obere*-*untere*-`platzierung`, wobei *obere* und *untere* die zu platzierenden Elemente darstellen. Jeder Abstand wird zwischen

den Referenzpunkten der beiden Elemente gemessen (siehe Beschreibung der Alistenstruktur oben). In diesen Variablenbezeichnungen bedeutet ‚markup‘ (Beschriftung) sowohl *Titelbeschriftungen* (bookTitleMarkup oder scoreTitleMarkup) als auch *Beschriftungen auf höchster Ebene* (siehe Abschnitt 19.5 [Die Dateistruktur], Seite 461). Alle Entfernungen werden in Systemzwischenräumen gemessen.

Standardwerte sind in ly/paper-defaults-init.ly definiert.

markup-system-spacing
: der Abstand zwischen einer (Titel-)Beschriftung (auf höchster Ebene) und der darauf folgenden Systemgruppe.

score-markup-spacing
: der Abstand zwischen dem letzten System einer Partitur und der darauf folgenden (Titel-)Beschriftung (auf höchster Ebene).

score-system-spacing
: der Abstand zwischen dem letzten System einer Partitur und dem ersten System der folgenden Partitur, wenn keine (Titel-)Beschriftung (auf höchster Ebene) dazwischen vorkommt.

system-system-spacing
: der Abstand zwischen zwei Systemgruppen der selben Partitur.

markup-markup-spacing
: der Abstand zwischen zwei (Titel-)Beschriftungen (auf höchster Ebene).

last-bottom-spacing
: der Abstand vom letzten System oder Beschriftung auf höchster Ebene auf einer Seite zum unteren Rand des bedruckbaren Bereichs (also bis zum Anfang des unteren Randes).

top-system-spacing
: der Abstand zwischen dem oberen Rand des bedruckbaren Bereichs (also dem Ende des oberen Rands) und dem ersten System auf der Seite, wenn keine (Titel-)Beschriftung (auf höchster Ebene) dazwischen kommt.

top-markup-spacing
: der Abstand vom oberen Rand des bedruckbaren Bereichs (also dem Ende des oberen Randes) zur ersten (Titel-)Beschriftung (auf höchster Ebene) auf einer Seite, wenn keine Systemgruppe dazwischen kommt.

Siehe auch

Notationsreferenz: Abschnitt 28.1 [Flexible vertikale Abstände in Systemgruppen], Seite 545.

Installierte Dateien: ly/paper-defaults-init.ly.

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

25.5 \paper-Variablen für horizontale Abstände

Achtung: Einige \paper-Dimensionen werden automatische entsprechend dem Papierformat skaliert und können deshalb ungewollte Resultate haben. Siehe Abschnitt 25.2.2 [Automatische Skalierung auf ein Papierformat], Seite 523.

25.5.1 \paper-Variablen für Breite und Ränder

Standardwerte (vor der Skalierung), die hier nicht aufgelistet sind, finden sich in `ly/paper-defaults-init.ly`.

`paper-width`
: Die Breite der Seite, standardmäßig nicht definiert. Während `paper-width` keine Auswirkungen auf die automatische Skalierung einiger horizontaler Dimensionen hat, beeinflusst es dennoch die `line-width`-Variable. Wenn sowohl `paper-width` als auch `line-width` definiert sind, dann werden auch `left-margin` und `right-margin` aktualisiert. Siehe auch `check-consistency`.

`line-width`
: Die horizontale Ausdehnung der Notenlinien in nicht eingerückten Systemen mit Ausgleich zum rechten Rand entspricht `(paper-width − left-margin − right-margin)`, wenn nicht definiert. Wenn `line-width` definiert ist und sowohl `left-margin` als auch `right-margin` nicht definiert sind, dann werden die Ränder aktualisiert, sodass die Systeme mittig auf der Seite zentriert werden. Siehe auch `check-consistency`. Diese Variable kann auch in der `\layout`-Umgebung definiert werden.

`left-margin`
: Der Rand zwischen der linken Papierkante und dem Beginn der Systeme ohne Einrückungen. Wenn das Papierformat verändert wird, wird auch der Standardwert dieser Dimension entsprechend skaliert. Wenn `left-margin` nicht definiert ist und sowohl `line-width` als auch `right-margin` definiert sind, dann wird `left-margin` auf den Wert `(paper-width − line-width − right-margin)` gesetzt. Wenn nur `line-width` definiert ist, dann werden beide Ränder auf den Wert `((paper-width − line-width) / 2)` gesetzt und die Systeme demzufolge auf der Seite zentriert. Siehe auch `check-consistency`.

`right-margin`
: Der Rand zwischen der rechten Papierkante und dem Ende der Systeme mit Randausgleich („Blocksatz“). Wenn das Papierformat geändert wird, wird auch der Standardwert dieser Dimension entsprechend skaliert. Wenn `right-margin` nicht definiert ist und sowohl `line-width` als auch `left-margin` definiert sind, dann wird `right-margin` auf den Wert `(paper-width − line-width − left-margin)` gesetzt. Wenn nur `line-width` definiert ist, dann werden beide Ränder auf den Wert `((paper-width − line-width) / 2)` gesetzt und die Systeme demzufolge auf der Seite zentriert. Siehe auch `check-consistency`.

`check-consistency`
: Wenn wahr, wird eine Warnung ausgegeben, sollten `left-margin`, `line-width` und `right-margin` zusammen nicht exakt den Wert von `paper-width` ergeben, und die Werte (außer `paper-width`) mit ihren Standardwerten belegt (wenn nötig auf das entsprechende Papierformat skaliert). Wenn falsch werden derartige Inkonsistenzen ignoriert und die Systeme dürfen auch über den Seitenrand hinausragen.

`ragged-right`
: Wenn wahr, werden Notensysteme nicht über die gesamte Zeilenbreite gestreckt, sondern sie enden horizontal entsprechend den enthaltenen Noten. Standard: `#t` (wahr) für Partituren mit einem System und `#f` (falsch) für Partituren mit zwei oder mehr Systemen. Diese Variable kann auch in der `\layout`-Umgebung definiert werden.

`ragged-last`
: Wenn wahr, wird das letzte Notensystem einer Partitur nicht über die gesamte Zeilenbreite gestreckt, sondern es endet horizontal entsprechend den enthaltenen Noten. Standard: `#f` (falsch). Diese Variable kann auch in der `\layout`-Umgebung definiert werden.

Siehe auch

Notationsreferenz: Abschnitt 25.2.2 [Automatische Skalierung auf ein Papierformat], Seite 523.

Installierte Dateien: `ly/paper-defaults-init.ly`.

Bekannte Probleme und Warnungen

Explizit definierte Papierformate überschreiben alle vom Benutzer erstellte Randeinstellung für die Ränder oben und unten.

25.5.2 \paper-Variablen für zweiseitigen Satz

Standardwerte (vor der Skalierung) sind definiert in `ly/paper-defaults-init.ly`.

`two-sided`
: Wenn auf wahr (`##t`) gesetzt, werden `inner-margin`, `outer-margin` und `binding-offset` zusammen benutzt, um die Ränder der Seite in Abhängigkeit von einer geraden oder ungeraden Seitennummer zu errechnen. Damit werden die Werte von `left-margin` und `right-margin` überschrieben. Standard: `##f`.

`inner-margin`
: Der Rand, den alle Seiten auf der Innenseite haben, wenn sie Teil eines Buches (`\book`) sind. Wenn das Papierformat verändert wird, wird der Standardwert dieser Dimension entsprechend skaliert. Funktioniert nur, wenn `two-sided` wahr ist.

`outer-margin`
: Der Rand, den alle Seiten auf der Außenseite haben, wenn sie Teil eines Buches sind. Wenn das Papierformat verändert wird, wird der Standardwert dieser Dimension entsprechend skaliert. Funktioniert nur, wenn `two-sided` wahr ist.

`binding-offset`
: Der Wert, um welchen `inner-margin` erhöht wird, um sicherzugehen, dass nichts in der Bindung verschwindet. Wenn das Papierformat verändert wird, wird der Standardwert dieser Dimension entsprechend skaliert. Funktioniert nur, wenn `two-sided` wahr ist.

Siehe auch

Notationsreferenz: Abschnitt 25.2.2 [Automatische Skalierung auf ein Papierformat], Seite 523.

Installierte Dateien: `ly/paper-defaults-init.ly`.

25.5.3 \paper-Variablen für Verschiebungen und Einrückungen

Standardwerte (vor der Skalierung), die hier nicht aufgeführt sind, sind definiert in `ly/paper-defaults-init.ly`.

`horizontal-shift`
: Der Wert, um den alle Systeme (und auch Überschriften und Systemtrenner) nach rechts verschoben werden. Standard: `0.0\mm`.

`indent`
: Der Einzug für das erste System einer Partitur. Wenn das Papierformat verändert wird, wird der Standardwert dieser Dimension entsprechend skaliert. Diese Variable kann auch in der `\layout`-Umgebung definiert werden.

`short-indent`
: Der Einzug für alle Systeme einer Partitur ausschließlich das erste System. Wenn das Papierformat verändert wird, wird der Standardwert dieser Dimension entsprechend skaliert. Diese Variable kann auch in der `\layout`-Umgebung definiert werden.

Siehe auch

Notationsreferenz: Abschnitt 25.2.2 [Automatische Skalierung auf ein Papierformat], Seite 523.

Installierte Dateien: `ly/paper-defaults-init.ly`.

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

25.6 Andere \paper-Variablen

25.6.1 \paper-Variablen für den Zeilenumbruch

`max-systems-per-page`
: Die maximale Anzahl an Notensystemgruppen, die auf einer Seite gesetzt werden. Das wird zur Zeit nur von dem `ly:optimal-breaking`-Algorithmus unterstützt. Standard: nicht gesetzt.

`min-systems-per-page`
: Die minimale Anzahl an Notensystemgruppen, die auf einer Seite gesetzt werden. Das kann dazu führen, dass Seiten zu dicht gefüllt werden, wenn der Wert zu groß gewählt wird. Die Option ist zur Zeit nur von dem `ly:optimal-breaking`-Algorithmus unterstützt. Standard: nicht gesetzt.

`systems-per-page`
: Die Anzahl an Systemen, die auf jede Seite gesetzt werden sollen. Diese Option wird zur Zeit nur von dem `ly:optimal-breaking`-Algorithmus unterstützt. Standard: nicht gesetzt.

`system-count`
: Die Anzahl der Systeme, auf denen eine Partitur gesetzt werden soll. Standard: nicht gesetzt. Diese Variablen kann auch in der `\layout`-Umgebung definiert werden.

Siehe auch

Notationsreferenz: Abschnitt 27.1 [Zeilenumbrüche], Seite 536.

25.6.2 \paper-Variablen für den Seitenumbruch

Standardwerte, die hier nicht aufgelistet sind, finden sich in `ly/paper-defaults-init.ly`

`blank-after-score-page-penalty`
: Die Strafpunke, die erteilt werden, wenn eine leere Seite nach einer Partitur und vor der nächsten vorkommt. Der Standardwert hiervon ist kleiner als `blank-page-penalty`, sodass leere Seiten nach einer Partitur leeren Seiten innerhalb einer Partitur vorgezogen werden.

`blank-last-page-penalty`
: Die Strafpunkte, wenn eine Partitur auf einer ungeraden Seite beendet wird. Standard: `0`.

`blank-page-penalty`
: Die Strafpunkte, wenn eine leere Seite mitten in einer Partitur auftritt. Das wird nicht benutzt von `ly:optimal-breaking`, weil hiermit niemals leere Seiten mitten in einer Partitur zugelassen werden.

`page-breaking`
: Der Algorithmus, der für Seitenumbrüche eingesetzt wird. Mögliche Algorithmen sind: `ly:minimal-breaking` (minimale Umbrüche), `ly:page-turn-breaking` (Umbrüche an guten Stellen zum Umblättern) und `ly:optimal-breaking`.

`page-breaking-system-system-spacing`
: Überlistet die Seitenumbruchfunktion, indem ihr ein anderer Wert für `system-system-spacing` mitgeteilt wird, als in Wirklichkeit eingestellt ist. Wenn beispielsweise `page-breaking-system-system-spacing #'padding` auf einen deutlich größeren Wert als `system-system-spacing #'padding` gesetzt wird, setzt die Seitenumbruchsfunktion weniger Systeme auf eine Seite. Standard: nicht gesetzt.

`page-count`
: Die Zahl der Seiten, die für eine Partitur benutzt werden sollen. Standard: nicht gesetzt.

Siehe auch

Notationsreferenz: Abschnitt 27.2 [Seitenumbrüche], Seite 538, Abschnitt 27.3 [Optimale Seitenumbrüche], Seite 539, Abschnitt 27.4 [Optimale Umbrüche zum Blättern], Seite 539, Abschnitt 27.5 [Minimale Seitenumbrüche], Seite 540, Abschnitt 27.6 [Eine-Seite-Seitenumbrüche], Seite 540.

Installierte Dateien: `ly/paper-defaults-init.ly`.

25.6.3 \paper-Variablen für Seitenzahlen

Standardwerte, die hier nicht aufgelistet sind, finden sich in `ly/paper-defaults-init.ly`

`auto-first-page-number`
: Der Seitenumbruchsalgorithmus wird davon beeinflusst, ob die erste Seitenzahl gerade oder ungerade ist. Wenn die Variable auf wahr gesetzt wird, entscheidet der Seitenumbruchsalgorithmus selber, ob die Noten auf einer geraden oder ungeraden Seite beginnen sollen. Das hat dann zur Folge, dass die erste Seite entweder bleibt wie sie ist oder um eins erhöht wird. Standard: `#f`.

`first-page-number`
: Der Wert der Seitenzahl auf der ersten Seite.

`print-first-page-number`
: Wenn wahr, wird auch auf der ersten Seite die Seitenzahl ausgegeben. Standard: `#f`.

`print-page-number`
: Wenn falsch, werden Seitenzahlen nicht ausgegeben.

Siehe auch

Installierte Dateien: `ly/paper-defaults-init.ly`.

Bekannte Probleme und Warnungen

Ungrade Seitenzahlen befinden sich immer auf der rechten Seite. Wenn Sie die Noten auf Seite 1 beginnen lassen wollen, müssen Sie eine leere Seite nach dem Deckblatt einfügen, damit die Noten auf der rechten Seite mit Seite 1 beginnen.

25.6.4 Verschiedene \paper-Variablen

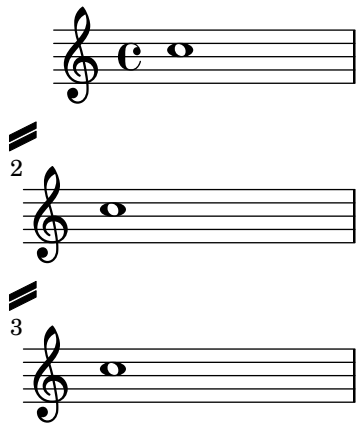
`page-spacing-weight`
: Die relative Gewichtung von (vertikalem) Abstand auf der Seite und (horizontalem) Abstand innerhalb der Zeilen. Hohe Werte gewichten die vertikalen Abstände mehr. Standard: `10`.

`print-all-headers`
: Wenn wahr, werden alle Einträge des Titelfeldes (`\header`-Umgebung) für jede Partitur (`\score`) ausgegeben. Normalerweise wird nur die Satzbezeichnung und die Opuszahl (`piece` und `opus`) ausgegeben. Standard: `#f`.

`system-separator-markup`
: Ein Beschriftungsobjekt, das zwischen zwei Systeme gesetzt wird. Das wird oft in Orchesterpartituren eingesetzt. Standard: nicht gesetzt. Der Beschriftungsbefehl `\slashSeparator`, definiert in `ly/titling-init.ly`, kann für einen Trenner benutzt werden, etwa so:

```
#(set-default-paper-size "a8")

\book {
  \paper {
    system-separator-markup = \slashSeparator
  }
  \header {
    tagline = ##f
  }
  \score {
    \relative { c''1 \break c1 \break c1 }
  }
}
```



Siehe auch

Installierte Dateien: `ly/titling-init.ly`.

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

Bekannte Probleme und Warnungen

Die Standard-Kopfzeilendefinition setzt die Seitenzahl und das `instrument`-Feld aus der `\header`-Umgebung in eine Zeile.

26 Partiturlayout

26.1 Die \layout-Umgebung

Während die \paper-Umgebung Einstellungen für die Formatierung der Seiten eines gesamten Dokuments enthalten, enthält die \layout-Umgebung Einstellungen für einzelne Partituren. Um Layoutoptionen für Partituren global einzustellen, müssen sie in einer \layout-Umgebung gesetzt werden, die sich auf höchster Ebene in der Datei befindet. Um sie für einzelne Partituren festzulegen, muss die \layout-Umgebung innerhalb der \score-Umgebung nach den Noten eingetraten werden. Einstellungen, die in einer \layout-Umgebung vorkommen können, beinhalten:

- die layout-set-staff-size-Scheme-Funktion,
- Kontextveränderungen in \context-Umgebungen und
- \paper-Variablen, die das Aussehen einer Partitur beeinflussen.

Die layout-set-staff-size-Funktion wird im nächsten Abschnitt behandelt, Abschnitt 26.2 [Die Notensystemgröße einstellen], Seite 534. Kontextveränderungen werden in einem eigenen Kapitel behandelt, siehe Abschnitt 32.4 [Umgebungs-Plugins verändern], Seite 586, and Abschnitt 32.5 [Die Standardeinstellungen von Kontexten ändern], Seite 588. Die \paper-Variablen, die innerhalb der \layout-Umgebungen erlaubt sind, sind:

- line-width, ragged-right und ragged-last (siehe Abschnitt 25.5.1 [\paper-Variablen für Breite und Ränder], Seite 527)
- indent und short-indent (siehe Abschnitt 25.5.3 [\paper-Variablen für Verschiebungen und Einrückungen], Seite 528)
- system-count (siehe Abschnitt 25.6.1 [\paper-Variablen für den Zeilenumbruch], Seite 529)

Hier ist ein Beispiel für eine \layout-Umgebung:

```
\layout {
  indent = 2\cm
  \context {
    \StaffGroup
    \override StaffGrouper.staff-staff-spacing.basic-distance = #8
  }
  \context {
    \Voice
    \override TextScript.padding = #1
    \override Glissando.thickness = #3
  }
}
```

Mehrfache \layout-Umbgebungen können als Ausdrücke auf höchster Ebene eingegeben werden. Das kann beispielsweise nützlich sein, wenn unterschiedliche Einstellungen in extra Dateien gespeichert werden und optional eingefügt werden. Intern wird eine Kopie der aktuellen \layout-Konfiguration gemacht, wenn eine \layout-Umgebung ausgewertet wird, dann erst werden Änderungen aus der Umgebung angewendet und das Ergebnis als die neue aktuelle Konfiguration gespeichert. Aus der Sicht des Benutzers werden die \layout-Umgebungen kombiniert, aber in Konfliktsituationen (wenn die gleiche Eigenschaft in unterschiedlichen Umgebungen geändert wird), erhält die spätere Definition den Vorrang.

Wenn also diese Umgebung:

```
\layout {
  \context {
```

```
    \Voice
    \override TextScript.color = #magenta
    \override Glissando.thickness = #1.5
  }
}
```

nach der Umgebung aus dem vorherigen Beispiel geschrieben wird, werden die `'padding`- und `'color`-Einstellungen für `TextScript` kombiniert, aber die spätere `'thickness`-Einstellung für `Glissando` ersetzt (oder versteckt) die vorherige.

`\layout`-Umgebungen können einer Variable zur späteren Benutzung zugewiesen werden, aber die Funktionsweise dieser Zuweisung ist geringfügig aber deutlich unterschiedlich von der Variante, sie auszuschreiben.

Wenn eine Variable etwa so definiert wird:

```
layoutVariable = \layout {
  \context {
    \Voice
    \override NoteHead.font-size = #4
  }
}
```

dann enthält sie die aktuelle `\layout`-Konfiguration mit zusätzlicher Veränderung von `NoteHead #'font-size` (der Schriftgröße der Notenköpfe), aber diese Kombination wird *nicht* als nächste aktuelle Konfiguration gespeichert. Man muss sich im klaren sein, dass die „aktuelle Konfiguration“ gelesen wird, wenn die Variable erstellt wird, nicht wenn sie benutzt wird. Darum ist der Inhalt der Variable abhängig von ihrer Position in der Partitur.

Die Variable kann dann auch innerhalb einer anderen `\layout`-Umgebung eingesetzt werden, etwa:

```
\layout {
  \layoutVariable
  \context {
    \Voice
    \override NoteHead.color = #red
  }
}
```

Eine `\layout`-Umgebung, die eine Variable enthält wie im Beispiel oben, kopiert die aktuelle Konfiguration *nicht*, sondern benützt den Inhalt von `\layoutVariable` als Basiskonfiguration für die weiteren Veränderungen. Das heißt, dass Änderungen, die zwischen der Definition der Variable und ihrer Benutzung definiert wurden, verloren gehen.

Wenn `layoutVariable` kurz vor ihrer Benutzung definiert wird (oder mit `\include` eingefügt wird), entspricht ihr Inhalt der aktuellen Konfiguration plus die Änderungen, die die Variable definiert. Das obige Beispiel, das den Einsatz von `\layoutVariable` zeigt, würde in seiner finalen Version folgende `\layout`-Umgebung haben:

```
TextScript.padding = #1
TextScript.color = #magenta
Glissando.thickness = #1.5
NoteHead.font-size = #4
NoteHead.color = #red
```

plus die Veränderungen an Einrückung (`indent`) und `StaffGrouper`.

Aber wenn die Variable vor der ersten `\layout`-Umgebung definiert wird, würde die aktuelle Konfiguration nur enthalten:

```
NoteHead.font-size= #4 % (written in the variable definition)
```

```
NoteHead.color = #red % (added after the use of the variable)
```

Wenn man sorgfältig plant, können \layout-Variablen ein wertvolles Instrument sein, um das Layout-Design von Quellen zu strukturieren und auch dazu dienen, die Layout-Einstellungen an einer bestimmten Stelle wieder zurückzusetzen.

Siehe auch

Notationsreferenz: Abschnitt 32.5 [Die Standardeinstellungen von Kontexten ändern], Seite 588, Abschnitt 26.2 [Die Notensystemgröße einstellen], Seite 534, Abschnitt 32.4 [Umgebungs-Plugins verändern], Seite 586.

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

26.2 Die Notensystemgröße einstellen

Die Standardgröße der Notensysteme beträgt 20 Punkte (pt). Das kann auf zwei Arten geändert werden:

Um die Systemgröße global für alle Partituren einer Datei (bzw. einer \book-Umgebung) zu verändern, wird set-global-staff-size benutzt:

```
#(set-global-staff-size 14)
```

Hiermit wird die Standardhöhe der Notensysteme auf 14 pt gesetzt. Die Schriftarten werden entsprechend verkleinert.

Um die Systemhöhe für jede Partitur einzeln zu verändern, muss

```
\score{
  ...
  \layout {
    #(layout-set-staff-size 15)
  }
}
```

eingesetzt werden.

Die Feta-Glyphe stellt die Noten- und Musiksymbole für acht verschiedene Größen zur Verfügung. Jede Schriftgröße ist einer bestimmten Systemgröße angepasst: für kleinere Schriftgrößen werden die Zeichen etwas schwerer, um mit den ebenfalls dickeren Notenlinien zu harmonieren. Die empfohlenen Notensystemgrößen sind in der Tabelle aufgeführt:

Schriftbezeichnung	Höhe des Systems (pt)	Höhe des Systems (mm)	Benutzung
feta11	11.22	3.9	Taschenparituren
feta13	12.60	4.4	
feta14	14.14	5.0	
feta16	15.87	5.6	
feta18	17.82	6.3	Liederbücher
feta20	20	7.0	Orchesterstimmen
feta23	22.45	7.9	
feta26	25.2	8.9	

Diese Schriftarten sind in allen Größen erhältlich. Die Kontext-Eigenschaft fontSize und die Layout-Eigenschaft staff-space (in StaffSymbol) können benutzt werden, um die Schriftgröße für einzelne Systeme zu verändern. Die Größe von einzelnen Systemen ist relativ zur globalen Systemgröße.

Siehe auch

Notationsreferenz: Abschnitt 7.1.1 [Auswahl der Notations-Schriftgröße], Seite 214.
Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

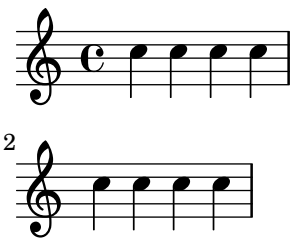
27 Umbrüche

27.1 Zeilenumbrüche

Zeilenumbrüche werden normalerweise automatisch erstellt. Sie werden so ausgewählt, dass die Zeilen weder gedrängt noch zu weit gespreizt wirken und aufeinander folgende Seiten einen ähnlichen Grauwert haben.

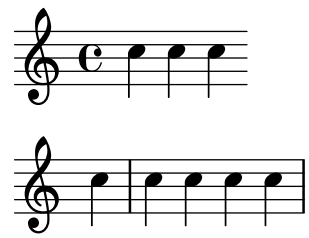
Einen manuellen Zeilenumbruch fügt man mit dem Befehl `\break` ein:

```
\relative c'' {
  c4 c c c | \break
  c4 c c c |
}
```
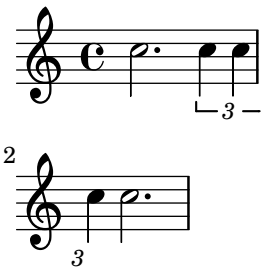


Normalerweise wird ein `\break` in der Mitte eines Takes ignoriert und eine Warnung ausgegeben. Um einen Zeilenumbruch in der Mitte eines Taktes zu erzwingen, können Sie mit `\bar "" ` eine unsichtbare Taktlinie hinzufügen, die dann den Zeilenumbruch erlaubt.

```
\relative c'' {
  c4 c c
  \bar "" \break
  c |
  c4 c c c |
}
```
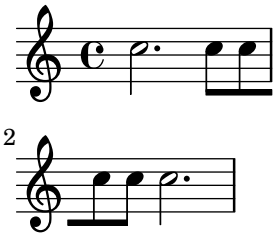


Ein `\break` an einem Taktstrich wird auch ignoriert, wenn der letzte Takt mitten in einer Note endet, wenn etwa eine N-tole in unterschiedlichen Takten beginnt und endet. Damit `\break` auch in derartigen Situationen funktioniert, muss `Forbid_line_break_engraver` aus der `Voice`-Umgebung entfernt werden. Dabei sollte beachtet werden, dass manuell hervorgerufene Umbrüche parallel mit den Noten hinzugefügt werden müssen.

```
\new Voice \with {
  \remove Forbid_line_break_engraver
} \relative {
  <<
    { c''2. \tuplet 3/2 { c4 c c } c2. | }
    { s1 | \break s1 | }
  >>
}
```



Genauso werden normalerweise Zeilenumbrüche auch verhindert, wenn Balken über die Taktenden hinausragen. Dieses Verhalten kann verändert werden, indem man `\override Beam.breakable = ##t` einstellt:

```
\relative c'' {
  \override Beam.breakable = ##t
  c2. c8[ c | \break
  c8 c] c2. |
}
```



Mit dem Befehl `\noBreak` wird ein Zeilenumbruch an dem entsprechenden Taktstrich verboten.

Die grundlegenden Einstellungen, die Einfluss auf die Zeilenlänge haben, sind `indent` (Einzug) und `line-width` (Zeilenbreite). Sie werden in der `\layout`-Umgebung eingestellt. Der erste Befehl bestimmt den Einzug der ersten Zeile, der zweite die Zeilenlänge der weiteren Notenzeilen.

Wenn `ragged-right` eingestellt ist (also in der `\layout`-Umgebung auf den Wert `#t` gesetzt wurde), werden die Systeme linksbündig gesetzt und nicht bis zum rechten Rand hin durchgezogen, sondern den Noten entsprechend gesetzt. Das ist oftmals nützlich für kleine Notenfragmente und um zu überprüfen, wie eng die Noten natürlicherweise gesetzt werden würden.

Die Option `ragged-last` verhält sich ähnlich zu `ragged-right`, aber wirkt sich nur auf die letzte Zeile eines Stückes aus.

```
\layout {
indent = #0
line-width = #150\mm
ragged-last = ##t
}
```

Um Zeilenumbrüche zu erzwingen, die in festgelegten Intervallen stattfinden, kann der Befehl `\break` in Kombination mit unsichtbaren Noten und einer Wiederholung (`\repeat`) eingesetzt werden. Das folgende Beispiel etwa setzt die nächsten 28 Takte (im 4/4-Takt) in Zeilen zu jeweils 4 Takten (die auch nur hier umgebrochen werden):

```
<<
  \repeat unfold 7 {
    s1 \noBreak s1 \noBreak
    s1 \noBreak s1 \break
  }
  { Hier die Noten... }
```

```
>>
```

Eine Zeilenumbruchkonfiguration kann auch als eine `.ly`-Datei automatisch gespeichert werden. Damit kann die vertikale Ausrichtung während eines zweiten Programmdurchlaufs angepasst werden um die Seiten besser zu füllen. Diese Eigenschaft ist recht neu und kompliziert. Mehr Einzelheiten finden sich in Abschnitt “Spacing” in *Schnipsel*.

Vordefinierte Befehle

`\break`, `\noBreak`.

Siehe auch

Notationsreferenz: Abschnitt 25.6.1 [\paper-Variablen für den Zeilenumbruch], Seite 529.

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

Referenz der Interna: Abschnitt “LineBreakEvent” in *Referenz der Interna*.

27.2 Seitenumbrüche

Die Standardseitenumbrüche können verändert werden, indem man die Befehle `\pageBreak` bzw. `\noPageBreak` benutzt. Sie verhalten sich analog zu den Befehlen `\break` und `\noBreak`. Sie sollten an einem Taktstrich notiert werden. Diese Befehle erzwingen bzw. verbieten einen Seitenumbruch. Mit dem `\pageBreak`-Befehl wird natürlich gleichzeitig auch ein Zeilenumbruch erzwungen.

Die `\pageBreak` und `\noPageBreak`-Befehle können auch auf der höchsten Ebene einer Datei benutzt werden, etwa zwischen Partituren und Textbeschriftungen.

Es gibt auch vertikale Gegenstücke zu den Variablen `ragged-right` und `ragged-last`: `ragged-bottom` und `ragged-last-bottom`. Wenn diese Variablen auf `#t` (wahr) gesetzt werden, werden im ersten Fall die Notensysteme auf allen Seiten eng nach oben orientiert gesetzt werden. Im zweiten Fall bezieht sich dies nur auf die letzte Seite. Zu Einzelheiten siehe Abschnitt 25.3 [Vertikale \paper-Variablen mit festen Abständen], Seite 523.

Seitenumbrüche werden von der `page-breaking`-Funktion errechnet. LilyPond kennt drei Algorithmen um Seitenumbrüche zu errechnen: `ly:optimal-breaking`, `ly:page-turn-breaking` und `ly:minimal-breaking`. Der Standard ist `ly:optimal-breaking`, aber der Wert kann in der `\paper`-Umgebung geändert werden:

```
\paper{
  #(define page-breaking ly:page-turn-breaking)
}
```

Wenn ein Buch (`\book`) viele Parituren und Seiten hat, kann die Seitenaufteilung schwer zu ermitteln sein und viel Zeit und Prozessorlast in Anspruch nehmen. Um den Seitenumbruchsprozess zu vereinfachen, werden `\bookpart`-Umgebungen benutzt, um das Buch in mehrere Teil zu trennen: Die Seitenumbrüche werden seperat für jeden Teil berechnet. Unterschiedliche Seitenumbruchsfunktionen können in unterschiedlichen Buchteilen benutzt werden.

```
\bookpart {
  \header {
    subtitle = "Vorwort"
  }
  \paper {
     %% In einem Abschnitt, der vor allem Text hat,
     %% funktioniert womöglich ly:minimal-breaking besser
     #(define page-breaking ly:minimal-breaking)
  }
  \markup { ... }
```

```
  ...
}
\bookpart {
  %% In diesem Abschnitt mit Noten wird
  %% die Standard-Seitenumbruchsfunktion benutzt.
  \header {
    subtitle = "Erster Satz"
  }
  \score { ... }
  ...
}
```

Vordefinierte Befehle

`\pageBreak`, `\noPageBreak`.

Siehe auch

Notationsreferenz: Abschnitt 25.6.2 [\paper-Variablen für den Seitenumbruch], Seite 529.

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

27.3 Optimale Seitenumbrüche

Die `ly:optimal-breaking`-Funktion ist die Standardmethode für LilyPond, um Seitenumbrüche zu errechnen. Hiermit wird versucht, Seitenumbrüche zu finden, die das Stauchen oder Strecken von Zeilen minimieren, sowohl horizontal als auch vertikal. Anders als die `ly:page-turn-breaking`-Funktion hat diese Methode keine Möglichkeit, Überlegungen zum Umblättern mit einzubeziehen.

Siehe auch

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

27.4 Optimale Umbrüche zum Blättern

Es ist oft nötig, die Seiten so umzubrechen, dass sich eine Pause am Ende jeder zweiten Seite befindet, damit der Musiker es leichter hat, die Seite umzublättern ohne das Spielen zu Unterbrechen. Die `ly:page-turn-breaking`-Funktion versucht, Seitenumbrüche zu finden, die das Stauchen oder Strecken von Zeilen minimieren und gleichzeitig auch noch Seitenumbrüchen an angegebenen Stellen den Vorrang zu geben.

Die Funktion wird in zwei Schritten eingesetzt. Zunächst muss sie in der `\paper`-Umgebung aktiviert werden, wie gezeigt in Abschnitt 27.2 [Seitenumbrüche], Seite 538. Dann muss noch angegeben werden, welche Stellen bevorzugt für Seitenumbrüche benutzt werden sollen.

Für diesen zweiten Schritt gibt es zwei Methoden. Am Einfachsten ist es, die möglichen Seitenumbrüche mit dem Befehl `\allowPageTurn` an jeder Stelle manuell anzugeben.

Wenn Ihnen das zu aufwändig ist, können Sie den `Page_turn_engraver` zu einem `Staff`- oder `Voice`-Kontext hinzufügen. Dieser Engraver durchsucht den entsprechenden Kontext nach Stellen ohne Noten. (Es wird also nicht nach Pausen gesucht, sondern nach Stellen ohne Noten. Dieses Verhalten verhindert, dass an polyphonen Stellen umgebrochen wird, wo nur in einer Stimme Pausen vorhanden sind.) Wenn eine derartige Stelle ohne Noten gefunden wird, fügt der Engraver den Befehl `\allowPageTurn` am letzten Taktstrich des Abschnitts ein. Wenn in dem Abschnitt ein besonderer Taktstrich vorkommt (wie etwa ein Doppelstrich), wird der Befehl nach diesem Taktstrich gesetzt.

Der `Page_turn_engraver` liest die Kontexteigenschaft `pageTurnMinimumRestLength` um zu erkennen, wie lang eine Stelle frei von Noten sein muss, damit ein Seitenumbruch in Frage kommt.

Der Standardwert hierfür ist 1. Wenn Sie Seitenumbrüche zum Umblättern ausschalten wollen, können Sie einen sehr großen Wert angeben.

```
\new Staff \with { \consists Page_turn_engraver }
{
  a4 b c d |
  R1 | % Ein Seitenumbruch zum Umblättern erlaubt
  a4 b c d |
  \set Staff.pageTurnMinimumRestLength = #5/2
  R1 | % Seitenumbruch nicht erlaubt
  a4 b r2 |
  R1*2 | % Seitenumbruch erlaubt
  a1
}
```

Der `Page_turn_engraver` erkennt Wiederholungen vom Typ `volta`. Ein Seitenumbruch zum Umblättern wird nur zugelassen, wenn vor und nach der Wiederholung genug Zeit ist, um die Seite wieder zurückzublättern. Wenn die Wiederholung sehr kurz ist, kann auch Umblättern verboten werden. Wenn Sie die Kontexteigenschaft `pageTurnMinimumRepeatLength` definieren, erlaubt der `Page_turn_engraver` nur Umblättern in Wiederholungen, deren Dauer länger als dieser Wert ist.

Die Seitenumblätter-Befehle `\pageTurn`, `\noPageTurn` und `\allowPageTurn` können auch auf oberster Dateiebene benutzt werden, etwa zwischen Partituren und Textabschnitten.

Vordefinierte Befehle

`\pageTurn`, `\noPageTurn`, `\allowPageTurn`.

Siehe auch

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

Bekannte Probleme und Warnungen

In einer Partitur sollte nur ein `Page_turn_engraver` vorkommen. Wenn mehr als einer definiert werden, stören sie sich gegenseitig.

27.5 Minimale Seitenumbrüche

Die `ly:minimal-breaking`-Funktion benötigt nur minimale Berechnungen, um die Seitenumbrüche zu bestimmen. Die Seite wird mit möglichst vielen Systemen gefüllt und dann zur nächsten Seite gewechselt. Die Funktion kann benutzt werden um Partituren mit vielen Seiten zu setzen, wenn die anderen Seitenumbruchsfunktionen zu langsam wären oder zu viel Speicher beanspruchen. Auch für Seiten mit viel Text ist die Funktion geeignet. Sie wird folgendermaßen aktiviert:

```
\paper {
  page-breaking = #ly:minimal-breaking
}
```

Siehe auch

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

27.6 Eine-Seite-Seitenumbrüche

Die Funktion `ly:one-line-breaking` ist ein besonderer Seitenumbruchalgorithmus, der jede Partitur (`score`) auf eine eigene Seite ausgibt, und in einer einzelnen Zeile. Diese Seitenumbruchfunktion gibt keine Titel oder Ränder aus, nur die Partitur wird dargestellt.

Die Seitenbreite wird angepasst, sodass die längeste Partitur auf eine Zeile passt. Die Variablen `paper-width`, `line-width` und `indent` in der \paper-Umgebung werden ignoriert, wenn auch `left-margin` und `right-margin` noch beachtet werden. Die Höhe der Seite wird nicht verändert.

27.7 Ausdrückliche Umbrüche

Es kann vorkommen, dass LilyPond direkte \break oder \pageBreak-Befehl nicht beachtet. Mit folgenden Einstellungen kann dieses Verhalten ausgeschaltet werden:

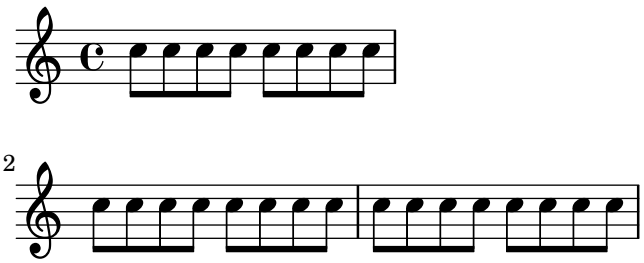
```
\override NonMusicalPaperColumn.line-break-permission = ##f
\override NonMusicalPaperColumn.page-break-permission = ##f
```

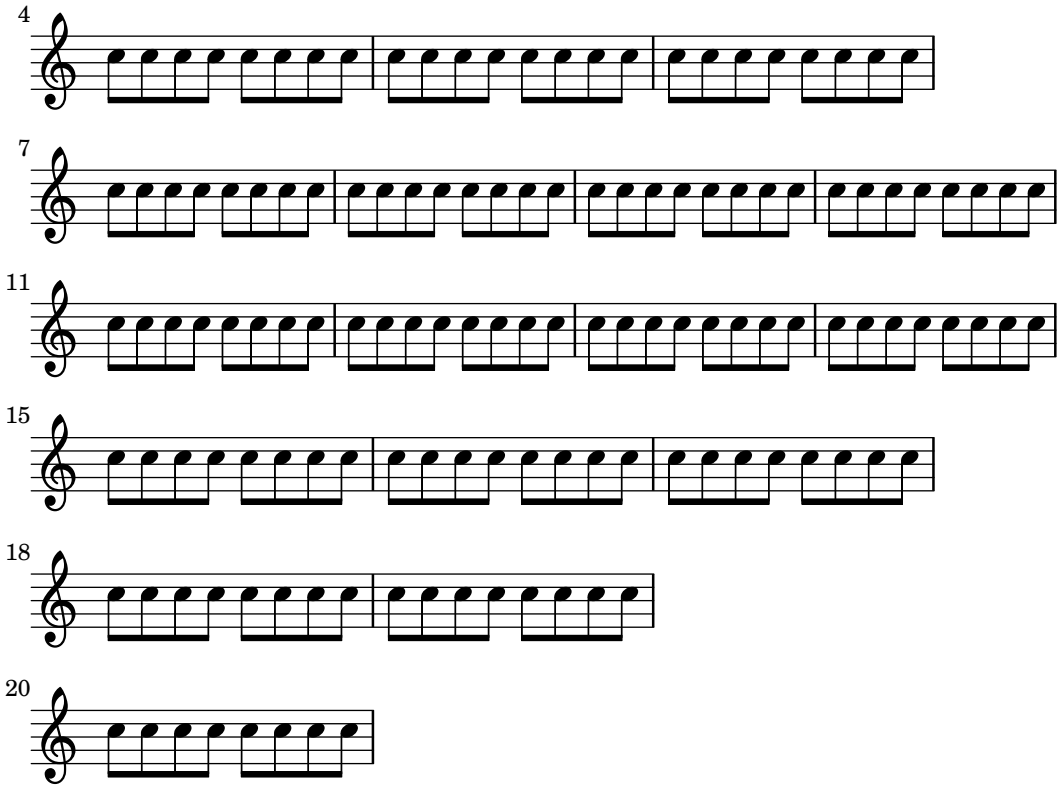
Wenn `line-break-permission` die Einstellung falsch (`##f`) hat, werden Zeilenumbrüche nur an den Befehlen \break eingefügt und nirgendwo anders. Wenn `page-break-permission` die Einstellung falsch (`##f`) hat, werden Seitenumbrüche nur an den Befehlen \pageBreak eingefügt und nirgendwo anders.

```
\paper {
  indent = #0
  ragged-right = ##t
  ragged-bottom = ##t
}

music = \relative { c''8 c c c }

\score {
  \new Staff {
    \repeat unfold 2 { \music } \break
    \repeat unfold 4 { \music } \break
    \repeat unfold 6 { \music } \break
    \repeat unfold 8 { \music } \pageBreak
    \repeat unfold 8 { \music } \break
    \repeat unfold 6 { \music } \break
    \repeat unfold 4 { \music } \break
    \repeat unfold 2 { \music }
  }
  \layout {
    \context {
      \Score
      \override NonMusicalPaperColumn.line-break-permission = ##f
      \override NonMusicalPaperColumn.page-break-permission = ##f
    }
  }
}
```





Siehe auch

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

27.8 Eine zusätzliche Stimme für Umbrüche benutzen

Zeilen- und Seitenumbruchbefehle werden normalerweise direkt zusammen mit den Noten eingegeben.

```
music = \relative { c''4 c c c }

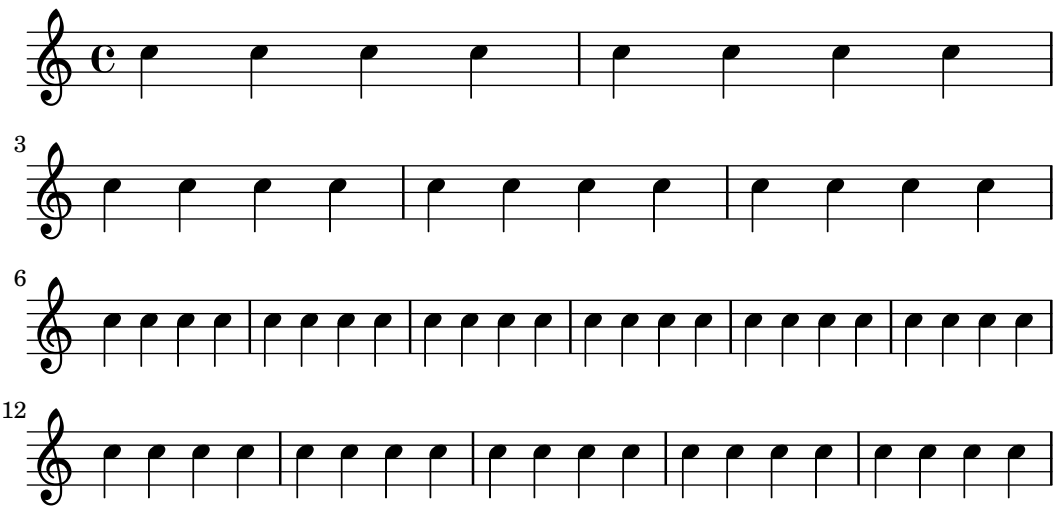
\score {
  \new Staff {
    \repeat unfold 2 { \music } \break
    \repeat unfold 3 { \music }
  }
}
```

Hierdurch sind zwar die Befehle `\break` und `\pageBreak` einfach zu notieren, es werden aber Informationen zur Notation mit Informationen zur Anordnung auf der Seite vermischt. Man kann diese Informationen auch voneinander trennen, indem man eine zusätzliche Stimme einfügt, in der Zeilen- und Seitenumbrüche vorgenommen werden. Diese zusätzliche Stimme enthält nur unsichtbare Noten und die Umbruchbefehle:

```
music = \relative { c''4 c c c }

\score {
  \new Staff <<
    \new Voice {
      s1 * 2 \break
      s1 * 3 \break
      s1 * 6 \break
```

```
      s1 * 5 \break
    }
    \new Voice {
      \repeat unfold 2 { \music }
      \repeat unfold 3 { \music }
      \repeat unfold 6 { \music }
      \repeat unfold 5 { \music }
    }
  >>
}
```



Mit dieser Herangehensweise kann der Code insbesondere dann klarer notiert werden, wenn man Einstellungen der `line-break-system-details`-Eigenschaft oder anderer Eigenschaften von `NonMusicalPaperColumnGrob` vornimmt (hierzu auch Kapitel 28 [Vertikale Abstände], Seite 545).

```
music = \relative { c''4 c c c }

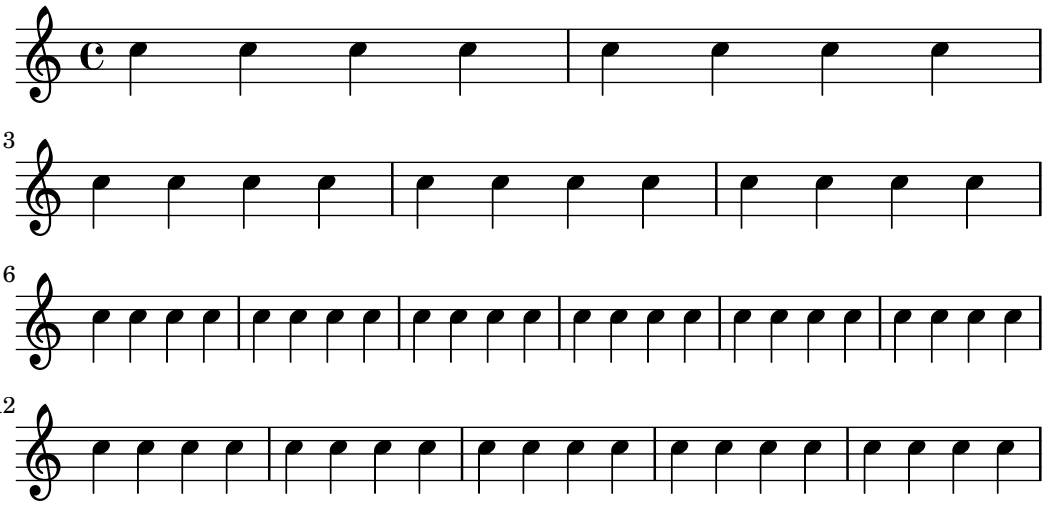
\score {
  \new Staff <<
    \new Voice {
      \overrideProperty Score.NonMusicalPaperColumn.line-break-system-details #'((Y-of
      s1 * 2 \break

      \overrideProperty Score.NonMusicalPaperColumn.line-break-system-details #'((Y-of
      s1 * 3 \break

      \overrideProperty Score.NonMusicalPaperColumn.line-break-system-details #'((Y-of
      s1 * 6 \break

      \overrideProperty Score.NonMusicalPaperColumn.line-break-system-details #'((Y-of
      s1 * 5 \break
    }
    \new Voice {
      \repeat unfold 2 { \music }
      \repeat unfold 3 { \music }
      \repeat unfold 6 { \music }
      \repeat unfold 5 { \music }
```

```
    }
  >>
}
```



Siehe auch

Notationsreferenz: Kapitel 28 [Vertikale Abstände], Seite 545.

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

28 Vertikale Abstände

Vertikale Abstände werden durch drei Eigenschaften bestimmt: wieviel Platz frei ist (etwa Papiergröße und Ränder), wieviel Platz zwischen Systemgruppen (engl. system) gesetzt werden soll und wieviel Platz zwischen Notensystemen (engl. staff, Pl. staves) innerhalb von Gruppen gesetzt wird.

28.1 Flexible vertikale Abstände in Systemgruppen

Drei unterschiedliche Mechanismen kontrollieren das flexible Abstandaufteilung in Systemgruppen, einer für jede der folgenden Kathegorien:

- *ungruppierte Systeme*,
- *Systemgruppen* (Systeme innerhalb einer `staff-group` wie etwa `ChoirStaff` usw.) und
- *Nicht-Notensystemzeilen* (wie etwa `Lyrics` (Gesangstext), `ChordNames` (Akkordbezeichnungen) usw.).

Die Höhe jeder Systemgruppe wird in zwei Schritten bestimmt. Zunächst werden alle Systeme anhand des vorhandenen Platzes aufgeteilt. Dann werden die nicht-Notensysteme (also Akkorde oder Gesangstext) zwischen den Systemen verteilt.

Es ist zu beachten, dass der Platzverteilungsmechanismus, der in diesem Abschnitt behandelt wird, nur die vertikale Platzierung von Systemen und nicht-Systemzeilen in einzelnen Systemgruppen behandelt. Die vertikale Platzierung zwischen einzelnen Systemgruppen, Partituren, Beschriftungen usw. und den Rändern wird durch \paper-Variablen kontrolliert, die Abschnitt 25.4 [Vertikale \paper-Variablen mit flexiblen Abständen], Seite 524.

28.1.1 Eigenschaften für Abstände innerhalb von Systemgruppen

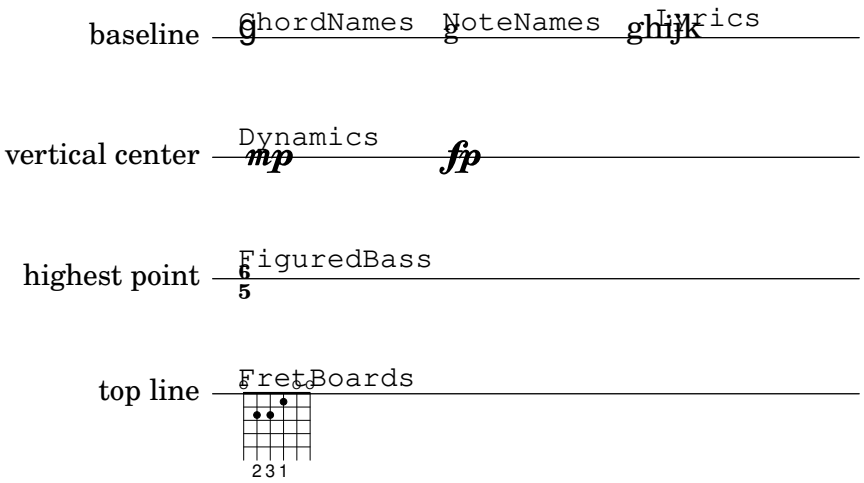
Der vertikalen Platzierungsmechanismen für Abstände innerhalb von Systemgruppen werden durch zwei Gruppen von Grob-Eigenschaften kontrolliert. Die erste Gruppe ist mit dem `VerticalAxisGroup`-Grob verknüpft, der von allen Notensystemen und Nicht-Notensystemzeilen erstellt wird. Die zweite Gruppe ist mit dem `StaffGrouper`-Grob verknüpft, der von Systemgruppen erstellt werden kann, aber nur, wenn das explizit verlangt wird. Die einzelnen Eigenschaften werden am Ende dieses Abschnitts beschrieben.

Die Bezeichnungen dieser Eigenschaften (mit Außnahmen von `staff-affinity`) haben das Format *`Element1-Element2`*`-spacing`, wobei *`Element1`* und *`Element2`* die Elemente sind, deren Abstände eingestellt werden sollen. Dabei ist allerdings zu beachten, dass *`Element2`* sich nicht notwendigerweise unterhalb von *`Element1`* befindet; beispielsweise `nonstaff-relatedstaff-spacing` (Nicht-Notensystem-verwandtesNotensystem) misst von dem Nicht-Notensystem nach oben, wenn `staff-affinity` (Richtung, an der sich ein System ausrichtet) auf `UP` (nach oben) eingestellt ist.

Jeder Abstand wird zwischen den *Referenzpunkten* der zwei Objekten gemessen. Der Referenzpunkt eines Notensystems ist die vertikale Mitte seines `StaffSymbol`-Objekts (also die Mittellinie, wenn `line-count` (Notenlinienzähler) ungrade ist, oder der mittlere Zwischenraum, wenn `line-count` grade ist). Die Referenzpunkte für einzelne Nicht-Notensystemzeilen ergibt sich aus der folgenden Tabelle:

Nicht-Notensystemzeile	**Referenzpunkt**
`ChordNames`	Grundlinie
`NoteNames`	Grundlinie
`Lyrics`	Grundlinie
`Dynamics`	vertikale Mitte
`FiguredBass`	höchster Punkt
`FretBoards`	Oberlinie

Im nächsten Bild zeigen horizontale Striche die Positionen dieser Referenzpunkte an:



Jeder der vertikalen Platzierungs-Grobeigenschaften (außer `staff-affinity`) benutzt die gleiche Alistenstruktur wie die \paper-Variablen, behandelt in Abschnitt 25.4 [Vertikale \paper-Variablen mit flexiblen Abständen], Seite 524. Besondere Methoden um Alisten zu verändern finden sich in Abschnitt 34.6 [Alisten verändern], Seite 607. Grob-Eigenschaften sollten mit dem \override-Befehle innerhalb einer \score- oder \layout-Umgebung angepasst werden, nicht innerhalb einer \paper-Umgebung.

Das folgende Beispiel zeigt die beiden Arten, Alisten zu modifizieren. Der erste Aufruf verändert nur einen Schlüsselwert einzlen, während der zweite die Eigenschaft komplett neu definiert:

```
\new Staff \with {
  \override VerticalAxisGroup.default-staff-staff-spacing.basic-distance = #10
} { ... }

\new Staff \with {
  \override VerticalAxisGroup.default-staff-staff-spacing =
    #'((basic-distance . 10)
       (minimum-distance . 9)
       (padding . 1)
       (stretchability . 10))
} { ... }
```

Um Platzierungseinstellungen global vorzunehmen, müssen sie in der \layout-Umgebung vorgenommen werden:

```
\layout {
  \context {
    \Staff
    \override VerticalAxisGroup.default-staff-staff-spacing.basic-distance = #10
  }
}
```

Standardeinstellungen für die vertikalen Platzierungs-Grobeigenscahften finden sich in Abschnitt “VerticalAxisGroup” in *Referenz der Interna* und Abschnitt “StaffGrouper” in *Referenz der Interna* aufgelistet. Standardveränderungen für bestimmte Typen von Nicht-Notensystemzeilen finden sich im relevanten Abschnitt in in Abschnitt “Contexts” in *Referenz der Interna* aufgelistet.

Eigenschaften des `VerticalAxisGroup`-Grobs

`VerticalAxisGroup`-Eigenschaften werden normalerweise mit einem `\override`-Befehl auf `Staff`-(Notensystem-)Ebene (oder entsprechend) vorgenommen.

`staff-staff-spacing`
: System-System-Platzierung

  Wird benutzt, um den Abstand zwischen dem aktuellen Notensystem und dem Notensystem direkt darunter in der gleichen Notensystemgruppe zu bestimmen, auch wenn eine oder mehrere Nicht-Notensystemzeilen (wie etwa `Lyrics`) dazwischen stehen. Gilt nicht für das unterste System einer Systemgruppe.

  Die Eigenschaft `staff-staff-spacing` einer `VerticalAxisGroup` ist eine Scheme-Funktion, welche die Eigenschaften vom `StaffGrouper`-Grob den Notensystemen zuweist, wenn sie einer Systemgruppe angehören, bzw. die `default-staff-staff-spacing`-Eigenschaft bei einem einzelnen Notensystem. Dadurch können Systeme unterschiedlich in Abhängigkeit von ihrer Zugehörigkeit zu einer Gruppe (StaffGroup) platziert werden. Wenn gleichmäßige Aufteilung ohne Berücksichtigung von Gruppierungen gewünscht ist, kann diese Funktion durch eine flexible Platzierungs-Aliste ersetzt werden, wobei man die vollständige Redefinition auf die oben gezeigte Weise vornehmen muss.

`default-staff-staff-spacing`
: Normale-System-System-Platzierung

  Eine flexible Platzierungs-Aliste, die den Wert von `staff-staff-spacing` für ungruppierte Notensysteme einstellt, es sei denn, `staff-staff-spacing` wirde explizit mit `\override` eingestellt.

`staff-affinity`
: System-Anziehung

  Die Richtung des Systems, die benutzt wird, um die aktuelle Nicht-Notensystemzeile zu platzieren. Mögliche Werte sind `UP` (nach oben), `DOWN` (nach unten) und `CENTER` (mittig). Wenn `CENTER` wird die Nicht-Notensystemzeile vertikal mittig zwischen den beiden nächsten Systemen oben und unten platziert, außer Zusammenstöße und andere Platzierungsprobleme verhindern das. Aufeinanderfolgende Nicht-Notensystemzeilen sollten nicht-aufsteigende `staff-affinity` von oben nach unten haben; also ein Nicht-Notensystemzeile mit `UP` sollte nicht direkt auf eine mit `DOWN` folgen. Nicht-Notensystemzeilen über einem Notensystem sollten `DOWN` benutzen, unter einem Notensystem dagegen `UP`. Wenn `staff-affinity` für eine Notensystem eingestellt wird, wird es wie eine Nicht-Notensystemzeile behandelt. Wenn `staff-affinity` auf `#f` gesetzt wird, wird eine Nicht-Notensystemzeile wie ein Notensystem behandelt. Wird `staff-affinity` auf `UP`, `CENTER` oder `DOWN` gesetzt, wird das Notensystem als Nicht-Notensystemzeile plaziert.

`nonstaff-relatedstaff-spacing`
: Nicht-Notensystem-verwandtesSystem-Platzierung

  Der Abstand zwischen der aktuellen Nicht-Notensystemzeile und dem nächsten Notensystem in der Richtung von `staff-affinity`, wenn keine Nicht-Notensystemzeilen dazwischen auftreten und `staff-affinity` entweder `UP` oder `DOWN` ist. Wenn `staff-affinity` `CENTER` ist, dann wird `nonstaff-relatedstaff-spacing` für die nächsten Notensysteme auf *beiden* Seiten benutzt, auch wenn andere Nicht-Notensystemzeilen zwischen der aktuellen und einem der Notensystem auftreten. Das heißt, dass die Platzierung eine Nicht-Notensystemzeile sowohl von den umliegenden Notensystemen als auch den umliegenden Nicht-Notensystemzeilen abhängt. Wenn `stretchability` einer dieser

Platzierungstypen auf einen kleinen Wert gesetzt wird, dominiert diese Platzierung. Wird es dagegen auf einen großen Wert gesetzt, hat die Platzierung dieses Objekts nur einen kleinen Einfluss.

`nonstaff-nonstaff-spacing`
: Nicht-Notensystemzeile-Nicht-Notensystemzeile-Platzierung

  Der Abstand zwischen der aktuellen Nicht-Notensystemzeile und der Nicht-Notensystemzeile in der Richtung von `staff-affinity`, wenn beide sich auf der gleichen Seite des verwandten Notensystems befinden und `staff-affinity` entweder `UP` oder `DOWN` ist.

`nonstaff-unrelatedstaff-spacing`
: Nicht-Notensystemzeile-Nicht-verwandtesSystem-Platzierung

  Der Abstand zwischen der aktuellen Nicht-Notensystemzeile und dem Notensystem in der gegenüberliegenden Richtung von `staff-affinity`, wenn keine anderen Nicht-Notensystemzeilen dazwischen auftreten und `staff-affinity` entweder `UP` oder `DOWN` ist. Das kann benutzt werden, um einen Minimalfüllabstand (padding) zwischen einer `Lyrics`-Gesangstextzeile und dem zugehörigen Notensystem zu verlangen.

Eigenschaften des `StaffGrouper`-Grobs

`StaffGrouper`-Eigenschaften werden normalerweise mit einem `\override`-Befehl auf `StaffGroup`-Ebene (oder entsprechend) eingestellt.

`staff-staff-spacing`
: Notensystem-Notensystem-Abstand

  Der Abstand zwischen zwei aufeinanderfolgenden Notensystemen in der aktuellen `StaffGroup`. Die `staff-staff-spacing`-Eigenschaft des `VerticalAxisGroup`-Grobs eines einzelnen Notensytems kann mit `\override` in andere Platzierungswerte für dieses Notensystem geändert werden.

`staffgroup-staff-spacing`
: Systemgruppe-System-Abstand

  Der Abstand zwischen dem letzten Notensystem der aktuellen `StaffGroup` und dem Notensystem dirket darunter in der selben Notensystemgruppe, auch wenn eine oder mehrere Nicht-Notensystemzeilen (wie etwa Gesangstext) zwischen den zwei Notensystemen vorkommen. Gilt nicht für das letzte Notensystem einer Systemgruppe. Die `staff-staff-spacing`-Eigenschaft des `VerticalAxisGroup`-Grobs individueller Notensysteme kann mit `\override` in andere Platzierungswerte für dieses Notensystem geändert werden.

Siehe auch

Notationsreferenz: Abschnitt 25.4 [Vertikale `\paper`-Variablen mit flexiblen Abständen], Seite 524, Abschnitt 34.6 [Alisten verändern], Seite 607.

Installierte Dateien: `ly/engraver-init.ly`, `scm/define-grobs.scm`.

Referenz der Interna: Abschnitt “Contexts” in *Referenz der Interna*, Abschnitt “VerticalAxisGroup” in *Referenz der Interna*, Abschnitt “StaffGrouper” in *Referenz der Interna*.

28.1.2 Abstände von nicht gruppierten Notensystemen

Notensysteme (wie etwa `Staff`, `DrumStaff`, `TabStaff` usw.) sind Kontexte, die eine oder mehrere Stimmen-Kontexte enthalten, aber keine anderen Notensysteme enthalten können.

Folgende Eigenschaften beeinflussen die Abstände von *nicht gruppierten* Notensystemen:

- `VerticalAxisGroup`-Eigenschaften:
  - `default-staff-staff-spacing`
  - `staff-staff-spacing`

Diese Eigenschaften sind einzlen oben behandelt worden, siehe Abschnitt 28.1.1 [Eigenschaften für Abstände innerhalb von Systemgruppen], Seite 545.

Zusätzliche Eigenschaften kommen hinzu für Notensysteme, die Teil einer Gruppieren (`StaffGroup`) werden, siehe Abschnitt 28.1.3 [Abstände von gruppierten Notensystemen], Seite 550.

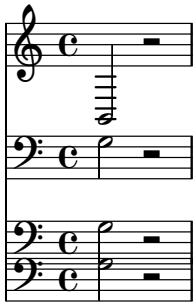
Folgendes Beispiel zeigt, wie die `default-staff-staff-spacing`-Eigenschaft sich auf die Platzierung von nicht-gruppierten Notensystemen auswirken kann. Wenn man die gleichen `\override`-Befehle auf `staff-staff-spacing` anwendet, ergäbe das den selben Effekt, würde sich aber auch in Fällen auswirken, in denen die Systeme gruppiert sind.

```
\layout {
  \context {
    \Staff
    \override VerticalAxisGroup.default-staff-staff-spacing =
      #'((basic-distance . 8)
         (minimum-distance . 7)
         (padding . 1))
  }
}

<<
  % The very low note here needs more room than 'basic-distance
  % can provide, so the distance between this staff and the next
  % is determined by 'padding.
  \new Staff { b,2 r | }

  % Here, 'basic-distance provides enough room, and there is no
  % need to compress the space (towards 'minimum-distance) to make
  % room for anything else on the page, so the distance between
  % this staff and the next is determined by 'basic-distance.
  \new Staff { \clef bass g2 r | }

  % By setting 'padding to a negative value, staves can be made to
  % collide.  The lowest acceptable value for 'basic-distance is 0.
  \new Staff \with {
    \override VerticalAxisGroup.default-staff-staff-spacing =
      #'((basic-distance . 3.5)
         (padding . -10))
  } { \clef bass g2 r | }
  \new Staff { \clef bass g2 r | }
>>
```



Siehe auch

Installierte Dateien: `scm/define-grobs.scm`.

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

Referenz der Interna: Abschnitt “VerticalAxisGroup” in *Referenz der Interna*.

28.1.3 Abstände von gruppierten Notensystemen

In Orchesterpartituren und anderen großen Partituren werden Notensysteme normalerweise in Gruppen zusammengefasst. Der Platz zwischen Gruppen ist normalerweise größer als der Zwischenraum zwischen einzelnen Notensystemen der gleichen Gruppe.

Gruppierte Notensysteme (wie `StaffGroup`, `ChoirStaff`, `GrandStaff` usw.) sind Kontexte, die mehr als ein Notensystem gleichzeitig enthalten können.

Folgende Eigenschaften beeinflussen die Platzierung von Notensystemen innerhalb von Gruppen:

- `VerticalAxisGroup`-Eigenschaften:
  - `staff-staff-spacing`
- `StaffGrouper`-Eigenschaften:
  - `staff-staff-spacing`
  - `staffgroup-staff-spacing`

Diese Grob-Eigenschaften sind weiter oben einzeln beschrieben, siehe Abschnitt 28.1.1 [Eigenschaften für Abstände innerhalb von Systemgruppen], Seite 545.

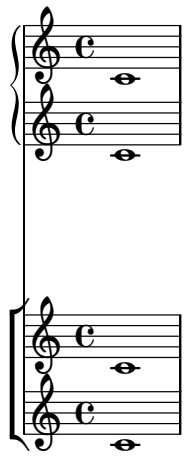
Das folgende Beispiel zeigt, wie Eigenscchaften des `StaffGrouper`-Grobs die Platzierung von gruppierten Notensystemen beeinflussen kann:

```
\layout {
  \context {
    \Score
    \override StaffGrouper.staff-staff-spacing.padding = #0
    \override StaffGrouper.staff-staff-spacing.basic-distance = #1
  }
}

<<
  \new PianoStaff \with {
    \override StaffGrouper.staffgroup-staff-spacing.basic-distance = #20
  } <<
    \new Staff { c'1 }
    \new Staff { c'1 }
  >>

  \new StaffGroup <<
```

```
    \new Staff { c'1 }
    \new Staff { c'1 }
  >>
>>
```



Siehe auch

Installierte Dateien: `scm/define-grobs.scm`.

Schnipsel: Abschnitt "Spacing" in *Schnipsel*.

Referenz der Interna: Abschnitt "VerticalAxisGroup" in *Referenz der Interna*, Abschnitt "StaffGrouper" in *Referenz der Interna*.

28.1.4 Abstände von nicht-Notensystemzeilen

Nicht-Notensystemzeilen (wie `Lyrics`, `ChordNames` usw.) sind Kontexte, deren Layoutobjekte wie Notensysteme gesetzt werden (also als horizontale Zeilen zwischen Notensystemen). Genau gesagt sind Nicht-Notensystemzeilen Nicht-Notensystemkontexte, die ein `VerticalAxisGroup`-Layoutobjekt erstellen.

Folgende Eigenschaften beeinflussen die Abstände von Nicht-Notensystemzeilen:

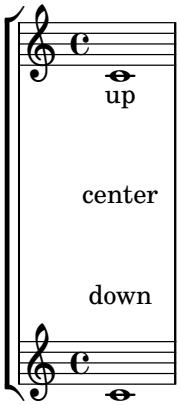
- `VerticalAxisGroup`-Eigenschaften:
  - `staff-affinity`
  - `nonstaff-relatedstaff-spacing`
  - `nonstaff-nonstaff-spacing`
  - `nonstaff-unrelatedstaff-spacing`

Diese Grob-Eigenschaften sind weiter oben einzlen beschrieben; siehe Abschnitt 28.1.1 [Eigenschaften für Abstände innerhalb von Systemgruppen], Seite 545.

Das folgende Beispiel zeigt, wie die `nonstaff-nonstaff-spacing`-Eigenschaft die Platzierung von aufeinanderfolgenden Nicht-Notensystemzeilen beeinflussen kann. Indem hier der Wert von `stretchability` auf einen sehr hohen Wert gesetzt wird, kann der Gesangstext sehr viel weiter als normal gespreizt werden:

```
\layout {
  \context {
    \Lyrics
    \override VerticalAxisGroup.nonstaff-nonstaff-spacing.stretchability = #1000
  }
}
```

```
\new StaffGroup
<<
  \new Staff \with {
    \override VerticalAxisGroup.staff-staff-spacing = #'((basic-distance . 30))
  } { c'1 }
  \new Lyrics \with {
    \override VerticalAxisGroup.staff-affinity = #UP
  } \lyricmode { up }
  \new Lyrics \with {
    \override VerticalAxisGroup.staff-affinity = #CENTER
  } \lyricmode { center }
  \new Lyrics \with {
    \override VerticalAxisGroup.staff-affinity = #DOWN
  } \lyricmode { down }
  \new Staff { c'1 }
>>
```



Siehe auch

Installierte Dateien: `ly/engraver-init.ly`, `scm/define-grobs.scm`.

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

Referenz der Interna: Abschnitt “Contexts” in *Referenz der Interna*, Abschnitt “VerticalAxisGroup” in *Referenz der Interna*.

28.2 Explizite Positionierung von Systemen

Man kann die flexiblen Einstellungen der vertikalen Abstände, wie sie im vorigen Abschnitt erklärt wurden, als eine Sammlung verschiedenerer Einstellmöglichkeiten verstehen, die vor allem die Größe des vertikalen Platzes zwischen Notensystemen und Gruppen auf der Seite kontrollieren.

Die vertikale Platzverteilung kann aber auch auf andere Weise eingestellt werden: mit den Optionen von `NonMusicalPaperColumn #'line-break-system-details`. Während der flexible vertikale Abstandsmechanismus vertikalen Füllplatz definiert, werden mit `NonMusicalPaperColumn #'line-break-system-details` absolute vertikale Positionen auf der Seite festgelegt.

`NonMusicalPaperColumn #'line-break-system-details` akzeptiert eine Liste aus drei unterschiedlichen Einstellungen:

- `X-offset`
- `Y-offset`
- `alignment-distances`

Veränderungen von Grobs (wozu auch NonMusicalPaperColumn gehört), können an drei unterschiedlichen Stellen in der Quelldatei vorgenommen werde:

- mitten im Notentext
- in einer \context-Umgebung
- in einer \with-Umgebung

Wenn der Grob NonMusicalPaperColumn verändert werden soll, wird der \override-Befehl in der \context oder \with-Umgebung eingesetzt. Wenn die Veränderungen aber mitten im Notentext stattfinden sollen, müssen Sie den Befehl \overrideProperty einsetzen. Einige Beispiele für eine Veränderungen von NonMusicalPaperColumn mit dem \overrideProperty-Befehl sind hier aufgelistet:

```
\overrideProperty NonMusicalPaperColumn.line-break-system-details #'((X-offset . 20))

\overrideProperty NonMusicalPaperColumn.line-break-system-details #'((Y-offset . 40))
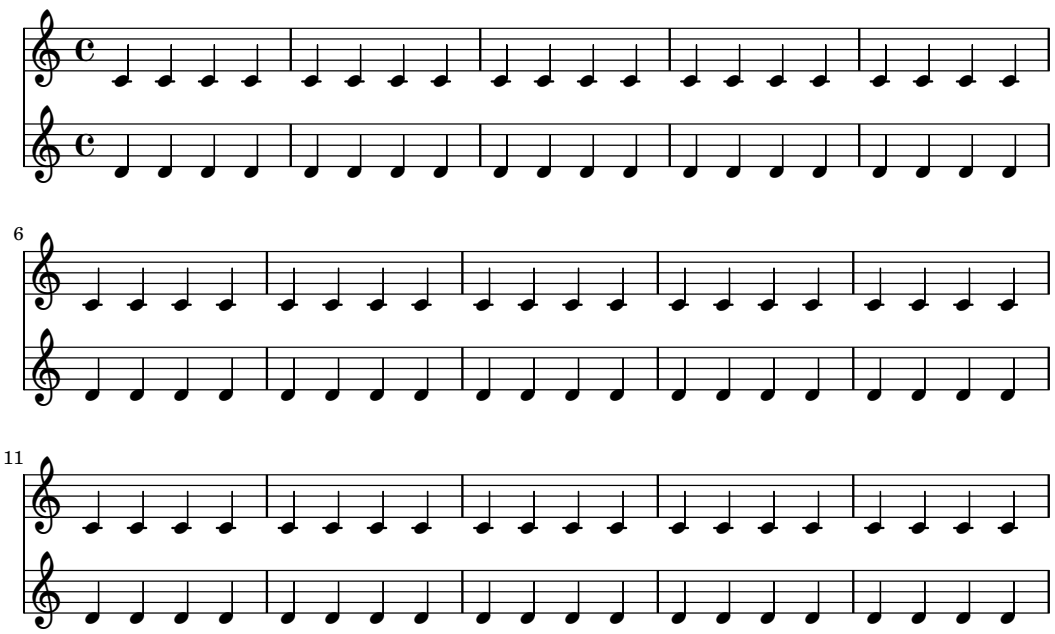
\overrideProperty NonMusicalPaperColumn.line-break-system-details #'((X-offset . 20)
                                  (Y-offset . 40))

\overrideProperty NonMusicalPaperColumn.line-break-system-details #'((alignment-distanc

\overrideProperty NonMusicalPaperColumn.line-break-system-details #'((X-offset . 20)
                                  (Y-offset . 40)
                                  (alignment-distances . (15)))
```

Um zu verstehen, wie jede dieser unterschiedlichen Einstellungen funktioniert, wollen wir uns ein Beispiel vornehmen, dass überhaupt keine Einstellungen (d.h. \override-Befehle) enthält:
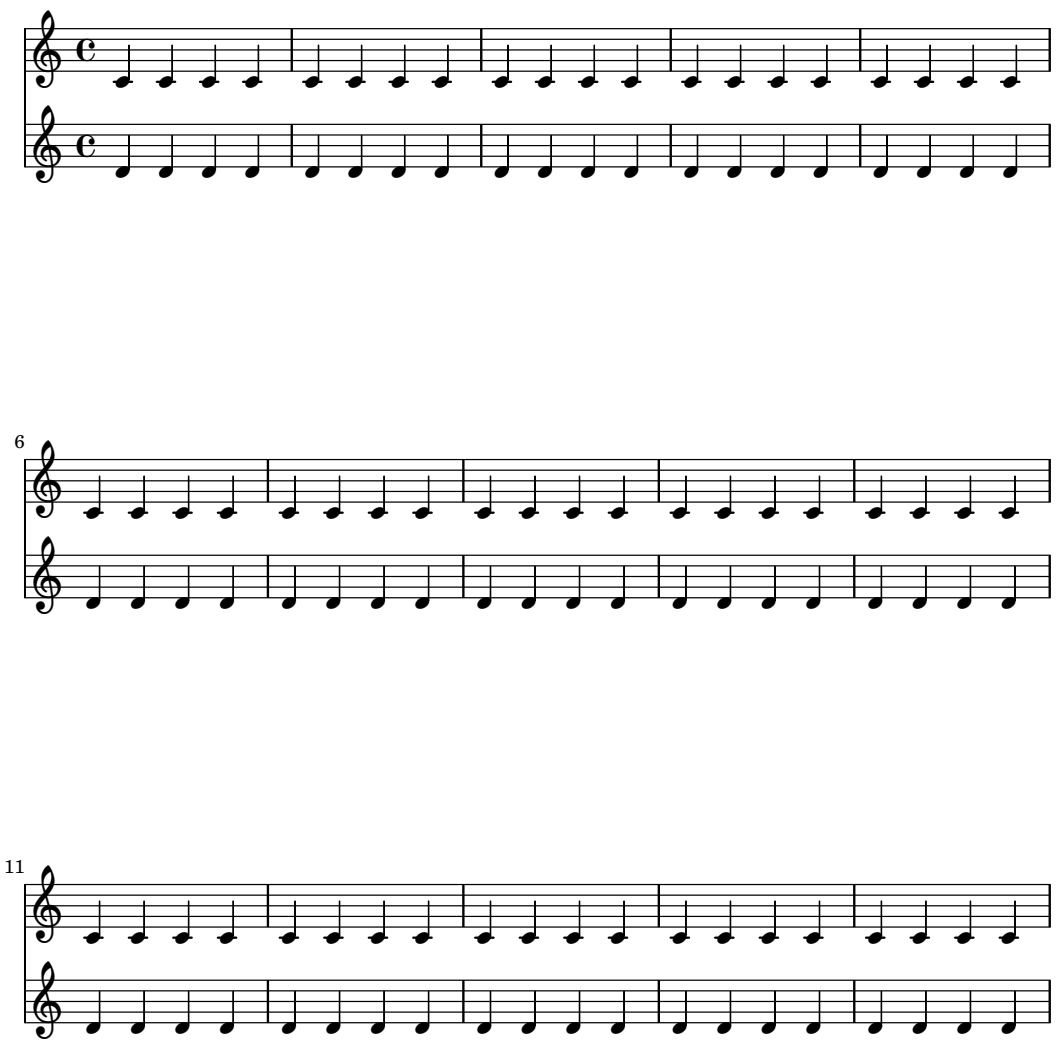
```
\header { tagline = ##f }
\paper { left-margin = 0\mm }
\book {
  \score {
    <<
      \new Staff <<
        \new Voice {
          s1*5 \break
          s1*5 \break
          s1*5 \break
        }
        \new Voice { \repeat unfold 15 { c'4 c' c' c' } }
      >>
      \new Staff {
        \repeat unfold 15 { d'4 d' d' d' }
      }
    >>
  }
}
```



Diese Partitur nimmt Zeilen- und Seitenumbruchinformationen in einer eigenen Stimme vor. Mit dieser Methode kann die Layout-Information einfach von den Noten getrennt werden, was sehr hilfreich ist, wenn das Beispiel komplizierter wird. Siehe auch Abschnitt 27.8 [Eine zusätzliche Stimme für Umbrüche benutzen], Seite 542.

Ausdrückliche `\break`-Befehle teilen die Noten in sechs Takte lange Zeilen. Die vertikale Platzverteilung wird von LilyPond errechnet. Um den vertikalen Beginn einer jeden Systemgruppe genau anzugeben, kann `Y-offset` in der `line-break-system-details`-Eigenschaft des `NonMusicalPaperColumn`-Grobs wie in dem Beispiel ersichtlich benutzt werden:

```
\header { tagline = ##f }
\paper { left-margin = 0\mm }
\book {
  \score {
    <<
      \new Staff <<
        \new Voice {
          \overrideProperty Score.NonMusicalPaperColumn.line-break-system-details #'((Y
          s1*5 \break
          \overrideProperty Score.NonMusicalPaperColumn.line-break-system-details #'((Y
          s1*5 \break
          \overrideProperty Score.NonMusicalPaperColumn.line-break-system-details #'((Y
          s1*5 \break
        }
        \new Voice { \repeat unfold 15 { c'4 c' c' c' } }
      >>
      \new Staff {
        \repeat unfold 15 { d'4 d' d' d' }
      }
    >>
  }
}
```
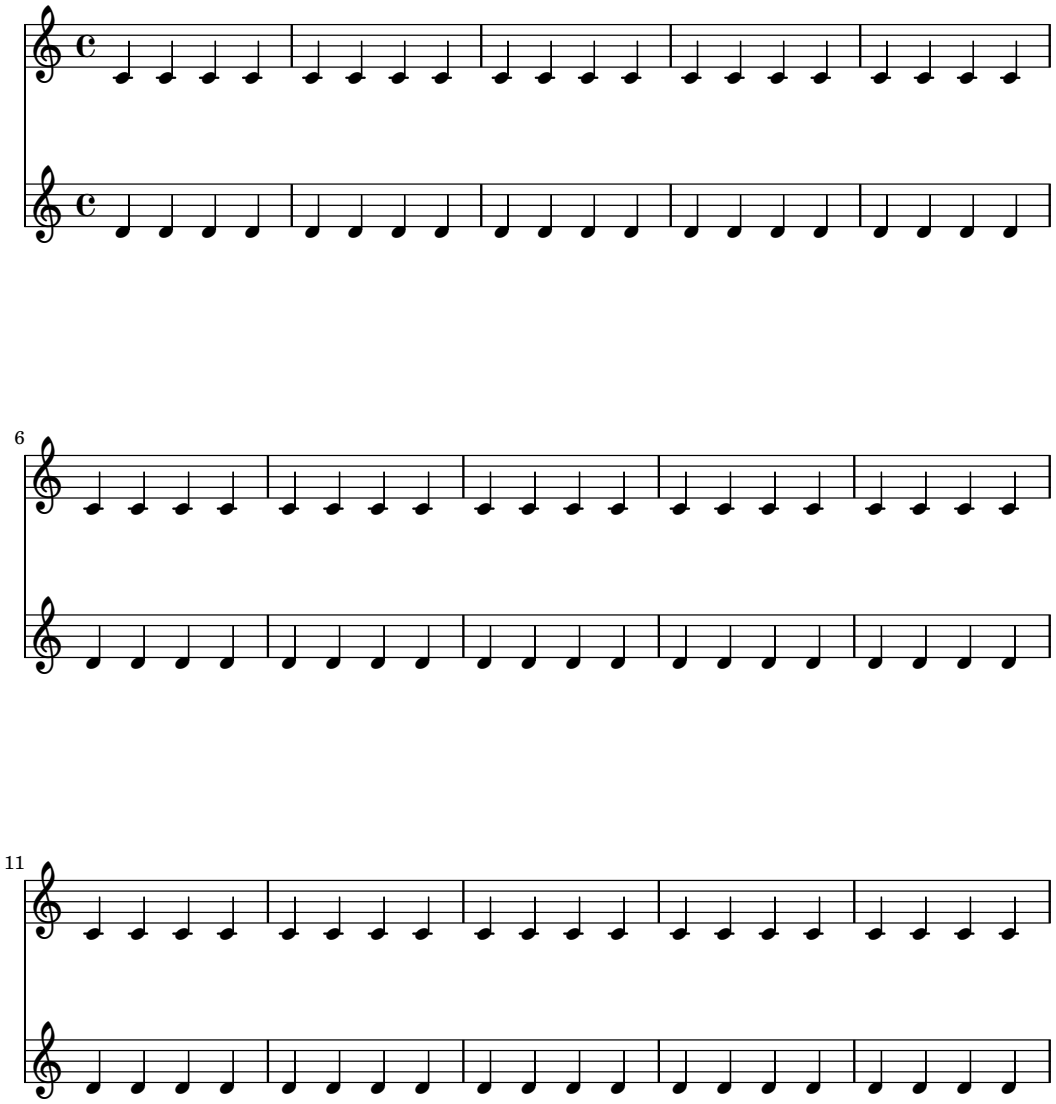


In der `line-break-system-details`-Eigenschaft kann eine Liste mit vielen Einstellungen eingegeben werden, aber hier wird nur eine Einstellung angegeben. Die `Y-offset`-Eigenschaft bestimmt hier die exakte vertikale Position auf der Seite, an welcher jede neue Systemgruppe begonnen wird.

Da jetzt der exakte Beginn einer jeden Systemgruppe explizit festgelegt wurde, können wir auch den exakten Beginn eines jeden Notensystems in der Gruppe festlegen. Dies geschieht mit der `alignment-distances`-Eigenschaft von `line-break-system-details`.
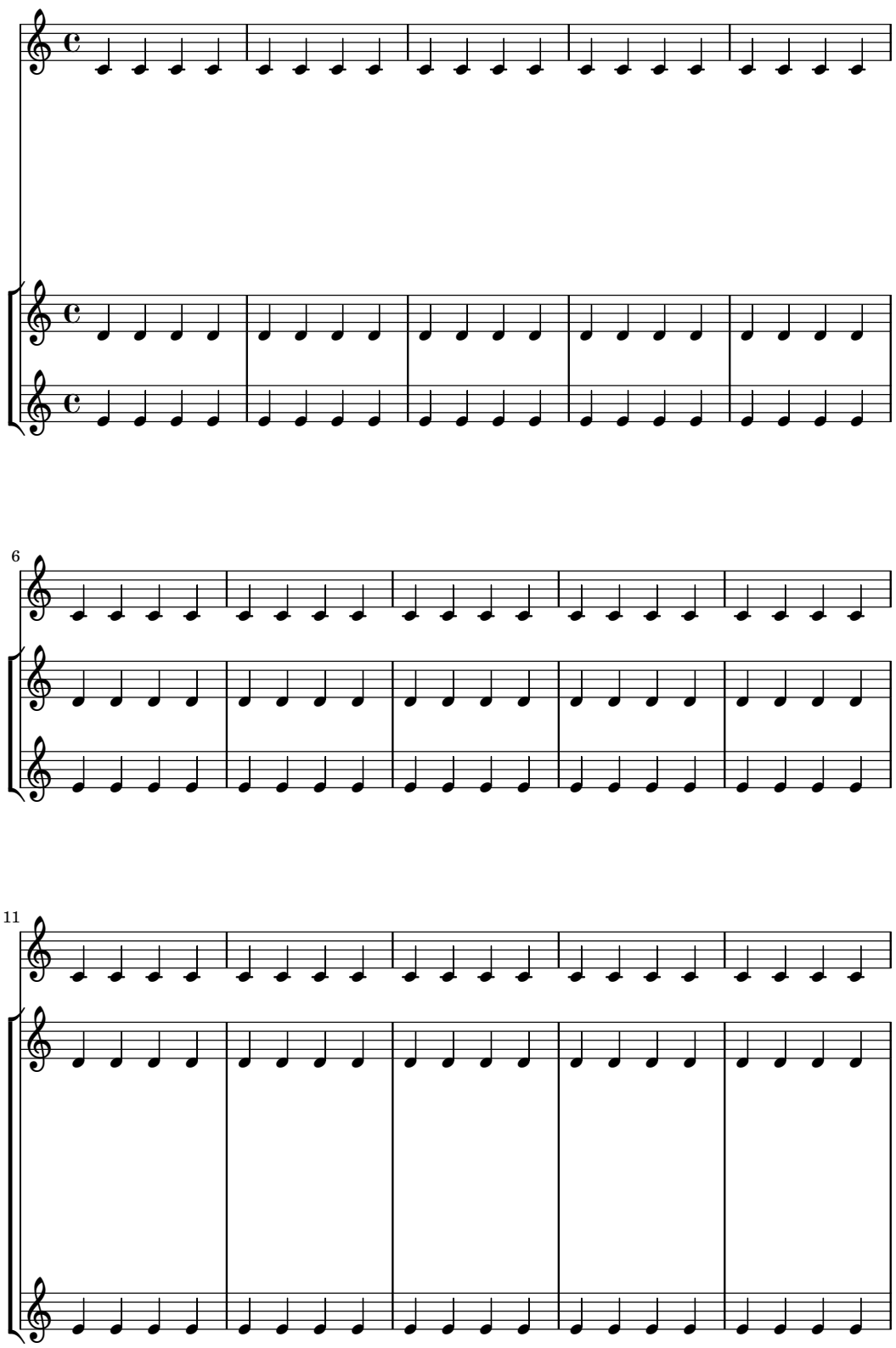
```
\header { tagline = ##f }
\paper { left-margin = 0\mm }
\book {
  \score {
    <<
      \new Staff <<
        \new Voice {
          \overrideProperty Score.NonMusicalPaperColumn.line-break-system-details #'((
                                        (alignment-distances . (15)))
          s1*5 \break
          \overrideProperty Score.NonMusicalPaperColumn.line-break-system-details #'((
                                        (alignment-distances . (15)))
          s1*5 \break
          \overrideProperty Score.NonMusicalPaperColumn.line-break-system-details #'((
```

```
                                        (alignment-distances . (15)))
            s1*5 \break
          }
          \new Voice { \repeat unfold 15 { c'4 c' c' c' } }
        >>
        \new Staff {
          \repeat unfold 15 { d'4 d' d' d' }
        }
      >>
    }
}
```



Dem `line-break-system-details`-Attribut des `NonMusicalPaperColumn`-Grobs werden zwei unterschiedliche Eigenschaften zugewiesen. Auch wenn die Aliste der Attribute von `line-break-system-details` sehr viel mehr Platzierungsparameter akzeptiert (wie etwa ein korrespondierendes `X-offset`-Paar), müssen hier nur die Parameter `Y-offset` und `alignment-distances` gesetzt werden, um den vertikalen Beginn jedes Systems und jeder Systemgruppe zu kontrollieren. `Y-offset` bestimmt also die vertikale Position von Systemgruppen und `alignment-distances` die vertikale Position von einzelnen Notensystemen.

```
\header { tagline = ##f }
\paper { left-margin = 0\mm }
\book {
  \score {
    <<
      \new Staff <<
        \new Voice {
          \overrideProperty Score.NonMusicalPaperColumn.line-break-system-details #'((
                                          (alignment-distances . (30 10)))
          s1*5 \break
          \overrideProperty Score.NonMusicalPaperColumn.line-break-system-details #'((
                                          (alignment-distances . (10 10)))
          s1*5 \break
          \overrideProperty Score.NonMusicalPaperColumn.line-break-system-details #'((
                                          (alignment-distances . (10 30)))
          s1*5 \break
        }
        \new Voice { \repeat unfold 15 { c'4 c' c' c' } }
      >>
      \new StaffGroup <<
        \new Staff { \repeat unfold 15 { d'4 d' d' d' } }
        \new Staff { \repeat unfold 15 { e'4 e' e' e' } }
      >>
    >>
  }
}
```



Einige Dinge sollten beachtet werden:

- Wenn `alignment-distances` benutzt wird, werden Gesangstextzeilen nicht als ein System gezählt.
- Die Einheiten der Zahlen, die für `X-offset`, `Y-offset` und `alignment-distances` benutzt werden, werden als Vielfaches des Abstandes zwischen zwei Notenlinien gewertet. Positive Werte verschieben Systeme und Gesangstext nach oben, negative Werte nach unten.

- Weil die Einstellungen von `NonMusicalPaperColumn #'line-break-system-details` es möglich machen, Notensysteme und Gruppen an beliebigen Stellen auf der Seite zu platzieren, kann man damit auch Ränder überschreiben oder sogar Notensysteme übereinander platzieren. Sinnvolle Werte für diese Parameter werden derartiges Verhalten vermeiden.

Siehe auch

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

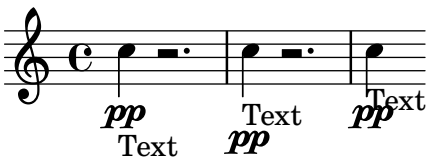
28.3 Vermeidung von vertikalen Zusammenstößen

Intuitiv gibt es in der Notation einige Objekte, die zu dem Notensystem gehören, und einige andere, die immer außerhalb des Notensystems positioniert werden sollten. Zu diesen letzteren gehören etwa Übungszeichen, Textbeschriftung und Dynamikbezeichnung (die als Objekte außerhalb des Systems bezeichnet werden können). LilyPonds Regeln um diese Objekte zu positionieren lautet: so nah am Notensystem wie möglich, aber gerade so weit weg, dass sie nicht mit anderen Objekten zusammenstoßen.

Dabei setzt LilyPond die `outside-staff-priority`-Eigenschaft ein um herauszufinden, ob ein Grob ein Objekt außerhalb des Systems ist: wenn `outside-staff-priority` eine Zahl ist, dann handelt es sich um ein Objekt außerhalb des Systems. Zusätzlich teilt `outside-staff-priority` noch mit, in welcher Reihenfolge die Objekte außerhalb des Systems gesetzt werden sollen.

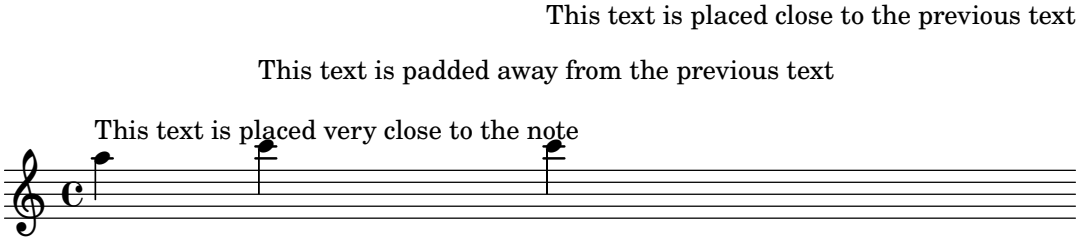
Zuerst werden alle Objekte gesetzt, die nicht außerhalb des Systems gehören. Dann werden die Objekte außerhalb des Systems nach dem Wert ihrer `outside-staff-priority` (in aufsteigender Anordnung) sortiert. Eins nach dem anderen werden diese Objekte schließlich genommen und so platziert, dass sie nicht mit den Objekten zusammenstoßen, die bereits platziert worden sind. Wenn also zwei Objekte außerhalb des Systems um den gleichen Platz streiten, wird das mit dem geringeren Wert von `outside-staff-priority` näher an das entsprechende Notensystem gesetzt.

```
\relative c'' {
  c4_"Text"\pp
  r2.
  \once \override TextScript.outside-staff-priority = #1
  c4_"Text"\pp % this time the text will be closer to the staff
  r2.
  % by setting outside-staff-priority to a non-number,
  % we disable the automatic collision avoidance
  \once \override TextScript.outside-staff-priority = ##f
  \once \override DynamicLineSpanner.outside-staff-priority = ##f
  c4_"Text"\pp % now they will collide
}
```



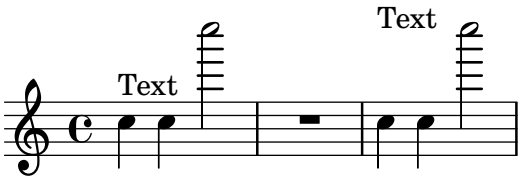
Der Platz, der zwischen einem Objekt außerhalb des Systems und dem vorhergehenden Objekt eingefügt werden kann (auch als padding bezeichnet), kann durch `outside-staff-padding` kontrolliert werden.

```
\once \override TextScript.outside-staff-padding = #0
a'^"This text is placed very close to the note"
\once \override TextScript.outside-staff-padding = #3
c^"This text is padded away from the previous text"
c^"This text is placed close to the previous text"
```



Standardmäßig werden Objekte außerhalb des Systems so gesetzt, dass sie eine horizontale Überschneidung mit einem der vorher gesetzten Grobs vermeiden. Das kann zu Situationen führen, in denen Objekte sehr dicht nebeneinander gesetzt werden. Der vertikale Platz zwischen Notensystemen kann auch gesetzt werden, sodass Objekte außerhalb des Systems ineinander greifen. Mit der Eigenschaft `outside-staff-horizontal-padding` können Objekte vertikal verschoben werden und derartige Situationen kommen nicht vor.

```
% the markup is too close to the following note
c4^"Text"
c4
c''2
% setting outside-staff-horizontal-padding fixes this
R1
\once \override TextScript.outside-staff-horizontal-padding = #1
c,,4^"Text"
c4
c''2
```



Siehe auch

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.
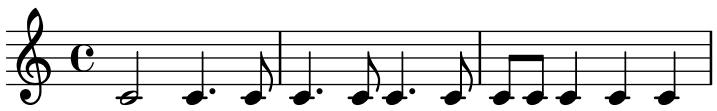
29 Horizontale Abstände

29.1 Überblick über horizontale Abstände

Die Setzmaschine interpretiert unterschiedliche Notendauern als dehnbare Abstände (engl. spring) unterschiedlicher Länge. Längere Dauern erhalten mehr Platz, kürzere weniger. Die kürzeste Dauer erhält eine feste Breite (die mit `shortest-duration-space` im `SpacingSpanner`-Objekt kontrolliert werden kann). Je länger die Dauer, umso mehr Platz erhält die Note: wenn ihre Dauer verdoppelt wird, wird ein bestimmter Platz hinzugefügt (dessen Breite durch `spacing-increment` bestimmt werden kann).

Das folgende Stück beispielsweise enthält Halbe, Viertel und Achtel. Die Achtelnote wird gefolgt von einem Notenkopfabstand (NKA). Die Viertel wird von 2 NKA gefolgt, die Halbe von 3 NKA usw.

```
c2 c4. c8 c4. c8 c4. c8 c8
c8 c4 c4 c4
```



Normalerweise ist `spacing-increment` definiert als 1.2 mal der Abstand zwischen zwei Notenlinien, was in etwa die Breite eines Notenkopfes ist. `shortest-duration-space` ist definiert als 2.0, was bedeutet, dass die kürzeste Note 2.4 Notenlinienabstände 2.0 mal der Wert von `spacing-increment`) horizontalen Abstand erhält. Der Abstand wird von der linken Kante des Symbols errechnet, so dass die kürzeste Note üblicherweise von 1 NKA Abstand gefolgt wird.
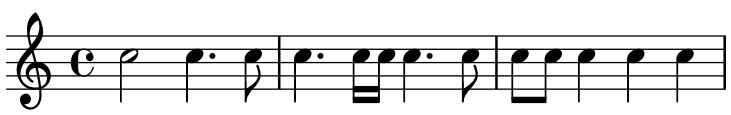
Wenn diese Herangehensweise konsequent angewandt würde, würde eine einzige Zweiunddreißigstel eine Partitur, in der vor allem Achtel und Sechzehntel vorkommen, sehr weit auseinanderdehnen. Die kürzeste Note wäre nun keine Sechzehntel mehr, sondern eine Zweiunddreißigstel, wodurch an jede Note der Wert von 1 NKA hinzugefügt würde. Um das zu vermeiden, ist die kürzeste Dauer für die Platzverteilung nicht die kürzeste Note einer Partitur, sondern die, die am häufigsten vorkommt.

Die Notendauer, die am häufigsten vorkommt, wird auf folgende Weise bestimmt: in jedem Takt wird die kürzeste Note bestimmt. Die häufigste kürzeste Note wird dann als Grundlage für die Platzverteilung der Noten herangezogen, mit der Bedingung, dass diese kürzeste Note immer ein Achtel oder kürzer sein soll. Die kürzeste Dauer wird ausgegeben, wenn `lilypond` mit der Option `--verbose` aufgerufen wird.

Diese Dauern können aber auch angepasst werden. Wenn Sie die Eigenschaft `common-shortest-duration` in dem `SpacingSpanner` setzen, dann wird hiermit die Grunddauer für die Platzverteilung eingestellt. Die maximale Dauer für diesen Grundwert (normalerweise eine Achtel) wird definiert mit `base-shortest-duration`.
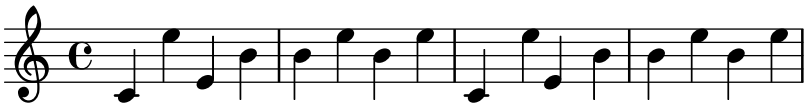
Noten, die noch kürzer sind als die häufigste kürzeste Note, werden durch einen Platz voneinander getrennt, der proportional zu ihrer Dauer in Beziehung zur häufigsten kürzesten Note ist. Wenn also nur ein paar Sechzehntel zu dem obigen Beispiel hinzugefügt werden, würden sie von 1/2 NKA gefolgt werden:

```
c2 c4. c8 c4. c16[ c] c4. c8 c8 c8 c4 c4 c4
```



In dem *Aufsatz zum automatisierten Notensatz* wurde erklärt, dass die Richtung der Notenhälse die Platzverteilung beeinflusst (siehe Abschnitt “Optischer Ausgleich” in *Aufsatz*). Das

wird kontrolliert durch die stem-spacing-correction-Eigenschaft in dem NoteSpacing-Objekt. Dieses Objekt wird für jeden Voice-Kontext erstellt. Das StaffSpacing-Objekt (in einem Staff-Kontext erstellt) enthält die gleiche Eigenschaft, um die Verteilung von Hälsen neben Taktlinien zu kontrollieren. In dem folgenden Beispiel werden diese Einstellungen gezeigt, einmal mit den Standardwerten und dann mit größeren Werten, damit man sie besser sieht:



Proportionale Notation ist unterstützt, siehe Abschnitt 29.5 [Proportionale Notation], Seite 565.

Siehe auch

Aufsatz über den automatischen Notensatz: Abschnitt “Optischer Ausgleich” in *Aufsatz*.

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

Referenz der Interna: Abschnitt “SpacingSpanner” in *Referenz der Interna*, Abschnitt “NoteSpacing” in *Referenz der Interna*, Abschnitt “StaffSpacing” in *Referenz der Interna*, Abschnitt “NonMusicalPaperColumn” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Es gibt keine sinnvolle Möglichkeit, die horizontale Verteilung der Noten zu unterdrücken. Die folgende Problemumgehung, mit der dehnbare Abstände (padding) eingesetzt werden, kann benutzt werden, um zusätzlichen Platz in eine Partitur einzufügen.

```
\override Score.NonMusicalPaperColumn.padding = #10
```
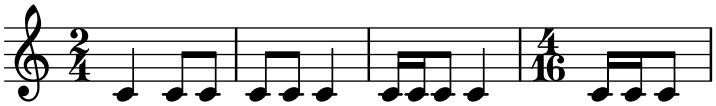
Es gibt derzeit keine Möglichkeit, den Platz zu verringern.

29.2 Eine neuer Bereich mit anderen Abständen

Neue Abschnitte mit unterschiedlichen Notenabstandsparametern können mit dem Befehl newSpacingSection begonnen werden. Das ist hilfreich, wenn in verschiedenen Abschnitten die Verhältnisse von kurzen und langen Noten sehr unterschiedlich ausfallen.

Im folgenden Beispiel wird durch die neue Taktart ein neuer Abschnitt begonnen, in dem die Sechzehntel weiter außeinander gesetzt werden sollen.

```
\time 2/4
c4 c8 c
c8 c c4 c16[ c c8] c4
\newSpacingSection
\time 4/16
c16[ c c8]
```



Der \newSpacingSection-Befehl erstellt ein neues SpacingSpanner-Objekt, weshalb auch neue Anpassungen mit dem \override-Befehl an dieser Stelle eingesetzt werden können.

Siehe auch
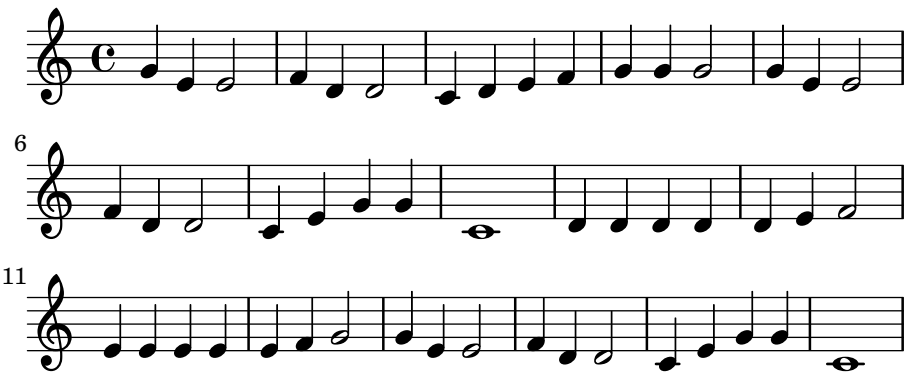
Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

Referenz der Interna: Abschnitt “SpacingSpanner” in *Referenz der Interna*.

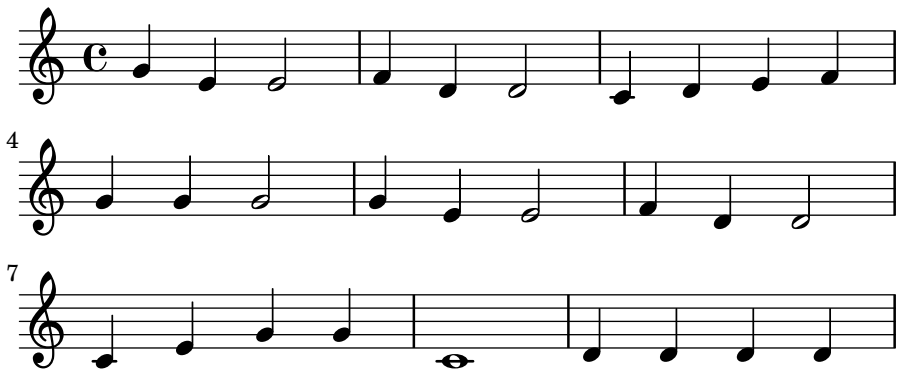
29.3 Horizontale Abstände verändern

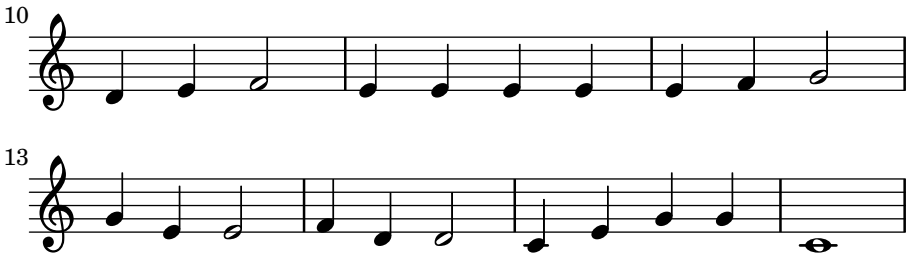
Die horizontalen Abstände können mit der `base-shortest-duration`-Eigenschaft verändert werden. In den folgenden Beispielen werden die gleichen Noten eingesetzt, zuerst ohne die Eigenschaft zu verändern, im zweiten Beispiel dann mit einem anderen Wert. Größere Werte für `ly:make-moment` ergeben dichtere Noten. `ly:make-moment` erstellt eine Dauer, die als Bruch notiert wird, sodass `1 4` eine größere Dauer ist als `1 16`.

```
\score {
  \relative {
    g'4 e e2 | f4 d d2 | c4 d e f | g4 g g2 |
    g4 e e2 | f4 d d2 | c4 e g g | c,1 |
    d4 d d d | d4 e f2 | e4 e e e | e4 f g2 |
    g4 e e2 | f4 d d2 | c4 e g g | c,1 |
  }
}
```



```
\score {
  \relative {
    g'4 e e2 | f4 d d2 | c4 d e f | g4 g g2 |
    g4 e e2 | f4 d d2 | c4 e g g | c,1 |
    d4 d d d | d4 e f2 | e4 e e e | e4 f g2 |
    g4 e e2 | f4 d d2 | c4 e g g | c,1 |
  }
  \layout {
    \context {
      \Score
      \override SpacingSpanner.base-shortest-duration = \musicLength 16
    }
  }
}
```

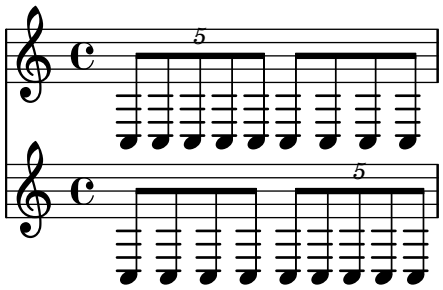




Ausgewählte Schnipsel

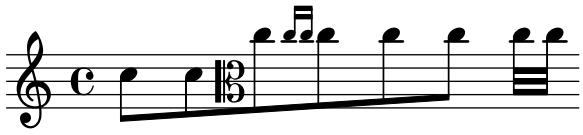
Standardmäßig wird die Platzverteilung in Triolen und andern rhythmischen Aufteilungen nach verschiedenen nicht von der Dauer abgeleiteten Faktoren (wie Versetzungszeichen, Schlüsselwechseln usw.) berechnet. Um diese Symbole zu ignorieren und eine gleichmäßige Verteilung der Noten zu erzwingen, kann die gleichmäßige Dehnung (engl. uniform stretching) zu Beginn einer Partitur mit `Score.SpacingSpanner #'uniform-stretching` eingeschaltet werden:

```
\score {
  <<
    \new Staff {
      \tuplet 5/4 {
        c8 c8 c8 c8 c8
      }
      c8 c8 c8 c8
    }
    \new Staff {
      c8 c8 c8 c8
      \tuplet 5/4 {
        c8 c8 c8 c8 c8
      }
    }
  >>
  \layout {
    \context {
      \Score
      \override SpacingSpanner.uniform-stretching = ##t
    }
  }
}
```



Wenn `strict-note-spacing` eingestellt ist, werden Noten gesetzt, ohne dass Schlüssel, Taktlinie oder Verzierungsnoten zusätzlichen Platz erhalten.

```
\override Score.SpacingSpanner.strict-note-spacing = ##t
\new Staff { c8[ c \clef alto c \grace { c16 c } c8 c c]  c32[ c32] }
```



Siehe auch

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

29.4 Zeilenlänge

Die grundlegenden Einstellungen, die Einfluss auf die Zeilenlänge haben, sind `indent` (Einzug) und `line-width` (Zeilenbreite). Sie werden in der `\layout`-Umgebung eingestellt. Der erste Befehl bestimmt den Einzug der ersten Zeile, der zweite die Zeilenlänge der weiteren Notenzeilen.

Wenn `ragged-right` eingestellt ist (als in der `\layout`-Umgebung auf den Wert `#t` gesetzt wurde), werden die Systeme linksbündig gesetzt und nicht bis zum rechten Rand hin durchgezogen, sondern den Noten entsprechend gesetzt. Das ist oftmals nützlich für kleine Notenfragmente und um zu überprüfen, wie eng die Noten natürlicherweise gesetzt werden würden. Die normale Einstellung ist unwahr (`#f`), aber wenn eine Partitur nur aus einer Zeile besteht, ist der Standardwert wahr.

Die Option `ragged-last` verhält sich ähnlich zu `ragged-right`, aber wirkt sich nur auf die letzte Zeile eines Stückes aus. Für diese letzte Zeile gibt es keine Einschränkungen. Das Resultat erinnert an Textabsätze im Blocksatz, wo die letzte Zeile des Absatzes mit ihrer natürlichen Länge gesetzt wird.

```
\layout {
indent = #0
line-width = #150
ragged-last = ##t
}
```

Siehe auch

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

29.5 Proportionale Notation

LilyPond hat Unterstützung für proportionale Notation. Dabei handelt es sich um eine horizontale Platzverteilung, die jeder Note einen exakt ihrer Dauer entsprechenden Platz zuordnet. Man kann es vergleichen mit der Notenplatzierung auf einem Raster. In einigen Partituren des späten 20. und frühen 21. Jahrhunderts wird dies proportionale Notation benutzt, um sehr komplizierte rhythmische Verhältnisse klarer darzustellen, oder um einen Zeitstrahl oder ähnliche Graphiken direkt in die Partitur zu integrieren.
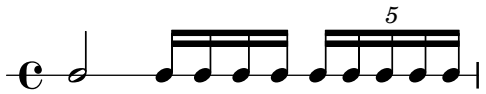
LilyPond hat Unterstützung für fünf verschiedene Einstellungen der proportionalen Notation, die alle zusammen oder jede für sich benutzt werden können:

- `proportionalNotationDuration` (proportionale Notendauer)
- `uniform-stretching` (gleichmäßige Dehnung)
- `strict-note-spacing` (strenge Notenverteilung)
- `\remove Separating_line_group_engraver` (entferne Liniengruppentrennungsengraver)
- `\override PaperColumn.used = ##t` (PapierSpalte benutzt = wahr)

In den Beispielen unten werden diese fünf unterschiedlichen Einstellungen für die proportionale Notation vorgestellt und ihre Wirkungen untereinander illustriert.

Es soll mit diesem 1 Takt langen Beispiel begonnen werden, in welchem die klassischen Abstände und Flattersatz (ragged-right) eingesetzt werden:

```
\score {
  <<
    \new RhythmicStaff {
      c'2
      c'16 c'16 c'16 c'16
      \tuplet 5/4 {
        c'16 c'16 c'16 c'16 c'16
      }
    }
  >>
}
```



Die Halbe, mit der der Takt beginnt, braucht weitaus weniger Platz als die Hälfte des Taktes. Gleichermaßen haben die Sechzehntel und die Sechzehntel-Quintolen (oder Zwanzigstel), mit denen der Takt endet, insgesamt weitaus mehr als die Hälfte der Taktbreite.

Im klassischen Notensatz kann dieses Verhalten genau das gewünschte Ergebnis bringen, weil dadurch horizontaler Platz von der Halben weggenommen werden kann und so insgesamt Platz in dem Takt eingespart wird.

Wenn allerdings ein Zeitstrahl oder andere zeitliche ablaufende Graphiken über oder unter dem Takt eingefügt werden soll, braucht man eine Notenplatzierung, die exakt der von ihnen eingenommenen Dauer entspricht. Auf folgende Art wird die proportionale Notation eingeschaltet:

```
\score {
  <<
    \new RhythmicStaff {
      c'2
      c'16 c'16 c'16 c'16
      \tuplet 5/4 {
        c'16 c'16 c'16 c'16 c'16
      }
    }
  >>
 \layout {
    \context {
      \Score
      proportionalNotationDuration = #1/20
    }
  }
}
```



Die Halbe zu Beginn des Taktes und die schnelleren Noten in der zweiten Takthälfte nehmen jetzt genau den gleichen horizontalen Platz ein. Jetzt könnte man einen Zeitstrahl mit dem Takt synchronisieren.

Die Einstellung von `proportionalNotationDuration` gehört zum `Score`-Kontext. Kontexteinstellungen können an drei verschiedenen Stellen in der Quelldatei geschrieben werden: in einer `\with`-Umgebung, in einer `\context`-Umgebung oder direkt in den Noten mit dem `\set`-Befehl. Alle drei Positionen sind gleichwertig und es hängt vom Benutzer ab, welche bevorzugt wird.

Die Eigenschaft `proportionalNotationDuration` braucht ein Argument, welches die Referenzdauer ist, anhand welcher alle Noten platziert werden. Hier wird die LilyPond Scheme-Funktion `make-moment` eingesetzt. Sie braucht zwei Argumente: einen Zähler und einen Nenner, die einen Bruch einer Ganzen darstellen. Die Funktion `(ly:make-moment 1/20)` ergibt also eine Referenzdauer von einer Zwanzigstel. Genauso gut können etwa die Dauern `(ly:make-moment 1/16)`, `(ly:make-moment 1/8)` oder `(ly:make-moment 3/97)` eingesetzt werden.

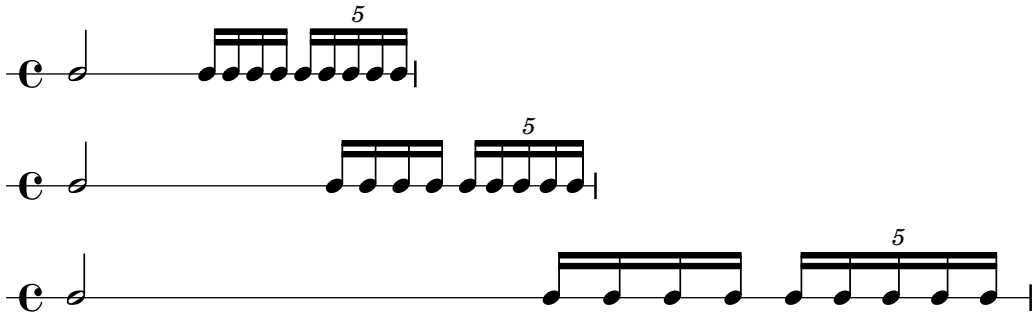
Die richtige Referenzdauer, mit der eine vernünftige Verteilung der Noten proportional möglich ist, muss durch Ausprobieren herausgefunden werden. Dabei sollte man mit einer Dauer beginnen, die der kleinsten Note des Stückes nahekommt. Kleine Referenzdauern lassen die Noten sehr gedehnt erscheinen, größere Referenzdauern zwingen sie dichter zusammen.

```
\score {
  <<
    \new RhythmicStaff {
      c'2
      c'16 c'16 c'16 c'16
      \tuplet 5/4 {
        c'16 c'16 c'16 c'16 c'16
      }
    }
  >>
  \layout {
    \context {
      \Score
      proportionalNotationDuration = #1/8
    }
  }
}

\score {
  <<
    \new RhythmicStaff {
      c'2
      c'16 c'16 c'16 c'16
      \tuplet 5/4 {
        c'16 c'16 c'16 c'16 c'16
      }
    }
  >>
  \layout {
    \context {
      \Score
      proportionalNotationDuration = #1/16
    }
  }
```

```
}

\score {
  <<
    \new RhythmicStaff {
      c'2
      c'16 c'16 c'16 c'16
      \tuplet 5/4 {
        c'16 c'16 c'16 c'16 c'16
      }
    }
  >>
  \layout {
    \context {
      \Score
      proportionalNotationDuration = #1/32
    }
  }
}
```

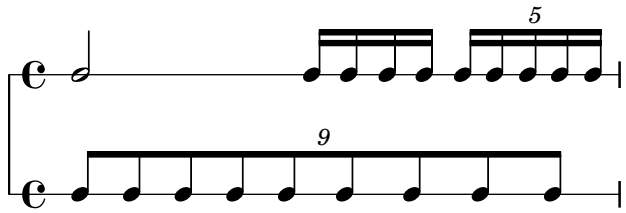


Man muss beachten, dass die Referenzdauer nicht zu groß ist (wie die Achtel in dem Beispiel oben), denn dadurch werden die Noten so dicht gesetzt, dass sich eventuell sogar Notenköpfe von sehr kleinen Notenwerten überschneiden können. Die proportionale Notation nimmt üblicherweise mehr Platz ein als die klassische Platzverteilung. Der rhythmischen Klarheit muss ein eng gesetztes Notenbild geopfert werden.

In Folgenden soll betrachtet werden, wie sich überlappende rhythmische Aufteilungen am besten positioniert werden. Als Referenz wird das erste Beispiel herangezogen, zu welchem ein zweites System mit anderen rhythmischen Werten hinzugefügt wird:

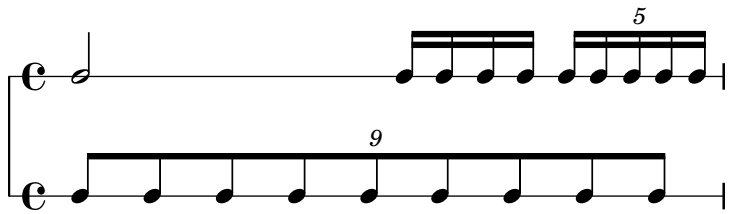
```
\score {
  <<
    \new RhythmicStaff {
      c'2
      c'16 c'16 c'16 c'16
      \tuplet 5/4 {
        c'16 c'16 c'16 c'16 c'16
      }
    }
    \new RhythmicStaff {
      \tuplet 9/8 {
        c'8 c'8 c'8 c'8 c'8 c'8 c'8 c'8 c'8
      }
    }
```

```
  >>
}
```



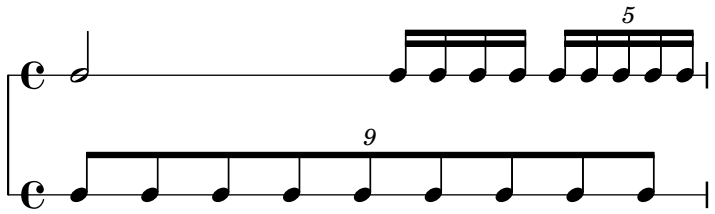
Die Platzaufteilung ist schlecht, weil die gleichlangen Noten des untersten Systems nicht gleichmäßig verteilt sind. Im klassischen Notensatz kommen komplexe rhythmische Verhältnisse wie dieses sehr selten vor, sodass der Notensatz nicht in Hinsicht auf sie optimiert ist. `proportionalNotationDuration` hilft in dieser Situation deutlich:

```
\score {
  <<
    \new RhythmicStaff {
      c'2
      c'16 c'16 c'16 c'16
      \tuplet 5/4 {
        c'16 c'16 c'16 c'16 c'16
      }
    }
    \new RhythmicStaff {
      \tuplet 9/8 {
        c'8 c'8 c'8 c'8 c'8 c'8 c'8 c'8 c'8
      }
    }
  >>
  \layout {
    \context {
      \Score
      proportionalNotationDuration = #1/20
    }
  }
}
```



Aber bei sehr genauer Betrachtung sind die Noten der zweiten Hälfte der Nonole doch immer noch eine Spur weiter gesetzt als die Noten der ersten Hälfte. Um wirklich gleichmäßige Abstände zu erzwingen, sollte auch noch die gleichmäßige Dehnung (`uniform-stretching`) angeschaltet werden, die eine Eigenschaft von `SpacingSpanner` ist:

```
\score {
  <<
    \new RhythmicStaff {
      c'2
      c'16 c'16 c'16 c'16
      \tuplet 5/4 {
        c'16 c'16 c'16 c'16 c'16
      }
    }
    \new RhythmicStaff {
      \tuplet 9/8 {
        c'8 c'8 c'8 c'8 c'8 c'8 c'8 c'8 c'8
      }
    }
  >>
  \layout {
    \context {
      \Score
      proportionalNotationDuration = #1/20
      \override SpacingSpanner.uniform-stretching = ##t
    }
  }
}
```



Das Beispiel mit den zwei Systemen ist nun exakt nach den rhythmischen Werten der Noten gesetzt, sodass ein Zeitstrahl oder ähnliches eingefügt werden könnte.

Alle Einstellungen zur proportionalen Notation erwarten, dass die `uniform-stretching`-Eigenschaft des `SpacingSpanner`-Objekts auf wahr (`#t`) gesetzt wird. Andernfalls kann es vorkommen, dass bestimmte Abstände (etwa von unsichtbaren Noten) nicht richtig gesetzt werden.

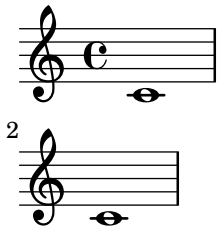
Das `SpacingSpanner`-Objekt ist ein abstraktes Grob, dass sich im `Score`-Kontext befindet. Genauso wie die Einstellungen von `proportionalNotationDuration` können auch diese Veränderungen an den drei Stellen in der Quelldatei vorkommen: in der `\with`-Umgebung innerhalb von `Score`, in einer `\context`-Umgebung oder direkt im Notentext.

Standardmäßig gibt es nur ein `SpacingSpanner` pro `Score`. Das heißt, dass `uniform-stretching` für die gesamte Partitur (d.h. für die Reichweite von `Score`) entweder an- oder ausgeschaltet ist. Man kann allerdings in einer Partitur unterschiedliche Abschnitte mit verschiedenem Platzierungsverhalten definieren. Hierzu ist der Befehl `\newSpacingSection` da. Siehe auch Abschnitt 29.2 [Eine neuer Bereich mit anderen Abständen], Seite 562.

Im Folgenden soll gezeigt werden, wie sich der `Separating_line_group_engraver` auswirkt und warum er normalerweise für proportionale Notation ausgeschaltet wird. In diesem Beispiel wird verdeutlicht, dass vor jeder ersten Note eines Notensystems immer etwas zusätzlicher Platz gesetzt wird:

```
\paper {
  indent = #0
}

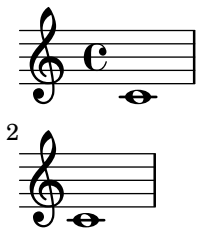
\new Staff {
  c'1
  \break
  c'1
}
```



Der gleiche horizontale zusätzliche Platz wird vor eine Noten gesetzt, wenn sie einer Taktart, einem Schlüssel oder einer Tonartbezeichnung folgt. Dieser Platz wird durch `Separating_line_group_engraver` eingefügt; wenn wir ihn aus der Partitur entfernen, entfällt auch dieser zusätzliche Platz:

```
\paper {
  indent = #0
}

\new Staff \with {
  \remove Separating_line_group_engraver
} {
  c'1
  \break
  c'1
}
```



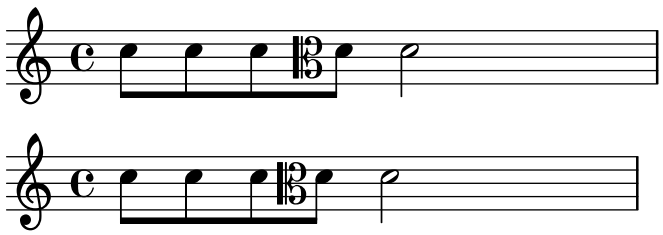
Nichtmusikalische Elemente wie Takt- und Tonartangaben, Schlüssel und Versetzungszeichen sind problematisch in proportionaler Notation. Keine dieser Elemente hat eine rhythmische Dauer, aber alle brauchen horizontalen Platz. Das Problem wird auf unterschiedliche Weise gelöst.

Es ist manchmal möglich, Probleme mit Tonarten zu lösen, indem keine benutzt werden. Das ist durchaus eine ernstzunehmende Option, weil die meisten Partituren mit proportionaler Notation für heutige Musik geschrieben werden. Ähnliches gilt für Taktarten, insbesondere, wenn ein Zeitstrahl in die Partitur eingearbeitet werden soll. In den meisten Partituren kommt jedoch irgendeine Taktart vor. Schlüssel und Versetzungszeichen sind noch wichtiger; auf sie kann selten verzichtet werden.

Eine Lösungsmöglichkeit ist es, die `strict-note-spacing`-Eigenschaft des `SpacingSpanner`-Objekts zu benutzen. Zum Vergleich die beiden Partituren unten:

```
\new Staff {
  \set Score.proportionalNotationDuration = #1/16
  c''8
  c''8
  c''8
  \clef alto
  d'8
  d'2
}

\new Staff {
  \set Score.proportionalNotationDuration = #1/16
  \override Score.SpacingSpanner.strict-note-spacing = ##t
  c''8
  c''8
  c''8
  \clef alto
  d'8
  d'2
}
```



Bei beiden handelt es sich um proportionale Notation, aber die Platzverteilung im oberen Beispiel ist zu weit wegen des Schlüsselwechsels. Die Platzverteilung des zweiten Beispiels dagegen bleibt rhythmisch korrekt. `strict-note-spacing` bewirkt, dass Takt- und Tonartbezeichnungen, Schlüssel und Versetzungszeichen keine Rolle bei der Berechnung der Abstände spielen.

Zusätzlich zu den hier vorgestellten Einstellungen gibt es noch eine Reihe von Möglichkeiten, die oft in proportionaler Notation benutzt werden. Dazu gehören:

- `\override SpacingSpanner.strict-grace-spacing = ##t`
- `tupletFullLength = ##t`
- `\override Beam.breakable = ##t`
- `\override Glissando.breakable = ##t`
- `\override TextSpanner.breakable = ##t`
- `\remove Forbid_line_break_engraver in the Voice context`

Diese Einstellungen bewirken, dass auch Verzierungsnoten proportional gesetzt werden, dass Klammern von rhythmischen Gruppen bis zu den Anfangs- und Endpunkten ausgedehnt werden und lassen dehnbare Objekte wie Balken und Glissandi auch über Taktstriche hinweg zu.

Siehe auch

Notationsreferenz: Abschnitt 29.2 [Eine neuer Bereich mit anderen Abständen], Seite 562.

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

30 Die Musik auf weniger Seiten zwingen

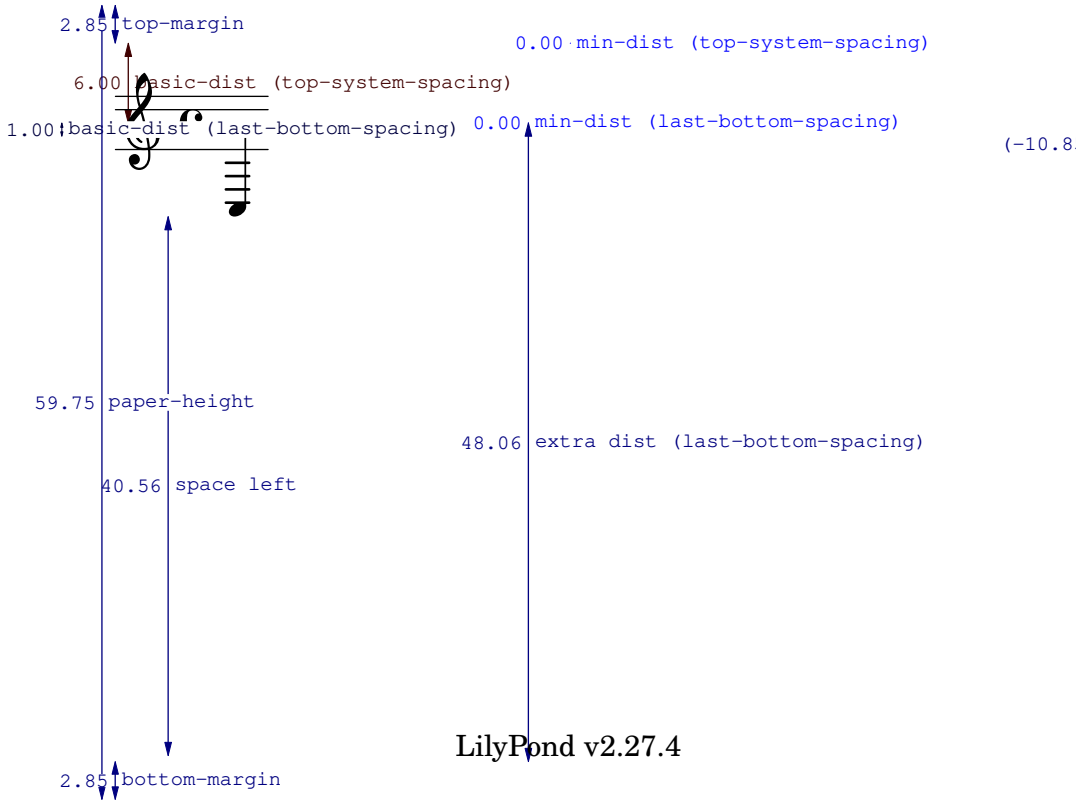
Manchmal kommt es vor, dass nur ein oder zwei Systeme auf die nächste Seite geraten, obwohl es so aussieht, als ob auf der vorigen Seite genügend Platz ist, um diese Systeme auch noch unterzubringen.

Wenn man derartige Platzierungsprobleme untersucht, ist die Funktion `annotate-spacing` von sehr großer Hilfe. Hiermit wird in den Musiksatz zusätzlich Information darüber ausgegeben, wieviel Platz bestimmten Parametern zugewiesen wird. Genaueres hierzu in Abschnitt 30.1 [Abstände anzeigen lassen], Seite 573.

30.1 Abstände anzeigen lassen

Die Dimensionen von vertikalen und horizontalen Platzierungsvariablen, die veränderbar sind, lassen sich mit ihren aktuellen Werten im Notentext anzeigen, wenn man die Funktion `annotate-spacing` in der \paper-Umgebung einschaltet:

```
#(set-default-paper-size "a6" 'landscape)
\book {
  \score { { c4 } }
  \paper { annotate-spacing = ##t }
}
```



Alle Layoutdimensionen werden in Notenlinienzwischenräumen aufgelistet, unabhängig von den Einheiten, mit denen sie in der \paper- oder \layout-Umgebung definiert worden sind. In dem letzten Beispiel hat `paper-height` einen Wert von 59.75 Notenlinienzwischenräumen und `staff-size` Systemhöhe) ist 20 Punkte. Dabei gilt:

1 Punkt = (25.4/72.27) mm

1 Notenlinienzwischenraum = (staff-size)/4 pts
= (staff-size)/4 * (25.4/72.27)
mm

In diesem Fall ist ein staff-space (Notenlinienzwischenraum) etwa gleich 1.757 mm. Deshalb entspricht der Wert von 95.75 staff-space für paper-height (Papierhöhe) 105 mm, die Höhe eines quer gelegten A6-Papiers. Die Paare (*a*,*b*) sind Intervalle, wobei *a* der untere Rand und *b* der obere Rand des Intervalls.

Siehe auch

Notationsreferenz: Abschnitt 26.2 [Die Notensystemgröße einstellen], Seite 534,

Schnipsel: Abschnitt "Spacing" in *Schnipsel*.

30.2 Abstände verändern

Die Ausgabe von annotate-spacing bietet sehr viele Details zu den vertikalen Dimensionen einer Partitur. Zu Information, wie Seitenränder und andere Layout-Variablen geändert werden können, siehe Kapitel 25 [Seitenlayout], Seite 521.

Neben Rändern gibt es einige weitere Optionen, Platz zu sparen:

- LilyPond kann die Systeme so dicht wie möglich platzieren (damit so viele Systeme wie möglich auf eine Seite passen), aber sie dann so anordnen, dass kein weißer Rand unten auf der Seite entsteht.

```
\paper {
  system-system-spacing = #'((padding . 0) (basic-distance . 0.1))
  ragged-last-bottom = ##f
  ragged-bottom = ##f
}
```

- Die Anzahl der Systeme kann erzwungen werden. Das kann auf zwei Arten helfen: wenn einfach nur ein Wert gesetzt wird, auch wenn es die gleiche Anzahl ist, die auch schon vorher von LilyPond erstellt wurde, kann manchmal dazu führen, dass mehr Systeme auf eine Seite gesetzt werden. Das liegt daran, dass ein Schritt im Notensatz ausgelassen wird, der die Seitenverteilung nur grob einschätzt, sodass eine bessere Seitenverteilung entsteht. Auch wenn man eine Verringerung der Anzahl an Systemen erzwingt, kann oft eine Seite eingespart werden. Wenn LilyPond die Musik etwa auf 11 Systeme verteilt, kann man die Benutzung von nur 10 Systemen erzwingen.

```
\paper {
  system-count = #10
}
```
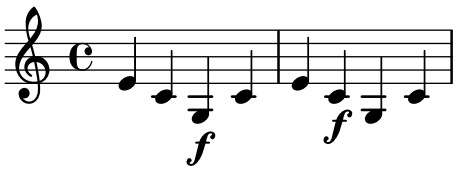
- Die Anzahl der Seiten kann erzwungen werden. Beispielsweise erzwingt folgender Code ein Layout mit zwei Seiten:

```
\paper {
  page-count = #2
}
```

- Vermeidung (oder Verminderung) von Objekten, die den vertikalen Abstand von Systemen vergrößern, hilft oft. Die Verwendung von Klammern bei Wiederholungen (oder alternativen Wiederholungen) etwa braucht mehr Platz. Wenn die Noten innerhalb der Klammern auf zwei Systeme verteilt sind, brauchen sie mehr Platz, als wenn sie nur auf einer Zeile gedruckt werden.
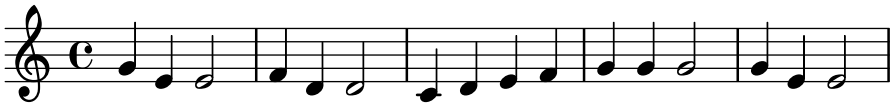
Ein anderes Beispiel ist es, Dynamik-Zeichen, die besonders weit „hervorstehen“, zu verschieben.

```
e4 c g\f c
e4 c g-\tweak X-offset #-2.7 -\tweak Y-offset #2.5 \f c
```
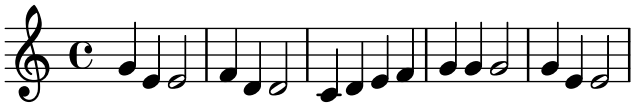


- Die horizontalen Abstände können mit der SpacingSpanner-Eigenschaft verändert werden. Siehe Abschnitt 29.3 [Horizontale Abstände verändern], Seite 563, für Einzelheiten. Dieses Beispiel zeigt die normalen Abstände:

```
\score {
  \relative {
    g'4 e e2 |
    f4 d d2 |
    c4 d e f |
    g4 g g2 |
    g4 e e2 |
  }
}
```



Das nächste Beispiel verändert common-shortest-duration (die häufigste kürzeste Note) von 1/4 zu 1/2. Die Viertelnote ist dennoch die häufigste Note in diesem Abschnitt, sodass der Notentext zusammengedrängt, wird, wenn eine Halbe als Standard angegeben wird:

```
\score {
  \relative {
    g'4 e e2 |
    f4 d d2 |
    c4 d e f |
    g4 g g2 |
    g4 e e2 |
  }
  \layout {
    \context {
      \Score
      \override SpacingSpanner.common-shortest-duration = \musicLength 2
    }
  }
}
```



Die common-shortest-duration-Eigenschaft kann nicht dynamisch verändert werden, darum muss sie immer in der \context-Umgebung definiert werden und wirkt sich somit auf eine ganze \score-Umgebung aus.

Siehe auch

Notationsreferenz: Kapitel 25 [Seitenlayout], Seite 521, Abschnitt 29.3 [Horizontale Abstände verändern], Seite 563.

Schnipsel: Abschnitt “Spacing” in *Schnipsel*.

Feinadjustierung des Notenbildes

31 Standardeinstellungen verändern

Das Ziel von LilyPonds Design ist es, von sich aus gut gesetzte Noten zu produzieren. Es kann aber trotzdem vorkommen, dass Sie diesen Standardsatz ändern wollen. Das Layout kann mithilfe einer recht großen Anzahl von „Schaltern und Knöpfen“ kontrolliert werden. Sie werden als „Eigenschaften“ (engl. properties) bezeichnet. Eine kurze Einführung und Übung, wie man auf diese Eigenschaften zugreifen kann und sie verändern kann, findet sich im Handbuch zum Lernen, siehe Abschnitt “Die Ausgabe verbessern” in *Handbuch zum Lernen*. Das Kapitel sollte zuerst gelesen werden. In diesem Kapitel werden die gleichen Themen behandelt, aber der Schwerpunkt liegt eher auf einer technischen Darstellung.

Die definitive Beschreibung der unterschiedlichen Einstellmöglichenkeiten findet sich in einem eigenen Dokument: Abschnitt “der Referenz der Interna” in *Referenz der Interna*. Diese Referenz zeigt alle Variablen, Funktionen und Optionen, die in LilyPond möglich sind. Es existiert als ein HTML-Dokumente, das sich on-line (`https://lilypond.org/doc/stable/Documentation/internals/`), aber auch lokal in das LilyPond-Dokumentationspaket integriert lesen lässt.

Intern benutzt LilyPond Scheme (ein LISP-Dialekt), um eine Infrastruktur zur Verfügung zu stellen. Wenn Layoutentscheidungen verändert werden sollen, müssen auf die programminternen Prozesse zugegriffen werden, wozu Scheme-Code benötigt wird. Scheme-Abschnitte werden in einer LilyPond-Quelldatei mit einer Raute # begonnen.[1]

[1] Abschnitt “Scheme-Tutorium” in *Extending* enthält eine kurze Übung, wie man Zahlen, Listen, Zeichenketten und Symbole in Scheme notiert.

32 Interpretationskontexte

Dieser Abschnitt erklärt, was Kontexte sind und wie man sie verändern kann.

Siehe auch

Handbuch zum Lernen: Abschnitt “Kontexte und Engraver” in *Handbuch zum Lernen*.

Installierte Dateien: `ly/engraver-init.ly`, `ly/performer-init.ly`.

Schnipsel: Abschnitt “Contexts and engravers” in *Schnipsel*.

Referenz der Interna: Abschnitt “Contexts” in *Referenz der Interna*, Abschnitt “Engravers and Performers” in *Referenz der Interna*.

32.1 Was sind Kontexte?

Kontexte sind hierarchisch geordnet:

32.1.1 Score – der Vater aller Kontexte

`Score` (Partitur) ist der höchste Notationskontext. Kein anderer Kontext kann einen `Score`-Kontext enthalten. Im Normalfall kümmert sich der `Score`-Kontext um die Verwaltung der Taktarten und sorgt dafür, dass Elemente wie Schlüssel und Taktart- oder Tonartbezeichnungen über die Systeme hinweg aneinander ausgerichtet sind.

Ein `Score`-Kontext wird eingerichtet, wenn eine `\score {...}` oder `\layout {...}`-Umgebung interpretiert wird.

32.1.2 Oberste Kontexte – Container für Systeme

Diese Kontexte fassen Systeme zu Gruppen zusammen und werden darum hier als Systemgruppen bezeichnet (engl. staffgroup).

StaffGroup

Gruppiert Systeme und fügt eine eckige Klammer auf der linken Seite hinzu. Die Taktstriche der enthaltenen Systeme werden vertikal miteinander verbunden. `StaffGroup` besteht nur aus einer Ansammlung von Systemen mit einer eckigen Klammer zu Beginn der Zeile und durchgezogenen Taktstriche.

ChoirStaff

Entspricht `StaffGroup`, außer dass die Taktstriche der enthaltenen Systeme nicht vertikal miteinander verbunden sind.

GrandStaff

Gruppiert Systeme mit einer geschweiften Klammer zur Linken. Die Taktlinien der enthaltenen Systeme werden vertikal verbunden.

PianoStaff

Entspricht `GrandStaff`, hat aber zusätzlich Unterstützung für Instrumentenbezeichnungen zu Beginn jeder Systemgruppe.

32.1.3 Mittlere Kontexte – Systeme

Diese Kontexte stellen verschiedene Arten einzelner Notationssysteme (engl. staff) dar.

Staff

Kümmert sich um Schlüssel, Taktstriche, Tonarten und Versetzungszeichen. Er kann `Voice`-Kontexte enthalten.

RhythmicStaff

Entspricht `Staff`, aber dient zur Notation von Rhythmen: Tonhöhen werden ignoriert und die Noten auf einer einzigen Linie ausgegeben.

TabStaff

Ein Kontext um Tabulaturen zu erstellen. Die Standardeinstellung ist eine Gitarrentabulatur mit sechs Notenlinien.

DrumStaff

Ein Kontext zur Notation von Perkussion. Er kann `DrumVoice`-Kontexte enthalten.

VaticanaStaff

Entspricht `Staff`, aber eignet sich besonders zum Notensatz des Gregorianischen Chorals.

MensuralStaff

Entspricht `Staff`, aber eignet sich zum Notensatz von Noten in der Mensuralnotation.

32.1.4 Unterste Kontexte – Stimmen

Stimmen-(`Voice`-Kontexte initialisieren bestimmte Eigenschaften und laden bestimmte Engraver. Weil es sich bei Stimmen um die untersten Kontexte handelt, können sie keine weiteren Kontexte enthalten.

Voice

Entspricht einer Stimme auf einem Notensystem. Der Kontext kümmert sich um die Umsetzung von Noten, Dynamikzeichen, Hälsen, Balken, diversen Texten, Bögen und Pausen. Wenn mehr als eine Stimme pro System benötigt wird, muss dieser Kontext explizit initialisiert werden.

VaticanaVoice

Entspricht `Voice`, aber eignet sich besonders zum Notensatz des Gregorianischen Chorals.

MensuralVoice

Entspricht `Voice`, aber mit Änderungen, um Mensuralnotation setzen zu können.

Lyrics

Entspricht einer Stimme mit Gesangstext. Kümmert sich um den Satz des Gesangstextes auf einer Zeile.

DrumVoice

Der Stimmenkontext in einem Perkussionssystem.

FiguredBass

Der Kontext, in dem Generalbassziffern (`BassFigure`-Objekte) gesetzt werden, die in der `\figuremode`-Umgebung notiert werden.

TabVoice

Dieser Stimmenkontext wird in einer Tabulatur (`TabStaff`-Kontext) benutzt. Er wird normalerweise implizit erstellt.

CueVoice

Ein Stimmenkontext, der Noten in reduzierter Größe ausgibt und vor allem dazu da ist, Stichnoten zu setzen. Siehe auch Abschnitt 6.3.3 [Stichnoten formatieren], Seite 208. Wird normalerweise implizit erstellt, wenn Stichnoten gesetzt werden.

ChordNames

Ausgabe von Akkordsymbolen.

32.2 Kontexte erstellen und referenzieren

In Partituren mit einer Stimme und einem System werden die Kontexte normalerweise automatisch erstellt. In komplizierteren Partituren muss man sie aber direkt erstellen. Es gibt drei Möglichkeiten, Kontexte zu erstellen:

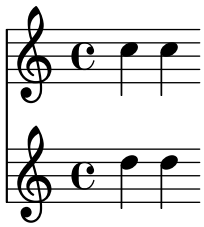
- Der einfachste Befehl ist `\new`. Er wird zusammen mit dem Kontextnamen vor einem musikalischen Ausdruck eingesetzt, etwa

  `\new` *`Kontext musik. Ausdruck`*

  wobei *Kontext* eine Kontextbezeichnung (wie `Staff` oder `Voice`) ist. Dieser Befehl erstellt einen neuen Kontext und beginnt mit der Auswertung von *musik. Ausdruck* innerhalb dieses Kontextes.

  Eine praktische Anwendung von `\new` ist eine Partitur mit vielen Systemen. Jede Stimme wird auf einem eigenen System notiert, das mit `\new Staff` begonnen wird.

  ```
  <<
    \new Staff { c4 c }
    \new Staff { d4 d }
  >>
  ```

  Der `\new`-Befehl kann den Kontext auch benennen:

  `\new` *`Kontext = ID musik. Ausdruck`*

  Dieser vom Benutzer definierte Name wird aber auch nur wirklich benutzt, wenn nicht vorher schon der gleiche Name definiert worden ist.

- Ähnlich dem `\new`-Befehl wird auch mit dem `\context`-Befehl ein musikalischer Ausdruck in einen Kontext umgeleitet. Diesem Kontext wird ein expliziter Name zugewiesen. Die Syntax lautet:

  `\context` *`Kontext = ID musik. Ausdruck`*

  Diese Art von Befehl sucht nach einem existierenden Kontext vom Typus *Kontext* mit der Bezeichnung *ID*. Wenn ein derartiger Kontext nicht existiert, wird ein neuer Kontext mit der entsprechenden Bezeichnung erstellt. Das ist nützlich, wenn auf den Kontext später zurückverwiesen werden soll. Um etwa Gesangstext zu einer Melodie hinzuzufügen, wird die Melodie in einem bezeichneten Kontext notiert:

  `\context Voice = "`**`Tenor`**`"` *`musik. Ausdruck`*

  sodass der Text an den Noten ausgerichtet werden kann:

  `\new Lyrics \lyricsto "`**`Tenor`**`"` *`Gesangstext`*
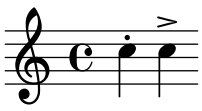
  Eine andere Möglichkeit für bezeichnete Kontexte ist es, zwei unterschiedliche musikalische Ausdrücke in einen Kontext zu verschmelzen. Im nächsten Beispiel werden Artikulationszeichen und Noten getrennt notiert:

  ```
  Noten = { c4 c4 }
  Artik = { s4-. s4-> }
  ```

  Dann werden sie kombiniert, indem sie dem selben `Voice`-Kontext zugewiesen werden:

  ```
  <<
    \new Staff \context Voice = "A" \Noten
  ```

```
  \context Voice = "A" \Artik
>>
```



Durch diesen Mechanismus ist es möglich eine Urtextausgabe zu erstellen, mit der optionalen Möglichkeit, bestimmte zusätzliche Artikulationszeichen zu den gleichen Noten hinzuzufügen und so eine editierte Ausgabe zu erhalten.

- Der dritte Befehl, um Kontexte zu erstellen, ist:

  ```
  \context Kontext musik. Ausdruck
  ```

  Dies entspricht dem `\context` mit `= ID`, aber hier wird ein beliebiger Kontext des Typs *Kontext* gesucht und der musikalische Ausdruck darin ausgewertet, unabhängig von der Bezeichnung, die dem Kontext gegeben wurde.

  Diese Variante wird bei musikalischen Ausdrücken benutzt, die auf verschiedenen Ebenen interpretiert werden können. Beispielsweise der `\applyOutput`-Befehl (siehe Abschnitt "Eine Funktion auf alle Layout-Objekte anwenden" in *Extending*). Ohne einen expliziten `\context` wird die Ausgabe normalerweise einem `Voice`-Kontext zugewiesen:

  ```
  \applyOutput #'Kontext #Funktion   % auf Voice anwenden
  ```

  Damit aber die Funktion auf `Score`- oder `Staff`-Ebene interpretiert wird, muss folgende Form benutzt werden:

  ```
  \applyOutput Score #Funktion
  \applyOutput Staff #Funktion
  ```

32.3 Kontexte am Leben halten

Kontexte werden normalerweise am ersten musikalischen Moment beendet, an dem sie nichts mehr zu tun haben. Ein `Voice`-Kontext stirbt also sofort, wenn keine Ereignisse mehr auftreten, `Staff`-Kontexte sobald alle in ihnen enthaltenen `Voice`-Kontexte keine Ereignisse mehr aufweisen usw. Das kann Schwierigkeiten ergeben, wenn auf frühere Kontexte verwiesen werden soll, die in der Zwischenzeit schon gestorben sind, beispielsweise wenn man Systemwechsel mit `\change`-Befehlen vornimmt, wenn Gesangstext einer Stimme mit dem `\lyricsto`-Befehl zugewiesen wird oder wenn weitere musikalische Ereignisse zu einem früheren Kontext hinzugefügt werden sollen.

Es gibt eine Ausnahme dieser Regel: genau ein `Voice`-Kontext innerhalb eines `Staff`-Kontextes oder in einer `<<...>>`-Konstruktion bleibt immer erhalten bis zum Ende des `Staff`-Kontextes oder der `<<...>>`-Konstruktion, der ihn einschließt, auch wenn es Abschnitte gibt, in der er nichts zu tun hat. Der Kontext, der erhalten bleibt ist immer der erste, der in der ersten enthaltenden `{...}`-Konstruktion angetroffen wird, wobei `<<...>>`-Konstruktionen ignoriert werden.

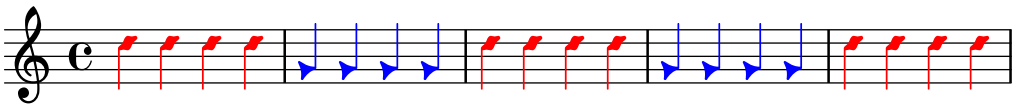
Jeder Kontext kann am Leben gehalten werden, indem man sicherstellt dass er zu jedem musikalischen Moment etwas zu tun hat. `Staff`-Kontexte werden am Leben gehalten, indem man sicherstellt, dass eine der enthaltenen Stimmen am Leben bleibt. Eine Möglichkeit, das zu erreichen, ist es, unsichtbare Pause zu jeder Stimme hinzuzufügen, die am Leben gehalten werden soll. Wenn mehrere Stimmen sporadisch benutzt werden sollen, ist es am sichersten, sie alle am Leben zu halten und sich nicht auf die Ausnahmeregel zu verlassen, die im vorigen Abschnitt dargestellt wurde.

Im folgenden Beispiel werden sowohl Stimme A als auch B auf diese Weise für die gesamte Dauer des Stückes am Leben gehalten.

```
musicA = \relative { d''4 d d d }
musicB = \relative { g'4 g g g }
keepVoicesAlive = {
  <<
    \new Voice = "A" { s1*5 }  % Keep Voice "A" alive for 5 bars
    \new Voice = "B" { s1*5 }  % Keep Voice "B" alive for 5 bars
  >>
}

music = {
  \context Voice = "A" {
    \voiceOneStyle
    \musicA
  }
  \context Voice = "B" {
    \voiceTwoStyle
    \musicB
  }
  \context Voice = "A" { \musicA }
  \context Voice = "B" { \musicB }
  \context Voice = "A" { \musicA }
}

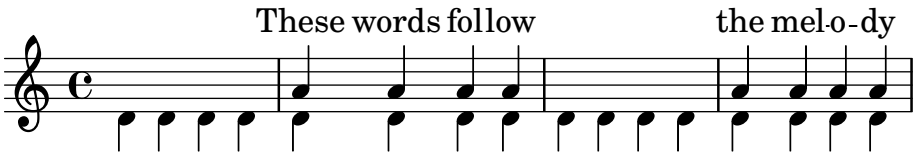
\score {
  \new Staff <<
    \keepVoicesAlive
    \music
  >>
}
```



Das nächste Beispiel zeigt eine Melodie, die zeitweise unterbrochen wird und wie man den entsprechenden Gesangstext mit ihr verknüpfen kann, indem man die Stimme am Leben hält. In wirklichen Situationen würden Begleitung und Melodie natürlich aus mehreren Abschnitten bestehen.

```
melody = \relative { a'4 a a a }
accompaniment = \relative { d'4 d d d }
words = \lyricmode { These words fol -- low the mel -- o -- dy }
\score {
  <<
    \new Staff = "music" {
      <<
        \new Voice = "melody" {
          \voiceOne
          s1*4  % Keep Voice "melody" alive for 4 bars
        }
        {
          \new Voice = "accompaniment" {
            \voiceTwo
```

```
            \accompaniment
          }
          <<
            \context Voice = "melody" { \melody }
            \context Voice = "accompaniment" { \accompaniment }
          >>
          \context Voice = "accompaniment" { \accompaniment }
          <<
            \context Voice = "melody" { \melody }
            \context Voice = "accompaniment" { \accompaniment }
          >>
        }
      >>
    }
    \new Lyrics \with { alignAboveContext = "music" }
    \lyricsto "melody" { \words }
  >>
}
```

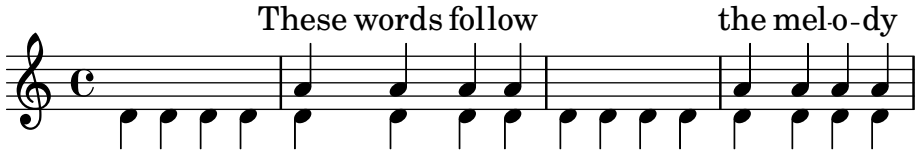


Eine Alternative, die in manchen Umständen besser geeignet sein kann, ist es, einfach unsichtbare Pausen einzufügen, um die Melodie mit der Begleitung passend auszurichten:

```
melody = \relative {
  s1  % skip a bar
  a'4 a a a
  s1  % skip a bar
  a4 a a a
}
accompaniment = \relative {
  d'4 d d d
  d4 d d d
  d4 d d d
  d4 d d d
}
words = \lyricmode { These words fol -- low the mel -- o -- dy }

\score {
  <<
    \new Staff = "music" {
      <<
        \new Voice = "melody" {
          \voiceOne
          \melody
        }
        \new Voice = "accompaniment" {
          \voiceTwo
          \accompaniment
        }
```

```
      >>
    }
    \new Lyrics \with { alignAboveContext = "music" }
    \lyricsto "melody" { \words }
  >>
}
```



32.4 Umgebungs-Plugins verändern

Notationskontexte (wie `Score` oder `Staff`) speichern nicht nur Eigenschaften, sie enthalten auch Plugins („engraver“ genannt), die die einzelnen Notationselemente erstellen. Ein `Voice`-Kontext enthält beispielsweise einen `Note_heads_engraver`, der die Notenköpfe erstellt, und ein `Staff`-Kontext einen `Key_engraver`, der die Vorzeichen erstellt.
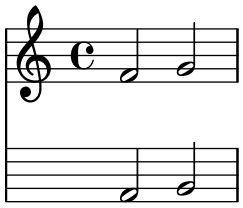
Eine vollständige Erklärung jedes Plugins findet sich in Referenz der Interna: ↦ Translation ↦ Engravers. Alle Kontexte sind erklärt in Referenz der Interna: ↦ Translation ↦ Context, wobei die in diesem Kontext vorkommenden Engraver aufgelistet sind.

Es kann teilweise nötig sein, diese Engraver umzupositionieren. Das geschieht, indem man einen neuen Kontext mit `\new` oder `\context` beginnt und ihn dann verändert:

```
\new context \with {
  \consists ...
  \consists ...
  \remove ...
  \remove ...
  etc.
}
{
  ..Noten..
}
```

`...` steht hier für die Bezeichnung des Engravers. `\consists` fügt einen Engraver hinzu und `\remove` entfernt ihn. Es folgt ein einfaches Beispiel, in dem der `Time_signature_engraver` (Engraver für den Takt) und der `Clef_engraver` (Engraver für den Schlüssel) aus dem `Staff`-Kontext entfernt werden:

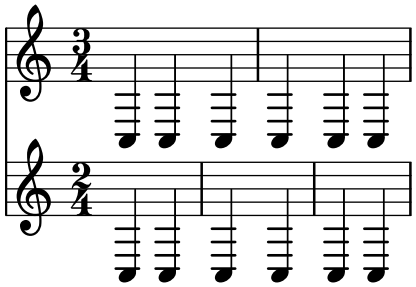
```
<<
  \new Staff \relative {
    f'2 g
  }
  \new Staff \with {
     \remove Time_signature_engraver
     \remove Clef_engraver
  } \relative {
    f'2 g2
  }
>>
```



Das zweite Notensystem enthält keine Taktangabe und keinen Notenschlüssel. Das ist eine recht brutale Methode, Objekte zu verstecken, weil es sich auf das gesamte System auswirkt. Diese Methode beeinflusst auch die Platzaufteilung, was erwünscht sein kann. Vielfältigere Methoden, mit denen Objekte unsichtbar gemacht werden können, finden sich in Abschnitt "Sichtbarkeit und Farbe von Objekten" in *Handbuch zum Lernen*.

Das nächste Beispiel zeigt eine Anwendung in der Praxis. Taktstriche und Taktart werden normalerweise in einer Partitur synchronisiert. Das geschieht durch `Timing_translator` und `Default_bar_line_engraver`. Diese Plugins sorgen sich um die Verwaltung der Taktzeiten und die Stelle innerhalb des Taktes, zu dem eine Note erscheint usw. Indem man diese Engraver aus dem `Score`-Kontext in den `Staff`-Kontext verschiebt, kann eine Partitur erstellt werden, in welcher jedes System eine unterschiedliche Taktart hat:

```
\score {
  <<
    \new Staff \with {
      \consists Timing_translator
    } {
        \time 3/4
        c4 c c c c c
    }
  \new Staff \with {
    \consists Timing_translator
  } {
      \time 2/4
      c4 c c c c c
  }
>>
\layout {
  \context {
    \Score
    \remove Timing_translator
    }
  }
}
```



Bekannte Probleme und Warnungen

Die Reihenfolge, in der die Engraver definiert werden, ist die Reihenfolge, in welcher sie aufgerufen werden, um ihre Verarbeitung vorzunehmen. Normalerweise spielt die Reihenfolge, in welcher die Engraver angegeben werden, keine Rolle, aber in einigen Spezialfällen ist die Reihenfolge sehr wichtig. Das kann beispielsweise vorkommen, wenn ein Engraver eine Eigenschaft erstellt und ein anderer von ihr liest, oder ein Engraver erstellt ein Grob und ein anderer wertet es aus.

Folgende Reihenfolgen müssen beachtet werden:

- der `Bar_engraver` muss normalerweise zuerst kommen,
- der `New_fingering_engraver` muss vor dem `Script_column_engraver` kommen,
- der `Timing_translator` muss vor dem `Bar_number_engraver` kommen.

Siehe auch

Installlierte Dateien: `ly/engraver-init.ly`.

32.5 Die Standardeinstellungen von Kontexten ändern

Kontext- und Grob-Eigenschaften können mit den Befehlen `\set` und `\override` verändert werden, wie beschrieben in Kapitel 34 [Eigenschaften verändern], Seite 601. Diese Befehle erstellen musikalische Ereignisse, damit die Veränderungen zum Zeitpunkt der Verarbeitung in den Noten erscheinen.

Dieser Abschnitt hingegen erklärt, wie man die *Standardwerte* von Kontext- und Grob-Eigenschaften zum Zeitpunkt, an dem der Kontext erstellt wird, verändert. Es gibt hierzu zwei Möglichkeiten. Die eine verändert die Standardeinstellungen aller Kontexte eines bestimmten Typs, die andere verändert die Standardwerte nur eines bestimmten Kontextes.

32.5.1 Alle Kontexte des gleichen Typs verändern

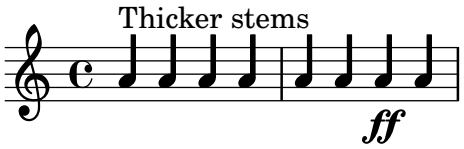
Die Kontexteinstellungen, die standardmäßig in `Score`, `Staff`, `Voice` und anderen Kontexten eingesetzt werden, können in einer `\context`-Umgebung innerhalb einer beliebigen `\layout`-Umgebung spezifiziert werden. Die `\layout`-Umgebung sollte innerhalb der `\score` (Partitur) stehen, auf die sie sich bezieht, nach den Noten.

```
\layout {
  \context {
    \Voice
    [Kontexteinstellungen für alle Voice-Kontexte]
  }
  \context {
    \Staff
    [Kontexteinstellungen für alle Staff-Kontexte]
  }
}
```

Folgende Einstellungstypen können angegeben werden:
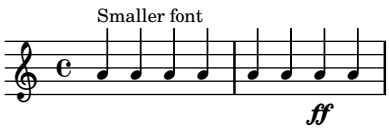
- Ein `\override`-Befehl, aber ohne die Kontextbezeichnung:

```
\score {
  \relative {
    a'4^"Thicker stems" a a a
    a4 a a\ff a
  }
  \layout {
    \context {
      \Staff
      \override Stem.thickness = #4.0
    }
  }
}
```



- Eine Kontexteigenschaft kann direkt gesetzt werden:

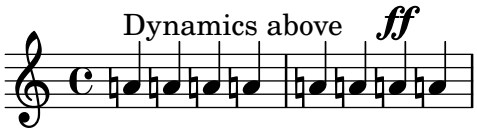
```
\score {
  \relative {
    a'4^"Smaller font" a a a
    a4 a a\ff a
  }
  \layout {
    \context {
      \Staff
      fontSize = #-4
    }
  }
}
```



- Ein vordefinierter Befehl wie etwa `\dynamicUp` oder ein musikalischer Ausdruck wie `\accidentalStyle dodecaphonic`:

```
\score {
  \relative {
    a'4^"Dynamics above" a a a
    a4 a a\ff a
  }
  \layout {
    \context {
      \Voice
      \dynamicUp
    }
    \context {
      \Staff
```

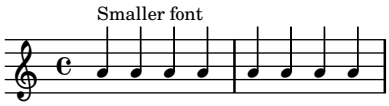
```
      \accidentalStyle dodecaphonic
    }
  }
}
```



- Eine vom Benutzer definierte Variable, die eine \with-Umgebung enthält; zu Details der \with-Umgebung, siehe Abschnitt 32.5.2 [Nur einen bestimmten Kontext verändern], Seite 591.

```
StaffDefaults = \with {
  fontSize = #-4
}
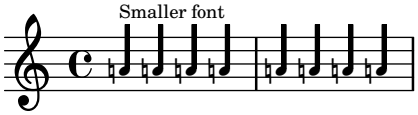
\score {
  \new Staff {
    \relative {
      a'4^"Smaller font" a a a
      a4 a a a
    }
  }
  \layout {
    \context {
      \Staff
      \StaffDefaults
    }
  }
}
```



Befehle, die die Eigenschaften verändern, können in einer \layout-Umgebung platziert werden, ohne von einer \context-Umgebung eingeschlossen zu werden. Derartige Einstellungen verhalten sich äquivalet wie Eigenschaftsveränderungen, die zu jedem Beginn eines bestimmten Kontextes angegeben werden. Wenn kein Kontext angegeben wird, wird *jeder* Kontext auf der untersten Ebene beeinflusst, siehe auch Abschnitt 32.1.4 [Unterste Kontexte – Stimmen], Seite 581. Die Syntax eines Befehls zu Einstellung von Eigenschaftseinstellungen in einer \layout-Umgebung ist die gleiche wie für den Befehl direkt zwischen den Noten geschrieben.

```
\score {
  \new Staff {
    \relative {
      a'4^"Smaller font" a a a
      a4 a a a
    }
  }
  \layout {
    \accidentalStyle dodecaphonic
    \set fontSize = #-4
```

```
    \override Voice.Stem.thickness = #4.0
  }
}
```



32.5.2 Nur einen bestimmten Kontext verändern

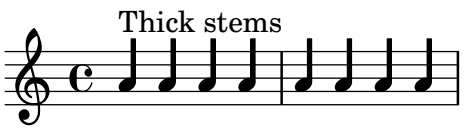
Die Kontexteigenschaften nur eines bestimmten Kontextes können mit einer `\with`-Umgebung geändert werden. Alle anderen Vorkommen des gleichen Kontexts behalten ihre Standardeinstellungen, möglicherweise durch Einstellungn in `\layout`-Umgebungen verändert. Die `\with`-Umgebung muss direkt nach dem Befehl `\new` *Kontext-Typ* gesetzt werden:

```
\new Staff
\with {
  [enthält Einstellungen nur für diesen spezifischen Kontext]
} {
...
}
```

Folgende Arten von Einstellungen können angegeben werden:

- Ein `\override`-Befehl, dessen Kontextbezeichnung ausgelassen wird:

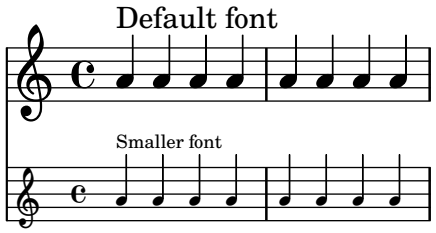
```
\score {
  \new Staff {
    \new Voice
    \with {
      \override Stem.thickness = #4.0
    }
    {
      \relative {
        a'4^"Thick stems" a a a
        a4 a a a
      }
    }
  }
}
```



- Eine Kontexeigenschaft direkt einstellen:

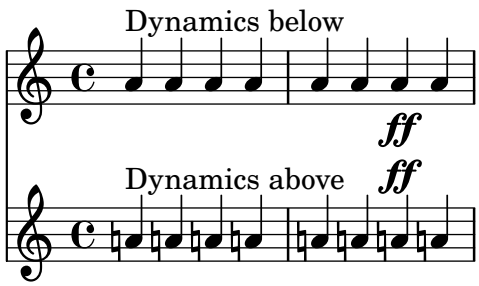
```
\score {
  <<
    \new Staff {
      \relative {
        a'4^"Default font" a a a
        a4 a a a
      }
    }
    \new Staff
    \with {
```

```
      fontSize = #-4
    } {
      \relative {
        a'4^"Smaller font" a a a
        a4 a a a
      }
    }
  >>
}
```



- Ein vordefinierter Befehl wie etwa \dynamicUp

```
\score {
  <<
    \new Staff {
      \new Voice {
        \relative {
          a'4^"Dynamics below" a a a
          a4 a a\ff a
        }
      }
    }
    \new Staff
    \with { \accidentalStyle dodecaphonic }
    {
      \new Voice
      \with { \dynamicUp }
      {
        \relative {
          a'4^"Dynamics above" a a a
          a4 a a\ff a
        }
      }
    }
  >>
}
```



32.5.3 Rangfolge von Kontextwerten

Der Wert einer Eigenschaft, die zu einer bestimmten Zeit aktiv ist, wird wie folgend bestimmt:

- wenn ein `\override`- oder `\set`-Befehl in der Eingabe aktiv ist, wird dieser Wert benützt,
- ansonsten wird der Standardwert aus einer `\with`-Umgebung zu Beginn des Kontextes benützt,
- ansonsten wied der Standardwert aus der letzten passenden `\context`-Umgebung in der letzten `\layout`-Umgebung benützt,
- ansonsten wird der Standardwert von LilyPond eingesetzt.

Siehe auch

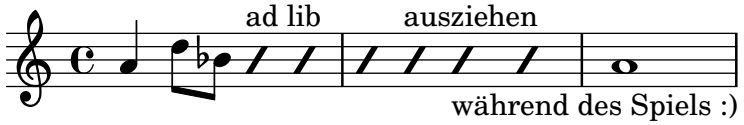
Handbuch zum Lernen: Abschnitt “Kontexteigenschaften verändern” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 32.1 [Was sind Kontexte?], Seite 580, Abschnitt 32.1.4 [Unterste Kontexte – Stimmen], Seite 581, Abschnitt 34.2 [Der `\set`-Befehl], Seite 601, Abschnitt 34.3 [Der `\\override`-Befehl], Seite 603, Abschnitt 26.1 [Die `\layout`-Umgebung], Seite 532.

32.6 Neue Kontexte definieren

Bestimme Kontexte, wie `Staff` oder `Voice`, werden erstellt, indem man sie mit einer Musikumgebung aufruft. Es ist aber auch möglich, eigene neue Kontexte zu definieren, in denen dann unterschiedliche Engraver benutzt werden.

Das folgende Beispiel zeigt, wie man etwa `Voice`-Kontexte von Grund auf neu bauen kann. Ein derartiger Kontext ähnelt `Voice`, es werden aber nur zentrierte Schrägstriche als Notenköpfe ausgegeben. Das kann benutzt werden, um Improvisation in Jazzmusik anzuzeigen.



Diese Einstellungen werden innerhalb der `\context`-Umgebung innerhalb der `\layout`-Umgebung definiert:

```
\layout {
  \context {
    ...
  }
}
```

Der Beispielcode des folgenden Abschnittest muss anstelle der Punkte im vorigen Beispiel eingesetzt werden.

Zuerst ist es nötig eine Bezeichnung für den neuen Kontext zu definieren:

```
\name ImproVoice
```

Weil dieser neue Kontext ähnlich wie `Voice` ist, sollen die Befehle, die in `Voice`-Kontexten funktionieren, auch in dem neuen Kontext funktionieren. Das wird erreicht, indem der Kontext als Alias `Voice` erhält:

```
\alias Voice
```

Der Kontext gibt Noten und Text aus, darum müssen wir die Engraver hinzufügen, die für diese Aktionen zuständig sind:

```
\consists Note_heads_engraver
\consists Text_engraver
```

aber die Noten sollen nur auf der mittleren Linie ausgegeben werden:

```
\consists Pitch_squash_engraver
squashedPosition = #0
```

Der `Pitch_squash_engraver` verändert Notenköpfe (die vom `Note_heads_engraver` erstellt werden) und setzt ihre vertikale Position auf den Wert von `squashedPosition`, in diesem Fall ist das die Mittellinie.

Die Noten sehen wie ein Querstrich aus und haben keine Hälse:

```
\override NoteHead.style = #'slash
\hide Stem
```

Alle diese Engraver müssen zusammenarbeiten, und das wird erreicht mit einem zusätzlichen Plugin, das mit dem Befehl `\type` gekennzeichnet werden muss. Dieser Typ solle immer `Engraver_group` lauten:

```
\type Engraver_group
```

Alles zusammen haben wir folgende Einstellungen:

```
\context {
  \name ImproVoice
  \type Engraver_group
  \consists Note_heads_engraver
  \consists Text_engraver
  \consists Pitch_squash_engraver
  squashedPosition = #0
  \override NoteHead.style = #'slash
  \hide Stem
  \alias Voice
}
```

Kontexte sind hierarchisch. Wie wollen, dass `ImproVoice` sich als Unterkontext von `Staff` erkennt, wie eine normale Stimme. Darum wird die Definition von `Staff` mit dem `\accepts`-Befehl verändert:

```
\context {
  \Staff
  \accepts ImproVoice
}
```

Das Gegenteil von `\accepts` ist `\denies` (verbietet), was manchmal gebraucht werden kann, wenn schon existierende Kontext-Definitionen wieder benutzt werden sollen.

Beide Definitionen müssen in die `\layout`-Umgebung geschrieben werden:

```
\layout {
  \context {
    \name ImproVoice
    ...
  }
  \context {
    \Staff
    \accepts ImproVoice
  }
}
```

Jetzt kann die Notation zu Beginn des Abschnitts folgendermaßen notiert werden:

```
\relative {
  a'4 d8 bes8
```
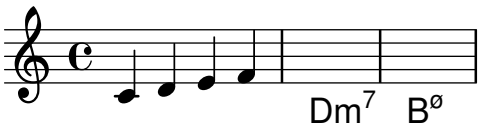
```
  \new ImproVoice {
    c4^"ad lib" c
    c4 c^"ausziehen"
    c c_"während des Spielens :)"
  }
  a1
}
```

32.7 Reihenfolge des Kontextlayouts
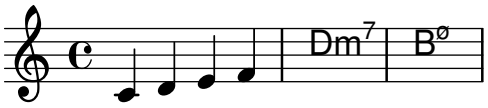
Kontexte werden in einer Systemgruppe normalerweise von oben nach unten positioniert in der Reihenfolge, wie sie in der Quelldatei auftreten. Wenn Kontext verschachtelt ewrden, enthält der äußere Kontexte geschachtelte innere Kontexte, wie in der Quelldatei angegeben, vorausgesetzt, die inneren Kontexte befinden sich auch in der „accepts“-Liste. Geschachtelte Kontexte, die nicht in dieser „accepts“-Liste enthalten sind, werden unterhalb des äußeren Kontextes neu positioniert, anstatt innerhalb von ihm gesetzt zu werden.

Die „accepts“-Liste eines Kontextes kann mit dem Befehlen `\accepts` und `\denies` verändert werden. `\accepts` fügt einen Kontext zur „accepts“-Liste, und `\denies` entfernt einen Kontext aus der Liste. Akkordbezeichnungen sollen beispielsweise normalerweise nicht innerhalb eines `Staff`-Kontextes geschachtelt werden, sodass der `ChordNames`-Kontext nicht automatisch in der „accepts“-Liste des `Staff`-Kontextes geführt ist. Wenn er aber benötigt wird, kann er hinzugefügt werden:

```
\score {
  \new Staff {
    c' d' e' f'
    \chords { d1:m7 b1:min7.5- }
  }
}
```




```
\score {
  \new Staff {
    c' d' e' f'
    \chords { d1:m7 b1:min7.5- }
  }
  \layout {
    \context {
      \Staff
      \accepts ChordNames
    }
  }
}
```



`\denies` wird vorrangig eingesetzt, wenn ein neuer Kontext basierend auf einem existierenden erstellt wird, aber sein Schachtelungsverhalten sich unterscheidet. Der `VaticanaStaff`-Kontext

beispielsweise basiert auf dem `Staff`-Kontext, hat aber den `VaticanaVoice`-Kontext anstellt des `Voice`-Kontexts in seiner „accepts“-Liste.

Zur Erinnerung: ein Kontext wird automatisch erstellt, wenn ein Befehl auftritt, der in den aktuellen Kontexten nicht enthalten sein kann. Dass kann zu unerwarteten neuen Systemgruppen oder Partituren führen.

Manchmal soll ein Kontext nur für einen kurzen Moment existieren, ein gutes Beispiel etwa ein System für ein Ossia. Das wird normalerweise erreicht, indem man die Kontextdefinition an der richtigen Stelle parallel mit dem existierenden Abschnitt der Hauptnoten anlegt. Standardmäßig wird der neue Kontext unter den existierenden Kontexten angelegt. Um ihn aber über dem Kontext mit der Bezeichnung „Hauptstimme“ zu positionieren, sollte er folgenderweise erstellt werden:

```
\new Staff \with { alignAboveContext = "Hauptstimme" }
```

Eine ähnliche Situation entsteht, wenn man einen zeitweiligen Gesangstext in einem Layout mit mehreren Notensystemen anlegen und positionieren will, etwa wenn eine zweite Strophe zu einem wiederholten Abschnitt in einem `ChoirStaff` hinzugefügt wird. Standardmäßig wird der neue Text unter dem untersten System angelegt. Wenn der Gesangstext mit der Eigenschaft `alignBelowContext` definiert wird, kann er korrekt unter dem (bezeichneten) Gesangstext positioniert werden, der die erste Strophe enthält.

Beispiele, die diese Neuordnung von temporären Kontexten zeigen, finden sich an anderen Stellen; siehe Abschnitt “Musikalische Ausdrücke ineinander verschachteln” in *Handbuch zum Lernen*, Abschnitt 6.2 [Einzelne Systeme verändern], Seite 191, und Abschnitt 9.2 [Techniken für die Gesangstextnotation], Seite 266.

Siehe auch

Handbuch zum Lernen: Abschnitt “Musikalische Ausdrücke ineinander verschachteln” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 6.2 [Einzelne Systeme verändern], Seite 191, Abschnitt 9.2 [Techniken für die Gesangstextnotation], Seite 266.

Handbuch zur Benutzung: Abschnitt “Ein zusätzliches System erscheint” in *Anwendungsbenutzung*.

Installierte Dateien: `ly/engraver-init.ly`.
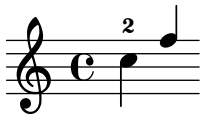
33 Die Referenz der Programminterna erklärt

33.1 Zurechtfinden in der Programmreferenz

Arbeit mit der Referenz der Interna soll hier an einigen Beispiel illustriert werden. Die Referenz der Interna existiert nur auf Englisch, darum sind auch die Beispiele dieses Abschnittes nicht übersetzt.

Folgende Aufgabe wird bearbeitet: Der Fingersatz aus dem Beispiel unten soll verändert werden:

```
c-2
\stemUp
f
```



In der Dokumentation über Fingersatz (in Abschnitt 7.1.2 [Fingersatzanweisungen], Seite 215) gibt es folgenden Abschnitt:

> **Siehe auch:**
>
> Referenz der Interna: Abschnitt “Fingering” in *Referenz der Interna*.

Die Referenz der Interna gibt es als HTML-Dokument. Sie sollten sie als HTML-Dokument lesen, entweder online oder indem Sie die HTML-Dokumentation herunterladen. Dieser Abschnitt ist sehr viel schwieriger zu verstehen, wenn Sie die PDF-Version verwenden.

Gehen Sie über diesen Link zum Abschnitt Abschnitt “Fingering” in *Referenz der Interna*. Oben auf der Seite findet sich:

> Fingering objects are created by the following engraver(s): Abschnitt “Fingering_engraver” in *Referenz der Interna* and Abschnitt “New_fingering_engraver” in *Referenz der Interna*.

Indem Sie die Links in der Referenz der Interna folgen, können Sie verfolgen, wie LilyPond intern arbeitet:

- Abschnitt “Fingering” in *Referenz der Interna*: Abschnitt “Fingering” in *Referenz der Interna* objects are created by the following engraver(s): Abschnitt “Fingering_engraver” in *Referenz der Interna*.
- Abschnitt “Fingering_engraver” in *Referenz der Interna*: Music types accepted: Abschnitt “fingering-event” in *Referenz der Interna*
- Abschnitt “fingering-event” in *Referenz der Interna*: Music event type `fingering-event` is in Music expressions named Abschnitt “FingeringEvent” in *Referenz der Interna*

Fingersatz-Objekte werden also durch den `Fingering_engraver` erstellt, welcher folgende Musikereignistypen akzeptiert: `fingering-event`. Ein Musikereignis vom Typ `fingering-event` ist ein musikalischer Ausdruck mit der Bezeichnung Abschnitt “FingeringEvent” in *Referenz der Interna*.

Dieser Pfad geht genau die entgegengesetzte Richtung von LilyPonds Wirkungsweise: er beginnt bei der graphischen Ausgabe und arbeitet sich voran zur Eingabe. Man könnte auch mit einem Eingabe-Ereignis starten und dann die Links zurückverfolgen, bis man zum Ausgabe-Objekt gelangt.

Die Referenz der Interna kann auch wie ein normales Dokument durchsucht werden. Sie enthält Kapitel über `Music definitions`, über Abschnitt “Translation” in *Referenz der Interna* und Abschnitt “Backend” in *Referenz der Interna*. Jedes Kapitel listet alle die Definitionen und Eigenschaften auf, die benutzt und verändert werden können.

33.2 Layout-Schnittstellen

Die HTML-Seite, die im vorigen Abschnitt betrachtet wurde, beschreibt ein Layoutobjekt mit der Bezeichnung `Fingering`. Ein derartiges Objekt ist ein Symbol in der Partitur. Es hat Eigenschaften, die bestimmte Zahlen speichern (wie etwa Dicke und Richtung), aber auch Weiser auf verwandte Objekte. Ein Layoutobjekt wird auch als „Grob“ bezeichnet, die Abkürzung für *Gr*aphisches *Ob*jekt. Mehr Information zu Grobs findet sich in Abschnitt “grob-interface” in *Referenz der Interna*.

Die Seite zu `Fingering` enthält Definitionen für das `Fingering`-Objekt. Auf der Seite steht etwa:

> `padding` (dimension, in staff space):
>
> `0.5`

was bedeutet, dass der Abstand zu anderen Objekten mindestens 0.5 Notenlinienabstände beträgt.

Jedes Layoutobjekt kann mehrere Funktionen sowohl als typographisches als auch als Notationselement einnehmen. Das Fingersatzobjekt beispielsweise hat folgende Aspekte:

- Seine Größe ist unabhängig von der horizontalen Platzaufteilung, anders als etwa bei Legatobögen.
- Es handelt sich um Text, normalerweise sehr kurz.
- Dieser Text wird durch ein Glyph einer Schriftart gesetzt, anders als bei Legatobögen.
- Der Mittelpunkt des Symbols sollte horizontal mit dem Mittelpunkt des Notenkopfes ausgerichtet werden.
- Vertikal wird das Objekt neben die Note und das Notensystem gesetzt.
- Die vertikale Position wird auch mit anderen Textelementen abgeglichen.

Jeder dieser Aspekte findet sich in sogenannten Schnittstellen (engl. interface), die auf der Abschnitt “Fingering” in *Referenz der Interna*-Seite unten aufgelistet sind:

> This object supports the following interfaces: Abschnitt “item-interface” in *Referenz der Interna*, Abschnitt “self-alignment-interface” in *Referenz der Interna*, Abschnitt “side-position-interface” in *Referenz der Interna*, Abschnitt “text-interface” in *Referenz der Interna*, Abschnitt “text-script-interface” in *Referenz der Interna*, Abschnitt “font-interface” in *Referenz der Interna*, Abschnitt “finger-interface” in *Referenz der Interna*, and Abschnitt “grob-interface” in *Referenz der Interna*.

Ein Klick auf einen der Links öffnet die Seite der entsprechenden Schnittstelle. Jede Schnittstelle hat eine Anzahl von Eigenschaften. Einige sind nicht vom Benutzer zu beeinflussen („interne Eigenschaften“), andere aber können verändert werden.

Es wurde immer von einem `Fingering`-Objekt gesprochen, aber eigentlich handelt es sich nicht um sehr viel. Die Initialisierungsdatei `scm/define-grobs.scm` zeigt den Inhalt dieses „Objekts“ (zu Information, wo diese Dateien sich finden siehe Abschnitt “Mehr Information” in *Handbuch zum Lernen*):

```
(Fingering
  . ((padding . 0.5)
     (avoid-slur . around)
     (slur-padding . 0.2)
     (staff-padding . 0.5)
     (self-alignment-X . 0)
     (self-alignment-Y . 0)
     (script-priority . 100)
     (stencil . ,ly:text-interface::print)
     (direction . ,ly:script-interface::calc-direction)
```
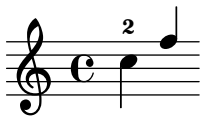
```
(font-encoding . fetaText)
(font-size . -5)  ; don't overlap when next to heads.
(meta . ((class . Item)
(interfaces . (finger-interface
               font-interface
               text-script-interface
               text-interface
               side-position-interface
               self-alignment-interface
               item-interface))))))
```

Wie man sehen kann, ist das Fingersatzobjekt nichts anderes als eine Ansammlung von Variablen, und die Internetseite der Referenz der Interna ist direkt aus diesen Anweisungen generiert.

33.3 Die Grob-Eigenschaften

Die Position der **2** aus dem Beispiel unten soll also geändert werden:

```
c-2
\stemUp
f
```



Weil die **2** vertikal an der zugehörigen Note ausgerichtet ist, müssen wir uns mit der Schnittstelle auseinander setzen, die diese Positionierung veranlasst. Das ist hier `side-position-interface`. Auf der Seite für diese Schnittstelle heißt es:

> `side-position-interface`
>
> Position a victim object (this one) next to other objects (the support). The property `direction` signifies where to put the victim object relative to the support (left or right, up or down?)

Darunter wird die Variable `padding` (Verschiebung) beschrieben:

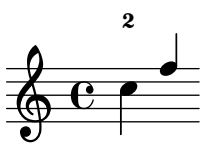
> `padding` (dimension, in staff space)
>
> Add this much extra space between objects that are next to each other.

Indem man den Wert von `padding` erhöht, kann die Fingersatzanweisung weiter weg von der Note gesetzt werden. Dieser Befehl beispielsweise fügt drei Notenlinienzwischenräume zwischen die Zahl und den Notenkopf:

```
\once \override Voice.Fingering.padding = #3
```

Wenn dieser Befehl in den Quelltext eingefügt wird, bevor der Fingersatz notiert ist, erhält man folgendes:

```
\once \override Voice.Fingering.padding = #3
c-2
\stemUp
f
```



In diesem Fall muss die Veränderung speziell für den `Voice`-Kontext definiert werden. Das kann auch aus der Referenz der Interna entnommen werden, da die Seite des Abschnitt "Fingering_engraver" in *Referenz der Interna* schreibt:

> Fingering_engraver is part of contexts: . . . Abschnitt "Voice" in *Referenz der Interna*

33.4 Benennungskonventionen

Die Bezeichnungen für Funktionen, Variablen, Engraver und Objekte folgen bestimmten Regeln:

- Scheme-Funktionen: kleinbuchstaben-mit-bindestrichen
- Scheme-Funktionen: ly:plus-scheme-stil
- Musikalische Ereignisse, Musikklassen und Musikeigenschaften: wie-scheme-funktionen
- Grob-Schnittstellen: scheme-stil
- backend-Eigenschaften: scheme-stil (aber X und Y)
- Kontexte: Großbuchstabe, oder GroßbuchstabeZwischenWörtern (CamelCase)
- Kontext-Eigenschaften: kleinbuchstabeMitFolgendenGroßbuchstaben
- Engraver: Großbuchstabe_gefolgt_von_kleinbuchstaben_mit_unterstrichen

34 Eigenschaften verändern

34.1 Grundlagen zum Verändern von Eigenschaften

Jeder Kontext ist verantwortlich für die Erstellung bestimmter graphischer Objekte. Die Einstellungen für diese Objekte werden auch in dem Kontext gespeichert. Wenn man diese Einstellungen verändert, kann die Erscheinung der Objekte geändert werden.

Es gibt zwei unterschiedliche Eigenschaftenarten, die in Kontexten gespeichert werden: Kontexteigenschaften und Grob-Eigenschaften. Kontexteigenschaften sind Eigenschaften, die sich auf den gesamten Kontext beziehen und seine Darstellung beinflussen. Grob-Eigenschaften dagegen wirken sich nur auf bestimmte graphische Objekte aus, die in einem Kontext dargestellt werden.

Die `\set`- und `\unset`-Befehle werden benutzt, um die Werte von Kontexteigenschaften zu ändern. Die Befehle `\override` und `\revert` hingegen verändern die Werte von Grob-Eigenschaften.

Siehe auch

Referenz der Interna: Abschnitt “OverrideProperty” in *Referenz der Interna*, Abschnitt “RevertProperty” in *Referenz der Interna*, Abschnitt “PropertySet” in *Referenz der Interna*, Abschnitt “Backend” in *Referenz der Interna*, Abschnitt “All layout objects” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

Das Back-end ist nicht sehr streng bei der Überprüfung der Typen von Objekteigenschaften. Auf sich selbst verweisende Bezüge in Scheme-Werten der Eigenschaften können Verzögerung oder einen Absturz des Programms hervorrufen.

34.2 Der `\set`-Befehl

Jeder Kontext kann unterschiedliche *Eigenschaften* besitzen, Variablen, die in diesem Kontext definiert sind. Sie können während der Interpretation des Kontextes verändert werden. Hierzu wird der `\set`-Befehl eingesetzt:
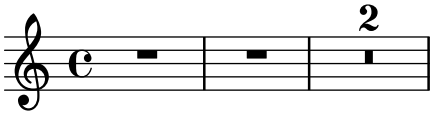
```
\set Kontext.Eigenschaft = #Wert
```

Wert ist ein Scheme-Objekt, weshalb ihm `#` vorangestellt werden muss.

Kontexteigenschaften werden üblicherweise mit `kleinGroßbuchstabe` benannt. Sie kontrollieren vor allem die Übersetzung von Musik in Notation, wie etwa `localAlterations`, welche bestimmt, wann ein Taktstrich gesetzt werden muss. Kontexteigenschaften können ihren Wert mit der Zeit ändern, während eine Notationsdatei interpretiert wird. Ein gutes Beispiel dafür ist `measurePosition`, was die Position der Noten im Takt angibt. Kontexteigenschaften werden mit dem `\set`-Befehl verändert.
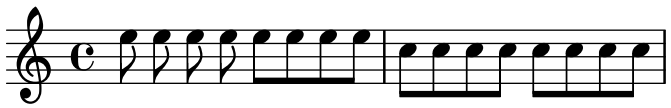
Mehrtaktpausen etwa können in einen Takt zusammengefasst werden, wenn die Kontexteigenschaft `skipBars` (Takte überspringen) auf `#t` (wahr) gesetzt wird:

```
R1*2
\set Score.skipBars = ##t
R1*2
```



Wenn das *Kontext*-Argument ausgelassen wird, bezieht sich der Befehl auf den gerade aktiven unterstmöglichen Kontext, üblicherweise `ChordNames`, `Voice` oder `Lyrics`.
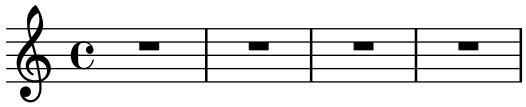
```
\set Score.autoBeaming = ##f
\relative {
  e''8 e e e
  \set autoBeaming = ##t
  e8 e e e
} \\
\relative {
  c''8 c c c c8 c c c
}
```



Die Änderung wird zur Laufzeit während der Musik interpretiert, sodass diese Einstellung sich nur auf die zweite Gruppe von Achteln auswirkt.

Dabei gilt zu beachten, dass der unterste Kontext nicht immer die Eigenschaft enthält, die verändert werden soll. Wenn man beispielsweise `skipBars` aus dem oberen Beispiel ohne Angabe des Kontextes zu verändern sucht, hat der Befehl keine Auswirkung, weil er sich auf den `Voice`-Kontext bezieht, die Eigenschaft sich aber im `Score`-Kontext befindet:

```
R1*2
\set skipBars = ##t
R1*2
```
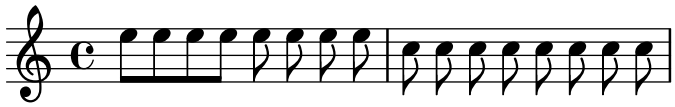


Kontexte sind hierarchisch angeordnet. Wenn ein übergeordneter Kontext angegeben wird, etwa `Staff`, dann beziehen sich die Änderungen auf alle Stimmen (`Voice`), die in diesem Kontext enthalten sind.

Es gibt auch einen `\unset`-Befehl:

```
\unset Kontext.Eigenschaft
```

der bewirkt, dass die vorgenommenen Definitionen für *Eigenschaft* entfernt werden. Dieser Befehl macht nur Einstellungen im richtigen Kontext rückgängig. Wenn also im `Staff`-Kontext die Bebalkung ausgeschaltet wird:

```
\set Score.autoBeaming = ##t
\relative {
  \unset autoBeaming
  e''8 e e e
  \unset Score.autoBeaming
  e8 e e e
} \\
\relative {
  c''8 c c c c8 c c c
}
```
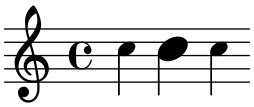


Wie für \set muss das *Kontext*-Argument für den untersten Kontext nicht mitangegeben werden. Die zwei Versionen

```
\set Voice.autoBeaming = ##t
\set autoBeaming = ##t
```

verhalten sich gleich, wenn die gegenwärtige Basis der Voice-Kontext ist.

Einstellungen, die nur einmal vorgenommen werden sollen, können mit \once notiert werden, etwa:

```
c''4
\once \set fontSize = #4.7
c''4
c''4
```



Eine vollständige Beschreibung aller vorhandenen Kontexteigenschaften findet sich in der Referenz der Interna, siehe „Translation ↦ Tunable context properties“.

Siehe auch

Internals Reference:

Abschnitt “Tunable context properties” in *Referenz der Interna*.

34.3 Der \\override-Befehl

Es gibt eine besondere Art von Kontexteigenschaft: die Grob-Beschreibung. Grob-Beschreibungen werden mit GroßGroßbuchstabe benannt. Sie enthalten „Standardeinstellungen“ für ein bestimmtes Grob als eine assoziative Liste. Siehe scm/define-grobs.scm für die Einstellungen aller Grob-Beschreibungen. Grob-Beschreibungen werden mit \override verändert.

\override ist eigentlich eine Kurzform, der Befehl

```
\override Kontext.GrobBezeichnung #'Eigenschaft = #Wert
```
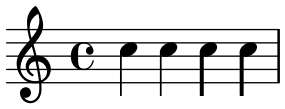
ist äquivalent zu

```
\set Kontext.GrobBezeichnung =
  #(cons (cons 'Eigenschaft Wert)
         <vorheriger Wert von Kontext.GrobBezeichnung>)
```

Der Wert von Kontext.GrobBezeichnung (die assoz. Liste „alist“) wird benutzt um die Eigenschaften von individuellen Grobs zu initialisieren. Grobs haben Eigenschaften, die im Scheme-Stil mit bindestrich-wörtern benannt sind. Diese Werte der Grob-Eigenschaften verändern sich während des Notensetzens: LilyPonds Notensatz heißt im Grunde, die Eigenschaften mit Callback-Fuktionen auszurechnen.
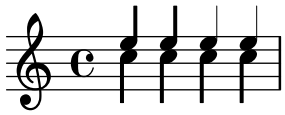
Beispielsweise kann die Dicke eines Notenhalses verändert werden, indem man die thickness-Eigenschaft des Stem-Objekts verändert:

```
c4 c
\override Voice.Stem.thickness = #3.0
c4 c
```
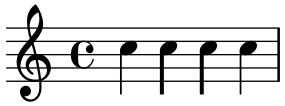


Wenn kein Kontext angegeben wird, wird der tiefste aktuelle Kontext benutzt:

```
{ \override Staff.Stem.thickness = #3.0
  <<
    {
      e4 e
      \override Stem.thickness = #0.5
      e4 e
    } \\ {
      c4 c c c
    }
  >>
}
```
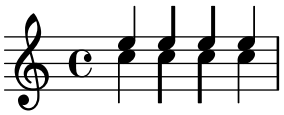


Die Auswirkungen von \override können mit \revert wieder rückgängig gemacht werden:

```
c4
\override Voice.Stem.thickness = #3.0
c4 c
\revert Voice.Stem.thickness
c4
```
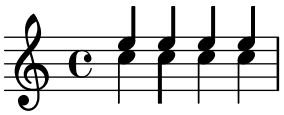


Die Auswirkungen von \override und \revert wirken sich auf alle Grobs im entsprechenden Kontext aber der Stelle aus, an der sie gesetzt werden:

```
{
  <<
    {
      e4
      \override Staff.Stem.thickness = #3.0
      e4 e e
    } \\ {
      c4 c c
      \revert Staff.Stem.thickness
      c4
    }
  >>
}
```



\once kann zusammen mit \override benutzt werden, um nur den aktuellen Zeitwert zu verändern:

```
{
  <<
    {
      \override Stem.thickness = #3.0
      e4 e e e
    } \\ {
      c4
      \once \override Stem.thickness = #3.0
      c4 c c
    }
  >>
}
```



Siehe auch

Referenz der Interna: Abschnitt “Backend” in *Referenz der Interna*.

34.4 Der \tweak-Befehl

Wenn man Grob-Eigenschaften mit \override verändert, verändern sich alle fraglichen Objekte zu dem gegebenen musikalischen Moment. Manchmal will man allerdings nur ein Grob verändern, anstatt allen Grobs des aktuellen Kontextes. Das kann mit dem \tweak-Befehl erreicht werden, mit dem man Optimierungen vornehmen kann:

```
\tweak Layout-Objekt #'grob-eigenschaft #Wert
```

Die Angabe von *Layout-Objekt* ist optional. Der \tweak-Befehl wirkt sich auf das musikalische Objekt aus, dass direkt auf Wert folgt.
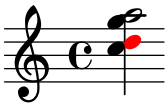
Eine Einleitung der Syntax und Benutzungen des \tweak-(Optimierungs)-Befehls findet sich in Abschnitt “Optimierungsmethoden” in *Handbuch zum Lernen*.

Wenn mehrere gleichartige Elemente zum gleichen musikalischen Moment auftreten, kann der \override-Befehl nicht benutzt werden, um nur einen von ihnen zu verändern: hier braucht man den \tweak-Befehl. Elemente, die mehrfach zum gleichen musikalischen Moment auftreten können sind unter Anderem:

- Notenköpfe von Noten innerhalb eines Akkordes
- Artikulationszeichen an einer einzelnen Note
- Bindebögen zwischen Noten eines Akkordes
- Llammern für rhythmische Verhältnisse (wie Triolen), die zur gleichen Zeit beginnen

In diesem Beispiel wird die Farbe eines Notenkopfes und die Art eines anderen Notenkopfes innerhalb eines Akkordes verändert:

```
< c''
  \tweak color #red
  d''
  g''
  \tweak duration-log #1
  a''
> 4
```
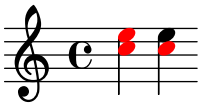


\tweak kann auch benutzt werden, um Bögen zu verändern:

```
\relative { c'-\tweak thickness #5 ( d e f) }
```



Damit der \tweak-Befehl funktioniert, muss er direkt vor dem Objekt stehen, auf das er sich bezieht. Einen ganzen Akkord kann man nicht mit \tweak verändern, weil der Akkord wie ein Kontainer ist, in dem alle Layoutelemente aus Ereignissen innerhalb von EventChord erstellt werden:
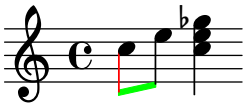
```
\tweak color #red <c e>4
<\tweak color #red c e>4
```



Der einfache \tweak-Befehl kann *nicht* eingesetzt werden, um Elemente zu verändern, die nicht direkt aus der Eingabe erstellt werden. Insbesondere Hälse, automatische Balken oder Versetzungszeichen lassen sich nicht beeinflussen, weil diese später durch die Layoutobjekte des Notenkopfs erstellt werden und nicht direkt durch den Quelltext.

Derartige indirekt erstellt Layoutobjekte können mit \tweak verändert werden, indem man die ausführliche Form des Befehls einsetzt:
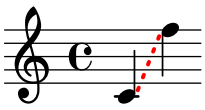
```
\tweak Stem.color #red
\tweak Beam.color #green c''8 e''
<c'' e'' \tweak Accidental.font-size #-3 ges''>4
```



\tweak kann auch nicht verwendet werden, um Schlüssel oder Taktarten zu verändern, denn sie werden von dem \tweak-Befehl während der Interpretation durch automatisches Einfügen von zusätzlichen Kontextelementen getrennt.

Mehrere \tweak-Befehle können vor ein Notationselement gesetzt werden und alle werden interpretiert:

```
c'
  -\tweak style #'dashed-line
  -\tweak dash-fraction #0.2
  -\tweak thickness #3
  -\tweak color #red
  \glissando
f''
```



Der Strom der musikalischen Ereignisse (engl. music stream), der aus dem Quelltext erstellt wird, und zu dem auch die automatisch eingefügten Elemente gehören, kann betrachtet werden, siehe Abschnitt “Musikalische Funktionen darstellen” in *Extending*. Das kann nützlich sein, wenn man herausfinden will, was mit dem `\tweak`-Befehl verändert werden kann.

Siehe auch

Handbuch zum Lernen: Abschnitt “Optimierungsmethoden” in *Handbuch zum Lernen*.

Erweitern: Abschnitt “Musikalische Funktionen darstellen” in *Extending*.

Bekannte Probleme und Warnungen

Der `\tweak`-Befehl kann nicht benutzt werden, um die Kontrollpunkte eines von mehreren Bindebögen eines Akkorden zu verändern. Anstelle dessen wird der erste Bogen verändert, der in der Eingabedatei auftritt.

34.5 \\set versus \\override

34.6 Alisten verändern

Einige vom Benutzer einstellbare Eigenschaften sind intern als *alists* (Assoziative Listen) dargestellt, die Paare von Schlüsseln und Werten speichern. Die Struktur einer Aliste ist:

```
'((Schlüssel1 . Wert1)
  (Schlüssel2 . Wert2)
  (Schlüssel3 . Wert3)
  ...)
```

Wenn eine Aliste eine Grob-Eigenschaft oder eine Variable der `\paper`-Umgebung ist, können ihre Schlüssel einzeln verändert werden, ohne andere Schlüssel zu beeinflussen.

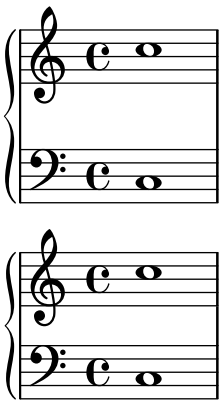
Um beispielsweise den Freiraum zwischen benachbarten Systemen in einer Systemgruppe zu verkleinern, kann man die `staff-staff-spacing`-Eigenschaft des `+StaffGrouper`-Grobs benutzen. Die Eigenschaft ist eine Aliste mit vier Schlüsseln: : `basic-distance` (Grund-Abstand), `minimum-distance` (minimaler Abstand), `padding` (Verschiebung) und `stretchability` (Dehnbarkeit). Die Standardwerte dieser Eigenschaft finden sich im Abschnitt „Backend“ der Referenz der Interna (siehe Abschnitt “StaffGrouper” in *Referenz der Interna*):

```
'((basic-distance . 9)
  (minimum-distance . 7)
  (padding . 1)
  (stretchability . 5))
```

Eine Möglichkeit, die Systemen dichter zueinander zu zwingen, ist es, der Wert des `basic-distance`-Schlüssels (`9`) zu verändern, sodass der den gleichen Wert wie `minimum-distance` (`7`) hat. Um einen einzelnen Schlüssel zu verändern, wird ein geschachtelter Aufruf benutzt:

```
% default space between staves
\new PianoStaff <<
  \new Staff { \clef treble c''1 }
  \new Staff { \clef bass   c1   }
>>

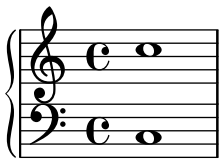
% reduced space between staves
\new PianoStaff \with {
  % this is the nested declaration
  \override StaffGrouper.staff-staff-spacing.basic-distance = #7
} <<
  \new Staff { \clef treble c''1 }
  \new Staff { \clef bass   c1   }
>>
```



Wenn man diese Art des geschachtelten Aufrufs einsetzt, wird der spezifische Schlüssel (`basic-distance` im obigen Beispiel) verändert, ohne dass sich andere Wert für die gleiche Eigenschaft ändern würden.

Nun sollen die Systeme so dicht wie möglich gesetzt werden, ohne das Überlappungen vorkommen. Die einfachste Möglichkeit, das zu tun, wäre es, alle vier Wert auf 0 zu setzen. Man muss jedoch nicht vier Werte definieren, sondern die Eigenschaft kann mit einem Aufruf als Aliste vollständig verändert werden:

```
\new PianoStaff \with {
  \override StaffGrouper.staff-staff-spacing =
    #'((basic-distance . 0)
       (minimum-distance . 0)
       (padding . 0)
       (stretchability . 0))
} <<
  \new Staff { \clef treble c''1 }
  \new Staff { \clef bass   c1   }
>>
```



Dabei sollte beachtet werden, dass alle Schlüssel, die bei dieser Weise des Aufrufs nicht explizit aufgelistet sind, auf den Standardwert gesetzt werden, den sie hätten, wenn sie nicht definiert

werden. Im Falle von staff-staff-spacing würden alle nicht genannten Schlüsselwerte auf 0 gesetzt (außer stretchability, welche immer den Wert von space hat, wenn sie nicht definiert ist). Somit sind folgende Aufrufe äquivalent:

```
\override StaffGrouper.staff-staff-spacing =
  #'((basic-distance . 7))

\override StaffGrouper.staff-staff-spacing =
  #'((basic-distance . 7)
     (minimum-distance . 0)
     (padding . 0)
     (stretchability . 7))
```

Eine möglicherweise ungewollte Konsequenz hiervon ist, dass alle Standardwerte, die etwa in einer Initialisierungsdatei zu Beginn einer LilyPond-Partitur geladen werden, nach dem Aufruf rückgängig gemacht werden. Im obigen Beispiel werden die initialisierten Standardwerte für padding und minimum-distance (definiert in scm/define-grobs.scm) auf den Standard zurückgesetzt, den sie uninitialisiert hätten (0 in beiden Fällen). Wenn eine Eigenschaft oder Variable in Form einer Aliste (jeder Größe) definiert wird, werden immer alle Schlüsselwerte auf den uninitialisierten Zustand zurückgesetzt. Es ist also sicherer, geschachtelte Aufrufe zu benutzen, wenn man nicht bewusst alle Werte zurücksetzen will.

Achtung: Geschachtelte Aufrufe funktionieren nicht mit Kontexteigenschaften (wie etwa beamExceptions, keyAlterations, timeSignatureSettings, usw.) Diese Eigenschaften können nur verändert werden, indem man sie vollständig als Alisten umdefiniert.

35 Nützliche Konzepte und Eigenschaften

35.1 Eingabe-Modi

Die Art, wie die Notation einer Eingabedatei interpretiert wird, hängt vom aktuellen Eingabemodus ab.

Chord (Akkordmodus)

Man erreicht ihn durch den Befehl `\chordmode`. Hierdurch wird die Eingabe entsprechend der Syntax der Akkordnotation interpretiert, siehe Kapitel 15 [Notation von Akkorden], Seite 397. Akkorde werden als Noten auf einem System dargestellt.

Der Akkordmodus wird auch mit dem Befehl `\chords` initiiert. Dadurch wird gleichzeitig ein neuer `ChordNames`-Kontext erstellt, die Eingabe entsprechend der Syntax der Akkordnotation interpretiert und als Akkordbezeichnungen in einem `ChordNames`-Kontext dargestellt. Siehe Abschnitt 15.2.1 [Akkordbezeichnungen drucken], Seite 403.

Drum (Schlagzeugmodus)

Man erreicht ihn mit dem Befehl `\drummode`. Die Eingabe wird entsprechend der Syntax der Schlagzeugnotation interpretiert, siehe Abschnitt 13.1.2 [Grundlagen der Schlagzeugnotation], Seite 375.

Der Schlagzeugmodus wird auch mit dem Befehl `\drums` aktiviert. Dadurch wird gleichzeitig ein neuer `DrumStaff`-Kontext erstellt, die Eingabe entsprechend der Syntax der Schlagzeugnotation interpretiert und als Schlagzeugsymbole auf einem Schlagzeugsystem dargestellt. Siehe Abschnitt 13.1.2 [Grundlagen der Schlagzeugnotation], Seite 375.

Figure (Ziffernmodus)

Man erreicht ihn mit dem Befehl `\figuremode`. Die Eingabe wird entsprechend der Syntax für Generalbass interpretiert, siehe Abschnitt 15.3.2 [Eingabe des Generalbass'], Seite 413.

Der Ziffernmodus wird auch mit dem Befehl `\figures` aktiviert. Dadurch wird gleichzeitig ein neuer `FiguredBass`-Kontext erstellt, die Eingabe entsprechend der Syntax für Generalbass interpretiert und als Generalbassziffern im `FiguredBass`-Kontext dargestellt. Siehe Abschnitt 15.3.1 [Grundlagen des Bezifferten Basses], Seite 412.

Fret/tab (Griffsymbol-/Tabulaturmodus)

Es gibt keinen besonderen Eingabemodus für Griffsymbole und Tabulaturen.

Um Tabulaturen zu erstellen, werden Noten oder Akkorde im Notenmodus notiert und dann in einem `TabStaff`-Kontext interpretiert, siehe Abschnitt 12.1.3 [Standardtabulaturen], Seite 330.

Um Griffsymbole oberhalb eines Notensystems zu erstellen, gibt es zwei Möglichkeiten. Man kann den `FretBoards`-Kontext einsetzen (siehe Abschnitt 12.1.7 [Automatische Bund-Diagramme], Seite 365) oder sie können als Beschriftung über den Noten eingefügt werden, indem man den `\fret-diagram`-Befehl einsetzt (siehe Abschnitt 12.1.5 [Bund-Diagramm-Beschriftung], Seite 346).

Lyrics (Gesangstextmodus)

Man erreicht ihn mit dem Befehl `\lyricmode`. Die Eingabe wird entsprechend der Syntax für Silben eines Gesangstextes interpretiert, wobei optional Dauern und verknüpfte Gesangstextveränderer möglich sind, siehe Kapitel 9 [Notation von Gesang], Seite 253.

Der Gesangstextmodus wird auch durch den Befehl `\addlyrics` aktiviert. Dadurch wird auch ein neuer `Lyrics`-Kontext erstellt und ein impliziter `\lyricsto`-Befehl, der den nachfolgenden Gesangstext mit der vorhergehenden Musik verknüpft.

Markup (Textbeschriftungsmodus)

Man erreicht ihn mit dem Befehl \markup. Die Eingabe wird entsprechend der Syntax für Textbeschriftung interpretiert, siehe Abschnitt A.10 [Textbeschriftungsbefehle], Seite 678.

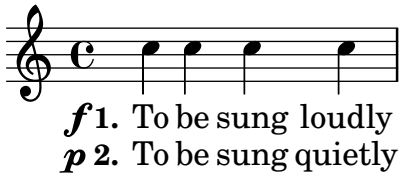
Note (Notenmodus)

Das ist der Standardmodus. Er kann auch mit dem Befehl \notemode gefordert werden. Die Eingabe wird als Tonhöhen, Dauern, Beschriftung usw. interpretiert und als musikalische Notation auf einem Notensystem gesetzt.

Es ist normalerweise nicht nötig, den Notenmodus extra anzugeben, aber es kann in bestimmten Situationen durchaus nützlich sein, etwa wenn man in einem Gesangstext-, Akkord- oder einem anderen Modus arbeitet aber ein Zeichen braucht, das nur im Notenmodus benutzt werden kann.

Um etwa Dynamikzeichen vor die Nummern von unterschiedlichen Strophen zu setzen, muss man den Notenmodus betreten:

```
{ c4 c4 c4 c4 }
\addlyrics {
  \notemode{ \stanza \markup{ \dynamic f 1. } }
  To be sung loudly
}
\addlyrics {
  \notemode{ \stanza \markup{ \dynamic p 2. } }
  To be sung quietly
}
```



35.2 Richtung und Platzierung

Die Platzierung und Richtung von Objekten ist im Notensatz oft durch eine enge Auswahl begrenzt: Notenhälse beispielsweise können entweder nach oben oder nach unten zeigen, Gesangstext, Dynamikzeichen und andere Ausdrucksbezeichnungen können über oder unter dem System gesetzt werden, Text kann rechts, links oder mittig ausgerichtet werden usw. Die meisten dieser Entscheidungen können LilyPond direkt überlassen werden; in einigen Fällen kann es allerdings nötig sein, eine bestimmte Richtung oder eine Position zu erzwingen.

Richtungseinstellung von Artikulationszeichen

Standardmäßig sind bestimmte Objekte immer nach oben oder unten ausgerichtet, wie Dynamikzeichen oder Fermaten, während andere Objekte zwischen oben und unten wechseln, was vor allem von der Richtung der Notenhälse abhängt und etwa Bögen und Akzente betrifft.

Die Standardeinstellungen können verändert werden, indem dem Artikulationszeichen ein Ausrichtungsmarkierer vorangeht. Drei derartige Ausrichtungsmarkierer sind vorhanden: ^ (bedeutet „nach oben“), _ (bedeutet „nach unten“) bzw. - (bedeutet „Standardrichtung“ benutzen) normalerweise weggelassen werden. In diesem Fall wird - angenommen. Eine Richtungsanweisung ist jedoch **immer** erforderlich vor

- \tweak-Befehlen
- \markup-(Textbeschriftungs-)Befehlen
- \tag-Befehlen
- Textbeschriftungen in reiner Textform, wie etwa -"string"
- Fingersatzanweisungen: -1

- Abkürzungen von Artikulationen, wie `-.`, `->`, `--`

Ausrichtungsmarkierer haben nur eine Auswirkung auf die nächste Note:

```
\relative {
  c''2( c)
  c2_( c)
  c2( c)
  c2^( c)
}
```



Die `direction`-(Richtungs-)Eigenschaft

Die Position oder Richtung vieler Layoutobjekte wird von der `direction`-Eigenschaft kontrolliert.

Der Wert der `direction`-Eigenschaft kann auf den Wert `1` gesetzt werden, was gleichbedeutend mit „nach oben“ bzw. „oberhalb“ ist, oder auf den Wert `-1`, was „nach unten“ bzw. „unterhalb“ bedeutet. Die Symbole `UP` und `DOWN` können anstelle von `1` und `-1` benutzt werden. Die Standardausrichtung kann angegeben werden, indem `direction` auf den Wert `0` oder `CENTER` gesetzt wird. In vielen Fällen bestehen auch vordefinierte Befehle, mit denen die Ausrichtung bestimmt werden kann. Sie haben die Form

`\xxxUp`, `xxxDown`, `xxxNeutral`

wobei `xxxNeutral` bedeutet: „Benutze die Standardausrichtung“. Siehe auch Abschnitt “within-staff (Objekte innerhalb des Notensystems)” in *Handbuch zum Lernen*.

In wenigen Fällen, von denen Arpeggio das einzige häufiger vorkommende Beispiel darstellt, entscheidet der Wert von `direction`, ob das Objekt auf der rechten oder linken Seite das Ursprungsobjektes ausgegeben wird. In diesem Fall bedeutet `-1` oder `LEFT` „auf der linken Seite“ und `1` oder `RIGHT` „auf der rechten Seite“. `0` oder `CENTER` bedeutet „benutze Standardausrichtung“.

Diese Ausrichtungsanzeigen wirken sich auf alle Noten aus, bis sie rückgängig gemacht werden:

```
\relative {
  c''2( c)
  \slurDown
  c2( c)
  c2( c)
  \slurNeutral
  c2( c)
}
```



In polyphoner Musik ist es normalerweise besser, eine explizite Stimme (`voice` zu erstellen, als die Richtung eines Objektes zu ändern. Zu mehr Information siehe Abschnitt 5.2 [Mehrere Stimmen], Seite 166.

Siehe auch

Handbuch zum Lernen: Abschnitt “within-staff (Objekte innerhalb des Notensystems)” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 5.2 [Mehrere Stimmen], Seite 166.

35.3 Abstände und Maße

In LilyPond gibt es zwei Arten von Abständen: absolute und skalierte.

Absolute Abstände werden benutzt, um Ränder, Einzüge und andere Einzelheiten des Seitenlayouts zu bestimmen. Sie sind in den Standardeinstellungen in Millimetern definiert. Abstände können auch in anderen Einheiten definiert werden, indem folgende Befehle auf die Zahl folgen: `\mm`, `\cm`, `\in` (Zoll=2,54 cm) und `\pt` (Punkte, 1/72.27 eines Zolls). Abstände des Seitenlayouts können auch in skalierbaren Einheiten (siehe folgenden Absatz) definiert werden, indem man den Befehl `\staff-space` an die Zahl hängt. Das Seitenlayout ist genauer beschrieben in Kapitel 25 [Seitenlayout], Seite 521.

Skalierbare Abstände werden immer in Einheiten von Notenlinienabständen angegeben, oder seltener in halben Notenlinienabständen. Ein Notenlinienabstand ist der Abstand zwischen zwei benachbarten Linien eines Notensystems. Der Standardwert dieser Einheit kann global geändert werden, indem man die globale Notensystemgröße ändert, oder sie kann lokal geändert werden, indem man die Eigenschaft `staff-space` des `StaffSymbol`-Objekts mit `\override` verändert. Skalierte Abstände verändern sich automatisch entsprechend, wenn der Notenlinienabstand entweder global oder lokal verändert wird, aber Schriftarten verändern ihre Größe nur, wenn der Notenlinienabstand global verändert wird. Mit dem globalen Notenlinienabstand kann man also auf einfach Art und Weise die gesamte Größe einer Partitur verändern. Zu Methoden, wie der globale Notenlinienabstand verändert werden kann, siehe Abschnitt 26.2 [Die Notensystemgröße einstellen], Seite 534.

Wenn nur eine ein Abschnitt einer Partitur in einer anderen Größe erscheinen soll, etwa ein Ossia-Abschnitt in einer Fußnote, kann die globale Notensystemgröße nicht einfach geändert werden, weil sich diese Änderung auf die gesamte Partitur auswirken würde. In derartigen Fällen muss die Größenänderung vorgenommen werden, indem man sowohl die `staff-space`-Eigenschaft von `StaffSymbol` als auch die Größe der Schriftarten verändert. Eine Scheme-Funktion, `magstep`, kann von einer Schriftartveränderung zu der entsprechenden Veränderung in `staff-space` (Notenlinienabständen) konvertieren. Zu einer Erklärung und Beispielen zu ihrer Verwendung siehe Abschnitt “Länge und Dicke von Objekten” in *Handbuch zum Lernen*.

Siehe auch

Handbuch zum Lernen: Abschnitt “Länge und Dicke von Objekten” in *Handbuch zum Lernen*.
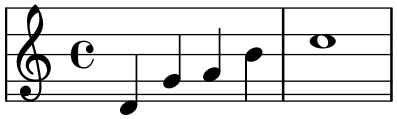
Notationsreferenz: Kapitel 25 [Seitenlayout], Seite 521, Abschnitt 26.2 [Die Notensystemgröße einstellen], Seite 534.

35.4 Eigenschaften des Staff-Symbols

Die vertikale Position der Notenlinien und die Anzahl der Notenlinien kann gleichzeitig definiert werden. Wie das folgende Beispiel zeigt, werden Notenpositionen nicht durch die Position der Notenlinien verändert:
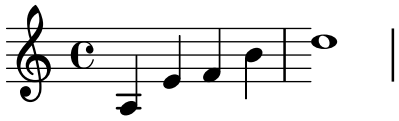
Achtung: Die `'line-positions`-Eigenschaft verändert die `'line-count`-Eigenschaft. Die Anzahl der Notenlinien wird implizit definiert durch die Anzahl der Elemente in der Liste der Werte von `'line-positions`.

```
\new Staff \with {
  \override StaffSymbol.line-positions = #'(7 3 0 -4 -6 -7)
}
\relative { a4 e' f b | d1 }
```



Die Breite eines Notensystems kann verändert werden. Die Einheit ist in Notenlinienabständen. Die Abstände von Objekten in diesem Notensystem wird durch diese Einstellung nicht beeinflusst.

```
\new Staff \with {
  \override StaffSymbol.width = #23
}
\relative { a4 e' f b | d1 }
```



35.5 Strecker

Viele Objekte der Musiknotation erstrecken sich über mehrere Objekte oder gar mehrere Takte. Beispiele hierfür sind etwa Bögen, Balken, Triolenklammern, Volta-Klamern in Wiederholungen, Crescendo, Triller und Glissando. Derartige Objekte werden als „Strecker“ bezeichnet. Sie haben spezielle Eigenschaften, mit welchen ihre Eigenschaften und ihr Verhalten beeinflusst werden kann. Einige dieser Eigenschaften gelten für alle Strecker, andere beschränken sich auf eine Untergruppe der Strecker.

Alle Strecker unterstützen das `spanner-interface` (Strecker-Schnittstelle). Ein paar, insbesondere die, die zwischen zwei Objekten eine gerade Linie ziehen, unterstützen auch das `line-spanner-interface` (Strecker-Linienschnittstelle).

Das `spanner-interface` benutzen

Diese Schnittstelle stellt zwei Eigenschaften zur Verfügung, die sich auf mehrere Strecker auswirken:

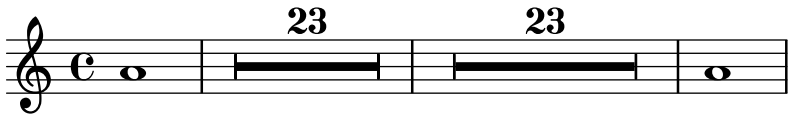
Die `minimum-length`*-Eigenschaft*

Die Mindestlänge eines Streckers wird durch die `minimum-length`-Eigenschaft definiert. Wenn diese Eigenschaft vergrößert wird, muss in den meisten Fällen auch der Abstand der Noten zwischen den zwei Endpunkten eines Streckers verändert werden. Eine Veränderung dieser Eigenschaft hat jedoch auf die meisten Strecker keine Auswirkung, weil ihre Länge aus anderen Berechnungen hervorgeht. Einige Beispiele, wo die Eigenschaft benutzt wird, sind unten dargestellt.
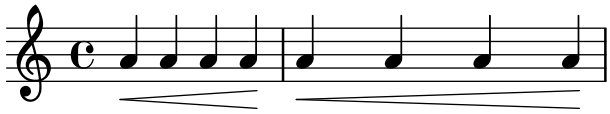
```
a~a
a
% increase the length of the tie
-\tweak minimum-length #5
~a
```



```
a1
\compressEmptyMeasures
R1*23
% increase the length of the rest bar
\once \override MultiMeasureRest.minimum-length = #20
R1*23
a1
```




```
a \< a a a \!
% increase the length of the hairpin
\override Hairpin.minimum-length = #20
a \< a a a \!
```



Diese Veränderung kann auch eingesetzt werden, um die Länge von Legato- und Phrasierungsbögen zu verändern:

```
a( a)
a
-\tweak minimum-length #5
( a)

a\( a\)
a
-\tweak minimum-length #5
\( a\)
```

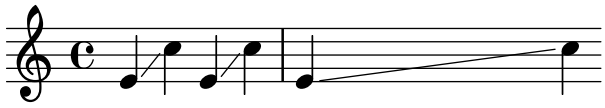


Im Falle einiger Layoutobjekte wirkt sich die `minimum-length`-Eigenschaft erst dann aus, wenn die `set-spacing-rods`-Prozedur explizit aufgerufen wird. Um das zu tun, sollte die `springs-and-rods`-Eigenschaft auf `ly:spanner::set-spacing-rods` gesetzt werden. Die Mindestlänge eines Glissandos etwa wird erst aktiv, wenn die `springs-and-rods`-Eigenschaft gesetzt ist:

```
% default
e' \glissando c''

% not effective alone
\once \override Glissando.minimum-length = #20
e' \glissando c''
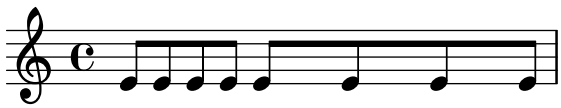
% effective only when both overrides are present
\once \override Glissando.minimum-length = #20
\once \override Glissando.springs-and-rods = #ly:spanner::set-spacing-rods
e' \glissando c''
```



Das gilt auch für das Beam-(Balken-)Objekt:

```
% not effective alone
\once \override Beam.minimum-length = #20
e'8 e' e' e'

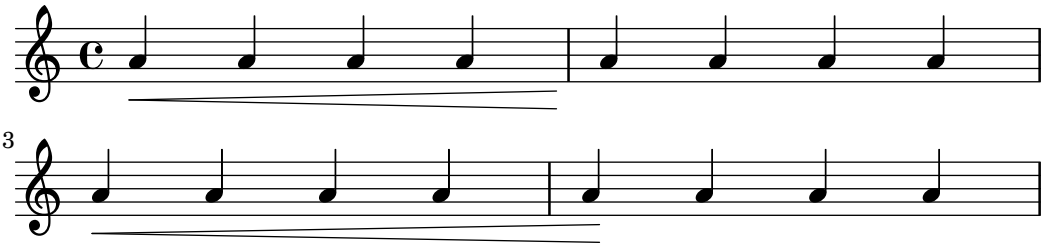
% effective only when both overrides are present
\once \override Beam.minimum-length = #20
\once \override Beam.springs-and-rods = #ly:spanner::set-spacing-rods
e'8 e' e' e'
```



Die to-barline*-Eigenschaft*

Die zweite nützliche Eigenschaft des spanner-interface ist to-barline (bis zum Taktstrich). In den Standardeinstellungen ist diese Eigenschaft auf „wahr“ gesetzt, was bedeutet, dass ein Strecker, etwa eine Crescendo-Klammer, der an der ersten Noten eines Taktes beendet wird, sich nur bis zum vorhergehenden Taktstrich erstreckt. Wenn die Eigenschaft auf „falsch“ gesetzt wird, erstrecken sich die Strecker entsprechend über die Taktlinie hinüber und enden erst an der entsprechenden Note:

```
\relative {
  a' \< a a a a \! a a a \break
  \override Hairpin.to-barline = ##f
  a \< a a a a \! a a a
}
```



Diese Eigenschaft wirkt sich nicht auf alle Strecker aus. Im Falle von Legato- oder Phrasierungsbögen etwa hat diese Eigenschaft keinen Effekt. Das gilt auch für alle anderen Streckern, bei denen es nicht sinnvoll währe, sie an einer Taktlinie abzuschließen.
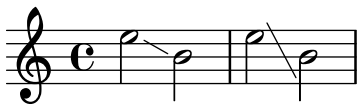
Das `line-spanner-interface` benutzen

Objekte, die das `line-spanner-interface` unterstützen, sind unter Anderem:

- `DynamicTextSpanner`
- `Glissando`
- `TextSpanner`
- `TrillSpanner`
- `VoiceFollower`

Die Routine, die das Setzen der Matrizen dieser Strecker hervorruft, ist `ly:line-interface::print`. Diese Routine bestimmt die exakte Position der zwei Endpunkte und zeichnet eine Linie zwischen ihnen, in dem erforderlichen Stil. Die Position der zwei Endpunkte des Streckers wird in Echtzeit errechnet, aber es ist möglich, ihre Y-Koordinaten zu verändern. Die Eigenschaften, die angegeben werden müssen, sind zwei Ebenen in der Objekthierarchie tiefer angeordnet, aber die Syntax des `\override`-Befehls ist ziemlich einfach:

```
e''2 \glissando b'
\once \override Glissando.bound-details.left.Y = #3
\once \override Glissando.bound-details.right.Y = #-2
e''2 \glissando b'
```

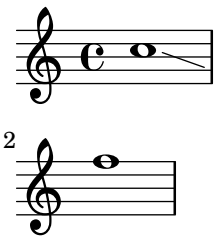


Die Einheiten für die `Y`-Eigenschaft werden in Notenlinienabständen angegeben, wobei die Mittellinie des Notensystems die Null darstellt. Für das Glissando ist der Wert von `Y` am entsprechenden X-Koordinatenpunkt entsprechend dem Mittelpunkt des Notenkopfes, wenn die Linie bis in die Noten hinein weitergeführt werden würde.

Wenn `Y` nicht gesetzt wird, wird der Wert aus der vertikalen Position des entsprechenden Anknüpfpunkts des Streckers errechnet.

Im Fall eines Zeilenumbruchs werden die Werte der Endpunkte in den Unterlisten `left-broken` bzw. `right-broken` von `bound-details` abgelegt. Zum Beispiel:

```
\override Glissando.breakable = ##t
\override Glissando.bound-details.right-broken.Y = #-3
c''1 \glissando \break
f''1
```



Eine Anzahl weitere Eigenschaft der `left-` und `right`-Unterlisten der `bound-details`-Eigenschaft kann auf gleiche Weise wie `Y` verändert werden:

`Y` Hiermit wird der Y-Koordinationspunkt des Endpunktes in Notenlinienabständen vom Mittelpunkt des Notensystems ausgehend angegeben. Der Endpunkt ist normalerweise der Mittelpunkt des Elternobjektes, sodass Glissandos vertikal auf den Mittelpunkt eines Notenkopfes weist.

Für horizontale Strecker, wie Textstrecker und Trillerstrecker ist sein Wert mit 0 definiert.
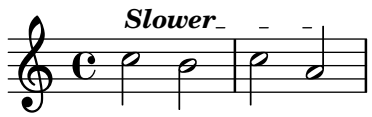
`attach-dir`
: Das entscheidet, wo die Linie auf der X-Achse beginnt und endet, relativ zum Elternobjekt. Ein Wert `-1` (oder `LEFT`) lässt die Linie an der linken Seite der Noten beginnen/enden, mit der sie verknüpft ist.

`X`
: Das ist der absolute X-Koordinatenpunkt des Endpunktes. Der Wert wird normalerweise in Echtzeit errechnet, und ihn zu verändern ist normalerweise nicht nützlich.

`stencil`
: Linienstrecker können Symbole am Ende oder zu Anfang des Streckers haben, die in dieser Untereigenschaft definiert werden. Die Eigenschaft ist für interne Benutzung, es wird empfohlen, die Eigenschaft `text` zu benutzen.

`text`
: Das ist eine Textbeschriftung, die ausgewertet wird und die `stencil`-Eigenschaft überschreibt. Sie wird eingesetzt, um *cresc.*, *tr* oder andere Texte an horizontale Strecker zu setzen.

```
\override TextSpanner.bound-details.left.text
   = \markup { \small \bold Slower }
\relative { c''2\startTextSpan b c a\stopTextSpan }
```
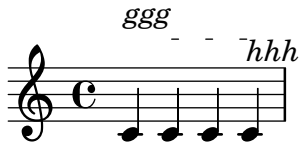


`stencil-align-dir-y`
`stencil-offset`
: Wenn keine dieser beiden Eigenschaften gesetzt wird, wird die Matrize (engl. stencil) einfach am Endpunkt des Streckers, auf seiner Mittellinie (wie durch `X` und `Y` definiert) zentriert, ausgegeben. Wenn entweder `stencil-align-dir-y` oder `stencil-offset` gesetzt werden, wird das Symbol am Rand vertikal entsprechend des Endpunktes der Linie verschoben:

```
\override TextSpanner.bound-details.left.stencil-align-dir-y = #-2
\override TextSpanner.bound-details.right.stencil-align-dir-y = #UP

\override TextSpanner.bound-details.left.text = "ggg"
\override TextSpanner.bound-details.right.text = "hhh"

\relative { c'4^\startTextSpan c c c \stopTextSpan }
```
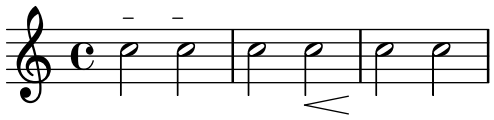


Dabei sollte beachtet werden, dass negative Werte das Objekt nach *oben* verschieben, anders als man erwarten könnte, weil der Wert `-1` oder `DOWN` bedeutet, dass die *Unterkante* des Textes mit der Streckerlinie ausgerichtet wird. Ein Wert `1` oder `UP` richtet die Oberkante des Textes mit der Streckerlinie aus.

`arrow`
: Wenn diese Untereigenschaft auf `#t` gesetzt wird, wird ein Pfeilkopf am Ende der Linie erstellt.

`padding`
: Diese Eigenschaft kontrolliert den Abstand zwischen dem angegebenen Endpunkt der Linie und dem wirklichen Ende. Ohne Füllung (engl. padding) würde ein Glissando in der Mitte eines Notenkopfes beginnen und enden.

Die musikalische Funktion `\endSpanners` beschließt den Strecker, der an der direkt folgenden Note beginnt, bevor er eigentlich zu ende wäre. Er wird exakt nach einer Note beendet, oder am nächsten Taktstrich, wenn `to-barline` auf wahr gesetzt ist und eine Taktlinie vor der nächsten Note erscheint.

```
\relative c'' {
  \endSpanners
  c2 \startTextSpan c2 c2
  \endSpanners
  c2 \< c2 c2
}
```



Wenn man `\endSpanners` benutzt, ist es nicht nötig, den Befehl \startTextSpan mit \stopTextSpan zu beenden, und es ist auch nicht nötig, Crescendo-Klammern mit `\!` zu beenden.

Siehe auch

Referenz der Interna: Abschnitt "TextSpanner" in *Referenz der Interna*, Abschnitt "Glissando" in *Referenz der Interna*, Abschnitt "VoiceFollower" in *Referenz der Interna*, Abschnitt "TrillSpanner" in *Referenz der Interna*, Abschnitt "line-spanner-interface" in *Referenz der Interna*.

35.6 Sichtbarkeit von Objekten

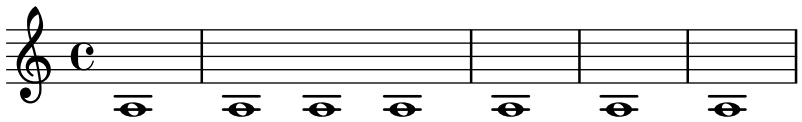
Die Sichtbarkeit von Layout-Objekten kann auf vier Arten kontrolliert werden: Ihre Matrizen (engl stencil) können entfernt werden, sie können unsichtbar gemacht werden, sie können weiß eingefärbt werden und ihre `break-visibility`-Eigenschaft kann verändert werden. Die ersten drei Möglichkeiten beziehen sich auf alle Layout-Objekte, die letzte nur auf einige wenige, nämlich die *zerteilbaren* Objekte. Das Handbuch zum Lernen führt in alle vier Möglichkeiten ein, siehe Abschnitt "Sichtbarkeit und Farbe von Objekten" in *Handbuch zum Lernen*.

Es gibt auch einige weitere Techniken, die sich nur auf bestimmte Layout-Objekte beziehen. Sie werden im letzten Abschnitt behandelt.

35.6.1 Einen `stencil` entfernen

Jedes Layout-Objekt hat eine Matrizen-(stencil)-Eigenschaft. Sie ist normalerweise definiert als die Funktion, die das entsprechende Objekt zeichnet. Wenn die Eigenschaft mit `\override` auf `#f` gesetzt wird, wird keine Funktion aufgerufen und also auch kein Objekt gezeichnet. Das Standardverhalten kann mit dem Befehl `\revert` wieder hergestellt werden.
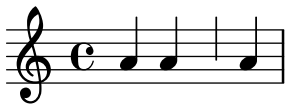
```
a1 a
\omit Score.BarLine
a a
\undo \omit Score.BarLine
a a a
```



35.6.2 Objekten unsichtbar machen

Jedes Layout-Objekt hat eine Durchsichtigkeits-Eigenschaft (`'transparent`), die normalerweise auf den Wert `#f` gesetzt ist. Wenn sie auf `#t` gesetzt wird, nimmt das Objekt immer noch den entsprechenden Platz ein, ist aber unsichtbar.

```
a'4 a'
\once \hide NoteHead
a' a'
```



35.6.3 Objekte weiß malen

Alle Layout-Objekte haben eine Farb-(color)-Eigenschaft, die normalerweise schwarz (`black`) definiert ist. Wenn sie nach weiß (`white`) verändert wird, kann man das Objekt nicht mehr vom weißen Hintergrund unterscheiden. Wenn das Objekt jedoch andere Objekte überschneidet, wird die Farbe der Überschneidungen von der Reihenfolge entschieden, in welcher die Objekte gesetzt werden. Es kann also vorkommen, dass man die Umrisse des weißen Objektes erahnen kann, wie in diesem Beispiel:

```
\override Staff.Clef.color = #white
a'1
```



Das kann man vermeiden, indem man die Satzreihenfolge der Objekte verändert. Alle Layout-Objekte haben eine `layer`-Eigenschaft, die auf eine ganze Zahl gesetzt sein muss. Objekte mit der niedrigsten Zahl in der `layer`-Eigenschaft werden zuerst gesetzt, dann die nächsten Objekte in ansteigender Ordnung. Objekte mit höheren Werten überschneiden also Objekte mit niedrigeren Werten. Die meisten Objekte bekommen den Wert `1` zugewiesen, einige wenige Objekte, unter die auch `StaffSymbol` (die Notenlinien) gehört, jedoch den Wert `0`. Die Reihenfolge, in der Objekte mit demselben Wert gesetzt werden, ist nicht definiert.

Im oberen Beispiel wird der weiße Schlüssel, der einen Wert von `1` für `layer` hat, nach den Notenlinien gesetzt (die einen Wert von `0` für `layer` haben) und überschneidet sie also. Um das zu ändern, muss dem `Clef`-Objekt (Notenschlüssel) ein niedrigerer Wert, etwa `-1`, gegeben werden, sodass es früher gesetzt wird:

```
\override Staff.Clef.color = #white
\override Staff.Clef.layer = #-1
a'1
```



35.6.4 `break-visibility` (unsichtbar machen) benutzen

Die meisten Layout-Objekte werden nur einmal gesetzt, aber einige, wie Taktstriche, Schlüssel, Taktartbezeichnung und Tonartvorzeichen, müssen mehrmals gesetzt werden, wenn die Zeile gewechselt wird: einmal am Ende des oberen Systems und ein zweites Mal zu Beginn des nächsten Systems. Derartige Objekte werden als *trennbar* bezeichnet und haben eine Eigenschaft, die `break-visibility`-Eigenschaft, mit der ihre Sichtbarkeit an allen drei Positionen, an denen sie

auftreten können, kontrolliert werden kann: zu Beginn einer Zeile, innerhalb einer Zeile, wenn sie verändert werden, und am Ende einer Zeile, wenn die Änderung hier stattfindet.

Die Taktart wird beispielsweise standardmäßig nur zu Beginn des ersten Systems gesetzt, aber an anderen Stellen nur, wenn sie sich ändert. Wenn diese Änderung am Ende eines Systems auftritt, wird die neue Taktart am Ende des aktuellen Systems als auch zu Beginn des nächsten Systems gesetzt.

Dieses Verhalten wird von der `break-visibility`-Eigenschaft kontrolliert, die erklärt wird in Abschnitt "Sichtbarkeit und Farbe von Objekten" in *Handbuch zum Lernen*. Die Eigenschaft braucht einen Vektor von drei Booleschen Werten, die in ihrer Reihenfolge bestimmte, ob das Objekt a) zu Ende der Zeile, b) innerhalb einer Zeile oder c) zu Beginn einer Zeile gesetzt wird. Oder, genauer gesagt, vor einem Zeilenumbruch, an Stellen, wo kein Zeilenumbruch auftritt oder nach einem Zeilenumbruch.

Die acht möglichen Kombinationen können auch durch vordefinierte Funktionen bestimmt werden, welche in der Datei `scm/output-lib.scm` definiert sind. Die letzten drei Spalten der folgenden Tabelle zeigen an, ob das Layout-Objekt an einer bestimmten Position sichtbar sein wird oder nicht:

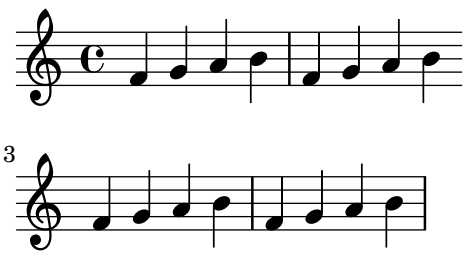
Funktion Form	**Vektor Form**	**Vor Umbruch**	**kein Umbruch**	**Nach Umbruch**
`all-visible`	`'#(#t #t #t)`	ja	ja	ja
`begin-of-line-visible`	`'#(#f #f #t)`	nein	nein	ja
`center-visible`	`'#(#f #t #f)`	nein	ja	nein
`end-of-line-visible`	`'#(#t #f #f)`	ja	nein	nein
`begin-of-line-invisible`	`'#(#t #t #f)`	ja	ja	nein
`center-invisible`	`'#(#t #f #t)`	ja	nein	ja
`end-of-line-invisible`	`'#(#f #t #t)`	nein	ja	ja
`all-invisible`	`'#(#f #f #f)`	nein	nein	nein

Die Standardeinstellungen von `break-visibility` hängen vom Layout-Objekt ab. Die folgende Tabelle zeigt alle wichtigen Layout-Objekte, die mit `break-visibility` verändert werden können und die jeweiligen Standardeinstellungen der Eigenschaft:

Layout-Objekt	**Normaler Kontext**	**Standardeinstellung**
`BarLine` (Taktstrich)	`Score`	calculated
`BarNumber` (Taktzahl)	`Score`	`begin-of-line-visible`
`BreathingSign` (Atemzeichen)	`Voice`	`begin-of-line-invisible`
`Clef` (Schlüssel)	`Staff`	`begin-of-line-visible`
`Custos`	`Staff`	`end-of-line-visible`
`DoublePercentRepeat` (Doppel-Prozent-Wiederholung	`Voice`	`begin-of-line-invisible`
`KeySignature` (Tonart)	`Staff`	`begin-of-line-visible`
`ClefModifier` (Oktavierungs-Acht)	`Staff`	`begin-of-line-visible`
`RehearsalMark` (Übungszeichen)	`Score`	`end-of-line-invisible`
`TimeSignature` (Taktart)	`Staff`	`all-visible`

Das Beispiel unten zeigt die Verwendung der Vektor-Form um die Sichtbarkeit von Taktlinien zu bestimmten:

```
f4 g a b
f4 g a b
% Remove bar line at the end of the current line
\once \override Score.BarLine.break-visibility = ##(#f #t #t)
\break
f4 g a b
f4 g a b
```



Obwohl alle drei Bestandteile des Vektors, mit denen `break-visibility` definiert wird, vorhanden sein müssen, haben nicht alle eine Auswirkung auf jedes Layout-Objekt, und einige Kombinationen können sogar Fehler hervorrufen. Es gelten die folgenden Einschränkungen:
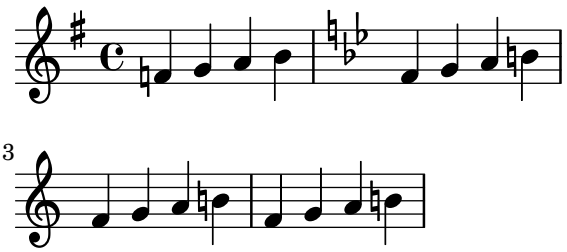
- Taktstriche können nicht zu Beginn einer Zeile gesetzt werden.
- Eine Taktzahl kann nicht zu Beginn der ersten Zeile gesetzt werden, außer wenn er nicht 1 ist.
- Schlüssel – siehe unten.
- Doppel-Prozent-Wiederholungen werden entweder alle ausgegeben oder alle unterdrückt. Mit `begin-of-line-invisible` werden sie ausgegeben, mit `all-invisible` unterdrückt.
- Tonart – siehe unten.
- Oktavierungs-Acht – siehe unten.

35.6.5 Besonderheiten

Sichtbarkeit nach expliziten Änderungen
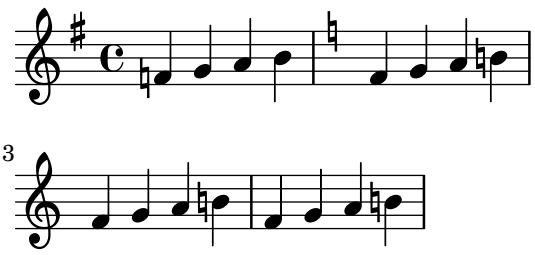
Die `break-visibility`-Eigenschaft kontrolliert die Sichtbarkeit von Tonarten und Schlüsseländerungen nur zu Beginn einer Zeile, d.h. nach einem Zeilenumbruch. Sie hat keinen Einfluss auf die Sichtbarkeit von Tonarten bzw. Schlüsseln, die nach einer expliziten Tonart- oder Schlüsseländerung in oder am Ende einer Zeile angezeigt werden. Im nächsten Beispiel ist die Tonartangabe nach dem expliziten Wechsel zu B-Dur immer noch sichtbar, obwohl `all-invisible` eingesetzt wurde:

```
\relative {
  \key g \major
  f'4 g a b
  % Try to remove all key signatures
  \override Staff.KeySignature.break-visibility = #all-invisible
  \key bes \major
  f4 g a b
  \break
  f4 g a b
  f4 g a b
}
```



Die Sichtbarkeit derartiger expliziter Tonart- und Schlüsseländerungen wird von den `explicitKeySignatureVisibility`- und `explicitClefVisibility`-Eigenschaften kontrolliert. Sie entsprechen der `break-visibility`-Eigenschaft und beide brauchen drei Boolesche Werte bzw. die oben aufgelisteten vordefinierten Funktionen als Argument, genau wie `break-visibility`. Beide sind Eigenschaft des `Staff`-Kontextes, nicht der Layout-Objekte selber, weshalb sie mit dem Befehl `\set` eingesetzt werden. Beide sind standardmäßig auf die Funktion `all-visible` gesetzt. Diese Eigenschaften kontrollieren nur die Sichtbarkeit von Tonarten bzw. Schlüssel, die von expliziten Änderungen herrühren, und haben keinen Einfluss auf Tonarten und Schlüssel zu Beginn einer Zeile – um diese zu beeinflussen, muss `break-visibility` benutzt werden.
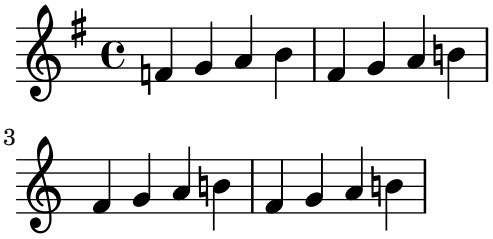
```
\relative {
  \key g \major
  f'4 g a b
  \set Staff.explicitKeySignatureVisibility = #all-invisible
  \override Staff.KeySignature.break-visibility = #all-invisible
  \key bes \major
  f4 g a b \break
  f4 g a b
  f4 g a b
}
```



Sichtbarkeit von erinnernden Versetzungszeichen

Um erinnernde Versetzungszeichen zu entfernen, die nach einer expliziten Tonartänderung auftreten, muss die `Staff`-Eigenschaft `printKeyCancellation` auf `#f` gesetzt werden:

```
\relative {
  \key g \major
  f'4 g a b
  \set Staff.explicitKeySignatureVisibility = #all-invisible
  \set Staff.printKeyCancellation = ##f
  \override Staff.KeySignature.break-visibility = #all-invisible
  \key bes \major
  f4 g a b \break
  f4 g a b
  f4 g a b
}
```



Mit diesen Veränderungen bleiben nur noch die Versetzungszeichen vor den Noten übrig um den Wechsel der Tonart anzuzeigen.

Automatische Takte

Ein Sonderfall sind die automatischen Taktstriche, die mit der Eigenschaft `automaticBars` im `Score`-Kontext ausgeschaltet werden können. Wenn sie auf `#f` gesetzt ist, werden Taktstrich nicht automatisch ausgegeben sondern müssen explizit mit dem `\bar`-Befehl eingegeben werden. Anders als bei dem `\cadenzaOn`-Befehl werden die Takte allerdings immer noch gezählt. Takterstellung wird später wieder mit diesem Zahl aufgenommen, wenn die Eigenschaft wieder auf `#t` gesetzt wird. Wenn sie den Wert `#f` hat, können Zeilenumbrüche nur an expliziten `\bar`-Befehlen auftreten.

Oktavierte Schlüssel

Das kleine Oktavierungssymbol von oktavierten Notenschlüsseln wird durch das `ClefModifier`-Layout-Objekt erstellt. Seine Sichtbarkeit wird automatisch vom `Clef`-Objekt geerbt, sodass Veränderungen von `break-visibility` des `ClefModifier`-Layout-Objekts nicht auch noch für unsichtbare Schlüssel zusätzlich vorgenommen werden müssen.

Bei expliziten Schlüsseländerungn kontrolliert die `explicitClefVisibility`-Eigenschaft wohl das Schlüsselsymbol als auch das damit verknüpfte Oktavierungssymbol.

Siehe auch

Handbuch zum Lernen: Abschnitt “Sichtbarkeit und Farbe von Objekten” in *Handbuch zum Lernen*.

35.7 Linienstile
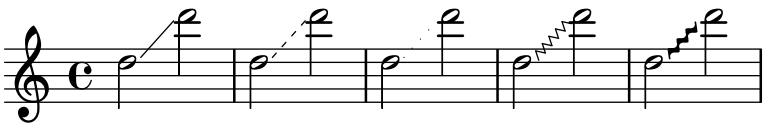
Einige Aufführungsanweisungen (z. B. *rallentando* und *accelerando* oder Triller werden als Text gesetzt und möglicherweise über mehrere Takte mit Linien fortgeführt, die teilweise gestrichelt oder gewellt sind.

Alle benutzen die gleichen Routinen wie das Glissando, um Text und Linien zu produzieren, weshalb auch eine Veränderungen der Erscheinung auf gleiche Weise vonstatten geht. Die Ausgabe erfolgt durch einen Strecker (engl. spanner), und die Routine, die ihn setzt, heißt `ly:line-interface::print`. Diese Routine bestimmt die exakte Position von zwei *Strecker-Punkten* und zeichnet eine Linie zwischen sie im gewünschten Linienstil.

Hier einige Beispiele, welche Linienstile möglich sind und wie sie verändert werden können:
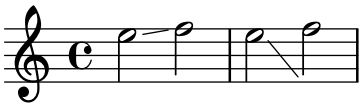
```
\relative {
  d''2 \glissando d'2
  \once \override Glissando.style = #'dashed-line
  d,2 \glissando d'2
  \override Glissando.style = #'dotted-line
  d,2 \glissando d'2
  \override Glissando.style = #'zigzag
  d,2 \glissando d'2
  \override Glissando.style = #'trill
  d,2 \glissando d'2
```

```
}
```



Die Position der Endpunkte des Streckers werden in Realzeit für jedes graphische Objekt errechnet, aber es ist möglich, sie manuell vorzugeben:

```
\relative {
  e''2 \glissando f
  \once \override Glissando.bound-details.right.Y = #-2
  e2 \glissando f
}
```



Der Wert von `Y` wird für den rechten Endpunkt auf `-2` gesetzt. Die linke Seite kann ähnlich angepasst werden, indem man `left` anstelle von `right` angibt.

Wenn `Y` nicht gesetzt ist, wird der Wert ausgehend von der vertikalen Position der linken und rechten Anbindepunkte des Streckers errechnet.

Andere Anpassungen der Strecker sind auch möglich, für Einzelheiten siehe Abschnitt 35.5 [Strecker], Seite 614.

35.8 Drehen von Objekten

Layout-Objekte und Textbeschriftungselemente können zu einem beliebigen Winkel um einen beliebigen Punkt herum gedreht werden, aber die Methode, mit der die Änderung vorgenommen werden muss, unterscheidet sich je nach Objekt.

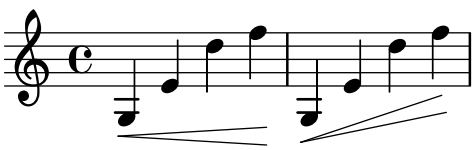
35.8.1 Drehen von Layout-Objekten

Alle Layout-Objekte, die das `grob-interface` unterstützen, können gedreht werden, indem man ihre `rotation`-Eigenschaft einstellt. Sie erhält eine Liste mit drei Einträgen: den Winkel der Drehung gegen den Uhrzeiger sowie die X- und Y-Koordinaten des Punktes relativ zum Referenzpunkt des Objekts, um welchen herum die Drehung stattfinden soll. Der Winkel der Drehung wird in Grad angegeben, die Koordinaten in Notenlinienzwischenräumen.

Der Winkel der Drehung und die Koordinaten des Drehpunktes müssen durch Ausprobieren herausgefunden werden.

Es gibt nur wenige Situationen, in welchen die Drehung eines Layout-Objektes sinnvoll ist. Das folgende Beispiel zeigt eine sinnvolle Anwendung:

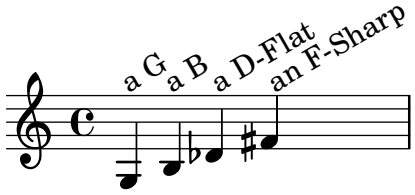
```
g4\< e' d'' f''\!
\override Hairpin.rotation = #'(15 -1 0)
g4\< e' d'' f''\!
```



35.8.2 Textbeschriftung drehen

Jede Textbeschriftung kann gedreht werden, indem vor die Anweisung der Befehl `\rotate` gesetzt wird. Der Befehl hat zwei Argumente: Den Winkel der Drehung in Grad gegen den Uhrzeiger und der Text, der gedreht dargestllt werden soll. Die Ausdehnung des Textes wird nicht gedreht, sie erhält ihren Wert von den Extrempunkten der x- und y-Koordinaten des gedrehten Textes. Im folgenden Beispiel wird die `outside-staff-priority`-Eigenschaft auf `#f` gesetzt, damit automatische Zusammenstöße nicht verhindert werden, wodurch andernfalls einige der Texte zu hoch geschoben werden würden.

```
\override TextScript.outside-staff-priority = ##f
g4^\markup { \rotate #30 "a G" }
b^\markup { \rotate #30 "a B" }
des'^\markup { \rotate #30 "a D-Flat" }
fis'^\markup { \rotate #30 "an F-Sharp" }
```



36 Fortgeschrittene Optimierungen

Dieser Abschnitt behandelt verschiedene Möglichkeiten, das Aussehen des Notenbildes zu polieren.

Siehe auch

Handbuch zum Lernen: Abschnitt “Die Ausgabe verbessern” in *Handbuch zum Lernen*, Abschnitt “Mehr Information” in *Handbuch zum Lernen*.

Notationsreferenz: Kapitel 33 [Die Referenz der Programminterna erklärt], Seite 597, Kapitel 34 [Eigenschaften verändern], Seite 601.

Erweitern: Abschnitt “Schnittstellen für Programmierer” in *Extending*.

Installierte Dateien: `scm/define-grobs.scm`.

Schnipsel: Abschnitt “Tweaks and overrides” in *Schnipsel*.

Referenz der Interna: Abschnitt “All layout objects” in *Referenz der Interna*.

36.1 Objekte ausrichten

Graphische Objekte, die das `self-alignment-interface` und/oder das `side-position-interface` unterstützen, können an einem vorher gesetzten Objekt auf verschiedene Weise ausgerichtet werden. Eine Liste derartiger Objekte findet sich in Abschnitt “self-alignment-interface” in *Referenz der Interna* und Abschnitt “side-position-interface” in *Referenz der Interna*.

Alle graphischen Objekte haben einen Referenzpunkt, eine horizontale Ausdehnung und eine vertikale Ausdehnung. Die horizontale Ausdehnung ist ein Zahlenpaar, mit dem die Verschiebung der rechten und linken Ecken ausgehend vom Referenzpunkt angegeben werden, wobei Verschiebungen nach links mit negativen Zahlen notiert werden. Die vertikale Ausdehnung ist ein Zahlenpaar, das die Verschiebung der unteren und oberen Ränder vom Referenzpunkt ausgehend angibt, wobei Verschiebungen nach unten mit negativen Zahlen notiert werden.

Die Position eines Objektes auf dem Notensystem wird mit Werten von `X-offset` und `Y-offset` angegeben. Der Wert von `X-offset` gibt die Verschiebung von der X-Koordinate des Referenzpunkts des Elternobjektes an, der Wert von `Y-offset` die Verschiebung ausgehend von der Mittellinie des Notensystemes. Die Werte von `X-offset` und `Y-offset` können direkt bestimmt werden oder durch Prozeduren errechnet werden, sodass eine Ausrichtung mit dem Elternobjekt erreicht werden kann.

> **Achtung:** Viele Objekte brauchen besondere Überlegungen zu ihrer Position, weshalb in manchen Fällen manuell gesetzte Werte von `X-offset` oder `Y-offset` ignoriert oder verändert werden können, obwohl das Objekt das `self-alignment-interface` unterstützt. Wenn man `X-offset` oder `Y-offset` auf einen festen Wert setzt, wird die entsprechende `self-alignment`-Eigenschaft ignoriert.

Ein Versetzungszeichen beispielsweise kann vertikal durch Veränderung von `Y-offset` verschoben werden, aber Änderungen von `X-offset` haben keine Auswirkung.

Übungszeichen können an trennbaren Objekten (wie Taktstrichen, Schlüsseln, Taktarten und Tonartvorzeichen) ausgerichtet werden. In `break-aligned-interface` finden sich besondere Eigenschaften, mit denen Übungszeichen an derartigen Objekten ausgerichtet werden können.

Siehe auch
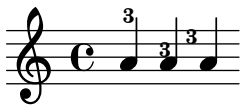
Notationshandbuch: Abschnitt 36.1.4 [Benutzung des break-alignable-interface], Seite 630.

Erweitern: Abschnitt “Callback-Funktionen” in *Extending*.

36.1.1 X-offset und Y-offset direkt setzen

Numerische Werte können den X-offset- und Y-offset-Eigenschaften vieler Objekte zugewiesen werden. Das folgende Beispiel zeigt drei Noten mit der Standardposition von Fingersatzanweisungen und die Positionen, wenn X-offset und Y-offset verändert werden.

```
a'-3
a'
-\tweak X-offset #0
-\tweak Y-offset #0
-3
a'
-\tweak X-offset #-1
-\tweak Y-offset #1
-3
```



36.1.2 Das side-position-interface benutzen

Ein Objekt, das die side-position-interface-Schnittstelle unterstützt, kann neben sein Elternobjekt gesetzt werden, sodass zwei definierte Enden der Objekte sich berühren. Das Objekt kann über, unter, rechts oder links vom Ursprungsobjekt positioniert werden. Das Ursprungsobjekt kann nicht definiert werden: es ergibt sich aus der Reihenfolge der Objekte in der Eingabe. Die meisten Objekte haben einen Notenkopf als Ursprung assoziiert.

Die Werte von side-axis und direction bestimmen, wo das Objekt platziert werden soll, wie in der Tabelle zu sehen:

side-axis- **Eigenschaft**	direction- **Eigenschaft**	**Platzierung**
0	-1	links
0	1	rechts
1	-1	unten
1	1	oben

Wenn side-axis gleich 0 ist, sollte X-offset auf die Prozedur ly:side-position-interface::x-aligned-side gesetzt werden. Diese Prozedur errechnet den richtigen Wert für X-offset, sodass das Objekt auf der rechten oder linken Seite des Ursprungs angeordnet wird, entsprechend dem Wert der direction-Eigenschaft.

Wenn side-axis gleich 1 ist, sollte Y-offset auf die Prozedur ly:side-position-interface::y-aligned-side gesetzt werden. Diese Prozedur errechnet den richtigen Wert für Y-offset, sodass das Objekt über oder unter dem Ursprungsobjekt angeordnet wird, entsprechend dem Wert der direction-Eigenschaft.

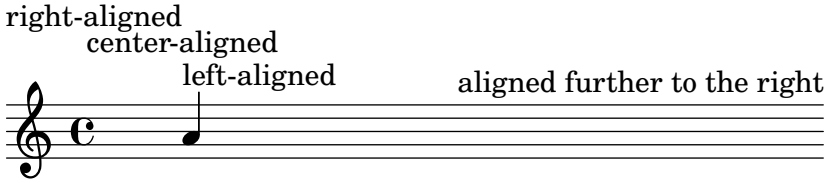
36.1.3 Das `self-alignment-interface` benutzen

Selbstausrichtende Objekte horizontal

Die horizontale Ausrichtung eines Objektes, das die `self-alignment-interface`-(Selbstausrichtungs)-Schnittstelle unterstützt, wird durch den Wert von `self-alignment-X` kontrolliert, vorausgesetzt die Eigenschaft `X-offset` des Objektes ist auf `ly:self-alignment-interface::x-aligned-on-self` gesetzt. `self-alignment-X` kann eine beliebige reale Zahl zugewiesen werden, in Einheiten der Hälfte der X-Gesamtausdehnung des Objekts. Negative Werte verschieben das Objekt nach rechts, positive nach links. Ein Wert von `0` zentriert das Objekt auf dem Referenzpunkt des Ursprungs, ein Wert von `-1` richtet die linke Ecke des Objekts am Referenzpunkt des Ursprungsobjektes aus, ein Wert von `1` richtet die rechte Ecke des Objektes am Referenzpunkt des Ursprungsobjektes aus. Die Symbole `LEFT`, `CENTER` und `RIGHT` können anstelle von `-1`, `0` und `1` eingesetzt werden.

Normalerweise würde der `\override`-Befehl benutzt werden, um die Werte von `self-alignment-X` zu verändern, aber der `\tweak`-Befehl kann benutzen, um verschiedene Anmerkungen an einer einzigen Note auszurichten:

```
a'
-\tweak self-alignment-X #-1
^"left-aligned"
-\tweak self-alignment-X #0
^"center-aligned"
-\tweak self-alignment-X #RIGHT
^"right-aligned"
-\tweak self-alignment-X #-2.5
^"aligned further to the right"
```



Objekte vertikal automatisch ausrichten
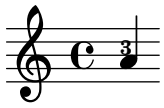
Objekte können auf ähnliche Weise auch vertikal aneinander ausgerichtet werden, wenn ihre `Y-offset`-Eigenschaft auf `ly:self-alignment-interface::y-aligned-on-self` gesetzt ist. Oft greifen jedoch auch andere Mechanismen bei der vertikalen Ausrichtung ein: Der Wert von `Y-offset` ist nur eine der Variablen, die für die Berechnung benutzt werden. Darum ist es kompliziert, den Wert für einige Objekte richtig anzupassen. Die Einheiten sind Halbe der vertikalen Ausdehnung des Objektes, welche normalerweise recht klein ist, sodass ziemlich große Werte erforderlich sein können. Der Wert `-1` richtet die untere Kante des Objekts am Referenzpunkt des Ursprungsobjektes aus, der Wert `0` richtet die Mitte des Objekts am Referenzpunkt des Ursprungsobjektes aus und der Wert `1` richtet die Oberkante des Objektes am Referenzpunkt des Ursprungsobjektes aus. Die Symbole `DOWN`, `CENTER` und `UP` können anstelle von `-1`, `0` und `1` benutzt werden.

Automatische Ausrichtung in beide Richtungen

Indem sowohl `X-offset` als auch `Y-offset` eingestellt werden, kann ein Objekt gleichzeitig in beiden Richtungen ausgerichtet werden.

Das folgende Beispiel zeigt, wie man eine Fingersatzanweisung so ausrichtet, dass sie nah am Notenkopf bleibt.

```
a'
-\tweak self-alignment-X #0.5  % move horizontally left
-\tweak Y-offset #ly:self-alignment-interface::y-aligned-on-self
-\tweak self-alignment-Y #-1  % move vertically up
-3  % third finger
```

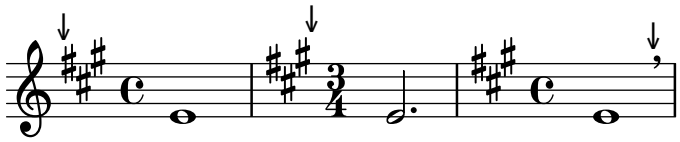


36.1.4 Benutzung des break-alignable-interface

Übungszeichen und Taktzahlen können an Notationsobjekten (ausschließlich Taktstriche) ausgerichtet werden. Zu diesen Objekten gehören ambitus, breathing-sign, clef, custos, staff-bar, left-edge, key-cancellation, key-signature und time-signature.

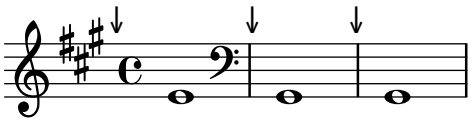
Standardmäßig werden Übungszeichen und Taktzahlen horizontal über dem Objekt zentriert:

```
% The RehearsalMark will be centered above the Clef
\override Score.RehearsalMark.break-align-symbols = #'(clef)
\key a \major
\clef treble
\mark "↓"
e1
% The RehearsalMark will be centered above the TimeSignature
\override Score.RehearsalMark.break-align-symbols = #'(time-signature)
\key a \major
\clef treble
\time 3/4
\mark "↓"
e2.
% The rehearsal mark will be centered above the Breath Mark
\override Score.RehearsalMark.break-align-symbols = #'(breathing-sign)
\key a \major
\clef treble
\time 4/4
e1
\breathe
\mark "↓"
```
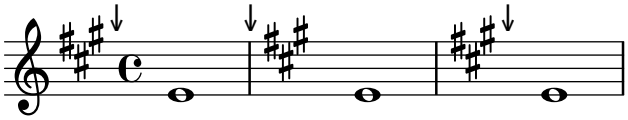


Eine Liste von möglichen Objekten zur Ausrichtung kann definiert werden. Wenn eins dieser Objekte an der aktuellen Stelle unsichtbar ist (etwa durch Einstellung von break-visibility oder die expliziten Sichtbarkeitseinstellungen von Taktart und Vorzeichen), werden Übungszeichen und Taktzahlen an dem ersten Objekt in der Liste ausgerichtet, dass sichtbar ist. Wenn keine Objekte in der Liste sichtbar sind, wird das Objekt am Taktstrich ausgerichtet. Wenn der Taktstrich unsichtbar ist, wird das Objekt an der Stelle ausgerichtet, an der sich der Taktstrich befinden würde.

```
% The RehearsalMark will be centered above the Key Signature
\override Score.RehearsalMark.break-align-symbols = #'(key-signature clef)
\key a \major
\clef treble
\mark "↓"
e1
% The RehearsalMark will be centered above the Clef
\set Staff.explicitKeySignatureVisibility = #all-invisible
\override Score.RehearsalMark.break-align-symbols = #'(key-signature clef)
\key a \major
\clef bass
\mark "↓"
gis,,1
% The rehearsal mark will be centered above the Bar Line
\set Staff.explicitKeySignatureVisibility = #all-invisible
\set Staff.explicitClefVisibility = #all-invisible
\override Score.RehearsalMark.break-align-symbols = #'(key-signature clef)
\key a \major
\clef treble
\mark "↓"
e''1
```



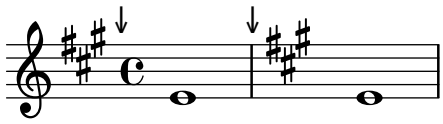
Die Ausrichtung des Übungszeichen relativ zum Notationsobjekt kann verändert werden, wie das nächste Beispiel zeigt. In einer Partitur mit vielen Systemen würde man diese Einstellung für alle Systeme vornehmen.

```
% The RehearsalMark will be centered above the KeySignature
\override Score.RehearsalMark.break-align-symbols = #'(key-signature)
\key a \major
\clef treble
\time 4/4
\mark "↓"
e1
% The RehearsalMark will be aligned with the left edge of the KeySignature
\once \override Score.KeySignature.break-align-anchor-alignment = #LEFT
\mark "↓"
\key a \major
e1
% The RehearsalMark will be aligned with the right edge of the KeySignature
\once \override Score.KeySignature.break-align-anchor-alignment = #RIGHT
\key a \major
\mark "↓"
e1
```



Das Übungszeichen kann auch nach rechts oder links um einen beliebigen Wert verschoben werden. Die Einheiten sind in Notenlinienzwischenräumen:

```
% The RehearsalMark will be aligned with the left edge of the KeySignature
% and then shifted right by 3.5 staff-spaces
\override Score.RehearsalMark.break-align-symbols = #'(key-signature)
\once \override Score.KeySignature.break-align-anchor = #3.5
\key a \major
\mark "↓"
e1
% The RehearsalMark will be aligned with the left edge of the KeySignature
% and then shifted left by 2 staff-spaces
\once \override Score.KeySignature.break-align-anchor = #-2
\key a \major
\mark "↓"
e1
```



36.2 Vertikale Gruppierung der grafischen Objekte („grob“s)

Die graphischen Objekte `VerticalAlignment` und `VerticalAxisGroup` funktionieren zusammen. `VerticalAxisGroup` gruppiert unterschiedliche Objekte wie Notensysteme, Gesangstext usw. zusammen. `VerticalAlignment` richtet die unterschiedlichen Objektgruppen dann aneinander aus. Es gibt normalerweise nur ein `VerticalAlignment` in einer Partitur, aber jedes Notensystem, Gesangstext usw. hat eine eigene `VerticalAxisGroup`.

36.3 stencils verändern
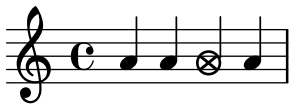
Alle Layout-Objekte haben eine `stencil`-(Stempel-)Eigenschaft, die ein Teil von `grob-interface` ist. Diese Eigenschaft ist normalerweise als eine Funktion definiert, die auf das jeweilige Objekt angepasst ist und das Symbol erstellt, dass dann im Druckbild erscheint. Beispielsweise die Standardeinstellung für die `stencil`-Eigenschaft von `MultiMeasureRest` (Ganztaktpausenobjekt) ist `ly:multi-measure-rest::print`.

Das Standardsymbol für jedes Objekt kann ersetzt werden, indem man die `stencil`-Eigenschaft verändert, sodass sie auf eine andere, speziell geschriebene Prozedur verweist. Das erfordert einen hohen Grad an Kenntnis der LilyPond-Interna, aber es gibt einen einfacheren Weg, mit dem man oft vergleichbarere Ergebnisse erzielen kann.

Dieser Weg besteht darin, die `stencil`-Eigenschaft auf die Prozedur zu verweisen, die Text ausgibt: `ly:text-interface::print` und eine `text`-Eigenschaft zu dem Objekt hinzuzufügen, in welcher dann die Textbeschriftung definiert wird, mit der das entsprechende Symbol dargestellt wird. Aufgrund der Flexibilität der Textbeschriftung ist hier sehr viel möglich. Siehe zu Details insbesondere Abschnitt 8.2.4 [Graphische Notation innerhalb einer Textbeschriftung], Seite 241.

Das folgende Beispiel zeigt diese Methode, indem das Symbol der Notenköpfe in ein Kreuz innerhalb eines Kreises umgewandelt wird.

```
XinO = {
  \once \override NoteHead.stencil = #ly:text-interface::print
  \once \override NoteHead.text = \markup {
    \combine
      \halign #-0.7 \draw-circle #0.85 #0.2 ##f
      \musicglyph "noteheads.s2cross"
  }
}
\relative {
  a' a \XinO a a
}
```



Alle Schriftzeichen in der Feta-Glyphe können mit dem `\musicglyph`-Befehl erreicht werden. Siehe auch Abschnitt A.8 [Die Emmentaler-Schriftart], Seite 658.

Siehe auch

Notationsreferenz: Abschnitt 8.2.4 [Graphische Notation innerhalb einer Textbeschriftung], Seite 241, Abschnitt 8.2 [Text formatieren], Seite 233, Abschnitt A.10 [Textbeschriftungsbefehle], Seite 678, Abschnitt A.8 [Die Emmentaler-Schriftart], Seite 658.

36.4 Formen verändern
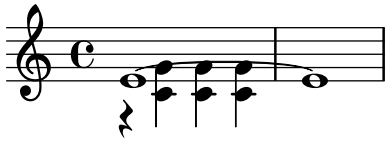
36.4.1 Bögen verändern

Binde-, Legato- und Phrasierungsbögen werden als Bézierkurven dritter Ordnung gezeichnet. Wenn die Form eines automatischen Bogens nicht optimal ist, kann sie manuell verändert werdne, indem man die vier erforderlichen Kontrollpunkte angibt.

Bézierkurven dritter Ordnung (auch als quadratische Bézierkurven bezeichnet) werden durch vier Kontrollpunkte definiert. Der erste und vierte Kontrollpunkt geben Beginn und Ende der Kurve an. Die zwei Punkte dazwischen werden benutzt, um die Form der Kurve zu bestimmen. Im Internet gibt es Animationen, die illustrieren, wie eine derartige Kurve gezeichnet wird, aber die folgende Beschreibung kann hilfreich sein. Die Kurve beginnt am ersten Kontrollpunkt in Richtung des zweiten, wobei sie sich schrittweise krümmt um zum dritten Kontrollpunkt zu gelangen, von wo aus sie sich weiter zum vierten Punkt hin krümmt. Die Form der Kurve wird vollständig von den vier Punkten definiert.

Hier ein Beispiel eines Falles, in dem der Bogen nicht optimal erscheint, und wo auch `\tieDown` das Problem nicht lösen würde.

```
<<
  { e'1~ 1 }
\\
  \relative { r4 <g' c,> <g c,> <g c,> }
>>
```
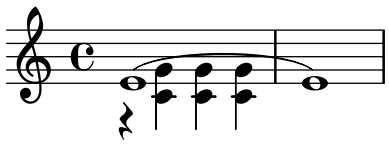


Eine Möglichkeit, diesen Bogen zu verbessern, ist es, seine Kontrollpunkte manuell zu verändern:

Die Koordinaten von Bézierkontrollpunkten werden in Notenlinienzwischenräumen angegeben. Die X-Achse ist relativ zum Referenzpunkt der Note, an die der Bogen angefügt wird, und die Y-Achse relativ zur Mittellinie des Notensystems. Die Koordinaten werden als eine Liste von vier Paaren an realen Dezimalzahlen eingegeben. Eine Möglichkeit ist es, die Koordinaten der zwei Endpunkte zu schätzen und dann die zwei Zwischenpunkte zu erraten. Die optimalen Werte können nur durch Ausprobieren gefunden werden.

Es lohnt sich daran zu denken, dass eine symmetrische Kurve symmetrische Kontrollpunkte benötigt, und dass Bézierkurven die nützliche Eigenschaft haben, dass eine Transformation der Kurve wie eine Übersetzung, Drehung oder Skalierung der Kurve erreicht werden kann, indem man die gleiche Skalierung auf die Kontrollpunkte anwendet.

In dem obigen Beispiel geben folgende Werte einen zufriedenstellenden Bogen – Achtung: der Befehl muss direkt vor dem Beginn der Note gesetzt werden, an die der (Binde-)Bogen angehängt wird.

```
<<
  {
  \once \override Tie.control-points = #'((1 . -1) (3 . 0.6) (12.5 . 0.6) (14.5 . -1))
  e1~ 1
  }
\\
  { r4 <g c,> <g c,> <g c,>  }
>>
```



Bekannte Probleme und Warnungen

Es ist nicht möglich, die Form von Bögen anhand ihrer `control-points`-Eigenschaft zu verändern, wenn mehrere Binde- oder Legatobögen zum gleichen musikalischen Moment auftreten, nicht einmal mit dem `\tweak`-Befehl. Die Eigenschaft `tie-configuration` von `TieColumn` kann jedoch verändert werden, sodass Startlinie und Richtung wie benötigt platziert werden.

Siehe auch

Referenz der Interna: Abschnitt “TieColumn” in *Referenz der Interna*.

36.5 Reine und unreine Container

Unreine und reine Container (engl. unpure/pure containers) sind nützlich, wenn man die Berechnungen der Platzierungen für die *Y-Achse* verändern will, insbesondere für `Y-offset` und `Y-extent`. Mit diesen Containern kann die Veränderung durch eine Scheme-Funktion anstelle einer direkten Zahl oder eines Paares vorgenommen werden.

Für bestimmte Grobs basiert die Eigenschaft `Y-extent` auf der `stencil`-Eigenschaft. Wenn diese mit `\override` verändert werden soll, braucht man eine zusätzliche Veränderung von `Y-extent` mit einem unreinen-reinen Container. Wenn eine Funktion `Y-offset` und/oder `Y-extent` verändert, wird angenommen, dass dadurch Zeilenumbruchsberechnungen zu früh während der Kompilation aufgerufen werden. Die Funktion wird also überhaupt nicht ausgewertet (und gibt also normalerweise den Wert ‘`0`’ oder ‘`'(0 . 0)`’ zurück), wodurch sich Zusammenstöße ergeben können. Eine „saubere“ Funktion beeinflusst keine Eigeschaften, Objekte oder Grob-Suizide, weshalb ihre Werte, die sich auf `Y-axis` beziehen, richtig berechnet werden.

Es gibt zuzeit etwa 30 Funktionen, die schon als „sauber“ erachtet werden, und unsauberesaubere Container sind eine Möglichkeit, auch Funktionen, die sich nicht auf dieser Liste befinden, als „sauber“ zu markieren. Die „saubere“ Funktion wird ausgewertet, *bevor* Seitenumbruch stattfindet, sodass die horizontale Platzierung „rechtzeitig“ stattfindet. Die „unsaubere“ Funktion wird dann *nach* dem Seitenumbruch ausgewertet.

> **Achtung:** Da es schwierig ist, immer sicher zu sein, welche Funktionen sich auf dieser Liste befinden, wird empfohlen, dass die selbsterstellten „sauberen“ Funktionen nicht die Grobs `Beam` oder `VerticalAlignment` einsetzen.

Ein unsauberer-sauberer Container wird wie folgend erstellt:

`(ly:make-unpure-pure-container f0 f1)`

wobei `f0` eine Fuktion ist, die *n* Arguments braucht (*n >= 1*) und deren erstes Argument immer der Grob sein muss. Das ist die Funktion, die das eigentliche Resultat ausgibt. *f1* ist die Funktion, die als „sauber“ bezeichnet wird, und braucht *n + 2* Argumente. Wiederum muss das erste Argument immer der Grob sein, aber das erste und zweite Argument sind „Beginn-“ und „Endeargumente“.

start (Beginn) und *end* (Ende) sind absichtlich nur Platzhalter, die nur für die Strecker gelten (etwa `Hairpin` oder `Beam`), die unterschiedliche Höhenberechnungen je nach beginnender und endender Note ausgeben können.

Der Rest sind andere Argumente für die erste Funktion (es können auch Null sein, wenn *n = 1*).

Die Ergebnisse der zweiten Funktion werden als Näherungswert des benötigten Wertes benutzt, welche dann von der ersten Funktion eingesetzt wird, um den wirklichen Wert auszugeben, mit dem dann sehr viel später im Layoutprozess die Platzierung justiert werden soll.
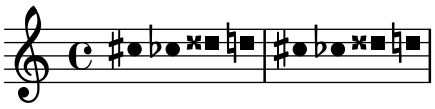
```
#(define (square-line-circle-space grob)
(let* ((pitch (ly:event-property (ly:grob-property grob 'cause) 'pitch))
       (notename (ly:pitch-notename pitch)))
 (if (= 0 (modulo notename 2))
     (make-circle-stencil 0.5 0.0 #t)
     (make-filled-box-stencil '(0 . 1.0)
                              '(-0.5 . 0.5)))))

squareLineCircleSpace = {
  \override NoteHead.stencil = #square-line-circle-space
}

smartSquareLineCircleSpace = {
  \squareLineCircleSpace
  \override NoteHead.Y-extent =
   #(ly:make-unpure-pure-container
      ly:grob::stencil-height
      (lambda (grob start end) (ly:grob::stencil-height grob)))
}

\new Voice \with { \remove Stem_engraver }
\relative c'' {
  \squareLineCircleSpace
  cis4 ces disis d
```

```
  \smartSquareLineCircleSpace
  cis4 ces disis d
}
```



Im ersten Takt weiß die Layoutmaschine ohne den unsauberen-sauberen Container nicht die Breite des Notenkopfes und lässt ihn deshalb mit den Versetzungszeichen zusammenstoßen. Im zweiten Takt, mit unreinen-reinen Containern, weiß die Layoutmaschine die Breite des Notenkopfes und vermeidet den Zusammenstoß, indem die Zeile entsprechend verlängert wird.

Normalerweise können für eine einfache Berechnungen fast identische Funktionen für den „unsauberen“ und „sauberen“ Teil benutzt werden, indem nur die Zahl der Argumente und die Reichweite der Funktion verändert wird.

Achtung: Wenn eine Funktion als „sauber“ bezeichnet ist und das aber nicht ist, können unerwartete Ergebnisse auftreten.

37 Musikfunktionen benutzen

Wenn Optimierungen von unterschiedlichen musikalischen Ausdrücken wiederverwendet werden sollen, bietet es sich oft an, den „Optimierungsanteil“ einer *musikalischen Funktion* zu erstellen. In diesem Abschnitt sollen nur *Ersetzungen* erklärt werden, wo es darum geht, eine Variable mit einem Stück LilyPond-Code zu ersetzen. Andere komplexere Funktionen werden beschrieben in Abschnitt “Musikalische Funktionen” in *Extending*.

37.1 Syntax der Ersetzungsfunktion

Es ist einfach eine Funktion zu erstellen, die eine Variable in LilyPond-Code umwandelt. Die generelle Form dieser Funktionen ist:

```
Funktion =
#(define-music-function
     (Arg1 Arg2 ...)
     (Typ1? Typ2? ...)
   #{
     ...Noten...
   #})
```

wobei

ArgN	*n*tes Argument
TypN?	ein Scheme *Typenprädikat*, für das *ArgN* den Wert `#t` ausgibt.
...Noten...	normale LilyPond-Eingabe, wobei `$` (wenn nur LilyPond-Konstruktionen erlaubt sind) oder `#` (um es als Scheme-Wert oder Argument einer musikalischen Funktionen oder als Noten innerhalb von Notenlisten einzusetzen) benutzt wird, um Argumente zu referenzieren (etwa ‘`$Arg1`’).

Die `parser` und `location`-Argumente sind zwingend und werden in einigen fortgeschrittenen Situationen benutzt, wie sie im „Erweitern“-Handbuch beschrieben werden (siehe Abschnitt “Musikalische Funktionen” in *Extending*). In Ersetzungsfunktionen gehen Sie einfach sicher, dass sie die beiden Wörter auch mit aufnehmen.

Die Liste der Typenprädikate ist auch notwendig. Einige der häufigsten Typenprädikate, die in musikalischen Funktionen benutzt werden, sind:

```
boolean?
cheap-list?  (benutze anstelle von ,list¿ für schnelleres Kompilieren)
ly:duration?
ly:music?
ly:pitch?
markup?
number?
pair?
string?
symbol?
```

Eine Liste aller Typprädikate findet sich unter Abschnitt A.18 [Vordefinierte Typprädikate], Seite 788. Eigene Typprädikate sind auch erlaubt.

Siehe auch

Notationsreferenz: Abschnitt A.18 [Vordefinierte Typprädikate], Seite 788.

Erweitern: Abschnitt “Musikalische Funktionen” in *Extending*.

Installierte Dateien: `lily/music-scheme.cc`, `scm/c++.scm`, `scm/lily.scm`.
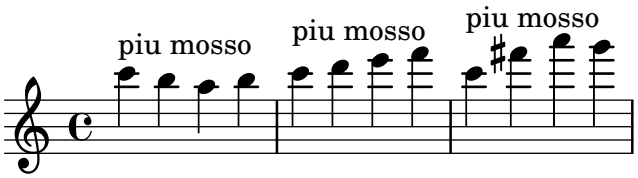
37.2 Beispiele der Ersetzungsfunktionen

Dieser Abschnitt zeigt einige Beispiele von Ersetzungsfunktionen. Sie sind nicht vollständig, sondern sollen einige der Möglichkeiten von einfachen Ersetzungsfunktionen aufzeigen.

Im ersten Beispiel wird eine Funktione definiert, die das Verschieben von `TextScript` erleichtert:

```
padText =
#(define-music-function
     (padding)
     (number?)
   #{
     \once \override TextScript.padding = #padding
   #})
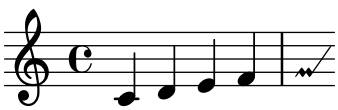
\relative {
  c'''4^"piu mosso" b a b
  \padText #1.8
  c4^"piu mosso" d e f
  \padText #2.6
  c4^"piu mosso" fis a g
}
```



Neben Zahlen können auch musikalische Ausdrücke wie Noten als Argumente für musikalische Funktionen eingesetzt werden:

```
custosNote =
#(define-music-function
     (note)
     (ly:music?)
   #{
     \tweak NoteHead.stencil #ly:text-interface::print
     \tweak NoteHead.text
        \markup \musicglyph "custodes.mensural.u0"
     \tweak Stem.stencil ##f
     #note
   #})

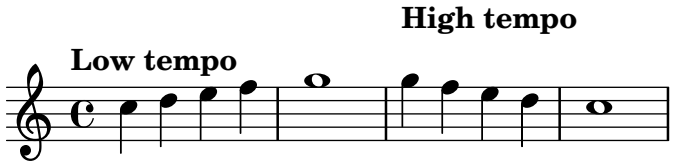
\relative { c'4 d e f \custosNote g }
```



Ersetzungsfunktionen mit mehrfachen Argumenten können definiert werden:

```
tempoPadded =
#(define-music-function
     (padding tempotext)
     (number? markup?)
   #{
     \once \override Score.MetronomeMark.padding = #padding
     \tempo \markup { \bold #tempotext }
   #})

\relative {
  \tempo \markup { "Low tempo" }
  c''4 d e f g1
  \tempoPadded #4.0 "High tempo"
  g4 f e d c1
}
```

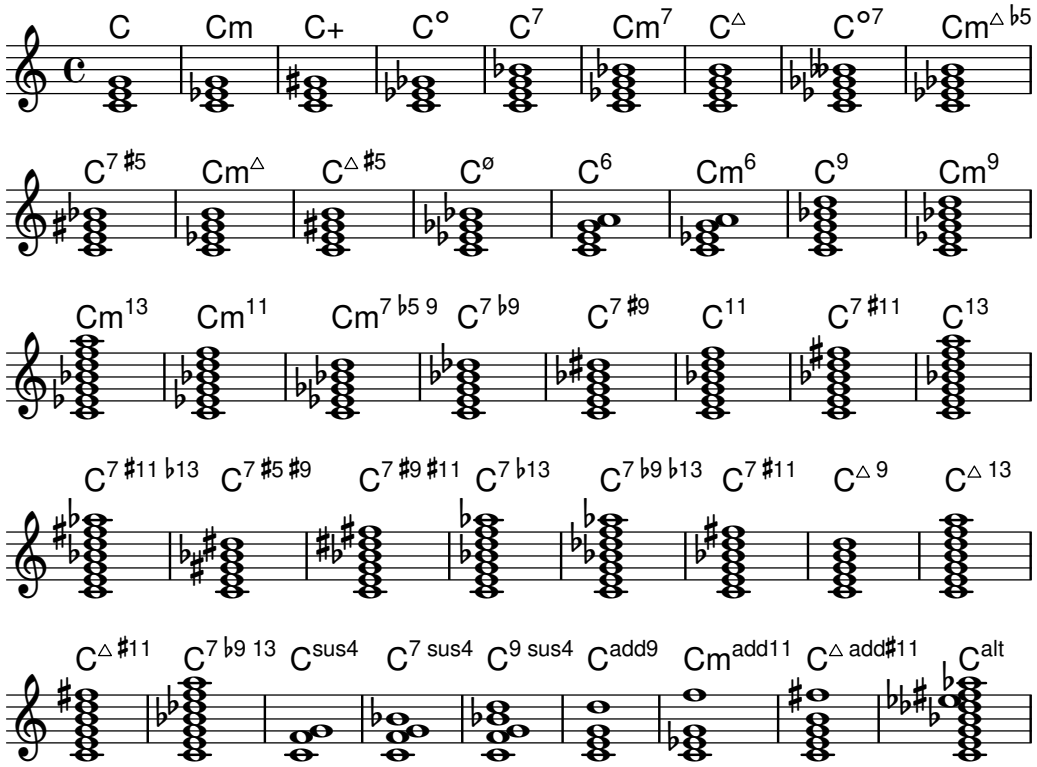


Anhänge

Anhang A Notationsübersicht

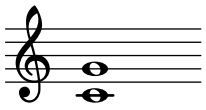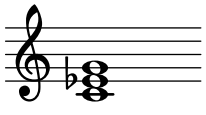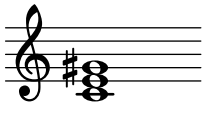
A.1 Liste der Akkordbezeichnungen

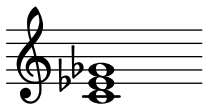
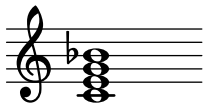
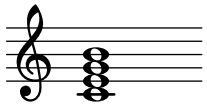
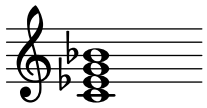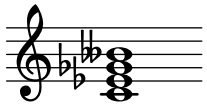
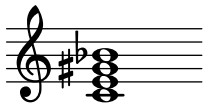
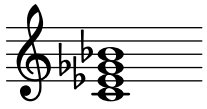
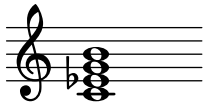
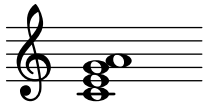
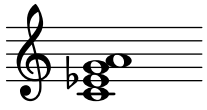
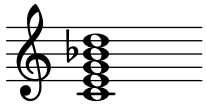
Die Tabelle zeigt die zwei üblichen Möglichkeiten, wie Akkordbezeichnungen ausgegeben werden. Es wird auch die entsprechende Note ausgegeben.

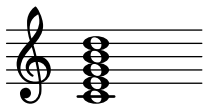
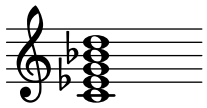
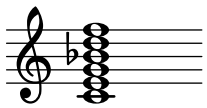
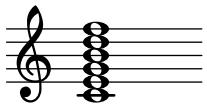
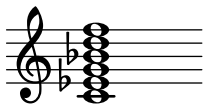
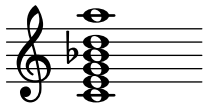
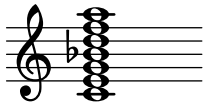
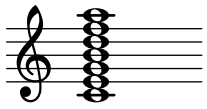
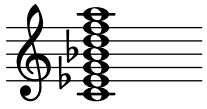
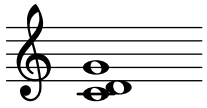
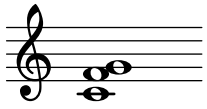


A.2 Übliche Akkord-Variablen

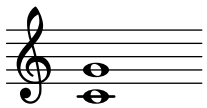
Die Tabelle zeigt Modifikatoren für Akkorde, die im `\chordmode`-Modus benutzt werden können, um übliche Akkordkonstrukte zu notieren.

Akkordtyp	**Intervalle**	**Modifikator(en)**	**Beispiel**
Dur	große Terz, Quinte	`5` oder nichts	
Moll	kleine Terz, Quinte	`m` oder `m5`	
Übermäßig	Große Terz, übermäßige Quinte	`aug`	

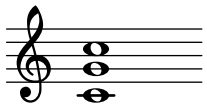
Vermindert	Kleine Terz, verminderte Quinte	`dim`	
Dominantsieben	Durdreiklang, kleine Septime	`7`	
Große Septime	Durdreiklang, große Septime	`maj7` oder `maj`	
Kleine Septime	Molldreiklang, kleine Septime	`m7`	
Verminderte Septime	Verminderter Dreiklang, verminderte Septime	`dim7`	
Übermäßige Septime	Übermäßiger Dreiklang, kleine Septime	`aug7`	
halbverminderte Septime	Verminderter Dreklang, kleine Sept	`m7.5-`	
Kleine MollSept	Molldreiklang, große Septime	`m7+`	
Große Sexte	Durdreiklang, Sexte	`6`	
Kleine Sexte	Molldreiklang, Sexte	`m6`	
Dominantnone	Dominantsept, große None	`9`	

Dur-None	Große None, große Septime	`maj9`	
Moll-None	Große None, kleine Septime	`m9`	
Dominantundezime	Dominantnone, reine Undezime	`11`	
Durundezime	Große None, reine Undezime	`maj11`	
Mollundezime	Kleine None, reine Undezime	`m11`	
Dominant-13	Dominantnone, große 13	`13`	
Dominant-13	Dominant-Undezime, große 13	`13.11`	
Dur-13	Große Undezime, große 13	`maj13.11`	
Moll-13	Kleine Undezime, große 13	`m13.11`	
Sekundakkord	große Sekunde, reine Quinte	`sus2`	
Quartakkord	reine Quarte, reine Quinte	`sus4`	

Powerakkord (zweistimmig)	Perfekte Quinte	`1.5`

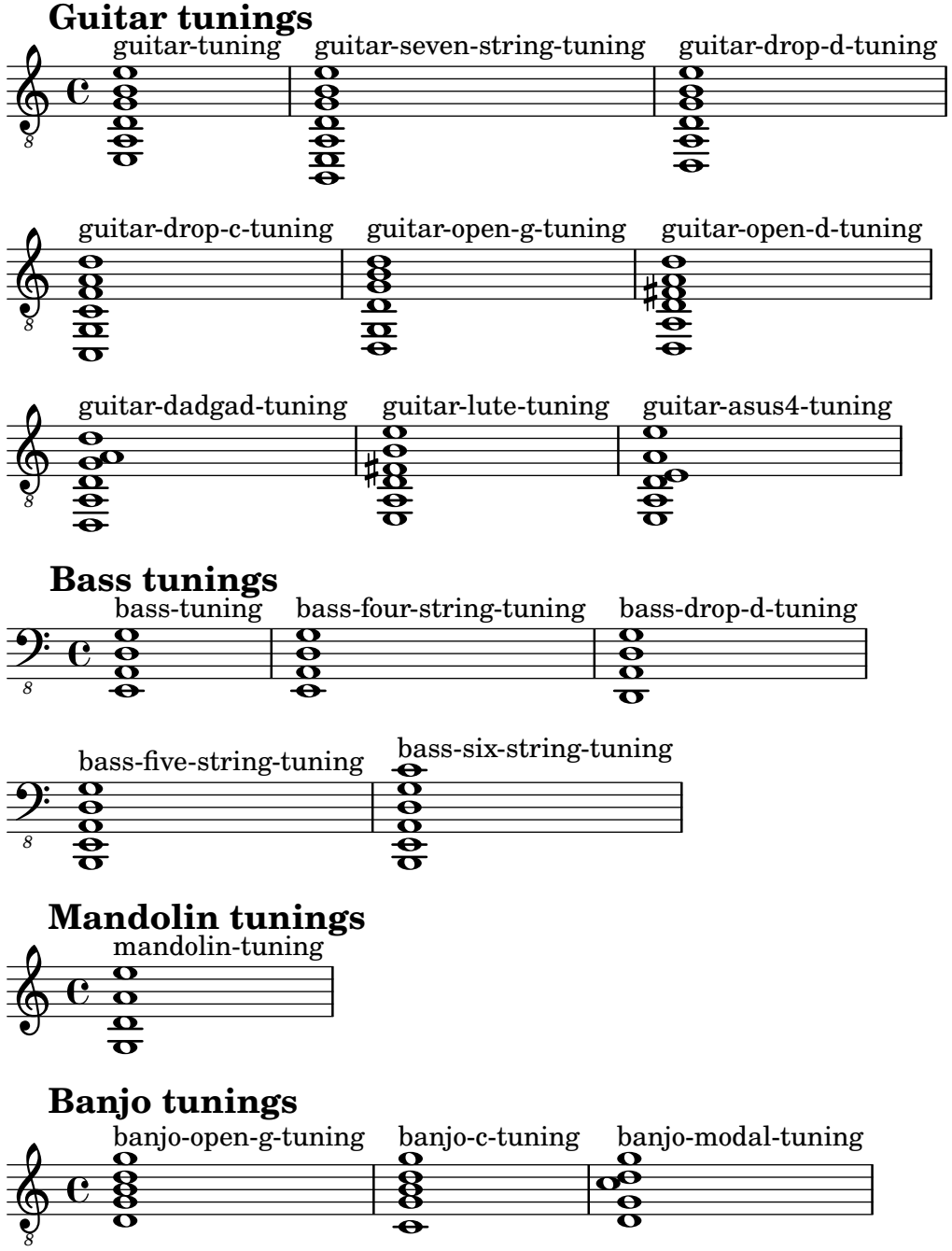


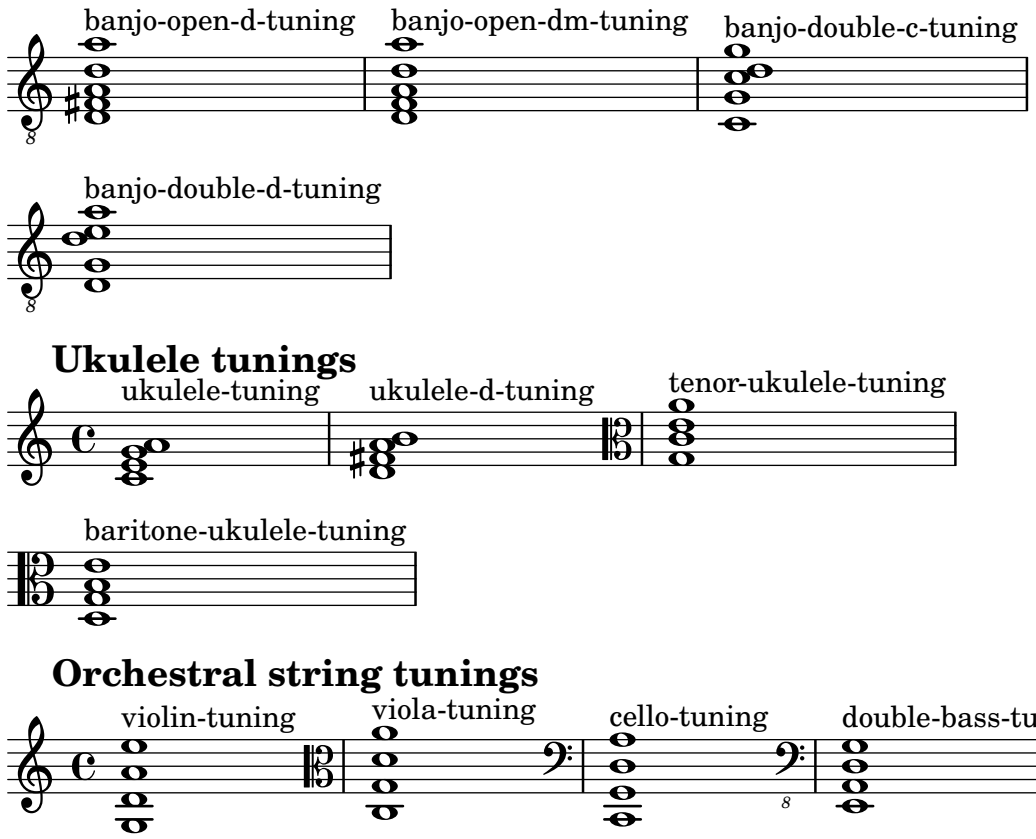
Powerakkord (dreistimmig)	Perfekte Quinte, Oktave	`1.5.8`



A.3 Vordefinierte Saitenstimmungen

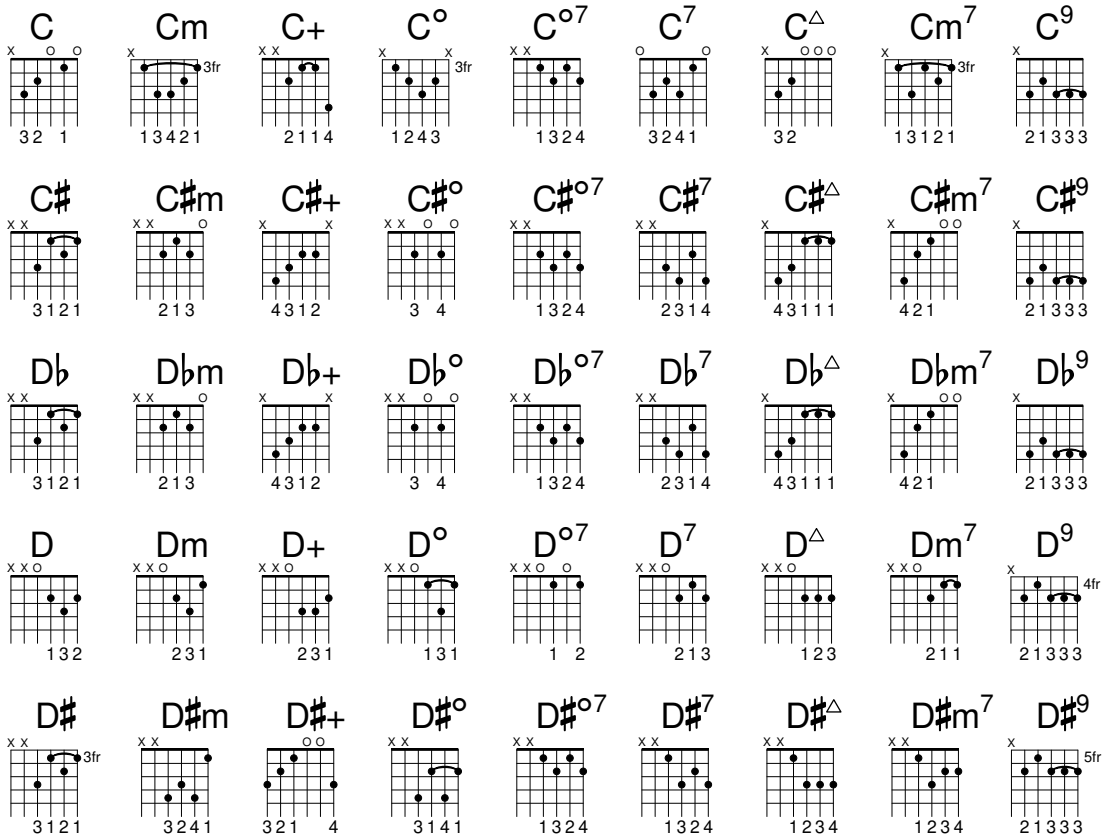
Die folgende Tabelle zeigt die vordefinierten Saitenstimmungen:

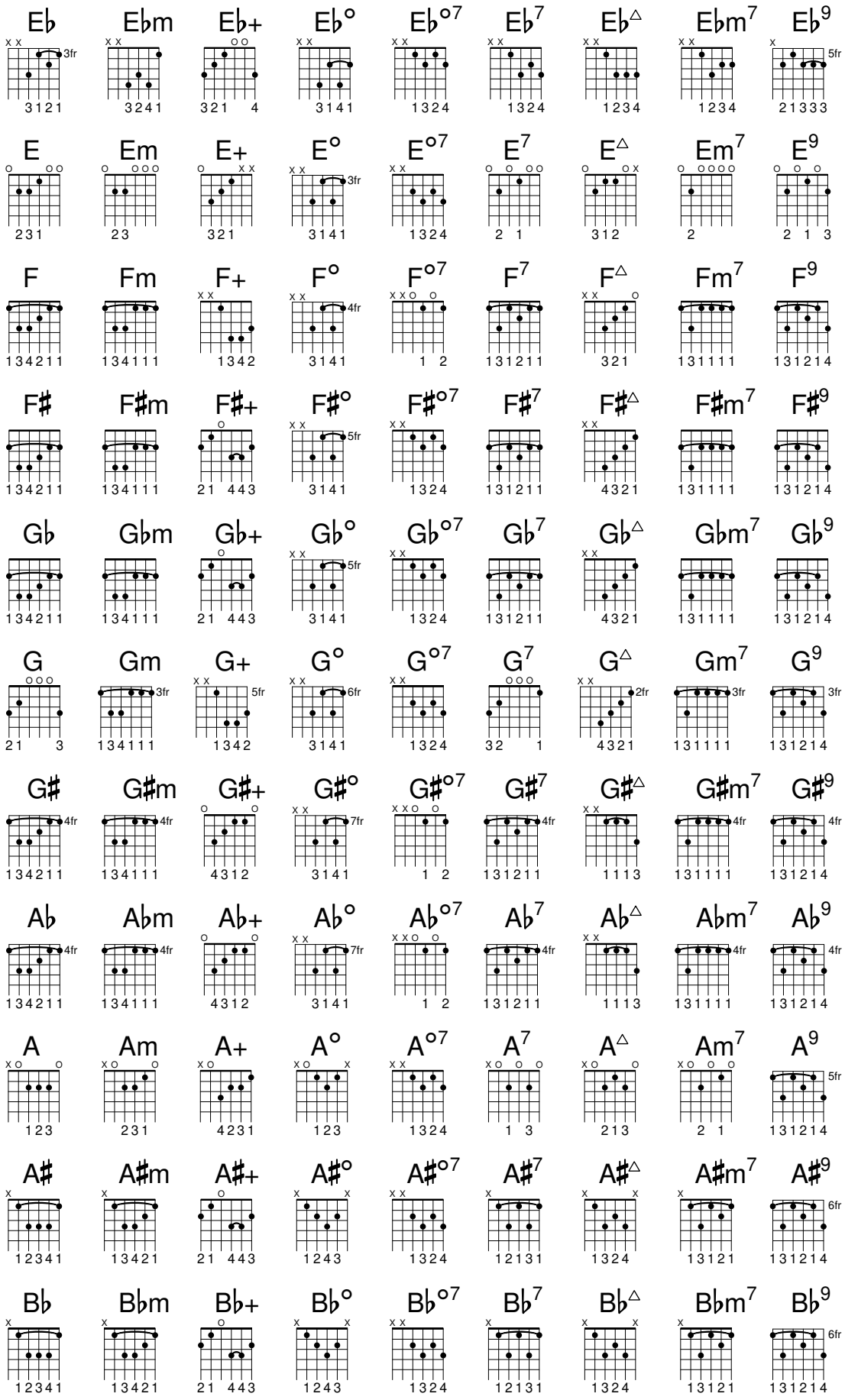


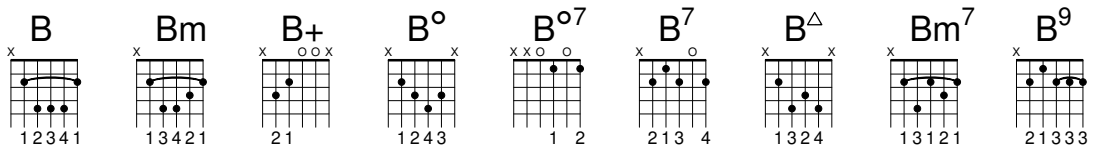


A.4 Die vordefinierten Bund-Diagramme

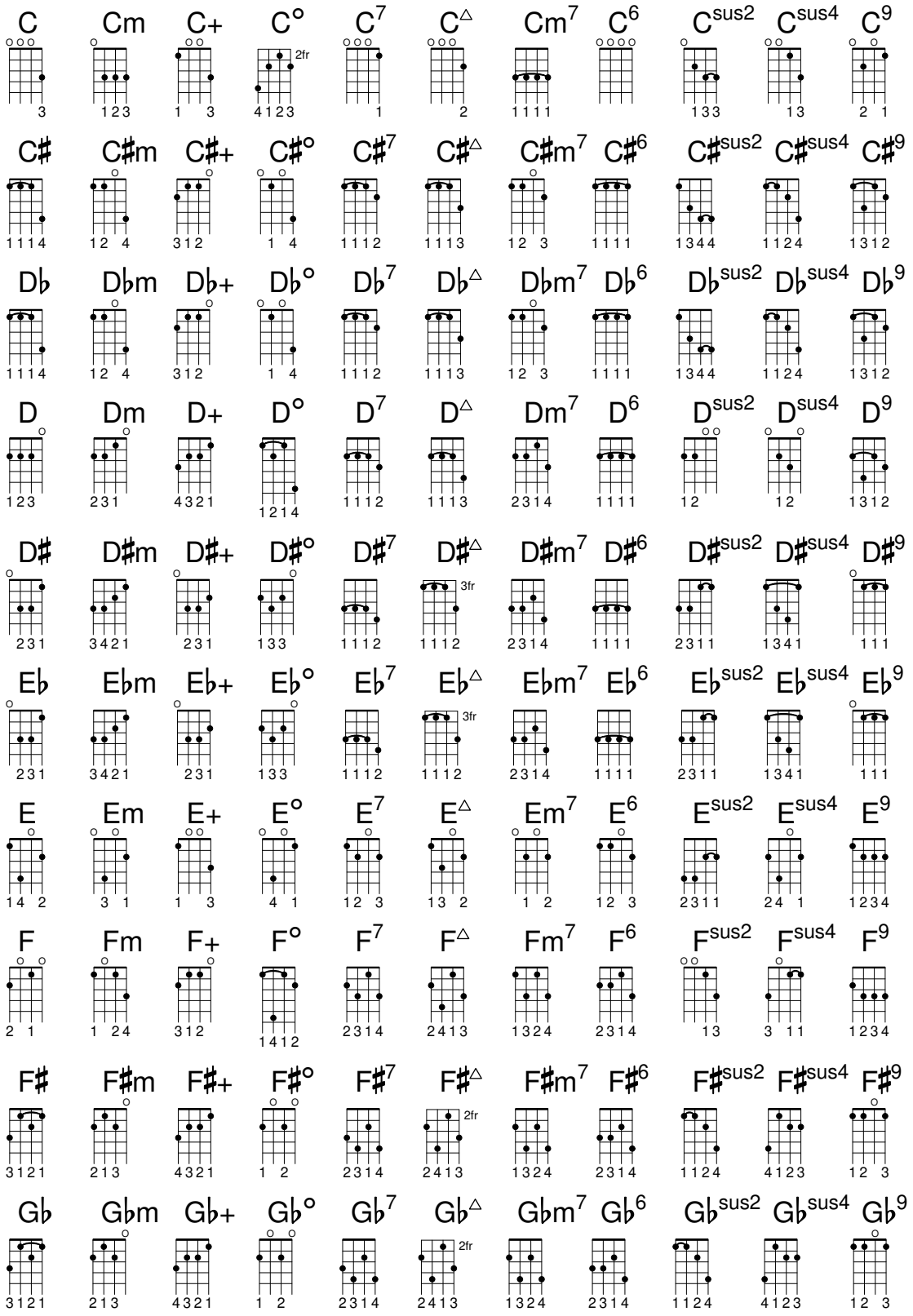
Die Tabelle zeigt alle vordefinierten Bunddiagramme für Gitarre.

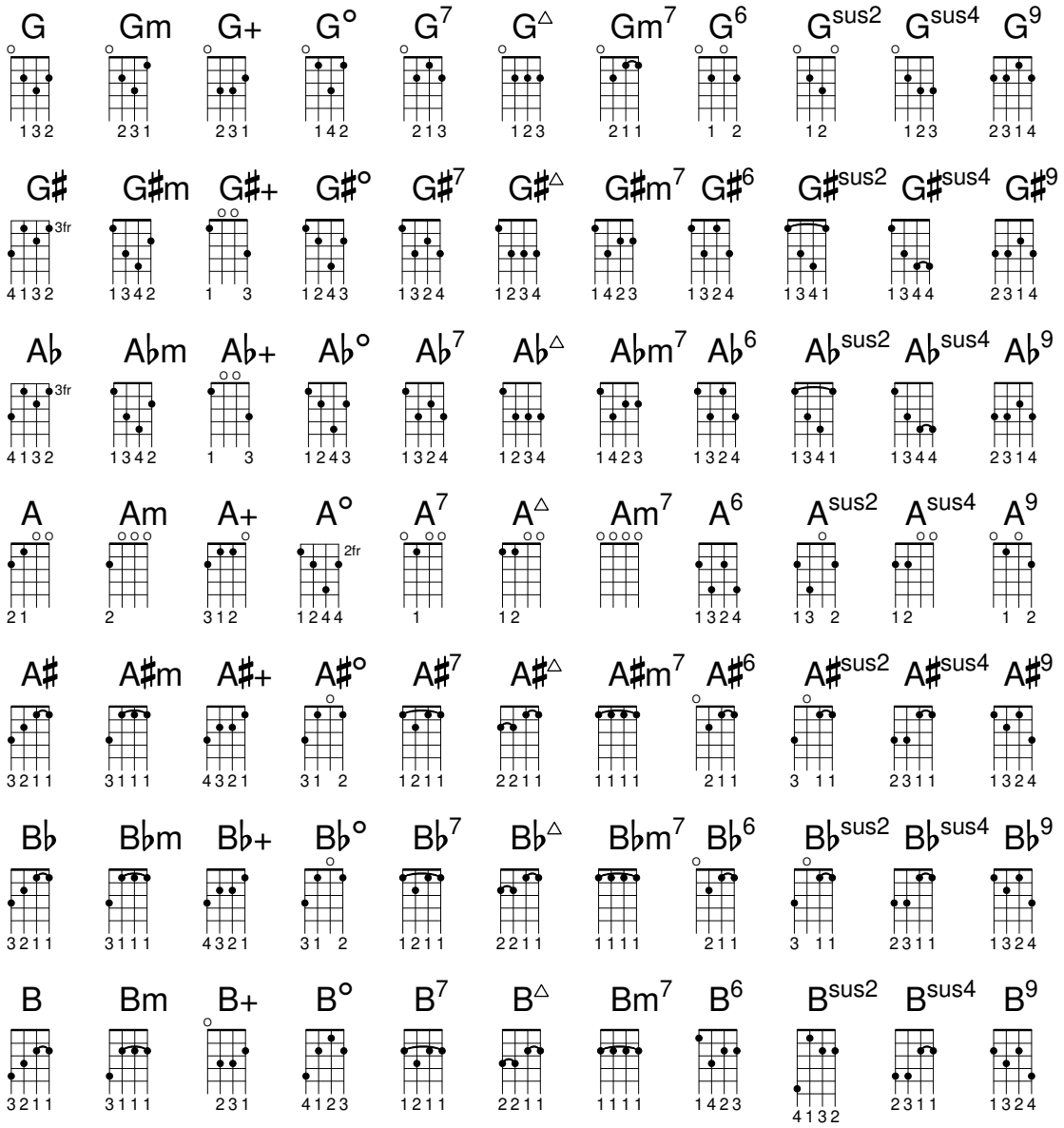



E♭ E♭m E♭+ E♭° E♭°7 E♭7 E♭△ E♭m7 E♭9
E Em E+ E° E°7 E7 E△ Em7 E9
F Fm F+ F° F°7 F7 F△ Fm7 F9
F♯ F♯m F♯+ F♯° F♯°7 F♯7 F♯△ F♯m7 F♯9
G♭ G♭m G♭+ G♭° G♭°7 G♭7 G♭△ G♭m7 G♭9
G Gm G+ G° G°7 G7 G△ Gm7 G9
G♯ G♯m G♯+ G♯° G♯°7 G♯7 G♯△ G♯m7 G♯9
A♭ A♭m A♭+ A♭° A♭°7 A♭7 A♭△ A♭m7 A♭9
A Am A+ A° A°7 A7 A△ Am7 A9
A♯ A♯m A♯+ A♯° A♯°7 A♯7 A♯△ A♯m7 A♯9
B♭ B♭m B♭+ B♭° B♭°7 B♭7 B♭△ B♭m7 B♭9

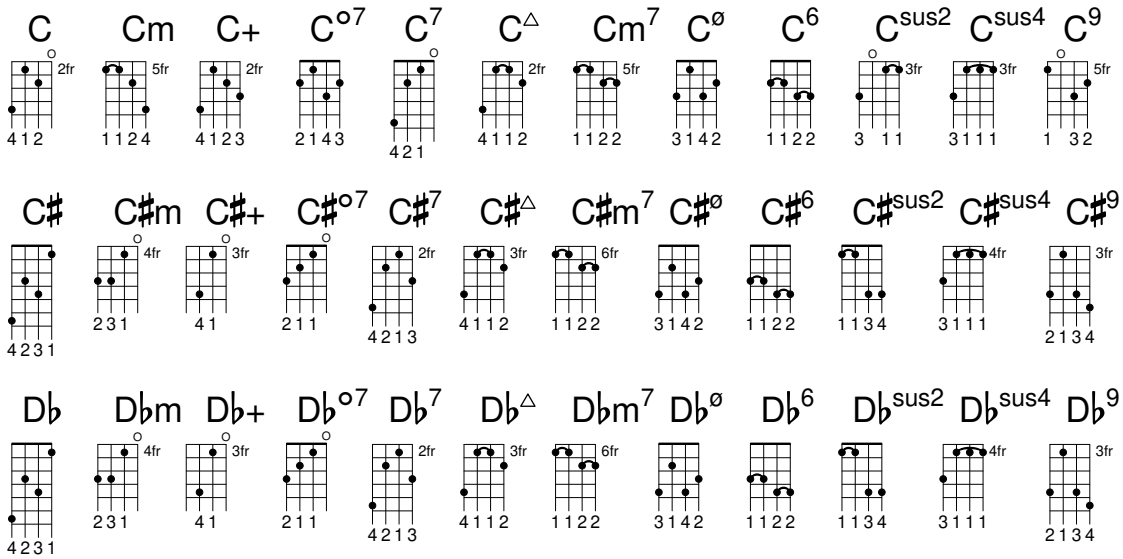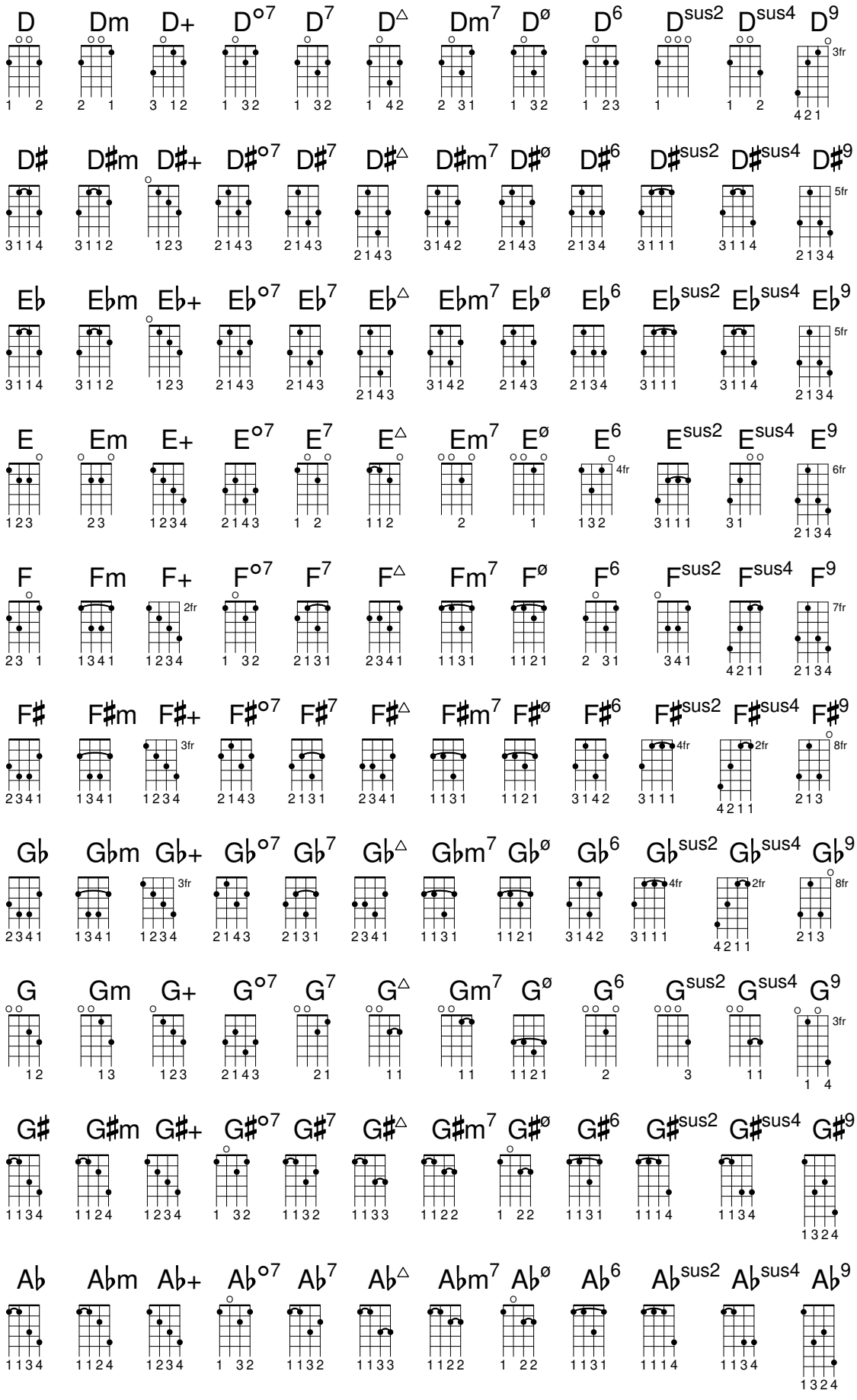


Die folgende Tabelle zeigt vordefinierte Bunddiagramme für Ukulele.

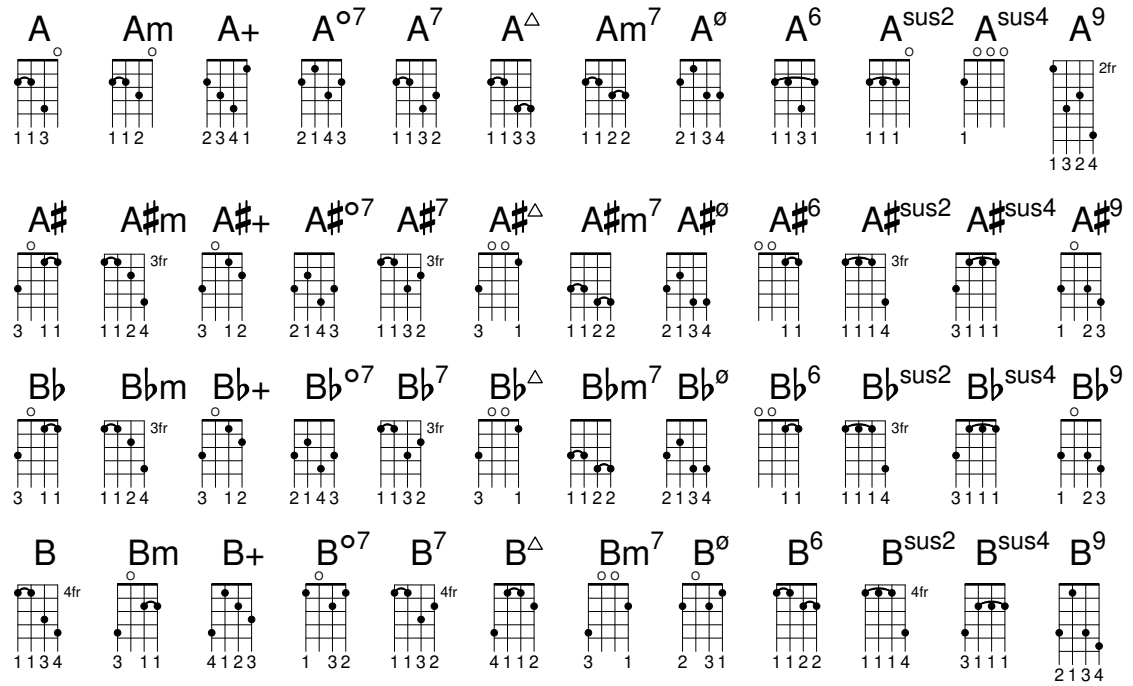




Die folgende Tabelle zeigt die vordefinierten Bunddiagramme für Mandoline.




D Dm D+ D°7 D7 D△ Dm7 Dø D6 Dsus2 Dsus4 D9
D♯ D♯m D♯+ D♯°7 D♯7 D♯△ D♯m7 D♯ø D♯6 D♯sus2 D♯sus4 D♯9
E♭ E♭m E♭+ E♭°7 E♭7 E♭△ E♭m7 E♭ø E♭6 E♭sus2 E♭sus4 E♭9
E Em E+ E°7 E7 E△ Em7 Eø E6 Esus2 Esus4 E9
F Fm F+ F°7 F7 F△ Fm7 Fø F6 Fsus2 Fsus4 F9
F♯ F♯m F♯+ F♯°7 F♯7 F♯△ F♯m7 F♯ø F♯6 F♯sus2 F♯sus4 F♯9
G♭ G♭m G♭+ G♭°7 G♭7 G♭△ G♭m7 G♭ø G♭6 G♭sus2 G♭sus4 G♭9
G Gm G+ G°7 G7 G△ Gm7 Gø G6 Gsus2 Gsus4 G9
G♯ G♯m G♯+ G♯°7 G♯7 G♯△ G♯m7 G♯ø G♯6 G♯sus2 G♯sus4 G♯9
A♭ A♭m A♭+ A♭°7 A♭7 A♭△ A♭m7 A♭ø A♭6 A♭sus2 A♭sus4 A♭9



A.5 Vordefinierte Papierformate

Papierformate sind definiert in `scm/paper.scm`

Die „ISO 216“ A-Serie (DinA)

`"a10"` (26 x 37 mm)

`"a9"` (37 x 52 mm)

`"a8"` (52 x 74 mm)

`"a7"` (74 x 105 mm)

`"a6"` (105 x 148 mm)

`"a5"` (148 x 210 mm)

`"a4"` (210 x 297 mm)

`"a3"` (297 x 420 mm)

`"a2"` (420 x 594 mm)

`"a1"` (594 x 841 mm)

`"a0"` (841 x 1189 mm)

Die „ISO 216“ B-Serie (DinB)

`"b10"` (31 x 44 mm)

`"b9"` (44 x 62 mm)

`"b8"` (62 x 88 mm)

`"b7"` (88 x 125 mm)

`"b6"` (125 x 176 mm)

`"b5"` (176 x 250 mm)

`"b4"` (250 x 353 mm)

`"b3"` (353 x 500 mm)

`"b2"` (500 x 707 mm)

`"b1"` (707 x 1000 mm)

`"b0"` (1000 x 1414 mm)

Zwei erweiterte Größen wie definiert in „DIN 476“

`"4a0"` (1682 x 2378 mm)

`"2a0"` (1189 x 1682 mm)

„ISO 269“ Standard-C-Serie (DinC)

`"c10"` (28 x 40 mm)

`"c9"` (40 x 57 mm)

`"c8"` (57 x 81 mm)

`"c7"` (81 x 114 mm)

`"c6"` (114 x 162 mm)

`"c5"` (162 x 229 mm)

`"c4"` (229 x 324 mm)

`"c3"` (324 x 458 mm)

`"c2"` (458 x 648 mm)

`"c1"` (648 x 917 mm)

`"c0"` (917 x 1297 mm)

Nordarmerikanische Papierformate

`"junior-legal"`
(8.0 x 5.0 in)

`"legal"` (8.5 x 14.0 in)

`"ledger"` (17.0 x 11.0 in)

`"letter"` (8.5 x 11.0 in)

`"tabloid"`
(11.0 x 17.0 in)

`"11x17"` (11.0 x 17.0 in)

`"17x11"` (17.0 x 11.0 in)

Government-letter der IEEE Printer Working Group, für Kinder zum Schreiben

`"government-letter"`
(8 x 10.5 in)

`"government-legal"`
(8.5 x 13.0 in)

`"philippine-legal"`
(8.5 x 13.0 in)

ANSI-Formate

`"ansi a"` (8.5 x 11.0 in)

`"ansi b"` (17.0 x 11.0 in)

`"ansi c"` (17.0 x 22.0 in)

`"ansi d"` (22.0 x 34.0 in)

`"ansi e"` (34.0 x 44.0 in)

`"engineering f"`
(28.0 x 40.0 in)

Nordamerikanische architektonische Formate

`"arch a"` (9.0 x 12.0 in)

`"arch b"` (12.0 x 18.0 in)

`"arch c"` (18.0 x 24.0 in)

`"arch d"` (24.0 x 36.0 in)

`"arch e"` (36.0 x 48.0 in)

`"arch e1"` (30.0 x 42.0 in)

Alte Formate, die noch in Großbritannien benützt werden

`"statement"`
(5.5 x 8.5 in)

`"half letter"`
(5.5 x 8.5 in)

`"quarto"` (8.0 x 10.0 in)

`"octavo"` (6.75 x 10.5 in)

`"executive"`
(7.25 x 10.5 in)

`"monarch"`
(7.25 x 10.5 in)

`"foolscap"`
(8.27 x 13.0 in)

`"folio"` (8.27 x 13.0 in)

`"super-b"`
(13.0 x 19.0 in)

`"post"` (15.5 x 19.5 in)

`"crown"` (15.0 x 20.0 in)

`"large post"`
(16.5 x 21.0 in)

`"demy"` (17.5 x 22.5 in)

`"medium"` (18.0 x 23.0 in)

`"broadsheet"`
(18.0 x 24.0 in)

`"royal"` (20.0 x 25.0 in)

`"elephant"`
(23.0 x 28.0 in)

`"double demy"`
(22.5 x 35.0 in)

`"quad demy"`
(35.0 x 45.0 in)

`"atlas"` (26.0 x 34.0 in)

`"imperial"`
(22.0 x 30.0 in)

`"antiquarian"`
(31.0 x 53.0 in)

Auf PA4 basierende Formate

`"pa0"` (840 x 1120 mm)

`"pa1"` (560 x 840 mm)

`"pa2"` (420 x 560 mm)

`"pa3"` (280 x 420 mm)

`"pa4"` (210 x 280 mm)

`"pa5"` (140 x 210 mm)

`"pa6"` (105 x 140 mm)

`"pa7"` (70 x 105 mm)

`"pa8"` (52 x 70 mm)

`"pa9"` (35 x 52 mm)

`"pa10"` (26 x 35 mm)

In Südostasien und Australien benützt

`"f4"` (210 x 330 mm)

Benützt für sehr kleine `@lilypond`-Beispiele in der Dokumentation, basierend auf A8 quer

`"a8landscape"`
(74 x 52 mm)

A.6 MIDI-Instrumente

Hier eine Liste von Musikinstrumentenbezeichnungen, die als Name für `midiInstrument` benutzt werden können. Die Anordnung der Instrumente entspricht den 128 Prgrammnummern des MIDI-Standards.

```
acoustic grand          contrabass          lead 7 (fifths)
bright acoustic         tremolo strings     lead 8 (bass+lead)
electric grand          pizzicato strings   pad 1 (new age)
honky-tonk              orchestral harp     pad 2 (warm)
electric piano 1        timpani             pad 3 (polysynth)
electric piano 2        string ensemble 1   pad 4 (choir)
harpsichord             string ensemble 2   pad 5 (bowed)
clav                    synthstrings 1      pad 6 (metallic)
celesta                 synthstrings 2      pad 7 (halo)
glockenspiel            choir aahs          pad 8 (sweep)
music box               voice oohs          fx 1 (rain)
```

```
vibraphone                synth voice          fx 2 (soundtrack)
marimba                   orchestra hit        fx 3 (crystal)
xylophone                 trumpet              fx 4 (atmosphere)
tubular bells             trombone             fx 5 (brightness)
dulcimer                  tuba                 fx 6 (goblins)
drawbar organ             muted trumpet        fx 7 (echoes)
percussive organ          french horn          fx 8 (sci-fi)
rock organ                brass section        sitar
church organ              synthbrass 1         banjo
reed organ                synthbrass 2         shamisen
accordion                 soprano sax          koto
harmonica                 alto sax             kalimba
concertina                tenor sax            bagpipe
acoustic guitar (nylon)   baritone sax         fiddle
acoustic guitar (steel)   oboe                 shanai
electric guitar (jazz)    english horn         tinkle bell
electric guitar (clean)   bassoon              agogo
electric guitar (muted)   clarinet            steel drums
overdriven guitar         piccolo              woodblock
distorted guitar          flute                taiko drum
guitar harmonics          recorder             melodic tom
acoustic bass             pan flute            synth drum
electric bass (finger)    blown bottle         reverse cymbal
electric bass (pick)      shakuhachi           guitar fret noise
fretless bass             whistle              breath noise
slap bass 1               ocarina              seashore
slap bass 2               lead 1 (square)      bird tweet
synth bass 1              lead 2 (sawtooth)    telephone ring
synth bass 2              lead 3 (calliope)    helicopter
violin                    lead 4 (chiff)       applause
viola                     lead 5 (charang)     gunshot
cello                     lead 6 (voice)
```

A.7 Liste der Farben

Normale Farben

Die Syntax zur Benutzung findet sich im Abschnitt Abschnitt 7.1.4 [Farbige Objekte], Seite 218.

```
black       white          red         green
blue        cyan           magenta     yellow
grey        darkred        darkgreen   darkblue
darkcyan    darkmagenta    darkyellow
```

X-Farbbezeichnungen

X-Farbbezeichnungen haben verschiedene Varianten:

Alle Bezeichnungen, die als einziges Wort mit Großbuchstaben geschrieben werden (bspw. ‚LightSlateBlue‘), können auch von Leerzeichen getrennt geschrieben werden (also ‚light slate blue‘).

Das Wort ‚grey‘ kann in jedem Fall auch ‚gray‘ geschrieben werden (bspw. ‚DarkSlateGray‘).

Manche Bezeichnungen können auch ein numerales Suffix tragen (etwa ‚LightSalmon4‘).

Farben ohne eine numerale Endung

```
snow            GhostWhite      WhiteSmoke      gainsboro       FloralWhite
OldLace         linen           AntiqueWhite    PapayaWhip      BlanchedAlmond
bisque          PeachPuff       NavajoWhite     moccasin        cornsilk
ivory           LemonChiffon    seashell        honeydew        MintCream
azure           AliceBlue       lavender        LavenderBlush   MistyRose
white           black           DarkSlateGrey   DimGrey         SlateGrey
LightSlateGrey  grey            LightGrey       MidnightBlue    navy
NavyBlue        CornflowerBlue  DarkSlateBlue   SlateBlue       MediumSlateBlue
LightSlateBlue  MediumBlue      RoyalBlue       blue            DodgerBlue
DeepSkyBlue     SkyBlue         LightSkyBlue    SteelBlue       LightSteelBlue
LightBlue       PowderBlue      PaleTurquoise   DarkTurquoise   MediumTurquoise
turquoise       cyan            LightCyan       CadetBlue       MediumAquamarine
aquamarine      DarkGreen       DarkOliveGreen  DarkSeaGreen    SeaGreen
MediumSeaGreen  LightSeaGreen   PaleGreen       SpringGreen     LawnGreen
green           chartreuse      MediumSpringGreen       GreenYellow     LimeGreen
YellowGreen     ForestGreen     OliveDrab       DarkKhaki       khaki
PaleGoldenrod   LightGoldenrodYellow    LightYellow     yellow  gold
LightGoldenrod  goldenrod       DarkGoldenrod   RosyBrown       IndianRed
SaddleBrown     sienna          peru            burlywood       beige
wheat           SandyBrown      tan             chocolate       firebrick
brown           DarkSalmon      salmon          LightSalmon     orange
DarkOrange      coral           LightCoral      tomato          OrangeRed
red             HotPink         DeepPink        pink            LightPink
PaleVioletRed   maroon          MediumVioletRed VioletRed       magenta
violet          plum            orchid          MediumOrchid    DarkOrchid
DarkViolet      BlueViolet      purple          MediumPurple    thistle
DarkGrey        DarkBlue        DarkCyan        DarkMagenta     DarkRed
LightGreen
```

Farben mit einer numeralen Endung

Für die folgenden Bezeichnungen kann das Suffix N durch eine Zahl von 1–4 ersetzt werden.

```
snowN           seashellN       AntiqueWhiteN   bisqueN         PeachPuffN
NavajoWhiteN    LemonChiffonN   cornsilkN       ivoryN          honeydewN
LavenderBlushN  MistyRoseN      azureN          SlateBlueN      RoyalBlueN
blueN           DodgerBlueN     SteelBlueN      DeepSkyBlueN    SkyBlueN
LightSkyBlueN   LightSteelBlueN LightBlueN      LightCyanN      PaleTurquoiseN
CadetBlueN      turquoiseN      cyanN           aquamarineN     DarkSeaGreenN
SeaGreenN       PaleGreenN      SpringGreenN    greenN          chartreuseN
OliveDrabN      DarkOliveGreenN khakiN          LightGoldenrodN LightYellowN
yellowN         goldN           goldenrodN      DarkGoldenrodN  RosyBrownN
IndianRedN      siennaN         burlywoodN      wheatN          tanN
chocolateN      firebrickN      brownN          salmonN         LightSalmonN
orangeN         DarkOrangeN     coralN          tomatoN         OrangeRedN
redN            DeepPinkN       HotPinkN        pinkN           LightPinkN
PaleVioletRedN  maroonN         VioletRedN      magentaN        orchidN
plumN           MediumOrchidN   DarkOrchidN     purpleN         MediumPurpleN
thistleN
```

Grauskala

Eine Grauskala kann mit der Bezeichnung

greyN

erstellt werden, wobei N eine Zahl von 0–100 darstellt.

A.8 Die Emmentaler-Schriftart

Die folgenden Symbole sind als Emmentaler-Schriftart verfügbar; auf sie kann direkt zugegriffen werden, indem man die übliche Textbeschriftung benutzt. \musicglyph greift direkt auf die Notationsschriftart zu

```
g^\markup {\musicglyph "scripts.segno" }
```

or

```
\markup {\musicglyph "five"}.
```

Siehe auch Abschnitt 8.2 [Text formatieren], Seite 233.

A.8.1 Notenschlüssel-Glyphen

clefs.C		clefs.C_change	
clefs.varC		clefs.varC_change	
clefs.F		clefs.F_change	
clefs.G		clefs.G_change	
clefs.GG		clefs.GG_change	
clefs.tenorG		clefs.tenorG_change	
clefs.percussion		clefs.percussion_change	
clefs.varpercussion		clefs .varpercussion_change	
clefs.tab		clefs.tab_change	

A.8.2 Taktart-Glyphen

timesig.C44		timesig.C44.text	
timesig.C22		timesig.C22.text	
timesig.X			

A.8.3 Zahlen-Glyphen

plus	+	comma	,
hyphen	-	equal	=
period	.	figuredash	‒
endash	–	parenleft	(
parenright	)	slash	/
zero	0	one	1
two	2	three	3
four	4	four.alt	4
five	5	six	6
seven	7	seven.alt	7
eight	8	nine	9
fixedwidth.zero	0	fixedwidth.one	1
fixedwidth.two	2	fixedwidth.three	3
fixedwidth.four	4	fixedwidth.four.alt	4
fixedwidth.five	5	fixedwidth.six	6
fixedwidth.seven	7	fixedwidth.seven.alt	7
fixedwidth.eight	8	fixedwidth.nine	9
fattened.zero	0	fattened.one	1

fattened.two 

fattened.three 

fattened.four 

fattened.four.alt 

fattened.five 

fattened.six 

fattened.seven 

fattened.seven.alt 

fattened.eight 

fattened.nine 

fattened.fixedwidth.zero 

fattened.fixedwidth.one 

fattened.fixedwidth.two 

fattened
.fixedwidth.three 

fattened.fixedwidth.four 

fattened.fixedwidth
.four.alt 

fattened.fixedwidth.five 

fattened.fixedwidth.six 

fattened
.fixedwidth.seven 

fattened.fixedwidth
.seven.alt 

fattened
.fixedwidth.eight 

fattened.fixedwidth.nine 

u2007

u2009

u200A

A.8.4 Versetzungszeichen-Glyphen

accidentals.sharp 

accidentals
.sharp.figbass 

accidentals
.sharp.arrowup 

accidentals
.sharp.arrowdown 

accidentals
.sharp.arrowboth 

accidentals.sharp
.slashslash.stem 

accidentals.sharp
.slashslashslash.stemstem 

accidentals.sharp
.slashslashslash.stem 

accidentals
.sharp.slash.stem 

accidentals.sharp
.slashslash.stemstemstem 

accidentals.doublesharp		accidentals .doublesharp.figbass	
accidentals.natural		accidentals .natural.figbass	
accidentals .natural.arrowup		accidentals .natural.arrowdown	
accidentals .natural.arrowboth		accidentals.flat	
accidentals.flat.figbass		accidentals.flat.arrowup	
accidentals .flat.arrowdown		accidentals .flat.arrowboth	
accidentals.flat.slash		accidentals.flat .slashslash	
accidentals .mirroredflat.flat		accidentals.mirroredflat	
accidentals .mirroredflat.backslash		accidentals.flatflat	
accidentals .flatflat.figbass		accidentals .flatflat.slash	
accidentals.sharp.sori		accidentals.flat.koron	
accidentals.rightparen	)	accidentals.leftparen	(

A.8.5 Standard-Notenkopf-Glyphen

noteheads.uM2		noteheads.dM2	
noteheads.sM1		noteheads.s0	
noteheads.s1		noteheads.s2	

A.8.6 Spezielle Notenkopf-Glyphen

noteheads.sM1double		noteheads.s0diamond	
noteheads.s1diamond		noteheads.s2diamond	
noteheads.s0triangle		noteheads.s1triangle	
noteheads.s2triangle		noteheads.s0slash	
noteheads.s1slash		noteheads.s2slash	
noteheads.s0cross		noteheads.s1cross	
noteheads.s2cross		noteheads.s2xcircle	
noteheads.d0arrow		noteheads.u0arrow	
noteheads.d2arrow		noteheads.u2arrow	
noteheads.s0harmonic		noteheads.s2harmonic	

A.8.7 Geformte Notenkopf-Glyphen

noteheads.s0do		noteheads.s1do	
noteheads.s2do		noteheads.s0doThin	
noteheads.s1doThin		noteheads.s2doThin	
noteheads.s0re		noteheads.s1re	
noteheads.s2re		noteheads.s0reThin	
noteheads.s1reThin		noteheads.s2reThin	
noteheads.s0mi		noteheads.s1mi	

noteheads.s2mi		noteheads.s0miMirror	
noteheads.s1miMirror		noteheads.s2miMirror	
noteheads.s0miThin		noteheads.s1miThin	
noteheads.s2miThin		noteheads.u0fa	
noteheads.d0fa		noteheads.u1fa	
noteheads.d1fa		noteheads.u2fa	
noteheads.d2fa		noteheads.u0faThin	
noteheads.d0faThin		noteheads.u1faThin	
noteheads.d1faThin		noteheads.u2faThin	
noteheads.d2faThin		noteheads.s0sol	
noteheads.s1sol		noteheads.s2sol	
noteheads.s0la		noteheads.s1la	
noteheads.s2la		noteheads.s0laThin	
noteheads.s1laThin		noteheads.s2laThin	
noteheads.s0ti		noteheads.s1ti	
noteheads.s2ti		noteheads.s0tiThin	
noteheads.s1tiThin		noteheads.s2tiThin	
noteheads.u0doFunk		noteheads.d0doFunk	
noteheads.u1doFunk		noteheads.d1doFunk	

noteheads.u2doFunk		noteheads.d2doFunk	
noteheads.u0reFunk		noteheads.d0reFunk	
noteheads.u1reFunk		noteheads.d1reFunk	
noteheads.u2reFunk		noteheads.d2reFunk	
noteheads.u0miFunk		noteheads.d0miFunk	
noteheads.u1miFunk		noteheads.d1miFunk	
noteheads.s2miFunk		noteheads.u0faFunk	
noteheads.d0faFunk		noteheads.u1faFunk	
noteheads.d1faFunk		noteheads.u2faFunk	
noteheads.d2faFunk		noteheads.s0solFunk	
noteheads.s1solFunk		noteheads.s2solFunk	
noteheads.s0laFunk		noteheads.s1laFunk	
noteheads.s2laFunk		noteheads.u0tiFunk	
noteheads.d0tiFunk		noteheads.u1tiFunk	
noteheads.d1tiFunk		noteheads.u2tiFunk	
noteheads.d2tiFunk		noteheads.s0doWalker	
noteheads.u1doWalker		noteheads.d1doWalker	
noteheads.u2doWalker		noteheads.d2doWalker	
noteheads.s0reWalker		noteheads.u1reWalker	

noteheads.d1reWalker		noteheads.u2reWalker	
noteheads.d2reWalker		noteheads.s0miWalker	
noteheads.s1miWalker		noteheads.s2miWalker	
noteheads.s0faWalker		noteheads.u1faWalker	
noteheads.d1faWalker		noteheads.u2faWalker	
noteheads.d2faWalker		noteheads.s0laWalker	
noteheads.s1laWalker		noteheads.s2laWalker	
noteheads.s0tiWalker		noteheads.u1tiWalker	
noteheads.d1tiWalker		noteheads.u2tiWalker	
noteheads.d2tiWalker			
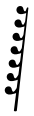
A.8.8 Pausen-Glyphen

rests.0		rests.1	
rests.0o		rests.1o	
rests.M3		rests.M2	
rests.M1		rests.M1o	
rests.2		rests.2classical	
rests.2z		rests.3	
rests.4		rests.5	
rests.6		rests.7	

rests.8



rests.9



rests.10



A.8.9 Fähnchen-Glyphen

flags.u3



flags.u4



flags.u5



flags.u6



flags.u7



flags.u8



flags.u9



flags.u10



flags.d3



flags.d4



flags.d5



flags.d6



flags.d7



flags.d8



flags.d9



flags.d10



flags.stackedu3



flags.stackedu4



flags.stackedu5



flags.stackedu6



flags.stackedu7



flags.stackedu8



flags.stackedu9 

flags.stackedu10 

flags.stackedd3 

flags.stackedd4 

flags.stackedd5 

flags.stackedd6 

flags.stackedd7 

flags.stackedd8 

flags.stackedd9 

flags.stackedd10 

flags.ugrace 

flags.dgrace 

A.8.10 Punkt-Glyphen

dots.dot

A.8.11 Dynamik-Glyphen

space

f 

m 

n 

p 

r 

s 

z 

A.8.12 Schrift-Glyphen

scripts.ufermata 

scripts.dfermata 

scripts
.uhenzeshortfermata 

scripts
.dhenzeshortfermata 

scripts
.uhenzelongfermata 

scripts
.dhenzelongfermata 

scripts.ushortfermata		scripts.dshortfermata	
scripts .uveryshortfermata		scripts .dveryshortfermata	
scripts.ulongfermata		scripts.dlongfermata	
scripts.uverylongfermata		scripts.dverylongfermata	
scripts.thumb		scripts.sforzato	
scripts.espr		scripts.staccato	
scripts.ustaccatissimo		scripts.dstaccatissimo	
scripts.tenuto		scripts.uportato	
scripts.dportato		scripts.umarcato	
scripts.dmarcato		scripts.open	
scripts.halfopen		scripts.halfopenvertical	
scripts.stopped		scripts.uupbow	
scripts.dupbow		scripts.udownbow	
scripts.ddownbow		scripts.reverseturn	
scripts.turn		scripts.slashturn	
scripts.haydnturn		scripts.trill	
scripts.upedalheel		scripts.dpedalheel	
scripts.upedaltoe		scripts.dpedaltoe	
scripts.pedalheelcircle		scripts.flageolet	

scripts.segno		scripts.varsegno	
scripts.coda		scripts.varcoda	
scripts.rcomma		scripts.lcomma	
scripts.rvarcomma		scripts.lvarcomma	
scripts.raltcomma		scripts.laltcomma	
scripts.arpeggio		scripts.trill_element	
scripts.arpeggio .arrow.M1		scripts.arpeggio.arrow.1	
scripts.prall		scripts.mordent	
scripts.prallprall		scripts.prallmordent	
scripts.upprall		scripts.upmordent	
scripts.prallup		scripts.downprall	
scripts.downmordent		scripts.pralldown	
scripts.lineprall		scripts.bachschleifer	
scripts.caesura.curved		scripts.caesura.straight	
scripts.tickmark		scripts.snappizzicato	
scripts.ictus		scripts.uaccentus	
scripts.daccentus		scripts.usemicirculus	
scripts.dsemicirculus		scripts.circulus	
scripts .usignumcongruentiae		scripts .dsignumcongruentiae	

A.8.13 Pfeilkopf-Glyphen

arrowheads.open.01		arrowheads.open.0M1	
arrowheads.open.11		arrowheads.open.1M1	
arrowheads.close.01		arrowheads.close.0M1	
arrowheads.close.11		arrowheads.close.1M1	

A.8.14 Klammerspitzen-Glyphen

brackettips.up		brackettips.down	

A.8.15 Pedal-Glyphen

pedal.*		pedal.M	
pedal..		pedal.P	
pedal.d		pedal.e	
pedal.Ped			

A.8.16 Akkordeon-Glyphen

accordion.discant		accordion.dot	
accordion.freebass		accordion.stdbass	
accordion.bayanbass		accordion.oldEE	
accordion.push		accordion.pull	

A.8.17 Bogen-Glyphen

ties.lyric.short		ties.lyric.default	

A.8.18 Vaticana-Glyphen

clefs.vaticana.do

clefs.vaticana.do_change

clefs.vaticana.fa



clefs.vaticana.fa_change

custodes.vaticana.u0

custodes.vaticana.u1

custodes.vaticana.u2

custodes.vaticana.d0

custodes.vaticana.d1

custodes.vaticana.d2

accidentals.vaticanaM1

accidentals.vaticana0

dots.dotvaticana

noteheads
.svaticana.punctum

noteheads.svaticana
.punctum.cavum

noteheads.svaticana
.linea.punctum

noteheads.svaticana
.linea.punctum.cavum

noteheads.svaticana
.inclinatum

noteheads.svaticana.lpes

noteheads
.svaticana.vlpes

noteheads.svaticana.upes

noteheads
.svaticana.vupes

noteheads
.svaticana.plica

noteheads
.svaticana.vplica

noteheads
.svaticana.epiphonus

noteheads.svaticana
.vepiphonus

noteheads.svaticana
.reverse.plica

noteheads.svaticana
.reverse.vplica

noteheads.svaticana
.inner.cephalicus

noteheads.svaticana
.cephalicus

noteheads
.svaticana.quilisma

A.8.19 Medicaea-Glyphen

clefs.medicaea.do 

clefs.medicaea.do_change 

clefs.medicaea.fa 

clefs.medicaea.fa_change 

custodes.medicaea.u0

custodes.medicaea.u1

custodes.medicaea.u2

custodes.medicaea.d0

custodes.medicaea.d1

custodes.medicaea.d2

accidentals.medicaeaM1

noteheads.smedicaea.inclinatum 

noteheads.smedicaea.punctum

noteheads.smedicaea.rvirga 

noteheads.smedicaea.virga 

A.8.20 Hufnagel-Glyphen

clefs.hufnagel.do 

clefs.hufnagel.do_change 

clefs.hufnagel.fa 

clefs.hufnagel.fa_change 

clefs.hufnagel.do.fa 

clefs.hufnagel.do.fa_change 

custodes.hufnagel.u0 

custodes.hufnagel.u1 

custodes.hufnagel.u2 

custodes.hufnagel.d0 

custodes.hufnagel.d1 

custodes.hufnagel.d2 

accidentals.hufnagelM1 

noteheads.shufnagel.punctum

noteheads.shufnagel.virga 

noteheads.shufnagel.lpes

A.8.21 Mensural-Glyphen

rests.M3mensural		rests.M2mensural	
rests.M1mensural		rests.0mensural	
rests.1mensural		rests.2mensural	
rests.3mensural		rests.4mensural	
clefs.mensural.c		clefs.mensural.c_change	
clefs.blackmensural.c		clefs.blackmensural .c_change	
clefs.mensural.f		clefs.mensural.f_change	
clefs.mensural.g		clefs.mensural.g_change	
custodes.mensural.u0		custodes.mensural.u1	
custodes.mensural.u2		custodes.mensural.d0	
custodes.mensural.d1		custodes.mensural.d2	
accidentals.mensural1		accidentals.mensuralM1	
flags.mensuralu03		flags.mensuralu13	
flags.mensuralu23		flags.mensurald03	
flags.mensurald13		flags.mensurald23	
flags.mensuralu04		flags.mensuralu14	
flags.mensuralu24		flags.mensurald04	
flags.mensurald14		flags.mensurald24	

flags.mensuralu05

flags.mensuralu15

flags.mensuralu25

flags.mensurald05 

flags.mensurald15

flags.mensurald25 

flags.mensuralu06

flags.mensuralu16

flags.mensuralu26

flags.mensurald06 

flags.mensurald16

flags.mensurald26 

timesig.mensural44 

timesig.mensural22 

timesig.mensural32 

timesig.mensural64 

timesig.mensural94 

timesig.mensural34 

timesig.mensural68 

timesig.mensural98 

timesig.mensural48 

timesig.mensural68alt 

timesig.mensural24 

noteheads.uM3mensural 

noteheads.dM3mensural 

noteheads.sM3ligmensural 

noteheads.uM2mensural 

noteheads.dM2mensural 

noteheads.sM2ligmensural 

noteheads.sM1mensural 

noteheads.urM3mensural 

noteheads.drM3mensural 

noteheads
.srM3ligmensural 

noteheads.urM2mensural 

noteheads.drM2mensural 

noteheads.srM1mensural 

noteheads .dM3semimensural 

noteheads .uM2semimensural 

noteheads .sM2semiligmensural 

noteheads .urM3semimensural 

noteheads .srM3semiligmensural 

noteheads .drM2semimensural 

noteheads .srM1semimensural 

noteheads .dM3blackmensural 

noteheads .uM2blackmensural 

noteheads .sM2blackligmensural 

noteheads.s0mensural

noteheads.s2mensural

noteheads .srM2ligmensural 

noteheads .uM3semimensural 

noteheads .sM3semiligmensural 

noteheads .dM2semimensural 

noteheads .sM1semimensural 

noteheads .drM3semimensural 

noteheads .urM2semimensural 

noteheads .srM2semiligmensural 

noteheads .uM3blackmensural 

noteheads .sM3blackligmensural 

noteheads .dM2blackmensural 

noteheads .sM1blackmensural 

noteheads.s1mensural

noteheads .s0blackmensural

A.8.22 Neomensural-Glyphen

rests.M3neomensural 

rests.M2neomensural

rests.M1neomensural

rests.0neomensural

rests.1neomensural

rests.2neomensural

rests.3neomensural

rests.4neomensural

clefs.neomensural.c		clefs.neomensural .c_change	
timesig.neomensural44		timesig.neomensural22	
timesig.neomensural32		timesig.neomensural64	
timesig.neomensural94		timesig.neomensural34	
timesig.neomensural68		timesig.neomensural98	
timesig.neomensural48		timesig.neomensural68alt	
timesig.neomensural24		noteheads.uM3neomensural	
noteheads.dM3neomensural		noteheads.uM2neomensural	
noteheads.dM2neomensural		noteheads.sM1neomensural	
noteheads .urM3neomensural		noteheads .drM3neomensural	
noteheads .urM2neomensural		noteheads .drM2neomensural	
noteheads .srM1neomensural		noteheads.s0neomensural	
noteheads.s1neomensural		noteheads.s2neomensural	

A.8.23 Petrucci-Glyphen

clefs.petrucci.c1		clefs.petrucci.c1_change	
clefs.petrucci.c2		clefs.petrucci.c2_change	
clefs.petrucci.c3		clefs.petrucci.c3_change	
clefs.petrucci.c4		clefs.petrucci.c4_change	

clefs.petrucci.c5 

clefs.petrucci.c5_change 

clefs.petrucci.f 

clefs.petrucci.f_change 

clefs.petrucci.g 

clefs.petrucci.g_change 

noteheads.s0petrucci 

noteheads.s1petrucci 

noteheads.s2petrucci 

noteheads
.s0blackpetrucci 

noteheads
.s1blackpetrucci 

noteheads
.s2blackpetrucci 

A.8.24 Solesmes-Glyphen

noteheads.ssolesmes
.incl.parvum

noteheads
.ssolesmes.auct.asc

noteheads
.ssolesmes.auct.desc

noteheads.ssolesmes
.incl.auctum

noteheads
.ssolesmes.stropha

noteheads.ssolesmes
.stropha.aucta

noteheads
.ssolesmes.oriscus

A.8.25 Glyphen der Kiever Notation

clefs.kievan.do 

clefs.kievan.do_change 

accidentals.kievan1 

accidentals.kievanM1 

scripts.barline.kievan 

dots.dotkievan

noteheads.sM2kievan 

noteheads.sM1kievan 

noteheads.s0kievan 

noteheads.d2kievan 

noteheads.u2kievan 

noteheads.s1kievan 
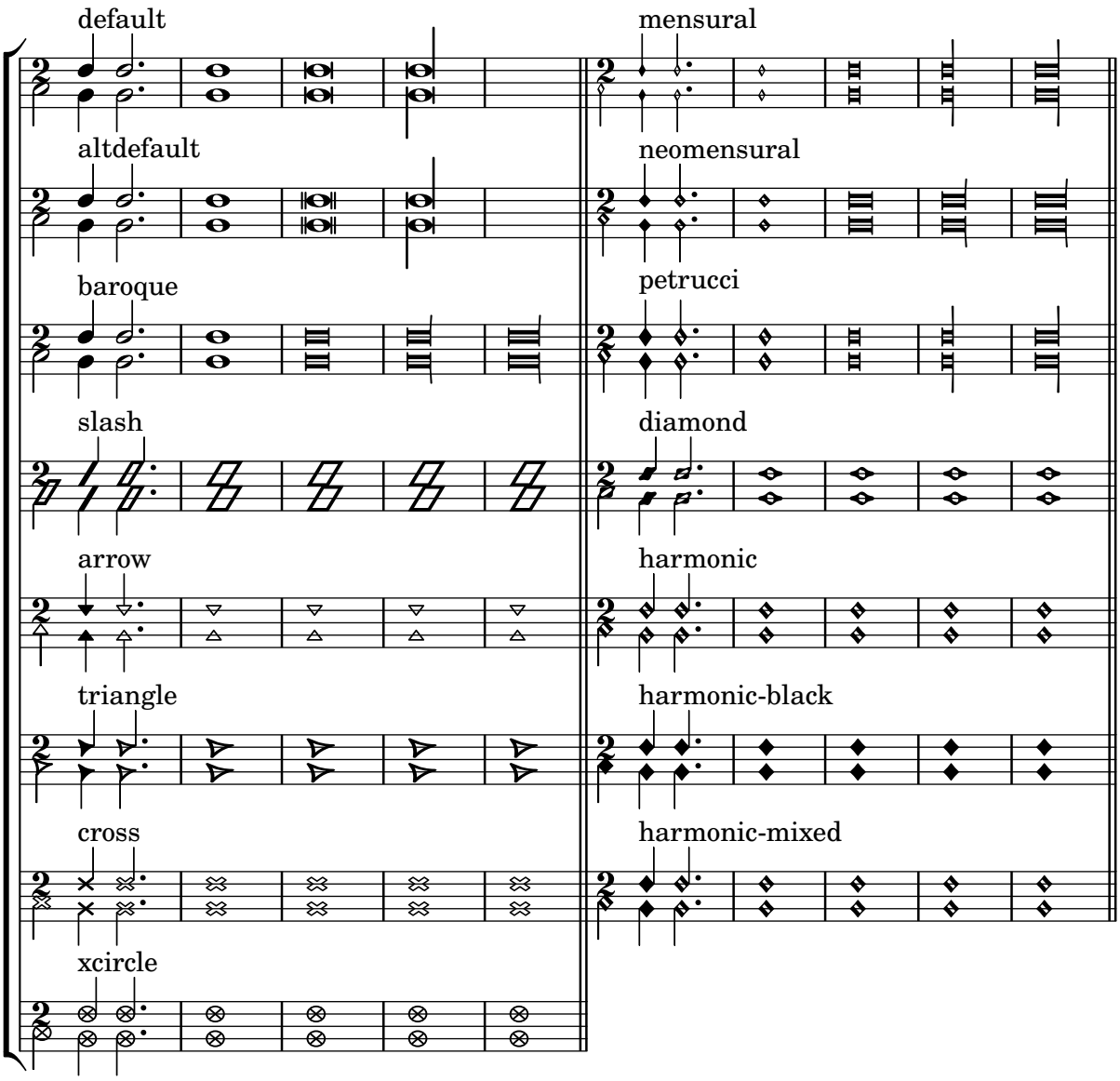
noteheads.sr1kievan 

noteheads.d3kievan 

noteheads.u3kievan 

A.9 Notenkopfstile

Folgende Stile können zur Darstellung der Notenköpfe verwendet werden:



A.10 Textbeschriftungsbefehle

The following commands can all be used inside `\markup { }`.

A.10.1 Font markup

`\abs-fontsize` *size* (number) *arg* (markup)

Use *size* as the absolute font size (in points) to display *arg*.

This function adjusts the `baseline-skip` and `word-space` properties accordingly.

```
\markup {
  default text font size
  \hspace #2
  \abs-fontsize #16 { text font size 16 }
  \hspace #2
  \abs-fontsize #12 { text font size 12 }
```

```
}
```

default text font size **text font size 16** text font size 12

Used properties:

- `baseline-skip` (3)
- `word-space` (0.6)

`\bold` *arg* (markup)
: Print *arg* with a bold face.

```
\markup {
  default
  \hspace #2
  \bold bold
}
```

default **bold**

The code `\markup \bold ...` is a shorthand for `\markup \override #'(font-series . bold) ...` – using the more verbose form, it is possible to obtain nuances such as semi-bold, if the text font has such variants. Refer to the documentation for the `font-series` properties (Abschnitt "User backend properties" in *Referenz der Interna*).

`\box` *arg* (markup)
: Draw a box around *arg*.

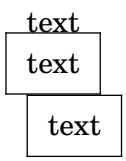
This function looks at the `thickness`, `box-padding`, and `font-size` properties to determine the line thickness and padding around the markup.

```
\markup {
  \override #'(box-padding . 0.5)
  \box \line { V. S. }
}
```



Note that the box does not horizontally displace its argument. Use markup commands like `\left-align` or `\table` to make LilyPond realign it.

```
\markup {
  \override #'(box-padding . 1.5)
  \column {
    "text"
    \box "text"
    \left-align \box "text"
  }
}
```



Used properties:

- `box-padding` (0.2)

- font-size (0)
- thickness (1)

\caps *arg* (markup)
: Print *arg* in (fake) small caps.

  This function is a copy of the \smallCaps command.

```
\markup {
  default
  \hspace #2
  \caps {
    Text \italic small caps
  }
}
```

default TEXT IN SMALL CAPS

  Use \fontCaps for real small caps (if the font provides it).

\dynamic *arg* (markup)
: Print *arg* using the (music) font for dynamics.

  This font only contains letters **f**, **m**, **n**, **p**, **r**, **s**, and **z**. When producing phrases like ‚più **f**', the normal words (like ‚più') should be done in a different font. The recommended font for this is bold and italic.

```
\markup {
  \dynamic {
    sfzp
  }
}
```

sfzp

\figured-bass *arg* (markup)
: Set *arg* as small numbers for figured bass.

  Specially slashed digits can be achieved with a trailing backslash (for numbers 6, 7, and 9) or a trailing plus (for numbers 2, 4, and 5).[1]

  The use of a backslash is in analogy to \figuremode (siehe Abschnitt 15.3.2 [Eingabe des Generalbass'], Seite 413). Note that to get a backslash character in markup it must be escaped by doubling it. Additionally, it must be put into double quotes.

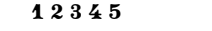
```
\markup {
  \figured-bass {
    2 3 4+ 7 "9\\"
  }
}
```

2 3 4+ 7 9

\finger *arg* (markup)
: Set *arg* as small numbers for fingering instructions.

[1] Internally, this works by activating the ‚dlig' OpenType feature of the Emmentaler font.

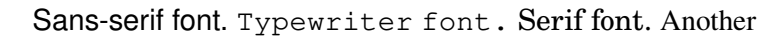
```
\markup {
  \finger {
    1 2 3 4 5
  }
}
```



\font-select *defs* (pair) *arg* (markup)
: Output the markup *arg* using fonts specified in *defs*.

  defs is either a single pair (`family . font-name`) or a non-empty list of such pairs. Here, `family` is usually `serif`, `sans` or `typewriter`; `font-name` is a string. Font families not mentioned in the call to `\font-select` remain unchanged.

```
\markup {
  \font-select #'((sans . "Liberation Sans")
                  (typewriter . "Liberation Mono"))
  {
    \sans { Sans-serif font. }
    \typewriter { Typewriter font. }
    Serif font.
    \font-select #'(serif . "Liberation Serif") {
      Another serif font.
    }
  }
}
```



\fontCaps *arg* (markup)
: Print *arg* in small caps.

  This command sets the `font-variant` property to `small-caps`.

  Unlike `\smallCaps`, which merely uses capital letters at a smaller font size, this uses the actual variant of the font for small caps. (As a consequence, if you configure a custom text font, this command has no effect if that font does not have a small caps variant.)

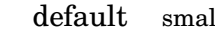
```
\markup \fontCaps "Small caps"
```



\fontsize *increment* (number) *arg* (markup)
: Increase current font size by *increment* to print *arg*.

  This function adjusts the `baseline-skip` and `word-space` properties accordingly.

```
\markup {
  default
  \hspace #2
  \fontsize #-1.5 smaller
}
```



  Used properties:

  - `baseline-skip` (2)
  - `word-space` (1)

- `font-size` (0)

`\huge` *arg* (markup)
: Set font size to value 2 to print *arg*.

```
\markup {
  default
  \hspace #2
  \huge huge
}
```

default **huge**

`\italic` *arg* (markup)
: Print *arg* in italics.

This command sets the `font-shape` property to `italic`.

```
\markup {
  default
  \hspace #2
  \italic italic
}
```

default *italic*

`\large` *arg* (markup)
: Set font size to value 1 to print *arg*.

```
\markup {
  default
  \hspace #2
  \large large
}
```

default large

`\larger` *arg* (markup)
: Increase current font size by 1 to print *arg*.

This function adjusts the `baseline-skip` and `word-space` properties accordingly.

```
\markup {
  default
  \hspace #2
  \larger larger
}
```

default larger

`\magnify` *sz* (number) *arg* (markup)
: Magnify current font by factor *sz* to print *arg*.

```
\markup {
  default
  \hspace #2
  \magnify #1.5 {
    50% larger
  }
}
```

default 50% larger

Note that magnification only works if a font name is explicitly selected. Use `\fontsize` otherwise.

`\normal-size-sub` *arg* (markup)
Set *arg* in subscript with a normal font size.

```
\markup {
  default
  \normal-size-sub {
    subscript in standard size
  }
}
```

default subscript in standard size

Used properties:

- `font-size` (0)

`\normal-size-super` *arg* (markup)
Set *arg* in superscript with a normal font size.

```
\markup {
  default
  \normal-size-super {
    superscript in standard size
  }
}
```

default superscript in standard size

Used properties:

- `font-size` (0)

`\normal-text` *arg* (markup)
Print *arg* with default text font.

This resets all font-related properties (except the size), no matter what font was used earlier.

```
\markup {
  \huge \bold \sans \fontCaps {
    huge bold sans caps
    \hspace #2
    \normal-text {
      huge normal
    }
    \hspace #2
    as before
  }
}
```

HUGE BOLD SANS CAPS huge normal **AS BEFORE**

`\normal-weight` *arg* (markup)
Switch to normal weight (in contrast to bold) to print *arg*.

This command sets the `font-series` property to `normal`.

```
\markup {
  \bold {
    some bold text
    \hspace #2
    \normal-weight {
      normal font series
    }
    \hspace #2
    bold again
  }
}
```

some bold text normal font series **bold again**

\normalsize *arg* (markup)
: Set font size to default (i.e., to value 0) to print *arg*.

```
\markup {
  \teeny {
    this is very small
    \hspace #2
    \normalsize {
      normal size
    }
    \hspace #2
    teeny again
  }
}
```

this is very small normal size teeny again

\number *arg* (markup)
: Print *arg* using the (music) font for numbers.

  This font also contains symbols for figured bass, some punctuation, spaces of various widths, some letters and text variants of accidentals, the common time, and the cut time symbol. Use \dynamic to access the (very small number of) letters. For accidentals you might use \number in combination with Unicode characters to access the text representation forms of accidental glyphs, as the following table shows.

Unicode value	Unicode character
U+266D	♭
U+266E	♮
U+266F	♯
U+1D12A	𝄪
U+1D12B	𝄫
U+1D134	𝄴

U+1D135 𝅝

Examples:

```
\number ♭ → ♭
\number { \char ##x266F } → ♯
```

To get accidentals protected against overrides of `font-name` it is preferable to use `\text-doubleflat`, `\text-flat`, `\text-natural`, `\text-sharp`, `\text-doublesharp`, or the general `\text-accidental` for the text variants of accidentals. For the time signature symbols, use `\text-common-time` and `\text-cut-time`.

The appearance of digits in the Emmentaler font can be controlled with four OpenType features: ‚tnum‘, ‚cv47‘, ‚ss01‘, and ‚kern‘, which can be arbitrarily combined.

tnum — If off (which is the default), glyphs ‚zero‘ to ‚nine‘ have no left and right side bearings. If on, the glyphs all have the same advance width by making the bearings non-zero.

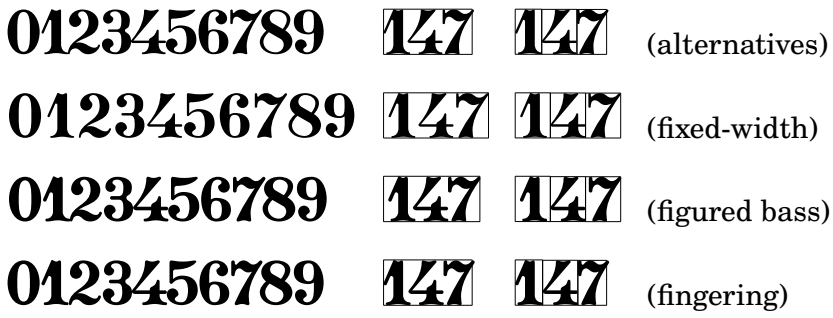
cv47 — If on, glyphs ‚four‘ and ‚seven‘ have shorter vertical strokes. Default is off.

ss01 — If on, glyphs ‚zero‘ to ‚nine‘ have a fatter design, making them more readable at small sizes. Default is off.

kern — If on (which is the default), provide pairwise kerning between (most) glyphs.

```
\markuplist
  \number
  \fontsize #4.5
  \override #'((padding . 2)
               (baseline-skip . 4)
               (box-padding . 0)
               (thickness . 0.1))
  \table #'(-1 -1 -1 -1) {
      0123456789 \box 147 \concat { \box 1 \box 4 \box 7 }
    \normal-text \normalsize "(time signatures)"
    \override #'(font-features .("cv47")) {
      0123456789 \box 147 \concat { \box 1 \box 4 \box 7 } }
    \normal-text \normalsize "(alternatives)"
    \override #'(font-features .("tnum" "cv47" "-kern")) {
      0123456789 \box 147 \concat { \box 1 \box 4 \box 7 } }
    \normal-text \normalsize "(fixed-width)"
    \override #'(font-features . ("tnum" "cv47" "ss01")) {
      0123456789 \box 147 \concat { \box 1 \box 4 \box 7 } }
    \normal-text \normalsize "(figured bass)"
    \override #'(font-features . ("cv47" "ss01")) {
      0123456789 \box 147 \concat { \box 1 \box 4 \box 7 } }
    \normal-text \normalsize "(fingering)"
  }
```





See also the markup commands `\figured-bass` and `\finger`, which set the font features accordingly.

`\overtie` *arg* (markup)

Print a tie over *arg*.

```
\markup \line {
  \overtie "overtied"
  \override #'((offset . 5) (thickness . 1))
  \overtie "overtied"
  \override #'((offset . 1) (thickness . 5))
  \overtie "overtied"
}
```



Used properties:

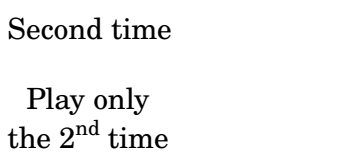
- `shorten-pair` (`(0 . 0)`)
- `height-limit` (`0.7`)
- `direction` (`1`)
- `offset` (`2`)
- `thickness` (`1`)

`\replace` *replacements* (list) *arg* (markup)

Use *replacements* to replace strings in *arg*.

Each (*key* . *value*) pair of the *replacements* alist specifies what should be replaced; *key* gets replaced by *value*. Note the quasi-quoting syntax with a backquote in the second example.

```
\markup \replace #'(("2nd" . "Second"))
  "2nd time"
\markup \replace
  #`(("2nd" . ,#{ \markup \concat { 2 \super nd } #}))
  \center-column {
    \line { Play only }
    \line { the 2nd time }
  }
```



Used properties:

- `replacement-alist`

\sans *arg* (markup)
: Print *arg* with a sans-serif font.

  This command sets the `font-family` property to `sans`.

```
\markup {
  default
  \hspace #2
  \sans {
    sans serif
  }
}
```

default sans serif

\serif *arg* (markup)
: Print *arg* with a serif font.

  This command sets the `font-family` property to `serif`.

```
\markup {
  \sans \bold {
    sans serif, bold
    \hspace #2
    \serif {
      text in serif font
    }
    \hspace #2
    return to sans
  }
}
```

sans serif, bold text in serif font return to sans

\simple *str* (string)
: Print string *str*.

  `\markup \simple "x"` is equivalent to `\markup "x"`. This command was previously used internally, but no longer is, and is being kept for backward compatibility only.

\small *arg* (markup)
: Set font size to value -1 to print *arg*.

```
\markup {
  default
  \hspace #2
  \small small
}
```

default small

\smallCaps *arg* (markup)
: Print *arg* in (fake) small caps.

  Unlike `\fontCaps`, which uses the actual small caps variant of the current font, this fakes small caps by using capital letters at a smaller font size. It can thus be used for fonts that don't have a small caps variant.

```
\markup {
  default
  \hspace #2
  \smallCaps {
    Text in small caps
  }
}
```

default TEXT IN SMALL CAPS

\smaller *arg* (markup)
: Decrease current font size by 1 to print *arg*.

  This function adjusts the `baseline-skip` and `word-space` properties accordingly.

```
\markup {
  \fontsize #3.5 {
    large text
    \hspace #2
    \smaller { smaller text }
    \hspace #2
    large text
  }
}
```

large text smaller text large text

\sub *arg* (markup)
: Set *arg* in subscript.

```
\markup { \concat { H \sub 2 O } }
```

H_2O

  See also \super.

  Used properties:

  - `font-size` (0)

\super *arg* (markup)
: Set *arg* in superscript.

```
\markup { E = \concat { mc \super 2 } }
```

$E = mc^2$

  See also \sub.

  Used properties:

  - `font-size` (0)

\teeny *arg* (markup)
: Set font size to value -3 to print *arg*.

```
\markup {
  default
  \hspace #2
  \teeny teeny
}
```

default teeny

\tie *arg* (markup)
: Add a horizontal bow at the bottom or top of *arg*.

  This function uses `make-tie-stencil` to create the bow; it looks at the `thickness` and `offset` properties to determine the line thickness and vertical offset, respectively. The added bow fits the extent of *arg*; use the `shorten-pair` property to modify this. The `direction` property may be set explicitly using `override` or direction modifiers, or implicitly by using `voiceOne`, etc.

```
\markup {
  \override #'(direction . 1)
  \tie "above"
  \override #'(direction . -1)
  \tie "below"
}
```

above below

  See also \undertie and \overtie, which are shorthands for this function.

  Used properties:

  - `shorten-pair` (`(0 . 0)`)
  - `height-limit` (`0.7`)
  - `direction` (`1`)
  - `offset` (`2`)
  - `thickness` (`1`)

\tiny *arg* (markup)
: Set font size to value -2 to print *arg*.

```
\markup {
  default
  \hspace #2
  \tiny tiny
}
```

default tiny

\typewriter *arg* (markup)
: Print *arg* in typewriter.

  This command sets the `font-family` property to `typewriter`, also switching off the ‚liga' OpenType feature to disable ligatures like ‚ff' or ‚fi'.

```
\markup {
  "default fi ff"
  \hspace #2
  \typewriter "typewriter fi ff"
}
```

default fi ff typewriter fi ff

\underline *arg* (markup)
: Underline *arg*.

  This function looks at the property `thickness` to determine the line thickness, at `offset` to determine the line's vertical offset from *arg*, and at `underline-skip` to determine the distance of additional lines from the others.

  The `underline-shift` property is used to make subsequent calls work correctly. Overriding it makes little sense since it would end up adding the provided value to the one of `offset`.

```
\markup \justify-line {
  \underline "underlined"
  \override #'(offset . 5)
  \override #'(thickness . 1)
  \underline "underlined"
  \override #'(offset . 1)
  \override #'(thickness . 5)
  \underline "underlined"
  \override #'(offset . 5)
  \override #'(underline-skip . 4)
  \underline \underline \underline "underlined thrice"
}
```

underlined underlined underlined underlined thrice

  Used properties:

  - `underline-skip` (2)
  - `underline-shift` (0)
  - `offset` (2)
  - `thickness` (1)

\undertie *arg* (markup)
: Print a tie under *arg*.

```
\markup \line {
  \undertie "undertied"
  \override #'((offset . 5) (thickness . 1))
  \undertie "undertied"
  \override #'((offset . 1) (thickness . 5))
  \undertie "undertied"
}
```

undertied undertied undertied

  Used properties:

  - `shorten-pair` ((0 . 0))
  - `height-limit` (0.7)

- `direction` (1)
- `offset` (2)
- `thickness` (1)

`\upright` *arg* (markup)
: Print *arg* upright.

  This command is the opposite of `\italic`; it sets the `font-shape` property to `upright`.

```
\markup {
  \italic {
    italic text
    \hspace #2
    \upright {
      upright text
    }
    \hspace #2
    italic again
  }
}
```

italic text upright text *italic again*

`\volta-number` *arg* (markup)
: Set *arg* in a font appropriate for volta numbers.

```
\markup \volta-number "4."
```

4.

`\with-string-transformer` *transformer* (procedure) *arg* (markup)
: Apply string transformer function *transformer* to *arg*.

  Whenever a string is interpreted inside *arg*, function *transformer* is called first, and its result is then interpreted. The arguments passed to *transformer* are the output definition, the property alist chain, and the markup *arg*. See Abschnitt “New markup command definition” in *Extending* about the two first arguments.

```
\markup \with-string-transformer
  #(lambda (layout props str)
     (string-upcase str))
  \concat { "abc" \larger "def" }
```

ABCDEF

A.10.2 Markup for text alignment

`\abs-hspace` *amount* (number)
: Create an invisible object taking up absolute horizontal space of *amount* points.

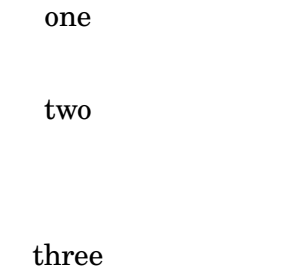
```
\markup {
  one
  \abs-hspace #20
  two
  \abs-hspace #40
  three
}
```

one two three

See also \hspace.

\abs-vspace *amount* (number)
: Create an invisible object taking up absolute vertical space of *amount* points.

```
\markup {
    \center-column {
    one
    \abs-vspace #20
    two
    \abs-vspace #40
    three
  }
}
```



See also \vspace.

\align-on-other *axis* (non-negative, exact integer) *other-dir* (boolean-or-number) *other* (markup) *self-dir* (boolean-or-number) *self* (markup)
: Align markup *self* on markup *other* along *axis*.

  This function uses *self-dir* and *other-dir* for mutual alignment of *self* and *other*, respectively, translating *self* as requested relative to its surroundings. *other* is not printed.

  If *self-dir* or *other-dir* is #f, use the reference point of *self* or *other*, respectively.

```
\markup \column {
  12
  \align-on-other #X #RIGHT 12
                     #LEFT 12345
  \align-on-other #X #RIGHT 123
                     #LEFT \fermata
  123
  \align-on-other #X #RIGHT 123
                     ##f \fermata
}
```



\center-align *arg* (markup)
: Align *arg* to its X center.

```
\markup {
  \column {
    one
    \center-align two
    three
  }
}
```

one
two
three

\center-column *args* (markup list)
: Put *args* into a centered column.

  See also \column.

```
\markup {
  \center-column {
    one
    two
    three
  }
}
```

one
two
three

  Used properties:

  - baseline-skip

\column *args* (markup list)
: Stack the markups in *args* vertically.

  The property baseline-skip determines the space between markups in *args* (to be more precise, the space between the baselines of the markups).

```
\markup {
  \column {
    one
    two
    three
  }
}
```

one
two
three

  The baseline of the output of \column is the baseline of its first line.

  Used properties:

  - baseline-skip

\combine *arg1* (markup) *arg2* (markup)
: Print *arg1*, then print *arg2* on top of it.

  Note: \combine cannot take a list of markups enclosed in curly braces as an argument; for this purpose use \overlay instead.

```
\markup {
  \fontsize #5
  \override #'(thickness . 2)
  \combine
    \draw-line #'(0 . 4)
    \arrow-head #Y #DOWN ##f
}
```



`\concat` *args* (markup list)

Concatenate *args* in a horizontal line, without spaces in between.

Strings are concatenated on the input level, allowing ligatures. For example, `\concat { "f" "i" }` is equivalent to `"fi"`.

```
\markup {
  \concat {
    one two three
  }
}
```

onetwothree

`\dir-column` *args* (markup list)

Make a column of *args*.

Depending on the setting of the `direction` layout property, the arguments are stacked upwards or downwards.

```
\markup {
  \override #`(direction . ,UP)
  \dir-column {
    going up
  }
  \hspace #1
  \dir-column {
    going down
  }
  \hspace #1
  \override #'(direction . 1)
  \dir-column {
    going up
  }
}
```

up going up
going down
going

The baseline of the output of `\dir-column` is the baseline of its first line.

Used properties:

- `baseline-skip`
- `direction`

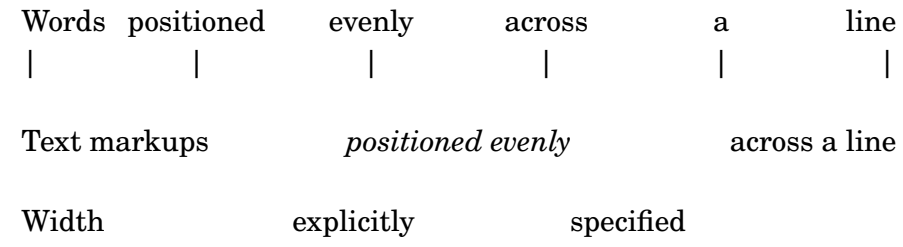
`\fill-line` *args* (markup list)

Put markups *args* into a horizontal line at fixed positions.

If there is a single argument, it gets centered. If there are two arguments, they get aligned to the left and right, respectively. Otherwise, if there are *n* arguments, the markups are aligned to *n* equally spaced positions, with the first markup left-aligned, the last markup right-aligned, and the remaining markups centered at the respective positions. If there are no arguments, return an empty stencil.

The width of the horizontal line can be modified by overriding the `line-width` property. The space between arguments is at least as large as the value of the `word-space` property, at the cost of lengthening the line if necessary.

```
\markup {
  \column {
    \fill-line { Words positioned evenly across a line }
    \fill-line { | | | | | | }
    \null
    \fill-line {
      \line { Text markups }
      \line { \italic { positioned evenly } }
      \line { across a line }
    }
    \null
    \override #'(line-width . 50)
    \fill-line { Width explicitly specified }
  }
}
```



See also `\justify-line`.

Used properties:

- `line-width` (`#f`)
- `word-space` (`0.6`)
- `text-direction` (`1`)

`\fill-with-pattern` *space* (number) *dir* (direction) *pattern* (markup) *left* (markup) *right* (markup)

Put *left* and *right* at the start and end of a line, respectively, and fill the space in between with repeated *pattern* markups.

Patterns are spaced apart by *space* and aligned to direction *dir*. The width of the line is given by the `line-width` property.

```
\markup \column {
  "right-aligned:"
  \fill-with-pattern #1 #RIGHT . first right
  \fill-with-pattern #1 #RIGHT . second right
  \null
  "center-aligned:"
  \fill-with-pattern #1.5 #CENTER - left right
  \null
  "left-aligned:"
  \override #'(line-width . 50) {
    \fill-with-pattern #2 #LEFT : left first
    \fill-with-pattern #2 #LEFT : left second
  }
}
```

right-aligned:
first . . . . . right
second . . . . . right

center-aligned:
left - right

left-aligned:
left : : : : : : : : : : : : : : : : : first
left : : : : : : : : : : : : : : : : : second

Used properties:

- `line-width`
- `word-space`

\general-align *axis* (integer) *dir* (number) *arg* (markup)
Align *arg* in *axis* direction to the *dir* side.

```
\markup {
  \column {
    one
    \general-align #X #LEFT two
    three
    \null
    one
    \general-align #X #CENTER two
    three
    \null
    \line {
      one
      \general-align #Y #UP two
      three
    }
    \null
    \line {
      one
      \general-align #Y #3.2 two
```

```
      three
    }
  }
}
```

one
two
three

one
two
three

one two three

one three
two

`\halign` *dir* (number) *arg* (markup)
: Print *arg* with horizontal alignment set to *dir*.

  If *dir* is -1, *arg* is left-aligned, while +1 makes it right-aligned. Values in between interpolate the alignment accordingly.

```
\markup {
  \column {
    one
    \halign #LEFT two
    three
    \null
    one
    \halign #CENTER two
    three
    \null
    one
    \halign #RIGHT two
    three
    \null
    one
    \halign #-5 two
    three
  }
}
```

one
two
three

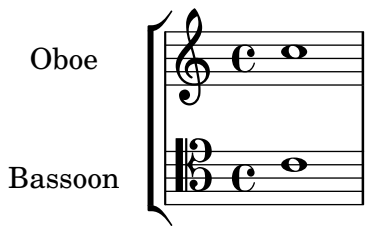
one
two
three

one
two
three

one
two
three

\hcenter-in *length* (number) *arg* (markup)
Center *arg* horizontally within a box of extending *length*/2 to the left and right.

```
\new StaffGroup <<
  \new Staff {
    \set Staff.instrumentName = \markup {
      \hcenter-in #12 Oboe
    }
    c''1
  }
  \new Staff {
    \set Staff.instrumentName = \markup {
      \hcenter-in #12 Bassoon
    }
    \clef tenor
    c'1
  }
>>
```



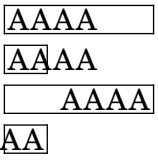
\hspace *amount* (number)
Create an invisible object taking up *amount* horizontal space.

```
\markup {
  one
  \hspace #2
  two
  \hspace #8
  three
}
```

one two three

amount can be also a negative value, which can be best visualized as if the current drawing point gets moved to the left.

```
\markup \concat {
  \hspace #4
  \column {
    \box \concat { AAAA \hspace #4 }
    \box \concat { AAAA \hspace #-4 }
    \box \concat { \hspace #4 AAAA }
    \box \concat { \hspace #-4 AAAA }
  }
}
```



See also `\abs-hspace`.

`\justify` *args* (markup list)

Print *args* as lines aligned both at the left and the right.

Like `\wordwrap`, but with lines stretched to justify the margins. Use `\override #'(line-width . X)` to set the line width; *X* is the number of staff spaces.

```
\markup {
  \justify {
    Lorem ipsum dolor sit amet, consectetur adipisicing elit,
    sed do eiusmod tempor incididunt ut labore et dolore
    magna aliqua.  Ut enim ad minim veniam, quis nostrud
    exercitation ullamco laboris nisi ut aliquip ex ea
    commodo consequat.
  }
}
```

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

The baseline of the output of `\justify` is the baseline of its first line.

Used properties:

- `text-direction` (`1`)
- `word-space`
- `line-width` (`#f`)
- `baseline-skip`

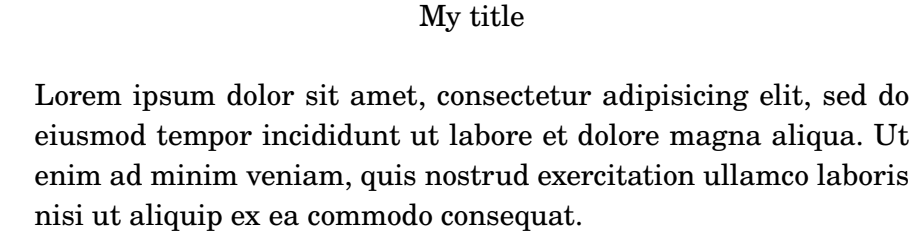
`\justify-field` *symbol* (symbol)

Justify the data that has been assigned to *symbol*.

```
\header {
  title = "My title"
  myText = "Lorem ipsum dolor sit amet, consectetur
    adipisicing elit, sed do eiusmod tempor incididunt
    ut labore et dolore magna aliqua.  Ut enim ad minim
    veniam, quis nostrud exercitation ullamco laboris
    nisi ut aliquip ex ea commodo consequat."
}

\paper {
  bookTitleMarkup = \markup {
    \column {
      \fill-line { \fromproperty #'header:title }
      \null
      \justify-field #'header:myText
    }
  }
}

\markup {
  \null
}
```



`\justify-line` *args* (markup list)

Put markups *args* into a horizontal line, equally spaced.

If there is a single argument, it gets centered. If there are two arguments, they get aligned to the left and right, respectively. Otherwise, the markups are spread to fill the entire line, separated by equally large spaces. If there are no arguments, return an empty stencil.

The width of the horizontal line can be modified by overriding the `line-width` property. The space between arguments is at least as large as the value of the `word-space` property, at the cost of lengthening the line if necessary.

```
\markup {
  \justify-line {
    Constant space between neighboring words
  }
}
```



See also `\fill-line`.

Used properties:

- `line-width` (`#f`)

- word-space (0.6)
- text-direction (1)

\justify-string *arg* (string)

Print string *arg* as lines aligned both at the left and the right.

Paragraphs are indicated by double newlines. Use \override #'(line-width . *X*) to set the line width; *X* is the number of staff spaces.

```
\markup {
  \override #'(line-width . 40)
  \justify-string "Lorem ipsum dolor sit amet, consectetur
      adipisicing elit, sed do eiusmod tempor incididunt ut
      labore et dolore magna aliqua.


      Ut enim ad minim veniam, quis nostrud exercitation
      ullamco laboris nisi ut aliquip ex ea commodo
      consequat.


      Excepteur sint occaecat cupidatat non proident, sunt
      in culpa qui officia deserunt mollit anim id est
      laborum"
}
```

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

The baseline of the output of \justify-string is the baseline of its first line.

Used properties:

- text-direction (1)
- word-space
- line-width
- baseline-skip

\left-align *arg* (markup)

Align *arg* on its left edge.

```
\markup {
  \column {
    one
    \left-align two
    three
  }
}
```

one
two
three

`\left-column` *args* (markup list)
Put *args* into a left-aligned column.

```
\markup {
  \left-column {
    one
    two
    three
  }
}
```

one
two
three

Used properties:

- `baseline-skip`

`\line` *args* (markup list)
Put *args* into a horizontal line.

The property `word-space` determines the space between markups in *args*. For right-to-left scripts like Hebrew, `text-direction` should be set to -1.

```
\markup
  \override #'(word-space . 3)
  \column {
    \line { "A B" "C D" "E F" }
    \override #'(text-direction . -1)
    \line { "A B" "C D" "E F" }
  }
```

A B C D E F
E F C D A B

Used properties:

- `text-direction` (1)
- `word-space`

`\lower` *amount* (number) *arg* (markup)
Lower *arg* by the distance *amount*.

A negative *amount* indicates raising; see also `\raise`.

The argument to `\lower` is the vertical displacement amount, measured in (global) staff spaces, which is independent of the markup's current font size. If you need vertical movement that takes the font size into account, use `\translate-scaled` instead.

This function is normally used to move one element inside of a markup relative to the other elements. When using it on the whole markup, bear in mind that spacing mechanisms that place the markup itself on the page could cancel this shift. Consider using grob properties such as `padding`, `Y-offset`, or `extra-offset`, or spacing variables such as `markup-system-spacing`.

```
\markup {
  one
  \lower #3 two
  three
}
```

one three
two

\overlay *args* (markup list)
Take a list of markups *args* and combine them.

```
\markup {
  \fontsize #5
  \override #'(thickness . 2)
  \overlay {
    \draw-line #'(0 . 4)
    \arrow-head #Y #DOWN ##f
    \translate #'(0 . 4) \arrow-head #Y #UP ##f
  }
}
```



\pad-around *amount* (number) *arg* (markup)
Add padding *amount* all around *arg*.
Identical to function \pad-markup.

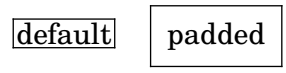
```
\markup {
  \box {
    default
  }
  \hspace #2
  \box {
    \pad-around #0.5 {
      padded
    }
  }
}
```

default padded

\pad-markup *amount* (number) *arg* (markup)
Add padding *amount* all around *arg*.
Identical to function \pad-around.

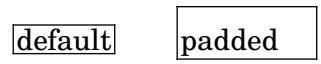
```
\markup {
  \box {
    default
  }
  \hspace #2
  \box {
    \pad-markup #1 {
      padded
```

```
    }
  }
}
```



\pad-to-box *x-ext* (pair of numbers) *y-ext* (pair of numbers) *arg* (markup)
Make *arg* take at least *x-ext*, *y-ext* space.

```
\markup {
  \box {
    default
  }
  \hspace #4
  \box {
    \pad-to-box #'(0 . 10) #'(0 . 3) {
      padded
    }
  }
}
```



\pad-x *amount* (number) *arg* (markup)
Add padding *amount* around *arg* in the X direction.

```
\markup {
  \box {
    default
  }
  \hspace #4
  \box {
    \pad-x #2 {
      padded
    }
  }
}
```



\pad-x-left *amount* (number) *arg* (markup)
Add padding *amount* to the left of *arg* in the X direction.

```
\markup {
  \box {
    default
  }
  \hspace #4
  \box {
    \pad-x-left #2 {
      padded
    }
  }
}
```



\pad-x-right *amount* (number) *arg* (markup)
Add padding *amount* to the right of *arg* in the X direction.

```
\markup {
  \box {
    default
  }
  \hspace #4
  \box {
    \pad-x-right #2 {
      padded
    }
  }
}
```



\put-adjacent *axis* (integer) *dir* (direction) *arg1* (markup) *arg2* (markup)
Put *arg2* next to *arg1* along *axis* to the *dir* side, without moving *arg1*.

```
\markup \column {
  text
  \put-adjacent #X #LEFT text *
  text
}
```



\raise *amount* (number) *arg* (markup)
Raise *arg* by the distance *amount*.

A negative *amount* indicates lowering, see also \lower.

The argument to \raise is the vertical displacement amount, measured in (global) staff spaces, which is independent of the markup's current font size. If you need vertical movement that takes the font size into account, use \translate-scaled instead.

This function is normally used to move one element inside of a markup relative to the other elements. When using it on the whole markup, bear in mind that spacing mechanisms that place the markup itself on the page could cancel this shift. Consider using grob properties such as padding, Y-offset, or extra-offset, or spacing variables such as markup-system-spacing.

```
\markup { C \small \bold \raise #1.0 9/7+ }
```



\right-align *arg* (markup)
Align *arg* on its right edge.

```
\markup {
  \column {
    one
    \right-align two
    three
  }
}
```

one
two
three

\right-column *args* (markup list)
: Put *args* into a right-aligned column.

```
\markup {
  \right-column {
    one
    two
    three
  }
}
```

one
two
three

Used properties:

- `baseline-skip`

\rotate *ang* (number) *arg* (markup)
: Rotate *arg* by *ang* degrees around its center.

```
\markup {
  default
  \hspace #2
  \rotate #45
  \line {
    rotated 45°
  }
}
```

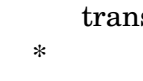
default rotated 45°

\translate *offset* (pair of numbers) *arg* (markup)
: Translate *arg* relative to its surroundings.

offset is a pair of numbers representing the displacement in the X and Y axes. See also `\translate-scaled`.

This function is normally used to move one element inside of a markup relative to the other elements. When using it on the whole markup, bear in mind that spacing mechanisms that place the markup itself on the page could cancel this shift. Consider using grob properties such as `padding`, `X-offset`, `Y-offset` or `extra-offset`, or spacing variables such as `markup-system-spacing`.

```
\markup {
  *
  \translate #'(2 . 3)
    \line { translated two spaces right, three up }
}
```

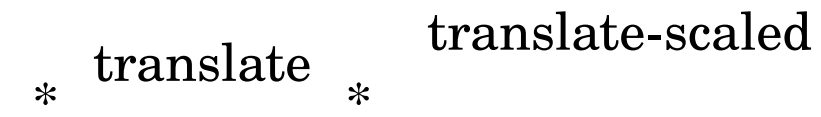


\translate-scaled *offset* (pair of numbers) *arg* (markup)
: Translate *arg* by *offset*, scaling the offset by the font size.

  This function is normally used to move one element inside of a markup relative to the other elements. When using it on the whole markup, bear in mind that spacing mechanisms that place the markup itself on the page could cancel this shift. Consider using grob properties such as `padding`, `X-offset`, `Y-offset` or `extra-offset`, or spacing variables such as `markup-system-spacing`.

  See also `\translate`.

```
\markup {
  \fontsize #5 {
    * \translate #'(2 . 3) translate
    \hspace #2
    * \translate-scaled #'(2 . 3) translate-scaled
  }
}
```

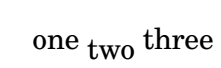


  Used properties:

  - `font-size` (0)

\vcenter *arg* (markup)
: Align *arg* to its Y center.

```
\markup {
  one
  \vcenter two
  three
}
```



\vspace *amount* (number)
: Create an invisible object taking up vertical space of *amount* multiplied by 3.

```
\markup {
    \center-column {
    one
    \vspace #1
    two
    \vspace #3
    three
  }
}
```

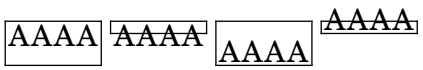
one

two

three

amount can be also a negative value, which can be best visualized as if the current drawing point gets moved up.

```
\markup {
  \vspace #1
  \box \column { AAAA \vspace #0.4 }
  \box \column { AAAA \vspace #-0.4 }
  \box \column { \vspace #0.4 AAAA }
  \box \column { \vspace #-0.4 AAAA }
}
```



See also `\abs-vspace`.

`\wordwrap` *args* (markup list)
Print *args* as left-aligned lines.

This function provides simple word-wrap. Use `\override #'(line-width . X)` to set the line width; *X* is the number of staff spaces.

```
\markup {
  \wordwrap {
    Lorem ipsum dolor sit amet, consectetur adipisicing elit,
    sed do eiusmod tempor incididunt ut labore et dolore
    magna aliqua.  Ut enim ad minim veniam, quis nostrud
    exercitation ullamco laboris nisi ut aliquip ex ea
    commodo consequat.
  }
}
```

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

The baseline of the output of `\wordwrap` is the baseline of its first line.

Used properties:

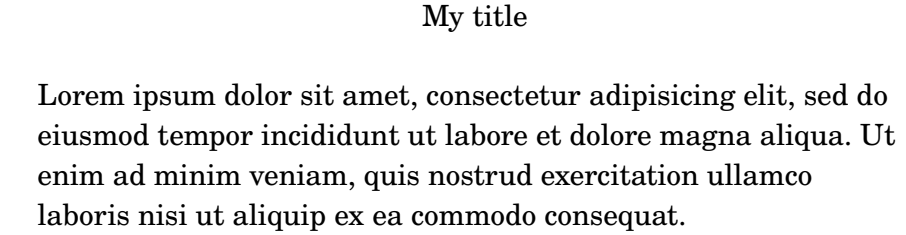
- `text-direction` (`1`)
- `word-space`
- `line-width` (`#f`)
- `baseline-skip`

`\wordwrap-field` *symbol* (symbol)
Word-wrap the data that has been assigned to *symbol*.

```
\header {
  title = "My title"
  myText = "Lorem ipsum dolor sit amet, consectetur
    adipisicing elit, sed do eiusmod tempor incididunt ut
    labore et dolore magna aliqua.  Ut enim ad minim
    veniam, quis nostrud exercitation ullamco laboris nisi
    ut aliquip ex ea commodo consequat."
}

\paper {
  bookTitleMarkup = \markup {
    \column {
      \fill-line { \fromproperty #'header:title }
      \null
      \wordwrap-field #'header:myText
    }
  }
}

\markup {
  \null
}
```



`\wordwrap-string` *arg* (string)

Print string *arg* as left-aligned lines.

Paragraphs are indicated by double newlines. Use `\override #'(line-width . X)` to set the line width; *X* is the number of staff spaces.

```
\markup {
  \override #'(line-width . 40)
  \wordwrap-string "Lorem ipsum dolor sit amet,
      consectetur adipisicing elit, sed do eiusmod tempor
      incididunt ut labore et dolore magna aliqua.


      Ut enim ad minim veniam, quis nostrud exercitation
      ullamco laboris nisi ut aliquip ex ea commodo
      consequat.


      Excepteur sint occaecat cupidatat non proident,
      sunt in culpa qui officia deserunt mollit anim id
      est laborum"
}
```

Lorem ipsum dolor sit amet,
consectetur adipisicing elit, sed do
eiusmod tempor incididunt ut labore et
dolore magna aliqua.
Ut enim ad minim veniam, quis
nostrud exercitation ullamco laboris
nisi ut aliquip ex ea commodo
consequat.
Excepteur sint occaecat cupidatat non
proident, sunt in culpa qui officia
deserunt mollit anim id est laborum

The baseline of the output of `\wordwrap-string` is the baseline of its first line.

Used properties:

- `text-direction` (1)
- `word-space`
- `line-width`
- `baseline-skip`

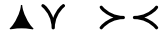
A.10.3 Graphical markup

`\arrow-head` *axis* (integer) *dir* (direction) *filled* (boolean)

Print an arrow head along *axis* in direction *dir*.

Fill the head if *filled* is set to `#t`.

```
\markup {
  \fontsize #5 {
    \general-align #Y #DOWN {
      \arrow-head #Y #UP ##t
      \arrow-head #Y #DOWN ##f
      \hspace #2
      \arrow-head #X #RIGHT ##f
      \arrow-head #X #LEFT ##f
    }
  }
}
```



`\beam` *width* (number) *slope* (number) *thickness* (number)

Draw a beam with given *width*, *slope*, and *thickness*.

```
\markup {
  \beam #5 #1 #2
}
```



`\bracket` *arg* (markup)

Draw vertical brackets around *arg*.

```
\markup {
  \bracket {
    \note {2.} #UP
  }
}
```

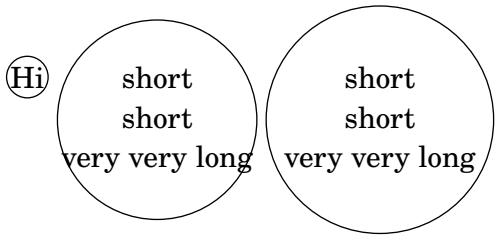


\circle *arg* (markup)

Draw a circle around *arg*.

Use properties `thickness`, `circle-padding`, and `font-size` to set the line thickness and padding around the markup. If `bbox` is set to `#t`, make the circle enclose the bounding box of *arg*, otherwise use either the width or the height of *arg* (whatever is larger) as the diameter.

```
\markup {
  \circle {
    Hi
  }
  \circle {
    \center-column { "short" "short" "very very long" }
  }
  \override #'(bbox . #t) \circle {
    \center-column { "short" "short" "very very long" }
  }
}
```



Note that the circle does not horizontally displace its argument. Use markup commands like `\left-align` or `\table` to make LilyPond realign it.

Used properties:

- `bbox` (`#f`)
- `circle-padding` (`0.2`)
- `font-size` (`0`)
- `thickness` (`1`)

\draw-circle *radius* (number) *thickness* (number) *filled* (boolean)

Draw a circle with given *radius* and *thickness*.

Fill the circle if *filled* is set to `#t`.

```
\markup {
  \draw-circle #2 #0.5 ##f
  \hspace #2
  \draw-circle #2 #0 ##t
}
```



`\draw-dashed-line` *dest* (pair of numbers)

Draw a dashed line along vector *dest*.

Properties `on` and `off` give the length of a dash and the space between two dashes, respectively; `phase` shortens the first dash by the given amount.

If the `full-length` property is set to `#t` (which is the default), the value of property `off` (and `on` under some boundary conditions) gets adjusted so that there is neither whitespace at the end of the line nor the last dash truncated.

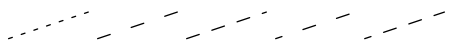
```
\markup {
  \override #'((on . 0.3) (off . 0.5))
  \draw-dashed-line #'(6 . 2)

  \draw-dashed-line #'(6 . 2)

  \override #'(full-length . #f)
  \draw-dashed-line #'(6 . 2)

  \override #'(phase . 0.5)
  \draw-dashed-line #'(6 . 2)

  \override #'((full-length . #f) (phase . 0.5))
  \draw-dashed-line #'(6 . 2)
}
```



Used properties:

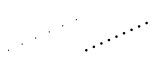
- `full-length` (`#t`)
- `phase` (`0`)
- `off` (`1`)
- `on` (`1`)
- `thickness` (`1`)

`\draw-dotted-line` *dest* (pair of numbers)

Draw a dotted line along vector *dest*.

Property `off` gives the space between two dots; its value gets adjusted so that the first and last dot exactly start and end the line, respectively. `phase` shifts all dots along the vector by the given amount.

```
\markup {
  \draw-dotted-line #'(5.1 . 2.3)
  \override #'((thickness . 2) (off . 0.2))
  \draw-dotted-line #'(5.1 . 2.3)
}
```



Used properties:

- `phase` (`0`)
- `off` (`1`)
- `thickness` (`1`)

`\draw-hline`

Draw a horizontal line.

The property `span-factor` sets the length of the line as a multiple of the `line-width` property.

```
\markup {
  \column {
    \draw-hline
    \override #'(span-factor . 1/3)
    \draw-hline
  }
}
```

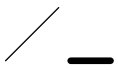


Used properties:

- `span-factor` (1)
- `line-width`
- `thickness` (1)

`\draw-line` *dest* (pair of numbers)
Draw a line along vector *dest*.

```
\markup {
  \draw-line #'(4 . 4)
  \override #'(thickness . 5)
  \draw-line #'(-3 . 0)
}
```



Used properties:

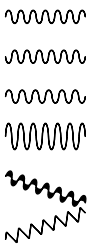
- `thickness` (1)

`\draw-squiggle-line` *sq-length* (number) *dest* (pair of numbers) *eq-end?* (boolean)
Draw a squiggled line along vector *dest*.

sq-length is the length of the first bow; this value gets always adjusted so that an integer number of squiggles is printed. If *eq-end?* is set to `#t`, the squiggled line ends with a bow in the same direction as the starting one.

The appearance of the squiggled line may be customized by overriding the `thickness`, `angularity`, `height`, and `orientation` properties.

```
\markup
  \column {
    \draw-squiggle-line #0.5 #'(6 . 0) ##t
    \override #'(orientation . -1)
    \draw-squiggle-line #0.5 #'(6 . 0) ##t
    \draw-squiggle-line #0.5 #'(6 . 0) ##f
    \override #'(height . 1)
    \draw-squiggle-line #0.5 #'(6 . 0) ##t
    \override #'(thickness . 5)
    \draw-squiggle-line #0.5 #'(6 . -2) ##t
    \override #'(angularity . 2)
    \draw-squiggle-line #0.5 #'(6 . 2) ##t
  }
```



Used properties:

- `orientation` (`1`)
- `height` (`0.5`)
- `angularity` (`0`)
- `thickness` (`0.5`)

`\ellipse` *arg* (markup)

Draw an ellipse around *arg*.

Use properties `thickness`, `x-padding`, `y-padding`, and `font-size` to set the line thickness and padding around the markup.

This is the same as function `\oval` but with different padding defaults.

```
\markup {
  \ellipse {
    Hi
  }
}
```



Note that the ellipse does not horizontally displace its argument. Use markup commands like `\left-align` or `\table` to make LilyPond realign it.

Used properties:

- `y-padding` (`0.2`)
- `x-padding` (`0.2`)
- `font-size` (`0`)
- `thickness` (`1`)

`\epsfile` *axis* (number) *size* (number) *file-name* (string)

Inline an image *file-name*, scaled along *axis* to *size*.

See `\image` for details on this command; calling

```
\markup \epsfile axis size file-name
```

is the same as

```
\markup
  \override #'(background-color . #f)
  \image axis size file-name
```

`\filled-box` *xext* (pair of numbers) *yext* (pair of numbers) *blot* (number)

Draw a box of dimensions *xext* and *yext*, with rounded corners given by *blot*.

For example,

```
\filled-box #'(-.3 . 1.8) #'(-.3 . 1.8) #0
```

creates a box extending horizontally from -0.3 to 1.8 and vertically from -0.3 up to 1.8, with corners formed from a circle of diameter 0 (i.e., sharp corners).

```
\markup {
  \filled-box #'(0 . 4) #'(0 . 4) #0
  \filled-box #'(0 . 2) #'(-4 . 2) #0.4
  \combine
    \filled-box #'(1 . 8) #'(0 . 7) #0.2
    \with-color #white
      \filled-box #'(3.6 . 5.6) #'(3.5 . 5.5) #0.7
}
```



\hbracket *arg* (markup)
: Draw horizontal brackets around *arg*.

```
\markup {
  \hbracket {
    \line {
      one two three
    }
  }
}
```

one two three

\image *axis* (number) *size* (number) *file-name* (string)
: Inline an image *file-name*, scaled along *axis* to *size*.

  The image format is determined based on the extension of *file-name*, which should be `.png` for a PNG image, or `.eps` for an EPS image (`.PNG` and `.EPS` are allowed as well).

  EPS files are only supported in the PostScript backend – for all output formats –, and in the Cairo backend – when creating PostScript or EPS output.

  If the image has transparency, it will appear over a colored background with the color specified by the `background-color` property, which defaults to `"white"`.

  To disable the colored background, set `background-color` to `#f`. For EPS images, this always works (where EPS images work in the first place). On the other hand, for PNG images, it works in the Cairo backend (which can output all supported formats), as well as in the SVG backend, but *not* in the PostScript backend, which is the default. See Abschnitt “Advanced command-line options for LilyPond” in *Anwendungsbenutzung* for how to activate the Cairo backend.

  Use `\withRelativeDir` as a prefix to *file-name* if the file should be found relative to the input file.

  Used properties:

  - `background-color` (`"white"`)

\oval *arg* (markup)
: Draw an oval around *arg*.

  Use properties `thickness`, `x-padding`, `y-padding`, and `font-size` to set the line thickness and padding around the markup.

  This is the same as function `\ellipse` but with different padding defaults.

```
\markup {
  \oval {
    Hi
  }
}
```



Note that the oval does not horizontally displace its argument. Use markup commands like `\left-align` or `\table` to make LilyPond realign it.

Used properties:

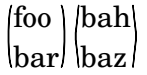
- `y-padding` (`0.75`)
- `x-padding` (`0.75`)
- `font-size` (`0`)
- `thickness` (`1`)

`\parenthesize` *arg* (markup)

Draw parentheses around *arg*.

This is useful for parenthesizing a column containing several lines of text.

```
\markup {
  \parenthesize
  \column {
    foo
    bar
  }
  \override #'(angularity . 2)
  \parenthesize
  \column {
    bah
    baz
  }
}
```



Used properties:

- `width` (`0.25`)
- `line-thickness` (`0.1`)
- `thickness` (`1`)
- `size` (`1`)
- `padding`
- `angularity` (`0`)

`\path` *thickness* (number) *commands* (list)

Draw a path with line *thickness* according to the directions given in *commands*.

commands is a list of lists where the `car` of each sublist is a drawing command and the `cdr` comprises the associated arguments for each command.

There are seven commands available to use in *commands*: `moveto`, `rmoveto`, `lineto`, `rlineto`, `curveto`, `rcurveto`, and `closepath`. Note that the commands that begin

with 'r' are the relative variants of the other three commands. You may also use the standard SVG single-letter equivalents: `moveto` = `M`, `lineto` = `L`, `curveto` = `C`, `closepath` = `Z`. The relative commands are written lowercase: `rmoveto` = `r`, `rlineto` = `l`, `rcurveto` = `c`.

The commands `moveto`, `rmoveto`, `lineto`, and `rlineto` take 2 arguments, namely the X and Y coordinates of the destination point.

The commands `curveto` and `rcurveto` create cubic Bézier curves, and take 6 arguments; the first two are the X and Y coordinates for the first control point, the second two are the X and Y coordinates for the second control point, and the last two are the X and Y coordinates for the destination point.

The `closepath` command takes zero arguments and closes the current subpath in the active path.

Line-cap styles and line-join styles may be customized by overriding the `line-cap-style` and `line-join-style` properties, respectively. Available line-cap styles are `butt`, `round`, and `square`. Available line-join styles are `miter`, `round`, and `bevel`.

The property `filled` specifies whether or not the path is filled with color.

```
samplePath =
  #'((lineto -1 1)
     (lineto 1 1)
     (lineto 1 -1)
     (curveto -5 -5 -5 5 -1 0)
     (closepath))

\markup \scale #'(2 . 2) {
  \path #0.25 #samplePath

  \override #'(line-join-style . miter)
  \path #0.25 #samplePath

  \override #'(filled . #t)
  \path #0.25 #samplePath
}
```



Used properties:

- `filled` (`#f`)
- `line-join-style` (`round`)
- `line-cap-style` (`round`)

`\polygon` *points* (list of number pairs)

A polygon delimited by the list of *points*.

Property `extroversion` defines how the shape of the polygon is adapted to its thickness: if it is 0, the polygon is traced as-is. If it is -1, the outer side of the line is just on the given points. If set to 1, the line has its inner side on the points. The `thickness` property controls the thickness of the line; for filled polygons, this means the diameter of the blot.

```
regularPentagon =
  #'((1 . 0) (0.31 . 0.95) (-0.81 . 0.59)
     (-0.81 . -0.59) (0.31 . -0.95))

\markup \scale #'(2 . 2) {
  \polygon #'((-1 . -1) (0 . -3) (2 . 2) (1 . 2))
  \override #'(filled . #f)
    \override #'(thickness . 2)
      \combine
        \with-color #(universal-color 'blue)
          \polygon #regularPentagon
        \with-color #(universal-color 'vermillion)
          \override #'(extroversion . 1)
            \polygon #regularPentagon
}
```



Used properties:

- `thickness` (1)
- `filled` (#t)
- `extroversion` (0)

`\postscript` *str* (string)

Insert *str* directly into the output as a PostScript command string.

This command is meant as a *last resort*. Almost all needs are better fulfilled by other markup commands (see, for example, `\path` and `\draw-line`). If you do use this command, keep the following points in mind:

- `\postscript` does not work in SVG output.
- Only a subset of the PostScript language is supported during the conversion from PostScript to PDF.
- There are no stability guarantees on the details of how LilyPond produces its own output (i.e., the context into which the PostScript code is inserted). They may change substantially across versions.
- LilyPond cannot understand the shape of the drawing, leading to suboptimal spacing. Usually, it is necessary to explicitly set up dimensions with a command like `\with-dimensions`.
- Depending on how you install LilyPond, the version of the PostScript interpreter (Ghostscript) can vary, and some of its features may be disabled.

str is processed with the following constraints.

- The string is embedded into the (intermediate) output file with the PostScript commands

  ```
  gsave currentpoint translate 0.1 setlinewidth
  ```

  before and

  ```
  grestore
  ```
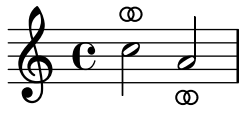
  after it.

- After these preceding commands (i.e., `currentpoint translate`) the origin of the current transformation is the reference point of `\postscript`. Scale and rotation of the current transformation reflect the global staff line distance and (if applied) other transformation markup commands (e.g., `\scale` and `\rotate`) encapsulating the `\postscript` command.
- The current point is set to the coordinate (0, 0).
- If an unwanted line appears at the beginning of your PostScript code, you are probably missing a call to `newpath`.

```
ringsps = "
  0.15 setlinewidth
  0.9 0.6 moveto
  0.4 0.6 0.5 0 361 arc
  stroke
  1.0 0.6 0.5 0 361 arc
  stroke
  "

rings = \markup {
  \with-dimensions #'(-0.2 . 1.6) #'(0 . 1.2)
  \postscript #ringsps
}

\relative c'' {
  c2^\rings
  a2_\rings
}
```
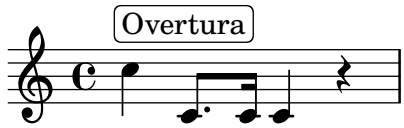


`\rounded-box` *arg* (markup)

Draw a box with rounded corners around *arg*.

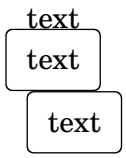
This function looks at properties `thickness`, `box-padding`, and `font-size` to determine the line thickness and padding around the *arg*. The `corner-radius` property defines the radius of the round corners (default value is 1).

```
c4^\markup {
  \rounded-box {
    Overtura
  }
}
c,8. c16 c4 r
```



Note that the box does not horizontally displace its argument. Use markup commands like `\left-align` or `\table` to make LilyPond realign it.

```
\markup {
  \override #'(box-padding . 1.5)
  \column {
    "text"
    \rounded-box "text"
    \left-align \rounded-box "text"
  }
}
```



Used properties:

- `box-padding` (`0.5`)
- `font-size` (`0`)
- `corner-radius` (`1`)
- `thickness` (`1`)

`\scale` *factor-pair* (pair of numbers) *arg* (markup)

Scale *arg*.

factor-pair is a pair of numbers representing the scaling factor of the X and Y axes. Negative values may be used to produce mirror images.

```
\markup {
  \line {
    \scale #'(2 . 1)
    stretched
    \scale #'(1 . -1)
    mirrored
  }
}
```



`\triangle` *filled* (boolean)

Draw a triangle.

Fill the triangle if *filled* is set to `#t`.

The appearance can be controlled with properties `extroversion`, `font-size`, and `thickness`.

```
\markup {
  \triangle ##t
  \triangle ##f
  \override #'(font-size . 5)
  \override #'(thickness . 5) {
    \override #'(extroversion . 1)
    \triangle ##f
    \override #'(extroversion . -1)
    \triangle ##f
  }
}
```



Used properties:

- `thickness` (1)
- `font-size` (0)
- `extroversion` (0)

`\with-url` *url* (string) *arg* (markup)
: Add a link to URL *url* around *arg*.

  This only works in the PDF backend.[2]

```
\markup {
  \with-url "https://lilypond.org/" {
    LilyPond ... \italic {
      music notation for everyone
    }
  }
}
```

LilyPond ... *music notation for everyone*

A.10.4 Markup for music and musical symbols

`\accidental` *alteration* (an exact rational number)
: Select an accidental glyph for *alteration*, given as a rational number.

  Use `\text-accidental` instead if you need glyph representation forms that fit and align well with text.

```
\markup {
  text
  \tiny { \accidental #1/2 \accidental #-1/2 }
  text
  \tiny { \text-accidental #1/2 \text-accidental #-1/2 }
  text
}
```

text ♯𝄲 text ♯𝄲 text

  Used properties:

  - `alteration-glyph-name-alist`

`\bar-line` *strg* (string)
: Print a bar line in markup.

  The allowed characters for input string *strg* are ‘`;|.:!S[]{}`’, having the same meaning as with the `\bar` command. The additional characters ‘`{`’ and ‘`}`’ denote left and right braces, respectively.

  The output is vertically centered.

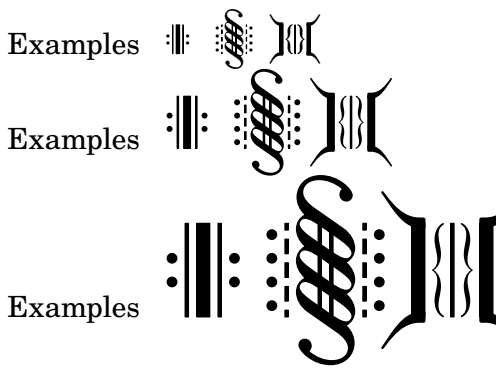
  Changes of `font-size` are respected.

  The default of `height` is 4 staff-space units. Apart from the bracket tips of a bracket bar line and the segno bar line all other bar lines are scaled with `height`. We don’t scale bracket tips and segno to meet the behavior of `SystemStartBracket` and the segno bar line.

[2] Due to technical limitations the link doesn’t work here in the Notation Reference.

\bar-line is further customizable by overriding dot-count and dash-count for dotted and dashed bar lines. The values for hair-thickness, kern and thick-thickness are customizable as well; defaults are the same as the values of the corresponding BarLine grob properties.

```
\markup {
   \override #'(word-space . 2)
   \column {
      \line {
        Examples
        \fontsize #-5 \translate-scaled #'(0 . 2) {
          \bar-line ":|.|:"
          \bar-line ";!S!;"
          \bar-line "]{|}["
        }
      }
      \line {
        Examples
        \fontsize #0 \translate-scaled #'(0 . 2) {
          \bar-line ":|.|:"
          \bar-line ";!S!;"
          \bar-line "]{|}["
        }
      }
      \line {
        Examples
        \fontsize #5 \translate-scaled #'(0 . 2) {
          \bar-line ":|.|:"
          \bar-line ";!S!;"
          \bar-line "]{|}["
        }
      }
   }
}
```



Used properties:

- thick-thickness (6.0)
- kern (3.0)
- hair-thickness (1.9)
- dash-count (5)
- dot-count (4)

- `height` (4)
- `font-size` (0)

`\coda` Draw a coda sign.

```
\markup {
  \coda
}
```

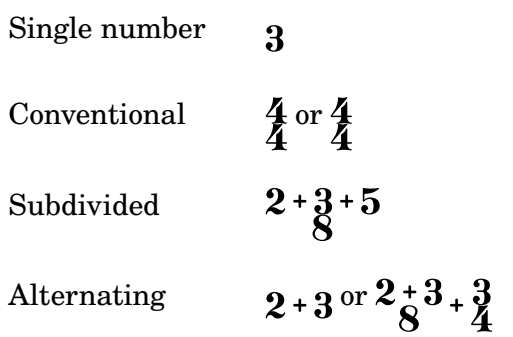


`\compound-meter` *time-sig* (number or pair)

Draw a numeric time signature based on *time-sig*.

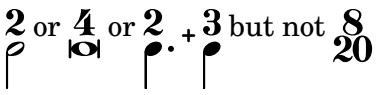
time-sig can be a single number, a pair of numbers, a simple list, or a list of lists, as the following example demonstrates.

```
\markuplist {
  \override #'(baseline-skip . 4.5)
  \override #'(padding . 4.5)
  \table #'(-1 -1) {
    "Single number"  \compound-meter #3
    "Conventional"   \line {
                       \compound-meter #'(4 . 4) or
                       \compound-meter #'(4 4)
                     }
    "Subdivided"     \compound-meter #'(2 3 5 8)
    "Alternating"    \line {
                       \compound-meter #'((2) (3)) or
                       \compound-meter #'((2 3 8) (3 4))
                     }
  }
}
```



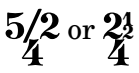
Setting the `denominator-style` property to `note` prints denominators as a note and dots when exact representation is possible. Example:

```
\markup {
  \override #'(denominator-style . note)
  \line {
    \compound-meter #'(2 2) or
    \compound-meter #'(4 1/2) or
    \compound-meter #'((2 8/3) (3 4)) but not
    \compound-meter #'(8 20)
  }
}
```



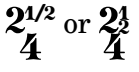
The `nested-fraction-mixed` property controls whether fractional parts are printed as mixed numbers or as common fractions. Example:

```
\markup {
  \override #'(nested-fraction-mixed . #f)
  \compound-meter #'(5/2 4) or
  \override #'(nested-fraction-mixed . #t)
  \compound-meter #'(5/2 4)
}
```



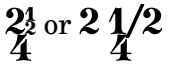
The `nested-fraction-orientation` property controls how nested fractions are arranged. Supported values are `horizontal` and `vertical`. Example:

```
\markup {
  \override #'(nested-fraction-orientation . horizontal)
  \compound-meter #'(5/2 4) or
  \override #'(nested-fraction-orientation . vertical)
  \compound-meter #'(5/2 4)
}
```



The `nested-fraction-relative-font-size` property controls the size of the numerals in nested fractions. Recommended values are -5.5 and 0. Using large numerals may take precedence over related properties. Example:

```
\markup {
  \override #'(nested-fraction-relative-font-size . -5.5)
  \compound-meter #'(5/2 4) or
  \override #'(nested-fraction-relative-font-size . 0)
  \compound-meter #'(5/2 4)
}
```



Used properties:

- `note-staff-position` (-2)
- `note-head-style` (())
- `note-flag-style` (())
- `note-dots-direction` (0)
- `nested-fraction-relative-font-size` (())
- `nested-fraction-orientation` (default)
- `nested-fraction-mixed` (#t)
- `font-size` (0)
- `denominator-style` (default)

\customTabClef *num-strings* (integer) *staff-space* (number)

Draw a clef in sans-serif style for a tablature with *num-strings* lines spaced by *staff-space*.

This markup command is used to implement \clef moderntab within a TabStaff context.

```
\markup {
  \customTabClef #4 #1
}
```



\doubleflat

Draw a double flat symbol.

```
\markup {
  \doubleflat
}
```



\doublesharp

Draw a double sharp symbol.

```
\markup {
  \doublesharp
}
```



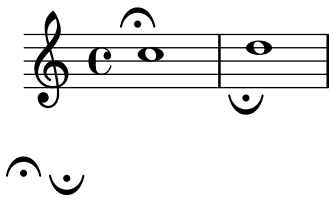
\fermata Create a fermata glyph.

If property direction is DOWN, use an inverted glyph.

Note that within music, one would normally use the \fermata articulation instead of a markup.

```
{ c''1^\markup \fermata d''1_\markup \fermata }

\markup { \fermata \override #`(direction . ,DOWN) \fermata }
```



Used properties:

- direction (1)

\flat Draw a flat symbol.

```
\markup {
  \flat
}
```



\multi-measure-rest-by-number *length* (non-negative, exact integer)
Return a multi-measure rest symbol for *length* measures.

If the number of measures is greater than the number given by `expand-limit` a thick horizontal line is printed. For every multi-measure rest lasting more than one measure a number is printed on top. However, if property `multi-measure-rest-number` is set to `#t`, this number gets suppressed.

```
\markup {
  Multi-measure rests may look like
  \multi-measure-rest-by-number #12
  or
  \override #'(multi-measure-rest-number . #f)
    \multi-measure-rest-by-number #7
  (church rests)
}
```



Used properties:

- `multi-measure-rest-number` (`#t`)
- `width` (`8`)
- `expand-limit` (`10`)
- `hair-thickness` (`2.0`)
- `thick-thickness` (`6.6`)
- `word-space`
- `style` (`()`)
- `font-size` (`0`)

\musicglyph *glyph-name* (string)
Print music symbol *glyph-name*.

See Abschnitt “The Emmentaler font” in *Notationsreferenz* for a complete listing of the possible glyph names.

```
\markup {
  \musicglyph "f"
  \musicglyph "rests.2"
  \musicglyph "clefs.G_change"
}
```



\natural Draw a natural symbol.

```
\markup {
  \natural
}
```



\note *duration* (duration) *dir* (number)

Draw a note with a given duration and a stem.

The note duration is specified with *duration* (setting both the note head type and the number of augmentation dots). The stem direction and length is given by *dir*.

This function is wrapper around `\note-by-number`; its documentation gives more details on the available properties.

```
\markup {
  \note {4..} #UP
  \hspace #2
  \override #'(style . cross)
    \note {4..} #0.75
  \hspace #2
  \note {\breve} #0
}
```



Used properties:

- `style` (`default`)
- `dots-direction` (`0`)
- `flag-style` (`()`)
- `font-size` (`0`)

\note-by-number *log* (number) *dot-count* (number) *dir* (number)

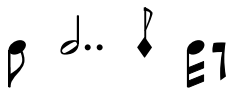
Draw a note with a given length, a number of dots, and a stem.

The note length is specified with *log*, the number of dots with *dot-count*, and the stem direction and length with *dir*. Fractional values for *dir* can be used to control the length of the stem. Value 0 suppresses the stem completely; this is useful for note head styles that don't take stems or come with built-in stems (like Kievan).

Ancient note head styles (via the `style` property, siehe Abschnitt A.9 [Notenkopfstile], Seite 678) get mensural-style flags by default; use the `flag-style` property to override this. Supported flag styles are `default`, `old-straight-flag`, `modern-straight-flag`, `flat-flag`, `stacked`, `mensural`, and `neomensural`. The last flag style is the same as `mensural` and provided for convenience.

```
\markup {
  \note-by-number #3 #0 #DOWN
  \hspace #2
  \note-by-number #1 #2 #0.8
  \hspace #2
  \override #'(style . petrucci)
    \note-by-number #3 #0 #UP
  \hspace #2
  \override #'(flag-style . modern-straight-flag)
    \note-by-number #4 #0 #DOWN
  \override #'(style . kievan)
```

```
    \note-by-number #2 #0 #0
}
```



See also function \note.

Used properties:

- style (default)
- dots-direction (0)
- flag-style (())
- font-size (0)

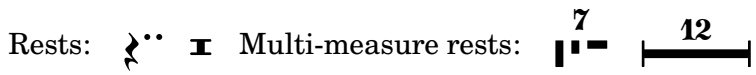
\rest *duration* (duration)

Return a rest symbol with length *duration*.

If the multi-measure-rest property is set to #t, a multi-measure rest symbol may be returned. In this case the duration needs to be entered as { 1*N* } to get a multi-measure rest for *N* bars. Actually, only the scaling factor (i.e., the number after '*') determines the length; the basic duration is disregarded.

See also functions \rest-by-number and \multi-measure-rest-by-number for more information on the used properties.

```
\markup {
  Rests:
  \hspace #2
  \rest { 4.. }
  \hspace #2
  \rest { \breve }
  \hspace #2
  Multi-measure rests:
  \override #'(multi-measure-rest . #t)
  {
    \hspace #2
    \rest { 1*7 }
    \hspace #2
    \rest { 1*12 }
  }
}
```



Used properties:

- multi-measure-rest-number (#t)
- width (8)
- expand-limit (10)
- hair-thickness (2.0)
- thick-thickness (6.6)
- word-space
- style (())
- font-size (0)

- `style` (())
- `ledgers` ((-1 0 1))
- `font-size` (0)

`\rest-by-number` *log* (integer) *dot-count* (integer)
: Draw a rest of length *log*, with *dot-count* dots.

  For duration logs that appear in the `ledgers` property, rest symbols with ledger lines are selected.

```
\markup {
  \rest-by-number #3 #2
  \hspace #2
  \rest-by-number #0 #1
  \hspace #2
  \rest-by-number #-1 #0
  \hspace #2
  \override #'(ledgers . ())
  \rest-by-number #-1 #0
}
```
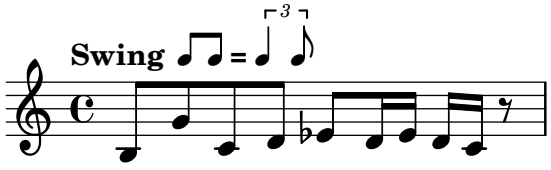


  Used properties:

  - `style` (())
  - `ledgers` ((-1 0 1))
  - `font-size` (0)

`\rhythm` *music* (music)
: Draw embedded rhythmic pattern as specified by *music*.

```
\relative {
  \tempo \markup {
    Swing
    \hspace #0.4
    \rhythm { 8[ 8] } = \rhythm { \tuplet 3/2 { 4 8 } }
  }
  b8 g' c, d ees d16 ees d c r8
}
```



  Within `\rhythm`, there is no time signature and no division in measures (as with `\cadenzaOn`, siehe Abschnitt 2.3.4 [Musik ohne Metrum], Seite 73). Beaming must be added explicitly with the syntax explained in Abschnitt 2.4.3 [Manuelle Balken], Seite 94.

```
\markup {
  The rhythmic pattern \rhythm { 16[ 8 16] } is
  a type of syncopation.
}
```

The rhythmic pattern  is a type of syncopation.

\stemDown can be used to flip the stems.

```
\markup \rhythm { \stemDown 8 16 8 }
```



\rhythm works by creating a StandaloneRhythmVoice context, which is enclosed in a StandaloneRhythmStaff context, which is enclosed in a StandaloneRhythmScore context. It is possible to apply global tweaks to the output by using a \layout block.

```
\layout {
  \context {
    \StandaloneRhythmVoice
    \xNotesOn
  }
}

\markup \rhythm { 8 16 8 }
```



Used properties:

- font-size (-2)

\score *score* (score)
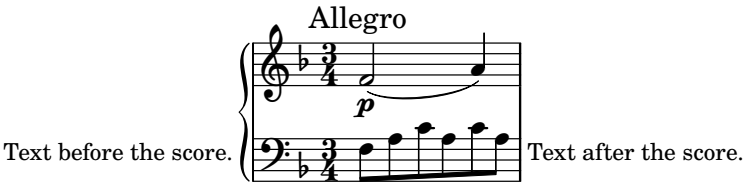
Inline an image of music as specified by *score*.

The reference point (usually the middle staff line) of the lowest staff in the top system is placed on the baseline.

No page breaks and no MIDI output, i.e., both \pageBreak commands and \midi{} blocks get ignored.

```
\markup {
  Text before the score.
  \score {
    \new PianoStaff <<
      \new Staff \relative c' {
        \key f \major
        \time 3/4
        \mark \markup { Allegro }
        f2\p( a4)
      }
      \new Staff \relative c {
        \clef bass
        \key f \major
        \time 3/4
        f8( a c a c a
      }
    >>
```

```
      \layout {
        indent = 0.0\cm
      }
    }
    Text after the score.
}
```



Used properties:

- `baseline-skip`

`\segno` Draw a segno symbol.

```
\markup {
  \segno
}
```



`\semiflat`

Draw a semiflat symbol.

```
\markup {
  \semiflat
}
```



`\semisharp`

Draw a semisharp symbol.

```
\markup {
  \semisharp
}
```



`\sesquiflat`

Draw a 3/2 flat symbol.

```
\markup {
  \sesquiflat
}
```



`\sesquisharp`

Draw a 3/2 sharp symbol.

```
\markup {
  \sesquisharp
}
```



\sharp Draw a sharp symbol.

```
\markup {
  \sharp
}
```



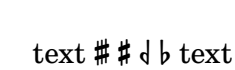
\text-accidental *alteration* (an exact rational number)

Get an accidental for *alteration* that aligns well with text.

alteration must be a rational number; the function uses the `alteration-glyph-name-alist` property to map this number to an accidental glyph.

If no text variant glyph for the accidental is found, markup is constructed to fittingly scale and position the (non-text) accidental glyph.

```
\markup {
  text
  \tiny { \text-accidental #3/4 \text-accidental #1/2
          \text-accidental #-1/4 \text-accidental #-1/2 }
  text
}
```



Used properties:

- `alteration-glyph-name-alist`

\text-common-time

Draw a common time symbol for text.

```
\markup {
  \text-common-time
}
```



\text-cut-time

Draw a cut time symbol for text.

```
\markup {
  \text-cut-time
}
```



\text-doubleflat

Draw a double flat symbol for text.

```
\markup {
  \text-doubleflat
}
```



`\text-doublesharp`
: Draw a double sharp symbol for text.

```
\markup {
  \text-doublesharp
}
```



`\text-flat`
: Draw a flat symbol for text.

```
\markup {
  \text-flat
}
```



`\text-natural`
: Draw a natural symbol for text.

```
\markup {
  \text-natural
}
```



`\text-sharp`
: Draw a sharp symbol for text.

```
\markup {
  \text-sharp
}
```



`\tied-lyric` *str* (string)
: Replace ‚~‘ tilde symbols with tie characters in *str*.

```
\markup \column {
  \tied-lyric
    "Siam navi~all'onde~algenti Lasciate~in abbandono"
  \tied-lyric
    "Impetuosi venti I nostri~affetti sono"
  \tied-lyric
    "Ogni diletto~e scoglio Tutta la vita~e~un mar."
}
```

Siam navi all'onde algenti Lasciate in abbandono
Impetuosi venti I nostri affetti sono
Ogni diletto e scoglio Tutta la vita e un mar.

Used properties:

- `word-space`

`\varcoda` Draw a varcoda sign.

```
\markup {
  \varcoda
}
```

𝄋

A.10.5 Conditional markup

`\if` *condition?* (procedure) *argument* (markup)

Test *condition?*, and only insert *argument* if it is true.

The condition is provided as a procedure taking an output definition and a property alist chain. The procedure is applied, and its result determines whether to print the markup. This command is most useful inside `oddHeaderMarkup` or similar. Here is an example printing page numbers in bold:

```
\paper {
  oddHeaderMarkup =
    \markup \fill-line {
      ""
      \if #print-page-number
           \bold \fromproperty #'page:page-number-string
    }
  evenHeaderMarkup =
    \markup \fill-line {
      \if #print-page-number
           \bold \fromproperty #'page:page-number-string
      ""
    }
}
```

`\unless` *condition?* (procedure) *argument* (markup)

Test *condition?*, and only insert *argument* if it is false.

This function is similar to `\if`; the following example shows how to print the copyright notice on all pages but the last instead of just the first page.

```
\paper {
  oddFooterMarkup = \markup {
    \unless #on-last-page-of-part \fill-line {
      \fromproperty #'header:copyright
    }
  }
}

\header {
  copyright = "© LilyPond Authors. License: GFDL."
  tagline = "© LilyPond Authors.  Documentation placed
under the GNU Free Documentation License
version 1.3."
}
```

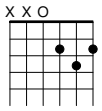
A.10.6 Instrument-specific markup

`\fret-diagram` *definition-string* (string)

Make a (guitar) fret diagram based on *definition-string*.

For example, say

```
\markup \fret-diagram "s:1.25;6-x;5-x;4-o;3-2;2-3;1-2;"
```



for fret spacing 5/4 of staff space, D chord diagram.

Syntax rules for *definition-string*:

– Diagram items are separated by semicolons.

– Possible items:

- `s:`*number* – Set the fret spacing of the diagram (in staff spaces). Default: 1.
- `t:`*number* – Set the line thickness (relative to normal line thickness). Default: 0.5.
- `h:`*number* – Set the height of the diagram in frets. Default: 4.
- `w:`*number* – Set the width of the diagram in strings. Default: 6.
- `f:`*number* – Set fingering label type (0 = none, 1 = in circle on string, 2 = below string). Default: 0.
- `d:`*number* – Set radius of dot, in terms of fret spacing. Default: 0.25.
- `p:`*number* – Set the position of the dot in the fret space. 0.5 is centered; 1 is on lower fret bar, 0 is on upper fret bar. Default: 0.6.
- `c:`*string1-string2-fret* – Include a barre mark from *string1* to *string2* on *fret*.
- *string-fret* – Place a dot on *string* at *fret*. If *fret* is '`o`', *string* is identified as open. If *fret* is '`x`', *string* is identified as muted.
- *string-fret-fingering* – Place a dot on *string* at *fret*, and label with *fingering* as defined by the `f:` code.

– Note: There is no limit to the number of fret indications per string.

Used properties:

- `thickness` (`0.5`)
- `fret-diagram-details`
- `size` (`1.0`)
- `align-dir` (`-0.4`)

`\fret-diagram-terse` *definition-string* (string)

Make a fret diagram markup using terse string-based syntax.

For example,

```
\markup \fret-diagram-terse "x;x;o;2;3;2;"
```



displays a D chord diagram.

Syntax rules for *definition-string*:

- Strings are terminated by semicolons; the number of semicolons is the number of strings in the diagram.
- Mute strings are indicated by '`x`'.
- Open strings are indicated by '`o`'.
- A number indicates a fret indication at that fret.

- If there are multiple fret indicators desired on a string, they should be separated by spaces.
- Fingerings are given by following the fret number with a '`-`' followed by the finger indicator, e.g., '`3-2`' for playing the third fret with the second finger.
- Where a barre indicator is desired, follow the fret (or fingering) symbol with `-(` to start a barre and `-)` to end the barre.

Used properties:

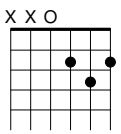
- `thickness` (`0.5`)
- `fret-diagram-details`
- `size` (`1.0`)
- `align-dir` (`-0.4`)

`\fret-diagram-verbose` *marking-list* (pair)

Make a fret diagram containing the symbols indicated in *marking-list*.

The following example produces a standard D chord diagram without fingering indications.

```
\markup \scale #'(1.5 . 1.5)
  \fret-diagram-verbose
    #'((mute 6) (mute 5) (open 4)
       (place-fret 3 2) (place-fret 2 3) (place-fret 1 2))
```



Possible elements in *marking-list*:

`(mute` *`string-number`*`)`

Place a small ‚x‘ at the top of string *string-number*.

`(open` *`string-number`*`)`

Place a small ‚o‘ at the top of string *string-number*.

`(barre` *`start-string end-string fret-number`*`)`

Place a barre indicator (much like a tie) from string *start-string* to string *end-string* at fret *fret-number*.

`(capo` *`fret-number`*`)`

Place a capo indicator (a large solid bar) across the entire fretboard at fret location *fret-number*. Also, set fret *fret-number* to be the lowest fret on the fret diagram.

`(place-fret` *`string-number fret-number`* `[`*`finger-value`*`] [`*`color-modifier`*`] [`*`color`*`] ['parenthesized ['default-paren-color]])`

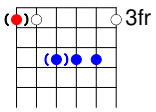
Place a fret playing indication on string *string-number* at fret *fret-number* with an optional fingering label *finger-value*, an optional color modifier *color-modifier*, an optional color *color*, an optional parenthesis `'parenthesized` and an optional parenthesis color `'default-paren-color`.

By default, the fret playing indicator is a solid dot. This can be globally changed by setting the value of the property `dot-color` or for a single dot by setting the value of *color*. The dot can be parenthesized by adding `'parenthesized`. By default the color for the parenthesis is taken from

the dot. Adding `'default-paren-color` will take the parenthesis color from the global `dot-color` property; as a fallback black will be used. Setting *color-modifier* to `inverted` inverts the dot color for a specific fingering.

The values for *string-number*, *fret-number*, and the optional *finger* should be entered first in that order. The order of the other optional arguments does not matter. If the *finger* part of the `place-fret` element is present, *finger-value* will be displayed according to the setting of the variable *finger-code*. There is no limit to the number of fret indications per string.

```
\markup \scale #'(1.5 . 1.5)
  \fret-diagram-verbose #'(
    (place-fret 6 3 1 red parenthesized default-paren-color)
    (place-fret 5 3 1 inverted)
    (place-fret 4 5 2 blue parenthesized)
    (place-fret 3 5 3 blue)
    (place-fret 2 5 4 blue)
    (place-fret 1 3 1 inverted)
  )
```



Used properties:

- `thickness` (`0.5`)
- `fret-diagram-details`
- `size` (`1.0`)
- `align-dir` (`-0.4`)

`\harp-pedal` *definition-string* (string)

Make a harp pedal diagram containing the symbols indicated in *definition-string*.

Possible elements in *definition-string*:

`^`	pedal is up
`-`	pedal is neutral
`v`	pedal is down
`|`	vertical divider line
`o`	the following pedal should be circled (indicating a change)

```
\markup \harp-pedal "^-v|--ov^"
```



The function also checks whether the string has the typical form of three pedals, then the divider, and then the remaining four pedals, printing a warning otherwise (without preventing the non-standard order).

Use the `size` property to control the overall size, and the `thickness` property for the line thickness of the horizontal line and the divider.

The remaining configuration is done via the `harp-pedal-details` property; it contains the following elements:

`box-offset`
: vertical shift of the box center for up/down pedals

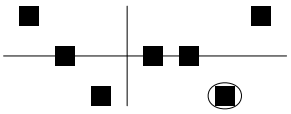
`box-width`
: box width

`box-height`
: box height

`space-before-divider`
: spacing between two boxes before the divider

`space-after-divider`
: spacing between two boxes after the divider

```
\markup {
  \override #'((size . 1.5)
               (harp-pedal-details . ((box-width . 1)
                                      (box-offset . 2))))
  \harp-pedal "^-v|--ov^"
}
```



For global modification of `harp-pedal-details`, i.e., outside of the current `\markup` block, you can also use code similar to

```
\override Voice.TextScript.harp-pedal-details.box-width = 1
```

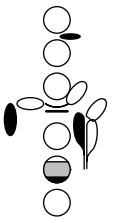
Used properties:

- `thickness` (`0.5`)
- `harp-pedal-details` (`()`)
- `size` (`1.2`)

`\woodwind-diagram` *instrument* (symbol) *user-draw-commands* (list)
Make a woodwind-instrument diagram for *instrument* using *user-draw-commands*.

user-draw-commands is a list of alists, specifying the left-hand keys, the elements on the central column, and the right-hand keys. For example, this diagram

```
\markup \woodwind-diagram
  #'oboe #'((lh . (d ees))
            (cc . (five3qT1q))
            (rh . (gis)))
```



shows an oboe with the left-hand d key, left-hand ees key, and right-hand gis key depressed, while the five-hole of the central column effectuating a trill between 1/4 and 3/4 is closed.

The following instruments are supported:

- piccolo
- flute
- oboe
- clarinet
- bass-clarinet
- saxophone
- bassoon
- contrabassoon

To see all of the callable keys for a given instrument, include the function call `(print-keys '`*`instrument`*`)` in your `.ly` file, where *instrument* is the instrument whose keys you want to print.

Certain keys allow for special configurations. The entire gamut of configurations possible is as follows:

`1q`	1/4 covered
`1h`	1/2 covered
`3q`	3/4 covered
`R`	ring depressed
`F`	fully covered; the default if no state put

Additionally, these configurations can be used in trills. So, for example, `three3qTR` effectuates a trill between 3/4 full and ring depressed on the three hole. As another example, `threeRT` effectuates a trill between R and open, whereas `threeTR` effectuates a trill between open and shut. To see all of the possibilities for all of the keys of a given instrument, invoke `(print-keys-verbose '`*`instrument`*`)`.

Lastly, substituting an empty list for the pressed-key alist results in a diagram with all of the keys drawn but none filled, for example

```
\markup \woodwind-diagram #'flute #'()
```



Used properties:

- `woodwind-diagram-details` (())
- `font-size` (0)
- `graphical` (#t)
- `thickness` (0.1)
- `size` (1)

A.10.7 Accordion registers

`\discant` *name* (string)

Generate a discant accordion register symbol for *name*.

To make it available,

```
#(use-modules (lily accreg))
```

is required near the top of your input file.

The register names in the default `\discant` register set have been modeled after the numeric Swiss notation (as depicted in `http://de.wikipedia.org/wiki/Register_%28Akkordeon%29`), omitting the slashes and dropping leading zeros.

The string *name* is basically a three-digit number with the lowest digit specifying the number of 16' reeds, the tens the number of 8' reeds, and the hundreds specifying the number of 4' reeds. Without modification, the specified number of reeds in 8' is centered in the symbol. Newer instruments may have registrations where 8' can be used either within or without a tone chamber, ‚cassotto'. Notationally, the central dot then indicates use of cassotto. One can suffix the tens' digits '`1`' and '`2`' with '`+`' or '`-`' to indicate clustering the dots at the right or left, respectively, rather than centered.

Some examples are



```
\discant "1"
```



```
\discant "1+0"
```



```
\discant "120"
```



```
\discant "131"
```

Used properties:

- `font-size` (`0`)

`\freeBass` *name* (string)

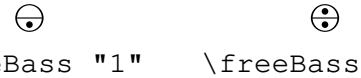
Generate a free bass/converter accordion register symbol for the usual two-reed layout as given by *name*.

To make it available,

```
#(use-modules (lily accreg))
```

is required near the top of your input file.

Available registrations are



```
\freeBass "1"
```



```
\freeBass "11"
```



```
\freeBass "10"
```

Used properties:

- `font-size` (`0`)

`\stdBass` *name* (string)

Generate a standard bass accordion register symbol for *name*.

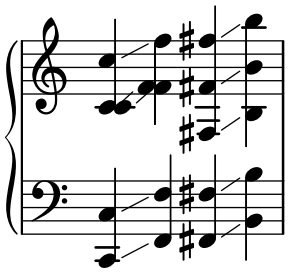
To make it available,

```
#(use-modules (lily accreg))
```

is required near the top of your input file.

The default bass register definitions have been modeled after the article `http://www.accordions.com/index/art/stradella.shtml` originally appearing in *Accord Magazine*.

The underlying register model is



This kind of overlapping arrangement is common for Italian instruments though the exact location of the octave breaks differ.

When not composing for a particular target instrument, using the five-reed definitions makes more sense than using a four-reed layout: in that manner, the '`Master`' register is unambiguous. This is rather the rule in literature bothering about bass registrations at all.

Available registrations are



`\stdBass "Soprano"`



`\stdBass "Soft Bass"`



`\stdBass "Alto"`



`\stdBass "Soft Tenor"`



`\stdBass "Tenor"`



`\stdBass "Bass/Alto"`



`\stdBass "Master"`

Used properties:

- `font-size` (0)

`\stdBassIV` *name* (string)
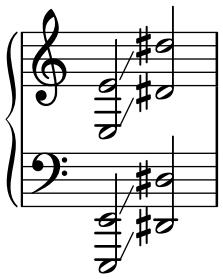
Generate a standard bass accordion register symbol for *name*.

To make it available,

```
#(use-modules (lily accreg))
```

is required near the top of your input file.

The main use is for four-reed standard bass instruments with reedbank layout



Notable instruments are Morino models with MIII (the others are five-reed instead) and the Atlantic IV. Most of those models have three register switches. Some newer Morinos with MIII might have five or even seven.

The prevalent three-register layout uses the middle three switches '`Tenor`', '`Master`', '`Soft Bass`'. Note that the sound is quite darker than the same registrations of '`c,`'-based instruments.

Available registrations are



`\stdBassIV "Soprano"`



`\stdBassIV "Soft Bass"`



`\stdBassIV "Alto"`



`\stdBassIV "Bass/Alto"`



`\stdBassIV "Tenor"`



`\stdBassIV "Soft Bass/Alto"`



`\stdBassIV "Master"`



`\stdBassIV "Soft Tenor"`

Used properties:

- `font-size` (0)

`\stdBassV` *name* (string)
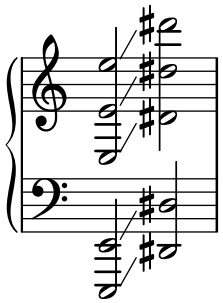
Generate a standard bass accordion register symbol for *name*.

To make it available,

```
#(use-modules (lily accreg))
```

is required near the top of your input file.

The main use is for five-reed standard bass instruments with reedbank layout



This tends to be the bass layout for Hohner's Morino series without converter or MIII manual.

With the exception of the rather new 7-register layout, the highest two chord reeds are usually sounded together. Older instruments offer 5 or 3 bass registers. The Tango VM offers an additional '`Solo Bass`' setting that mutes the chord reeds. The symbol on the register buttons of the Tango VM would actually match the physical five-octave layout reflected here, but it is not used in literature.

Composers should likely prefer the five-reed versions of these symbols. The mismatch of a four-reed instrument with five-reed symbols is easier to resolve for the player than the other way round.

Available registrations are



Used properties:

- `font-size` (0)

`\stdBassVI` *name* (string)

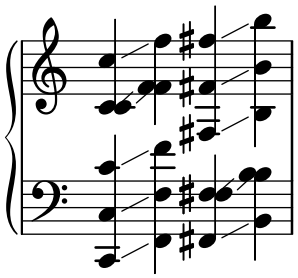
Generate a standard bass accordion register symbol for six-reed basses as given by *name*.

To make it available,

```
#(use-modules (lily accreg))
```

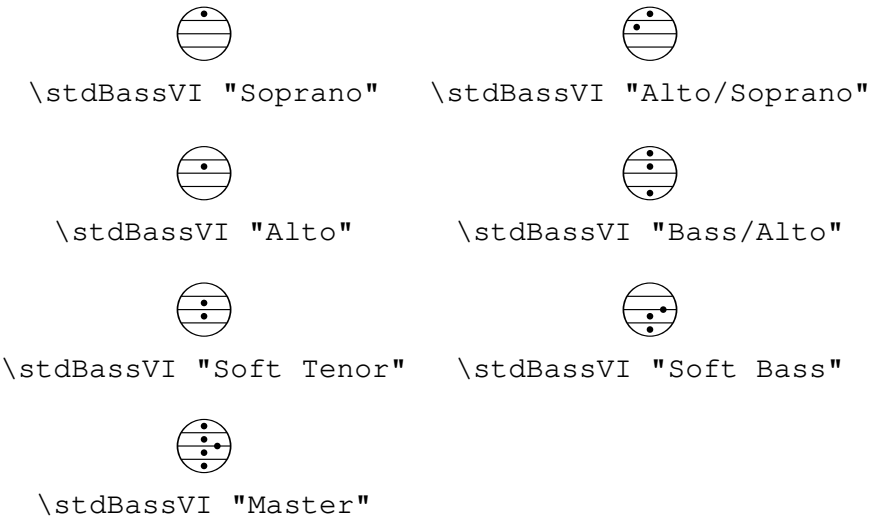
is required near the top of your input file.

This is primarily the register layout for the Hohner „Gola“ model. The layout is



The registers are effectively quite similar to that of \stdBass. An additional bass reed at alto pitch is omitted for aesthetical reasons from the 'Master' setting, so the symbols are almost the same except for the 'Alto/Soprano' register with bass notes at Alto pitch and chords at Soprano pitch.

Available registrations are



Used properties:

- font-size (0)

A.10.8 Other markup commands

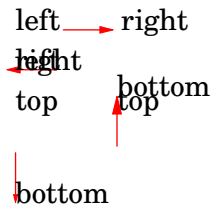
\annotate-moving *arg* (markup)

Indicate \vspace and \hspace movement with an arrow.

The arrow changes its size and thickness depending on the printed length; the maximum size of the arrow head can be controlled with the size property. If size exceeds a third of the length of the final arrow, it falls back to that third.

Note that the arrows do not reflect the actual extents of the objects created by \vspace and \hspace; you might use \box for that.

```
\markup
  \column {
    \line { left \annotate-moving \hspace #4 right }
    \line { left \annotate-moving \hspace #-4 right }
    \line {
      \column {
        top
        \override #'(size . 0.6)
          \annotate-moving \vspace #4/3 bottom
      }
      \column {
        top
        \override #'(size . 2.0)
          \annotate-moving \vspace #-4/3 bottom
      }
    }
  }
```



Used properties:

- `size` (`1`)
- `color` (`"red"`)

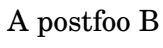
`\append-to-tag` *tag* (symbol) *more* (markup) *arg* (markup)

Append *more* to all markup in var *arg* tagged with *tag*.

It works similar to `\appendToTag` for music, but only with markups.

```
tagged = \markup {
  \tag #'foo A
  \tag #'bar B
}

\markup { \append-to-tag #'foo postfoo \tagged }
```



Used properties:

- `tags-with-appends-alist` (`()`)

`\auto-footnote` *mkup* (markup) *note* (markup)

Have footnote *note* act as an annotation to the markup *mkup*.

```
\markup {
  \auto-footnote a b
  \override #'(padding . 0.2)
  \auto-footnote c d
}
```



The footnote will be annotated automatically.

Used properties:

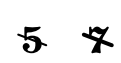
- `padding` (`0.0`)

- `raise` (`0.5`)

`\backslashed-digit` *num* (integer)
: Print number *num* with the Emmentaler font, crossed through with a backslash.

  This is for use in the context of figured bass notation.

```
\markup {
  \backslashed-digit #5
  \hspace #2
  \override #'(thickness . 3)
  \backslashed-digit #7
}
```



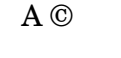
  Used properties:

  - `thickness` (`1.6`)
  - `font-size` (`0`)

`\char` *num* (integer)
: Produce a single Unicode character with code *num*.

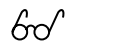
  Characters encoded in hexadecimal format require the prefix `#x`.

```
\markup {
  \char #65 \char ##x00a9
}
```



`\eyeglasses`
: Prints out eyeglasses, indicating strongly to look at the conductor.

```
\markup { \eyeglasses }
```



`\first-visible` *args* (markup list)
: Use the first markup in *args* that yields a non-empty stencil and ignore the rest.

```
\markup {
  \first-visible {
    \fromproperty #'header:composer
    \italic Unknown
  }
}
```



`\footnote` *mkup* (markup) *note* (markup)
: Have footnote *note* act as an annotation to the markup *mkup*.

```
\markup {
  \footnote a b
  \override #'(padding . 0.2)
  \footnote c d
}
```

a c

b
d

The footnote will not be annotated automatically.

`\fraction` *arg1* (markup) *arg2* (markup)
Make a fraction of markups *arg1* and *arg2*.

```
\markup {
  π ≈ \fraction 355 113
}
```

$$\pi \approx \frac{355}{113}$$

Used properties:

- `font-size` (0)

`\fromproperty` *symbol* (symbol)
Read *symbol* from the property settings and produce a stencil from the markup contained within.

If *symbol* is not defined or is not a markup, return an empty markup.

Currently, the following properties can be accessed.

- Within a `\paper` block defining titles, headers, or footers, or within a `\header` block: all fields from the `\header` block (that produce markup) are available, with `header:` as a name prefix.
- Within a `\paper` block defining headers or footers: the current page number (symbol `page:page-number-string`).
- Within the `tocItemMarkup` paper variable (or in custom-made Scheme code that uses function `add-toc-item!`) defining a table of contents entry: the entry's text and page number are available as `toc:text` and `toc:page`, respectively. An entry's indentation markup is available as `toc:indent`.

```
\header {
  myTitle = "myTitle"
  title = \markup {
    from
    \italic
    \fromproperty #'header:myTitle
  }
}
\markup {
```

```
  \null
}
```

from *myTitle*

\keep-with-tag *tags* (symbol list or symbol) *arg* (markup)

Keep markup from *arg* that is untagged or tagged with *tags*.

All other parts of *arg* that are using a different tag are replaced with empty stencils. It works similar to \keepWithTag for music, but only with markups.

```
tagged = \markup {
  untagged
  \tag #'foo A
  \tag #'bar B
}

\markup { \keep-with-tag #'bar \tagged }
\markup { \keep-with-tag #'foo \tagged }
```

untagged B

untagged A

Used properties:

- tags-to-keep (())

\left-brace *size* (number)

Print a brace from the music font, of height *size* (in points).

```
\markup {
  \left-brace #35
  \hspace #2
  \left-brace #45
}
```



\lookup *glyph-name* (string)

Print a brace glyph with name *glyph-name*.

This is a historical command; \left-brace (which looks up the glyph by absolute size and is independent of the font size) is recommended instead.

```
\markup \lookup "brace200"
```



\markalphabet *num* (integer)

Make a markup letter for *num*.

The letters start with A to Z and continue with double letters.

```
\markup {
  \markalphabet #8
  \hspace #2
  \markalphabet #26
}
```

H Z

`\markletter` *num* (integer)
: Make a markup letter for *num*.

  The letters start with A to Z (skipping letter I), and continue with double letters.

```
\markup {
  \markletter #8
  \hspace #2
  \markletter #26
}
```

H AA

`\null`
: An empty markup with extents of a single point.

```
\markup {
  \null
}
```

`\on-the-fly` *procedure* (procedure) *arg* (markup)
: Apply the *procedure* markup command to *arg*.

  procedure takes the same arguments as `interpret-markup` and returns a stencil.

`\override` *new-prop* (pair) *arg* (markup)
: Add the argument *new-prop* to the property list for printing *arg*.

  In general, any property may be overridden that is part of `font-interface` (Abschnitt "font-interface" in *Referenz der Interna*), `text-interface` (Abschnitt "text-interface" in *Referenz der Interna*), or `instrument-specific-markup-interface` (Abschnitt "instrument-specific-markup-interface" in *Referenz der Interna*). Additionally, various markup commands listen to other properties, too, as described in a markup function's documentation.

  new-prop is either a single alist pair or a non-empty list of alist pairs.

```
\markup {
  \undertie "undertied"
  \override #'(offset . 15)
  \undertie "offset undertied"
  \override #'((offset . 15) (thickness . 3))
  \undertie "offset thick undertied"
}
```

undertied offset undertied offset thick undertied

`\page-link` *page-number* (number) *arg* (markup)
: Add a link to a score's page *page-number* around *arg*.

  This only works in the PDF backend.

```
\markup {
  \page-link #2 { \italic { This links to page 2... } }
}
```

This links to page 2...

\page-ref *label* (symbol) *gauge* (markup) *default* (markup)
: Print a page number reference.

  label is the label set on the referenced page (using \label or \tocItem), *gauge* a markup used to estimate the maximum width of the page number, and *default* the value to display when *label* is not found.

  If the current book or book part is set to use roman numerals for page numbers, the reference will be formatted accordingly – in which case the *gauge*'s width may require additional tweaking.

  Used properties:

  - x-align (1)

\pattern *count* (non-negative, exact integer) *axis* (non-negative, exact integer) *space* (number) *pattern* (markup)
: Print a *pattern* markup *count* times.

  Patterns are spaced apart by *space* (defined as for \hspace or \vspace, respectively) and distributed on *axis*.

```
\markup \column {
  "Horizontally repeated:"
  \pattern #7 #X #2 \flat
  \null
  "Vertically repeated:"
  \pattern #3 #Y #0.5 \flat
}
```

Horizontally repeated:
♭ ♭ ♭ ♭ ♭ ♭ ♭

Vertically repeated:
♭
♭
♭

\property-recursive *symbol* (symbol)
: Print out a warning when header field markup in *symbol* contains some recursive markup definition.

\push-to-tag *tag* (symbol) *more* (markup) *arg* (markup)
: Prepend *more* to all markup in *arg* tagged with *tag*.

  It works similar to \pushToTag for music, but only with markups.

```
tagged = \markup {
  \tag #'foo A
  \tag #'bar B
}

\markup { \push-to-tag #'foo prefoo \tagged }
```

prefoo A B

Used properties:

- `tags-with-pushes-alist` (`()`)

`\qr-code` *width* (non-negative number) *str* (string)

Insert a QR code for string *str*, usually a URL, with a given *width*.

```
\markup \vcenter {
  \center-column { Engraved with LilyPond }
  \hspace #1.5
  \qr-code #10.0 "https://lilypond.org"
}
```

Engraved with LilyPond 

The `error-correction-level` property can be set to one of the symbols `low`, `medium`, `quarter`, or `high`. The higher the level of error correction is, the more the QR code contains redundancy, potentially helping detectors, e.g., in poor lighting conditions; however, a higher correction level also makes the code denser.

```
\markup \vcenter {
  \center-column { Engraved with LilyPond }
  \hspace #1.5
  \override #'(error-correction-level . high)
    \qr-code #10.0 "https://lilypond.org"
}
```

Engraved with LilyPond 

The `quiet-zone-size` property specifies the width of the „quiet zone“, namely the white area around the QR code. It is expressed as a multiple of the width of one little square inside the QR code. Use at least 4 for best results.

Used properties:

- `quiet-zone-size` (`4`)
- `error-correction-level` (`low`)

`\remove-with-tag` *tags* (symbol list or symbol) *arg* (markup)

Remove markup from *arg* that is tagged with *tags*.

The removed markup is replaced with empty stencils. It works similar to `\removeWithTag` for music, but only with markups.

```
tagged = \markup {
  \tag #'foo A
  \tag #'bar B
}

\markup { \remove-with-tag #'foo \tagged }
\markup { \remove-with-tag #'bar \tagged }
```

B

A

Used properties:

- `tags-to-remove` (())

`\replace-with-tag` *tag* (symbol) *replacement* (markup) *arg* (markup)

Replace tagged markups in *arg*.

Everything tagged with *tag* (including the tagging itself) gets replaced with *replacement*.

It works similar to `\replaceWithTag` for music, but only with markups.

```
tagged = \markup {
  \tag #'foo A
  \tag #'bar B
}

\markup { \replace-with-tag #'foo C \tagged }
```

C B

Used properties:

- `tags-with-replacement-alist` (())

`\right-brace` *size* (number)

A music brace in point size *size*, rotated 180 degrees.

```
\markup {
  \right-brace #45
  \hspace #2
  \right-brace #35
}
```

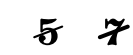


`\slashed-digit` *num* (integer)

Print number *num* with the Emmentaler font, crossed through with a slash.

This is for use in the context of figured bass notation.

```
\markup {
  \slashed-digit #5
  \hspace #2
  \override #'(thickness . 3)
  \slashed-digit #7
}
```



Used properties:

- `thickness` (`1.6`)
- `font-size` (`0`)

`\stencil` *stil* (stencil)
: Use stencil *stil* as markup.

```
\markup {
  \stencil #(make-circle-stencil 2 0 #t)
}
```



`\strut`
: Create a box of the same height as the space in the current font.

`\tag` *tags* (symbol list or symbol) *arg* (markup)
: Tag markup *arg* with *tag*.

tag can be one or multiple tags. This allows later on to reference *arg*; for example, to remove it or to add markup before or after the tagged markup. It works similar to `\tag` for music, but only with markups.

```
tagged = \markup {
  \tag #'foo A
  \tag #'bar B
}

\markup { \keep-with-tag #'bar \tagged }
\markup { \keep-with-tag #'foo \tagged }
```

B

A

Used properties:

- `tags-with-replacement-alist` (`()`)
- `tags-with-appends-alist` (`()`)
- `tags-with-pushes-alist` (`()`)
- `tags-to-remove` (`()`)
- `tags-to-keep` (`()`)

`\transparent` *arg* (markup)
: Make *arg* transparent.

```
\markup {
  \transparent {
    invisible text
  }
}
```

\verbatim-file *name* (string)
: Read the contents of file *name* and include it verbatim.

```
\markup {
  \verbatim-file "en/included/simple.ly"
}
```

```
% A simple piece in LilyPond, a scale.
\version "2.19.21"
\relative {
  c' d e f g a b c
}
```

Use \withRelativeDir as a prefix to *name* if the file should be found relative to the input file.

\whiteout *arg* (markup)
: Provide a white background for *arg*.

The shape of the white background is determined by the style property. The default is box which produces a rectangle. rounded-box produces a rounded rectangle, and outline approximates the outline of the markup.

The color of the background can be controlled with the color property, defaulting to "white".

```
\markup {
  \combine
    \filled-box #'(-1 . 62) #'(-3 . 4) #1
    \override #'(line-width . 60)
      \fill-line {
        \override #'(thickness . 1.5)
          \whiteout box
        \override #'((style . rounded-box) (thickness . 3))
          \whiteout rounded-box
        \override #'((style . outline) (thickness . 3))
          \whiteout outline
        \override #'((color . "red") (style . outline))
          \whiteout red-outline
      }
}
```



Used properties:

- color ("white")
- thickness (())

- `style` (box)

`\with-color` *col* (color) *arg* (markup)
Use color *col* to draw *arg*.

Siehe Abschnitt 7.1.4 [Farbige Objekte], Seite 218, for valid color specifications.

```
\markup {
  \with-color #red red
  \hspace #2
  \with-color #green green
  \hspace #2
  \with-color "#0000ff" blue
}
```

red green blue

`\with-dimension` *axis* (integer) *val* (pair of numbers) *arg* (markup)
Set the dimension of *arg* along *axis* to *val*.

If *axis* is equal to `X`, set the horizontal dimension. If *axis* is equal to `Y`, set the vertical dimension.

`\with-dimension-from` *axis* (integer) *arg1* (markup) *arg2* (markup)
Print *arg2* but replace the dimension along *axis* with the one from *arg1*.

If *axis* is set to `X`, replace the horizontal dimension. If *axis* is set to `Y`, replace the vertical dimension.

`\with-dimensions` *x* (pair of numbers) *y* (pair of numbers) *arg* (markup)
Set the horizontal and vertical dimensions of *arg* to *x* and *y*.

`\with-dimensions-from` *arg1* (markup) *arg2* (markup)
Print *arg2* with the horizontal and vertical dimensions of *arg1*.

`\with-link` *label* (symbol) *arg* (markup)
Add a link to the page holding label *label* around *arg*.

This only works in the PDF backend.

```
\markup {
  \with-link #'label {
    \italic { This links to the page
              containing the label... }
  }
}
```

`\with-outline` *outline* (markup) *arg* (markup)
Print *arg* with the outline and dimensions of *outline*.

The outline is used by skylines to resolve collisions (not for whiteout).

`\with-true-dimension` *axis* (integer) *arg* (markup)
Give *arg* its actual dimension (extent) on *axis*.

Sometimes, the extents of a markup's printed ink differs from the default extents. The main case is if glyphs are involved. By default, the extents of a glyph are based on the glyph's *metrics* (i.e., a default vertical and horizontal size for the glyph), which, for various reasons, are often not identical to its *bounding box* (i.e., the smallest rectangle that completely encompasses the glyph's outline) – in most cases, the outline protrudes the box spanned up by the metrics.

```
\markup {
  text
  \fontsize #10
  \override #'((box-padding . 0) (thickness . 0.2))
  \box
    \musicglyph "scripts.trill"
  text
}
```



For purposes other than setting text, this behavior may not be wanted. You can use `\with-true-dimension` in order to give the markup its actual printed extent.

```
\markup {
  text
  \fontsize #10
  \override #'((box-padding . 0) (thickness . 0.2))
  \box
    \with-true-dimension #X
    \musicglyph "scripts.trill"
  text
}
```



`\with-true-dimensions` *arg* (markup)
: Give *arg* its actual dimensions (extents).

  Calling

  ```
  \markup \with-true-dimensions arg
  ```

  is short for

  ```
  \markup
    \with-true-dimension #X
    \with-true-dimension #Y
    arg
  ```

  i.e., `\with-true-dimensions` has the effect of `\with-true-dimension` on both axes.

A.11 Textbeschriftungslistenbefehle

Folgende Befehle können mit dem Befehl `\markuplist` zusammen benutzt werden:

`\column-lines` *args* (markup list)
: Stack the markups in *args* vertically.

  Like `\column`, but return a list of lines instead of a single markup. The property `baseline-skip` determines the space between each markup in *args*.

  Used properties:

  - `baseline-skip`

`\justified-lines` *args* (markup list)
: Print *args* as lines aligned both at the left and the right.

Like \justify, but return a list of lines instead of a single markup. Use \override-lines #'(line-width . *X*) to set the line width; *X* is the number of staff spaces.

Used properties:

- text-direction (1)
- word-space
- line-width (#f)
- baseline-skip

\override-lines *new-prop* (pair) *args* (markup list)

Add the argument *new-prop* to the property list for printing *args*.

Like \override but for markup lists.

\score-lines *score* (score)

Inline an image of music as specified by *score*.

Like \score but return a list of lines instead of a single markup.

Used properties:

- tags-with-replacement-alist (())
- tags-with-appends-alist (())
- tags-with-pushes-alist (())
- tags-to-remove (())
- tags-to-keep (())

\string-lines *str* (string)

Split string *str* into lines.

The character to split at is specified by the property split-char, defaulting to #\newline. Surrounding whitespace is removed from every resulting string. The returned list of markups is ready to be formatted by other markup or markup list commands like \column, \line, etc.

```
\markup {
  \column
    \string-lines
     "foo, foo,
     bar, bar,
     buzz, buzz!"
}
```

foo, foo,
bar, bar,
buzz, buzz!

Used properties:

- split-char (#\newline)

\table *column-align* (number list) *lst* (markup list)

Print a table.

column-align specifies how each column is aligned; possible values are -1, 0, and 1. The number of elements in *column-align* determines how many columns will be printed.

The entries to print are given by *lst*, a markup list. If needed, the last row is filled up with point-stencils.

Override the padding property to increase the horizontal distance between columns. Override baseline-skip to increase the vertical distance between rows.

```
% A markup command to print a fixed-width number.
\markup fwnum =
  \markup \override #'(font-features . ("ss01" "-kern"))
    \number \etc

\markuplist {
  \override #'(padding . 2)
  \table #'(0 1 0 -1) {
    \underline { center-aligned right-aligned
                 center-aligned left-aligned }
    one      \fwnum    1 thousandth \fwnum 0.001
    eleven   \fwnum   11 hundredth  \fwnum 0.01
    twenty   \fwnum   20 tenth      \fwnum 0.1
    thousand \fwnum 1000 one        \fwnum 1.0
  }
}
```

center-aligned	right-aligned	center-aligned	left-aligned
one	1	thousandth	0.001
eleven	11	hundredth	0.01
twenty	20	tenth	0.1
thousand	1000	one	1.0

Used properties:

- baseline-skip
- padding (0)

\table-of-contents

Print a table of contents.

This function uses the paper variable tocTitleMarkup for the title; it then prints \tocItem entries line by line.

Siehe Abschnitt 20.5 [Inhaltsverzeichnis], Seite 485, for a complete discussion.

Used properties:

- baseline-skip

\tag-list *tags* (symbol list or symbol) *arg* (markup list)

Tag markup list *arg* with *tag*.

tag can be one or multiple tags. This allows later on to reference *arg*; for example, to remove it or to add markup before or after the tagged markup list.

It works like the \tag command for markups but with markup lists. You will need it if you have to reference a whole list; for example, to use \push-to-tag and \append-to-tag without pushing or appending before or after every single item of the list, but before or after the whole list instead.

```
tagged = \markuplist {
  \tag-list #'foo { foo bar }
}

\markup { \push-to-tag #'foo test \tagged }
```

test foo bar

Used properties:

- `tags-with-replacement-alist` (())
- `tags-with-appends-alist` (())
- `tags-with-pushes-alist` (())
- `tags-to-remove` (())
- `tags-to-keep` (())

`\wordwrap-lines` *args* (markup list)

Print *args* as left-aligned lines.

Like `\wordwrap`, but return a list of lines instead of a single markup. Use `\override-lines #'(line-width . X)` to set the line width, where X is the number of staff spaces.

Used properties:

- `text-direction` (1)
- `word-space`
- `line-width` (#f)
- `baseline-skip`

A.12 Liste der Sonderzeichen

Folgende Sonderezeichen-Bezeichnungen können benutzt werden, zu mehr Details siehe Abschnitt 21.3.3 [ASCII-Aliase], Seite 502.

die HTML-Synatx wird benutzt und die meisten der Bezeichnungen sind die gleichen wie für HTML. Der Rest ist durch LaTeX inspiriert.

Die Charakter haben einen Rahmen, sodass ihre Größe sichtbar ist. Etwas Verschiebung wurde zwischen Zeichen und Rahmen zur besseren Lesbarkeit eingefügt.

¡	¡	¿	¿	&solidus;	⁄	&flq;	‹		
&frq;	›	&flqq;	«	&frqq;	»	&glq;	‚		
&grq;	‘	&glqq;	„	&grqq;	“	&elq;	‘		
&erq;	’	&elqq;	“	&erqq;	”				
						&nnbsp;			
‍		‌		·	·	•	•		
&copyright;	©	&registered;	®	&trademark;	™	†	†		
‡	‡	№	№	ª	ª	º	º		
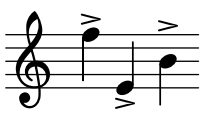
¶	¶	§	§	°	°	№	№		
‰	‰	¦	¦	´	´	&acutedbl;	˝		
`	`	˘	˘	ˇ	ˇ	&cedilla;	¸		
&circumflex;	^	&diaeresis;	¨	&macron;	¯	&aa;	å		
&AA;	Å	&ae;	æ	&AE;	Æ	ä	ä		
Ä	Ä	&dh;	ð	&DH;	Ð	&dj;	đ		
&DJ;	Đ	&l;	ł	&L;	Ł	&ng;	ŋ		
&NG;	Ŋ	&o;	ø	&O;	Ø	&oe;	œ		
&OE;	Œ	ö	ö	Ö	Ö	&s;	ſ		
&ss;	ß	&th;	þ	&TH;	Þ	ü	ü		
Ü	Ü	+	+	−	−	×	×		
÷	÷	¹	¹	²	²	³	³		
&sqrt;	√	&increment;	∆	&infty;	∞	∑	∑		
±	±	&bulletop;	∙	&partial;	∂	&neg;	¬		
&currency;	¤	$	$	€	€	&pounds;	£		
¥	¥	¢	¢						

A.13 Liste der Artikulationszeichen

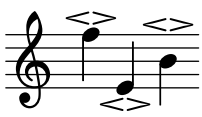
Die Skripte unten sind in der Feta-Glyphe definiert und können an Noten angehängt werden (etwa ‘c\accent’).

A.13.1 Artikulationsskripte
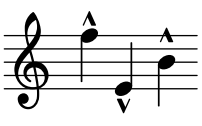
`\accent` or `->`
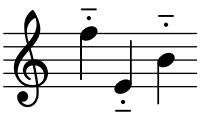


`\espressivo`
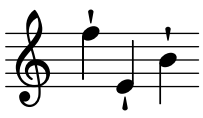


`\marcato` or `-^`
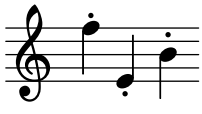


`\portato` or `-_`
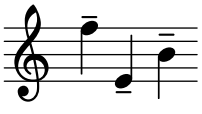


`\staccatissimo` or `-!`



`\staccato` or `-.`
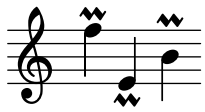


`\tenuto` or `--`
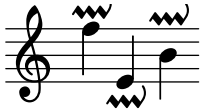


A.13.2 Ornamentale Skripte

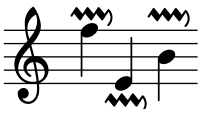
\prall
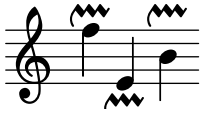


\prallup
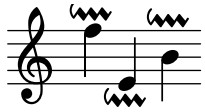


\pralldown



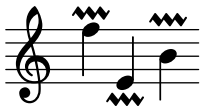
\upprall
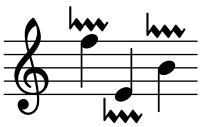


\downprall
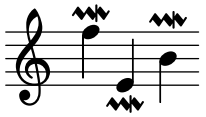


\prallprall
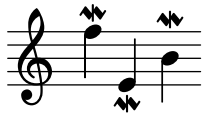


\lineprall
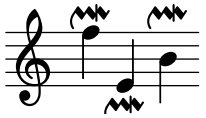


\prallmordent



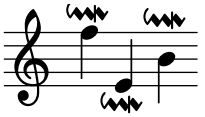
\mordent
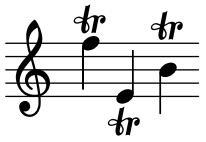


\upmordent
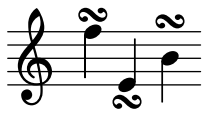


\downmordent
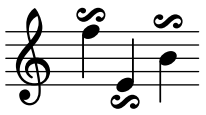


\trill
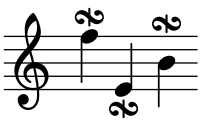


\turn



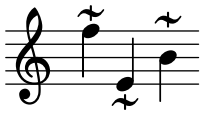
\reverseturn



\slashturn
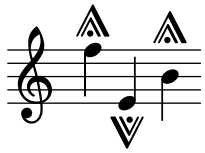


\haydnturn
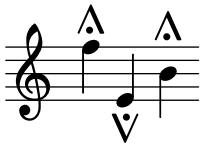


A.13.3 Fermatenskripte
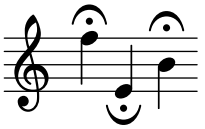
\veryshortfermata



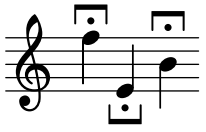
\shortfermata



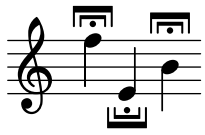
\fermata
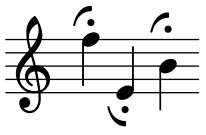


\longfermata
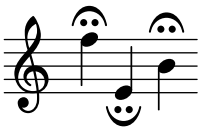


\verylongfermata



\henzeshortfermata

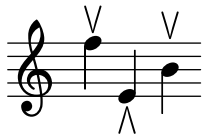


\henzelongfermata
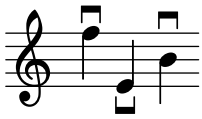


A.13.4 Instrumentenspezifische Skripte

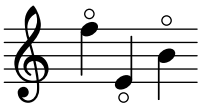
\upbow



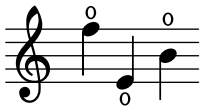
\downbow
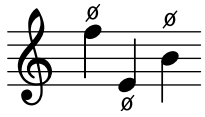


\flageolet



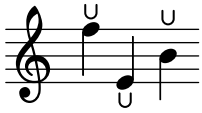
\open



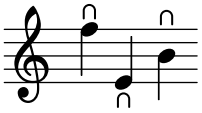
\halfopen



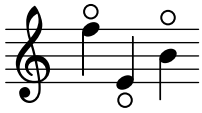
\heel
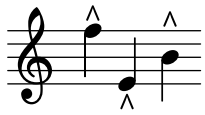


\varheel



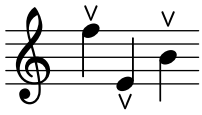
\heelcircle



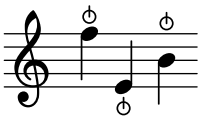
\toe
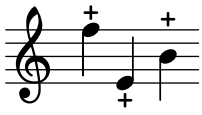


\vartoe



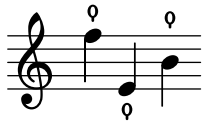
\snappizzicato



\stopped or -+
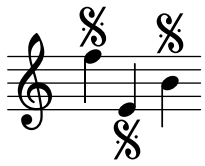


\thumb
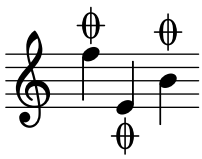


A.13.5 Wiederholungszeichenskripte

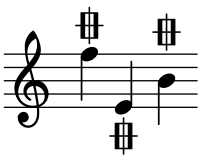
\segno



\coda



\varcoda

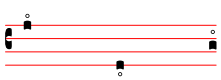


A.13.6 Skripte der alten Notation

\accentus



\circulus



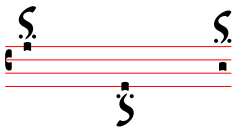
\ictus



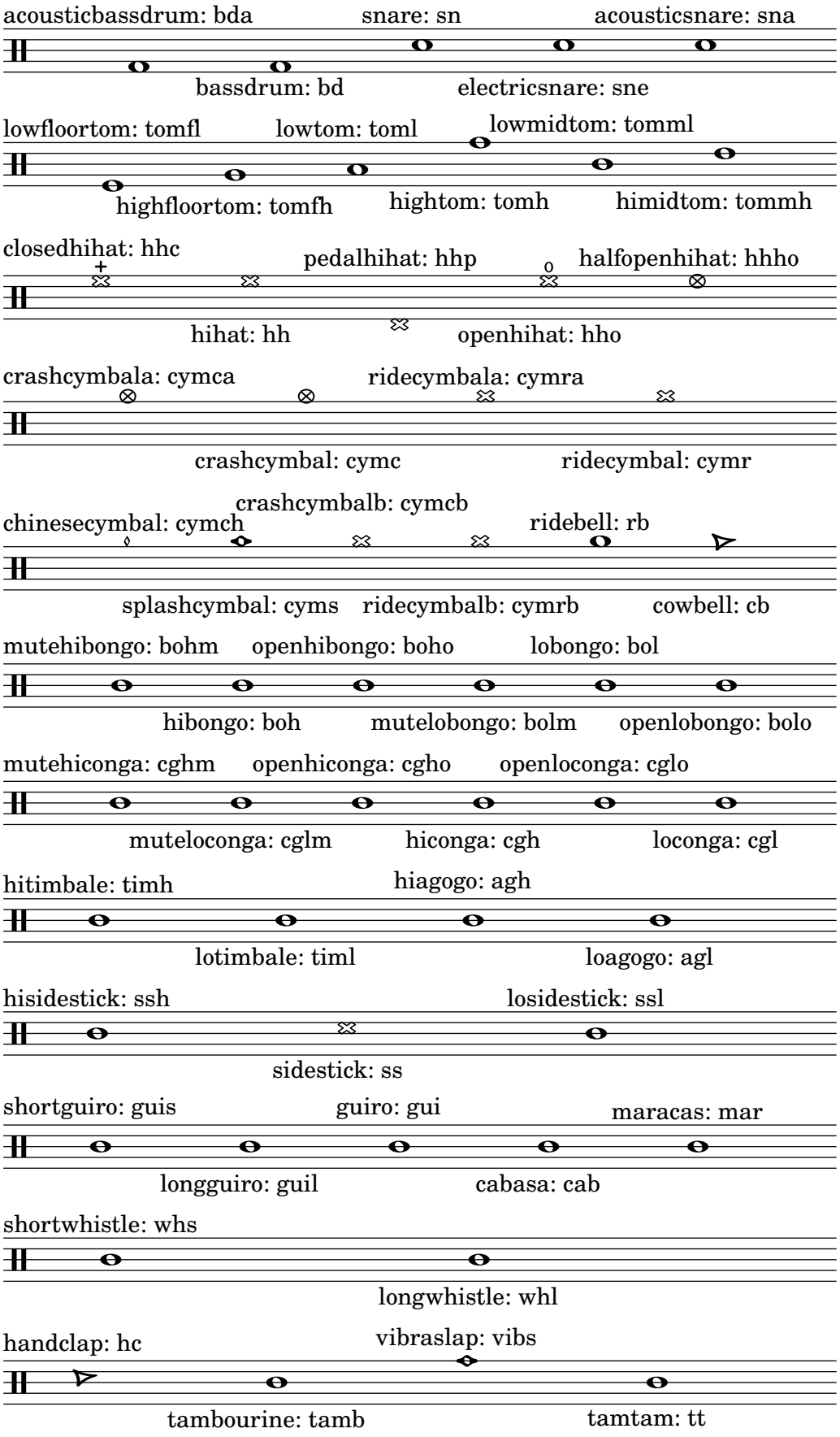
\semicirculus

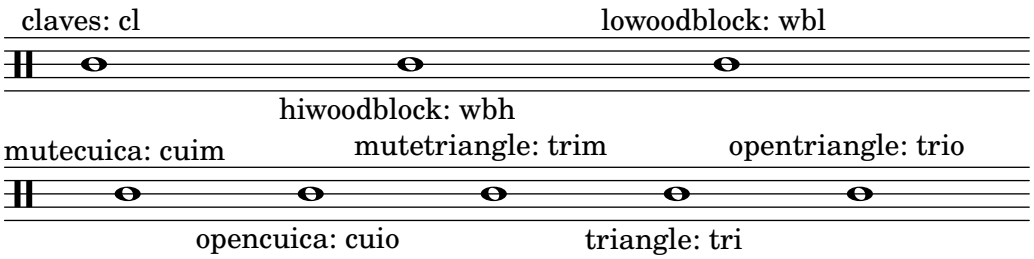


\signumcongruentiae



A.14 Schlagzeugnoten





A.15 Technisches Glossar

Ein Glossar der technischen Ausdrücke und Konzepte, die von LilyPond intern benutzt werden. Die Ausdrücke kommen in den Handbüchern, auf den Mailinglisten oder im Quellcode vor.

A.15.1 alist

Eine assoziative Liste oder **alist** in kurz ist ein Scheme-Paar, das einen Wert mit einem Schlüssel assoziiert: `(Schlüssel . Wert)`. In der Datei `scm/lily.scm` beispielsweise assoziiert die alist „type-p-name-alist“ bestimmte Prädikate (etwa `ly:music?`) mit Bezeichnungen (wie „music“) sodass Fehler der Typüberprüfung über eine Konsolennachricht mitgeteilt werden können, die auch die Bezeichnung des erwarteten Typprädikats mitteilt.

A.15.2 callback

Ein **callback** ist eine Routine, Funktion oder Methode, deren Referenz in einem Aufruf als Argument an eine andere Routine weitergereicht wird, sodass die aufgerufene Routine ermöglicht wird, das Argument zu aktivieren. Die Technik ermöglicht es einer niedrigeren Ebene des Programmes, eine Funktion aufzurufen, die auf höherer Ebene definiert wurde. Callbacks werden sehr ausgiebig in LilyPond eingesetzt, um es Scheme-Code auf der Benutzerebene zu erlauben, wie viele Funktionen der niedrigeren Ebene ausgeführt werden sollen.

A.15.3 closure

In Scheme entsteht ein **closure** (Abschluss), wenn eine Funktion, normalerweise ein Lambda-Ausdruck, als Variable weitergegeben wird. Das closure enthält den Code der Funktion plus Verweise zu den lexikalischen Verknüpfungen der freien Variablen der Funktion (also die Variablen, die in Ausdrücken benutzt werden, aber außerhalb von ihnen definiert werden). Wenn diese Funktion später einem anderen Argument zugewiesen wird, werden die freien Variabel-Verknüpfungend, die in das closure eingeschlossen sind, benutzt um die Werte der freien Variablen, die in der Rechnung benutzt werden sollen, zu errechnen. Eine nützliche Eigenschaft von closures ist, dass man interne variable Werte zwischen den Aufrufen wiederverwerten kann, sodass ein Status erhalten bleiben kann.

Ein **simple closure** (einfacher Abschluss) ist ein closure, dessen Ausdruck keine freien Variablen und auch keine freien Variablel-Verknüpfungen hat.

Ein simple closure wird in LilyPond von einem smob dargestellt, der den Ausdruck und eine Methode, wie der Ausdruck auf eine Liste von Argumenten angewendet werden soll, enthält.

A.15.4 glyph

Ein **glyph** (Glyphe) ist eine bestimmte graphische Repräsentation eines typographischen Charakters oder einer Kombination von zwei oder mehr Charakteren, die dann eine Ligatur bilden. Eine Gruppe an Glyphen des gleichen Stils bilden ein Font, und eine Gruppe an Schriftarten, die mehrere Stile darstellen, bilden eine Schriftfamilie (engl. typeface).

Siehe auch

Notationsreferenz: Abschnitt 8.3 [Schriftarten], Seite 247, Abschnitt 21.3 [Sonderzeichen], Seite 501.

A.15.5 grob

LilyPond-Objekte, die Elemente der Notation in der graphischen Ausgabe des Programmen darstellen, wie etwa Notenköpfe, Hälse, Bögen, Bindebögen, Fingersatz, Schlüssel usw., werden „Layout-Objekte“ genannt, auch oft als „GRaphische OBjekte“ bezeichnet, was dann zu **grob** abgekürzt wird.

Siehe auch

Handbuch zum Lernen: Abschnitt “Objekte und Schnittstellen” in *Handbuch zum Lernen*, Abschnitt “Regeln zur Benennung von Objekten und Eigenschaften” in *Handbuch zum Lernen*, Abschnitt “Eigenschaften von Layoutobjekten” in *Handbuch zum Lernen*.

Referenz der Interna: Abschnitt “All layout objects” in *Referenz der Interna*.

A.15.6 immutable

Ein **immutable** (unberührbares) Objekt ist ein, dessen Status nach der Erstellung nicht mehr verändert werden kann, entgegen einem mutable Objekt, das nach der Erstellung noch verändert werden kann.

In LilyPond sind unberührbare oder geteilte Eigenschaften das Standardverhalten von Grobs. Sie werden zwischen vielen Objekten geteilt. Entgegen ihrer Bezeichnung können sie jedoch mit `\override` und `\revert` verändert werden.

Siehe auch

Notationsreferenz: Abschnitt A.15.9 [mutable], Seite 765.

A.15.7 interface

Aktionen und Eigenschaften, die eine Gruppe von Grobs gemeinsam haben, werden in ein Objekt gesammelt, das als `grob-interface` oder auch „Schnittstelle“ (engl. interface) bezeichnet wird.

Siehe auch

Handbuch zum Lernen: Abschnitt “Objekte und Schnittstellen” in *Handbuch zum Lernen*, Abschnitt “Regeln zur Benennung von Objekten und Eigenschaften” in *Handbuch zum Lernen*, Abschnitt “Eigenschaften, die Schnittstellen besitzen können” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 33.2 [Layout-Schnittstellen], Seite 598.

Referenz der Interna: Abschnitt “Graphical Object Interfaces” in *Referenz der Interna*.

A.15.8 lexer

Ein **lexer** ist ein Programm, das eine Charaktersequenz in eines Sequenz von Tokens übersetzt. Dieser Prozess wird als lexikalische Analyse bezeichnet. Der LilyPond-Lexer konvertiert eine Eingabedatei (`.ly` in eine Datei mit Tokens, die sich besser für den nächsten Schritt der Verarbeitung, nämlich das Parsen, eignet. Siehe Abschnitt A.15.11 [parser], Seite 766.

A.15.9 mutable

Ein **mutable** (veränderbares) Objekt ist ein Objekt, dessen Status verändert werden kann, im Gegenteil zu einem immutable Objekt, dessen Status zur Entstehungszeit festgelegt ist.

In LilyPond enthalten mutable Eigenschaften Werte, die nur für einen Grob gelten. Normalerweise werden Listen von anderen Objekten oder Resultate einer Berechnung in mutablen Eigenschaften gespeichert.

Siehe auch

Notationsreferenz: Abschnitt A.15.6 [immutable], Seite 765.

A.15.10 output-def

Eine Instanz der `Output-def`-Klasse enthält die Methoden und Datenstruktur, die mit einem Ausgabeabschnitt assoziiert wird. Instanzen werden für `midi`, `layout` und `paper`-Umgebungen erstellt.

A.15.11 parser

Ein **parser** (Syntaxanalysierer) analysiert die Tokensequenzen, die von einem Lexer erstellt wurden, um deren grammatikalische Struktur zu entschlüsseln, wie sie von den Regeln des Eingabeformates vorgegeben werden. Dabei werden die Sequenzen in immer größere Gruppen entsprechend den grammatischen Regeln zusammengefasst. Wenn die Kette der Tokens gültig ist, ist das Endprodukt ein Token-Baum, dessen Wurzel das Startsymbol der Grammatik ist. Wenn dies nicht erreicht werden kann, ist die Datei nicht korrekt und ensprechende Fehlermeldungen werden ausgegeben. Die syntaktischen Gruppierungen und die Regeln, nach welchen die Gruppen aus ihren Einzelteilen nach der LilyPond-Syntax erstellt werden, finden sich in der Datei `lily/parser.yy`. Diese Datei wird benutzt, um den Parser während der Programmkompilation zu erstellen. Hierzu wird der Parser-Ersteller Bison verwendet. Er ist Teil des Quellcodes und nicht in die binäre Installation von LilyPond integriert.

A.15.12 parser variable

Diese Variablen werden direkt in Scheme definiert. Von ihrer direkten Benutzung durch den Benutzer wird streng abgeraten, weil ihre Semantikzuordnung sehr verwirrend sein kann.

Wenn der Wert einer derartigen Variable in einer `.ly`-Datei verändert wird, ist diese Änderung global, und wenn sie nicht explizit rückgängig gemacht wird, wird der neue Wert bis zum Ende der Datei gelten und dabei sowohl aufeinander folgende `\score`-Umgebungen als auch externe Dateien, die mit `\include` geladen werden, beeinflussen. Das kann zu nicht gewollten Konsequenzen führen, und in komplizierteren Projekten kann es sehr schwer sein, die immer wieder auftretenden Fehler zu beheben.

LilyPond benutzt folgende Parser-Variablen:

- afterGraceFraction
- musicQuotes
- mode
- output-count
- output-suffix
- partCombineListener
- pitchnames
- toplevel-bookparts
- toplevel-scores
- showLastLength
- showFirstLength

A.15.13 prob

Property OBjects, also Eigenschaftsobjekte, oder kurz **Prob**, sind Mitglieder der `Prob`-Klasse, eine einfache Basisklasse für Objekte, die mutable oder immutable alists haben und die Methoden, mit denen sie verändert werden können. Die `Music`- und `Stream_event`-Klassen stammen von `Prob` ab. Verkörperungen der `Prob`-Klasse werden auch erstellt, um formatierte Inhalte von Systemgrobs und Titelblöcken während der Erstellung des Seitenlayouts zu speichern.

A.15.14 simple closure

Siehe Abschnitt A.15.3 [closure], Seite 764.

A.15.15 smob

Smobs sind ScheMe-OBjekte, Teile des Mechanismus von Guile, um C- und C++-Ojekte in Scheme-Code zu exportieren. In LilyPond werden Smobs von C++-Objekten mithilfe von Makros erstellt. Es gibt zwei Arten von Smob-Objekten: einfache Smobs, die da sind für einfach immutable Objekte wie Nummern, und komplexe Smobs, benutzt für Objekte mit einer Identität. Wenn Sie auf die LilyPond-Quellen zurückgreifen können, findet sich mehr Information hierzu in `lily/includes/smob.hh`.

A.15.16 stencil

Eine Einheit der **stencil**-Klasse enthält die Information, die benötigt wird um ein typographisches Objekt zu setzen. Es handelt sich um einen sehr einfachen Smob, der eine begrenzende Box enthält, welche die vertikale und horizontale Ausdehnung des Objekt beschreibt, und einen Scheme-Ausdruck, der das Objekt ausgibt, nachdem es ausgewertet wurde. Stencil-Objekte können kombiniert werden, um komplexere Stencil zu bilden, die aus einem Baum von Scheme-Ausdrücken des Typs Stencil bestehen.

Die `stencil`-Eigenschaft, die einen Grob mit seinem Stencil verbindet, ist in der `grob-interface`-Schnittstelle definiert.

Siehe auch

Referenz der Interna: Abschnitt “grob-interface” in *Referenz der Interna*.

A.16 Erhältliche Musikfunktionen

`\absolute` *music* (music) ⇒ music
: Make *music* absolute.

  This does not actually change the music itself but rather hides it from surrounding `\relative` (Seite 781) and `\fixed` (Seite 772) commands.

`\acciaccatura` *music* (music) ⇒ music
: Create an acciaccatura from *music*.

`\accidentalStyle` *style* (symbol list) ⇒ music
: Set accidental style to *style*.

  style is a (predefined) symbol list like `piano-cautionary`; siehe Abschnitt 1.3.5 [Automatische Versetzungszeichen], Seite 26, for the available styles. If it is preceded by a context name, the settings are applied to that context (example: `Staff .piano-cautionary`). Otherwise, the context defaults to `Staff`, except for piano styles, which use `GrandStaff` as a context.

`\addChordShape` *key-symbol* (symbol) *tuning* (pair) *shape-definition* (string or pair) ⇒ void
: Add *shape-definition* as a chord shape.

  It gets added to the `chord-shape-table` hash with the key `(cons` *`key-symbol tuning`*`)`.

`\addInstrumentDefinition` *name* (string) *lst* (list) ⇒ void
: Create instrument *name* with properties *lst*.

  This function is deprecated.

`\addQuote` *name* (string) *music* (music) ⇒ void
: Define *music* as a quotable music expression named *name*.

`\addToTagGroup` *tag-group* (symbol list) *more-tags* (symbol list) ⇒ void
: Add *more-tags* to the tag group identified by *tag-group*.

`\after` *delta* (duration-or-music) *ev* (music) *mus* (music) ⇒ music
: Add music *ev* with a delay of *delta* after the onset of *mus*.

  ev is usually a post-event.

`\afterGrace` [*fraction* (non-negative rational, fraction, or moment)] *main* (music) *grace* (music) ⇒ music
: Create *grace* as grace notes after a *main* music expression.

  The musical position of the grace expression is after a given fraction of the main note's duration has passed. If optional argument *fraction* is omitted, the fraction value is taken from `afterGraceFraction`, defaulting to 3/4.

`\allowPageTurn` ⇒ music
: Allow a page turn.

  May be used at top level (i.e., between scores or markups), or inside a score.

`\allowVoltaHook` *bar* (string) ⇒ void
: Allow the volta bracket hook being drawn over bar line *bar*.

`\alterBroken` *property* (key list or symbol) *arg* (list) *target* (key list or music) ⇒ music
: Override *property* for pieces of broken spanner *target* with *arg*.

  target may either be music in the form of a starting spanner event, or a symbol list of the form *`Context.Grob`* or just *`Grob`*. If *target* is in the form of a spanner event, *property* may also have the form *`Grob.property`* for specifying a directed tweak.

  arg is a list of values, one for each broken piece.

`\ambitusAfter` *target* (symbol) ⇒ music
: Move the ambitus after the break-align symbol *target*.

`\appendToTag` *tag* (symbol) *more* (music) *music* (music) ⇒ music
: Append *more* to *music* tagged with *tag*.

  A post-event can be added to the articulations of rhythmic events or chords; other expressions may be added to chords, sequential or simultaneous music.

`\appendToTagMarkup` *tag* (symbol) *more* (markup) *music* (music) ⇒ music
: Append *more* to every markup in *music* tagged with *tag*.

`\applyContext` *proc* (procedure) ⇒ music
: Modify context properties with Scheme procedure *proc*.

`\applyMusic` *func* (procedure) *music* (music) ⇒ music
: Apply procedure *func* to *music*.

`\applyOutput` *target* (symbol list or symbol) *proc* (procedure) ⇒ music
: Apply function `proc` to every layout object matched by *target*.

  target takes the form *`Context`* or *`Context.Grob`*.

`\appoggiatura` *music* (music) ⇒ music
: Create an appoggiatura from *music*.

`\approximatePitch` *note* (music) ⇒ music
: Indicate that the pitch of *note* approximates a pitch that cannot be known exactly, such as the highest note a singer can sing.

`\arpeggioArrowDown` ⇒ music
: Let `\arpeggio` produce a wavy line with a down arrow.

`\arpeggioArrowUp` ⇒ music
: Let `\arpeggio` produce a wavy line with an up arrow.

`\arpeggioBracket` ⇒ music
: Let `\arpeggio` produce a square bracket.

  Note: For a bracket designating a non-arpeggiated chord, it is better to use `\nonArpeggiato` than to use `\arpeggio` and alter the appearance.

`\arpeggioNormal` ⇒ music
: Let `\arpeggio` produce a wavy line without an arrow.

`\arpeggioParenthesis` ⇒ music
: Let `\arpeggio` produce a vertical slur.

  Note: For a vertical slur designating a quasi-non-arpeggiated chord, it is better to use `\chordSlur` than to use `\arpeggio` and alter the appearance.

`\arpeggioParenthesisDashed` ⇒ music
: Let `\arpeggio` produce a dashed vertical slur.

  Note: For a vertical slur designating a quasi-non-arpeggiated chord, it is better to use `\chordSlur` than to use `\arpeggio` and alter the appearance.

`\assertBeamQuant` *l* (pair) *r* (pair) ⇒ music
: Testing function: check whether the beam quants *l* and *r* are correct.

`\assertBeamSlope` *comp* (procedure) ⇒ music
: Testing function: check whether the slope of the beam is the same as *comp*.

`\atLeft` *m* (music) ⇒ post-event
: Set `side-axis` to `X` and `direction` to `LEFT` for *mus*.

`\atRight` *m* (music) ⇒ post-event
: Set `side-axis` to `X` and `direction` to `RIGHT` for *mus*.

`\augmentum` *expr* (music) ⇒ music
: Add augmentum dots (*morae*) to Gregorian chant *expr*.

`\autoChange` [*pitch* (pitch)] [*clef-1* (context modification)] [*clef-2* (context modification)] *music* (music) ⇒ music
: Make voices for *music* that switch between staves automatically.

  The optional argument *pitch* specifies where to switch staves; the default is `c'`. Optional arguments *clef-1* and *clef-2* specify which clefs to use; this only works for implicitly instantiated staves.

  Example:

  ```
  \autoChange d' \with { \clef alto } { g4 d' g' }
  ```

`\balloonGrobText` *grob-name* (symbol) *offset* (pair of numbers) *text* (markup) ⇒ music
: Attach *text* to *grob-name* at offset *offset*.

  Use like `\once` (Seite 776).

`\balloonText` *offset* (pair of numbers) *text* (markup) ⇒ post-event
: Attach *text* at *offset*.

  Use like `\tweak` (Seite 785).

`\bar` *type* (string) ⇒ music
: Insert a bar line of type *type*, overriding any automatic bar lines.

`\barNumberCheck` *n* (integer) ⇒ music
: Print a warning if the current bar number is not *n*.

`\beamExceptions` *music* (music) ⇒ any type
: Set beam exceptions.

This function extracts a value suitable for setting `Timing.beamExceptions` from the given pattern with explicit beams in *music*. A bar check '`|`' has to be used between bars of patterns in order to reset the timing.

`\bendAfter` *delta* (real number) ⇒ post-event
Create a fall or doit of pitch interval *delta*.

`\bendHold` *mus* (music) ⇒ post-event
Set `BendSpanner.style` to `'hold` for *mus*.

`\bendStartLevel` *idx* (non-negative, exact integer) *mus* (music) ⇒ post-event
Set `Bendspanner.details.successive-level` to *idx* for *mus*.

`\bookOutputName` *newfilename* (string) ⇒ void
Direct output for the current book block to *newfilename*.

This is equivalent to setting `output-filename` in the current book's `\paper` block.

`\bookOutputSuffix` *newsuffix* (string) ⇒ void
Set the output file name suffix for the current book block to *newsuffix*.

This is equivalent to setting `output-suffix` in the current book's `\paper` block.

`\breakAlignInsert` [*positions* (vector)] *symbol-to-move* (symbol) *sorting-symbol* (symbol) *anchor-symbol* (symbol) ⇒ music
Re-order break-align symbols.

This function is for tasks like putting a clef change after the bar line, which would be accomplished by `\breakAlignInsert clef after staff-bar`.

sorting-symbol can be `before` or `after`. Depending on its value, the break-align symbol *symbol-to-move* gets moved directly before/after the symbol *anchor-symbol* in the `break-align-orders` vector of `Score.BreakAlignment`.

The available break-align symbols and their corresponding grobs are listed in Abschnitt "Grobs and their break-align symbols" in *Notationsreferenz*.

The optional argument *positions* determines whether the specified change should become visible at the beginning/the end/the middle of a line. Possible values are: `#all-visible` (default), `#begin-of-line-invisible`, `#center-invisible`, `#end-of-line-invisible`, `#begin-of-line-visible`, `#center-visible`, `#end-of-line-visible`, `#all-invisible`.

`\breathe` ⇒ music
Insert a breath mark.

`\cadenzaOff` ⇒ music
Resume advancing `Timing.measurePosition`.

This sets `Timing.timing` to `#t`.

`\cadenzaOn` ⇒ music
Stop advancing `Timing.measurePosition`.

This sets `Timing.timing` to `#f`.

Compare to `\senzaMisura` (Seite 782).

`\caesura` ⇒ music
Insert a caesura.

`\chordRepeats` [*event-types* (list)] *music* (music) ⇒ music
Extend '`q`' to also repeat articulation.

This function walks through *music*, putting the notes of the previous chord into repeat chords, as well as an optional list of *event-types* such as `#'(string-number-event)`.

`\clef` *type* (string) ⇒ music
Set the current clef to *type*.

`\codaMark` [*num* (non-negative, exact integer)] ⇒ music
Create a coda mark.

num may be 1 for the first mark, 2 for the second, etc., or it may be `\default` to use the next number in sequence automatically.

`\compressMMRests` *music* (music) ⇒ music
Convert empty bars to multi-measure rests in *music*.

`\contextPropertyCheck` *property-path* (symbol list or symbol) [*expected-value* (any type)] ⇒ music

Check that the context property identified by *property-path* is set to *expected-value* in that very context: being set in an enclosing context is insufficient. If *expected-value* is `\default`, check that the property is unset in that context.

If *property-path* does not name a context, the check occurs in the current context.

Print a warning if the requested context is not visible looking upward from the current context or if the state of the property in the requested context is unexpected.

Caveat: Like some other context-specific commands, a context given in *property-path* is ignored in certain cases, such as inside `\with`.

`\crossStaff` *notes* (music) ⇒ music
Create cross-staff stems for *notes*.

`\cueClef` *type* (string) ⇒ music
Set the current cue clef to *type*.

`\cueClefUnset` ⇒ music
Unset the current cue clef.

`\cueDuring` *what* (string) *dir* (direction) *main-music* (music) ⇒ music
Create a cue.

This function inserts the contents of quote *what* corresponding to *main-music*, in a `CueVoice` context called `cue` oriented by *dir*.

`\cueDuringWithClef` *what* (string) *dir* (direction) *clef* (string) *main-music* (music) ⇒ music
Create a cue with clef.

This function inserts the contents of quote *what* corresponding to *main-music*, in a `CueVoice` context called `cue` oriented by *dir* and using clef *clef*.

`\deadNote` *note* (music) ⇒ music
Print *note* with a cross-shaped note head.

`\defineBarLine` *bar* (string) *glyph-list* (list) ⇒ void
Define bar line settings for bar line *bar*.

The list *glyph-list* must have three entries, defining substitute glyphs for the end of a line, the beginning of a line, and a span bar, respectively. The substitute glyphs may be either strings or Booleans: `#t` calls for the same value as *bar* and `#f` calls for no glyph.

`\displayLilyMusic` [*port* (output port)] *music* (music) ⇒ music
Write LilyPond's input representation of *music*.

If *port* is omitted, the output defaults to the console (stdout).

`\displayMusic` [*port* (output port)] *music* (music) ⇒ music
Write the internal representation of *music*.

If *port* is omitted, the output defaults to the console (stdout).

\displayScheme [*port* (output port)] *expr* (any type) ⇒ any type
: Write the internal Scheme representation of *expr*.

  If *port* is omitted, the output defaults to the console (stdout).

\dropNote *num* (integer) *music* (music) ⇒ music
: ‚Drop' the *num*-th note in each chord of *music*.

  This function moves the affected notes down (usually by an octave) to be lower than the other notes of the chord. The position in a chord is counted downwards from the top.

  The opposite function is \raiseNote (Seite 780).

\enablePerStaffTiming ⇒ void
: Enable polymeter with unaligned measures.

  This function moves the timing management from Score to Staff-like contexts. This is done by removing the Timing_translator from Score, and adding it to all contexts having the Staff alias.

  Use this within an output definition.

\endSpanners *music* (music) ⇒ music
: Terminate spanners.

  This function prematurely ends all spanners in *music* by inserting an end-spanner event at the end of the argument, without the need of specific end-spanner commands.

\eventChords *music* (music) ⇒ music
: Compatibility function: Handle isolated rhythmic events in *music*.

  Use this to wrap EventChord around isolated rhythmic events occuring since version 2.15.28, after expanding repeat chords 'q'.

  Not needed for new code.

\featherDurations *scale* (non-negative rational, fraction, or moment) *music* (music) ⇒ music
: Adjust feathered beam durations in *music* by *scale*.

\finger *finger* (index or markup) ⇒ post-event
: Apply *finger* as a fingering indication.

\fixed *pitch* (pitch) *music* (music) ⇒ music
: Use the octave of *pitch* as the default octave for *music*.

\footnote [*mark* (markup)] *offset* (pair of numbers) *footnote* (markup) *item* (symbol list or music) ⇒ music
: Make the markup *footnote* a footnote on *item*.

  The footnote is marked with a markup *mark* moved by *offset* with respect to the marked music.

  If *mark* is not given or specified as \default, it is replaced by an automatically generated sequence number. If *item* is a symbol list of form *Grob* or *Context.Grob*, then grobs of that type are marked at the current time step in the given context (default Bottom).

  If *item* is music, the music gets a footnote attached to a grob immediately attached to the event, like \tweak does. For attaching a footnote to an *indirectly* caused grob, write \single\footnote, use *item* to specify the grob, and follow it with the music to annotate.

  Like with \tweak, if you use a footnote on a following post-event, the \footnote command itself needs to be attached to the preceding note or rest as a post-event with '-'.

\grace *music* (music) ⇒ music
: Insert *music* as grace notes.

\grobdescriptions *descriptions* (list) ⇒ any type
: Create a context modification from *descriptions*.

  The argument is a list in the format of `all-grob-descriptions`.

\harmonicByFret *fret* (number) *music* (music) ⇒ music
: Convert *music* into mixed harmonics.

  The resulting notes resemble harmonics played on a fretted instrument by touching the strings at *fret*.

\harmonicByRatio *ratio* (number) *music* (music) ⇒ music
: Convert *music* into mixed harmonics.

  The resulting notes resemble harmonics played on a fretted instrument by touching the strings at the point given through *ratio*.

\harmonicNote *note* (music) ⇒ music
: Print *note* with a diamond-shaped note head.

\harmonicsOn ⇒ music
: Set the default note head style to a diamond-shaped style.

\hide *item* (symbol list or music) ⇒ music
: Make *item* invisible while still retaining its dimensions.

  If *item* is a symbol list of form *`GrobName`* or *`Context.GrobName`*, the result is an override for the grob name specified by it. If *item* is a music expression, the result is the same music expression with an appropriate tweak applied to it.

  This function sets *item*'s `transparent` property to `#t`.

\incipit [*offset* (number)] *incipit-music* (music) ⇒ music
: Output *incipit-music* as an incipit.

  incipit-music is typeset within a `MensuralStaff` context; the result is positioned before the main staff (as part of an `InstrumentName` grob) to indicate the music's original notation. The optional argument *offset* specifies how much the incipit should be moved to the left (default value is 0).

  In the special case that *incipit-music* has the form `\new` *xxx* ..., where *xxx* is a context type not accepted by `MensuralStaff`, it is taken directly.

\inherit-acceptability *to* (symbol) *from* (symbol) ⇒ void
: Make two contexts ‚compatible'.

  When used in an output definition, modify all context definitions such that context *to* is accepted as a child by all contexts that also accept *from*.

\initialContextFrom *music* (music) ⇒ music
: Enter the initial context of *music* and ignore the rest of it.

  This is useful for prepending music while preserving the influence of the original music on the context.

  Example:

```
{
  \initialContextFrom \originalMusic
  \prependedMusic
  \originalMusic
  \appendedMusic
}
```

`\inStaffSegno` ⇒ music
: Put the segno variant `varsegno` at this position into the staff.

  This is compatible with the repeat command.

`\instrumentSwitch` *name* (string) ⇒ music
: Switch instrument to *name*.

  name must have been predefined with function `\addInstrumentDefinition` (Seite 767).

  This function is deprecated.

`\inversion` *around* (pitch) *to* (pitch) *music* (music) ⇒ music
: Invert *music* about *around* and transpose from *around* to *to*.

`\invertChords` *num* (integer) *music* (music) ⇒ music
: Invert any chords in *music* into their *num*-th position.

  Chord inversions may be directed downwards using negative integers.

`\jump` *text* (markup) ⇒ music
: Use *text* to mark a point of departure, e.g., ‚Gavotte I D.C.‘.

`\keepWithTag` *tags* (symbol list or symbol) *music* (music) ⇒ music
: Keep tagged music.

  This function only includes elements of *music* that are tagged with one of the tags in *tags*. *tags* may be either a single symbol or a list of symbols.

  Each tag may be declared as a member of at most one tag group (defined with `\tagGroup`). If none of a *music* element's tags share a tag group with one of the specified *tags*, the element is retained.

`\key` [*tonic* (pitch)] [*pitch-alist* (list of number pairs)] ⇒ music
: Set key to *tonic* and scale *pitch-alist*.

  If both arguments are omitted (i.e., replaced by `\default`), just generate a `KeyChangeEvent`, which prints the current key signature again.

`\killCues` *music* (music) ⇒ music
: Remove cue notes from *music*.

`\label` *label* (symbol) ⇒ music
: Create *label* as a referrable label.

  The value stored in *label* is the page number, which can be extracted with the `\page-ref` markup command later on.

`\language` *language* (string) ⇒ void
: Set note names for language *language*.

  This function updates the Scheme variables `pitchnames` and `input-language`. The former is an alist that holds the pitches of all note names for the selected language; the latter holds the language name itself (as a symbol).

`\languageRestore` ⇒ void
: Restore the previously-saved `pitchnames` alist.

`\languageSaveAndChange` *language* (string) ⇒ void
: Save current `pitchnames` alist and change note names to *language*.

`\ligature` *music* (music) ⇒ music
: Make a ligature from Gregorian Chant *music*.

  This is equivalent to enclosing *music* with `\[` and `\]`.

\magnifyMusic *mag* (positive number) *music* (music) ⇒ music
: Magnify the size of *music* by factor *mag*.

  This doesn't change the staff size. Stems, beams, slurs, ties, and horizontal spacing are adjusted automatically.

\magnifyStaff *mag* (positive number) ⇒ music
: Change the staff size by factor *mag*.

  This adjusts notation size and horizontal spacing automatically.

\makeClusters *arg* (music) ⇒ music
: Display chords in *arg* as clusters.

\makeDefaultStringTuning *symbol* (symbol) *pitches* (list) ⇒ void
: Define string tuning *symbol* by a list of *pitches*.

  symbol also gets registered in `defaultStringTunings` for documentation purposes.

\mark [*label* (index or markup)] ⇒ music
: Create a rehearsal mark.

  If *label* is an integer, create the rehearsal mark for the given sequence number. If *label* is `\default`, create the next sequential rehearsal mark. If *label* is markup, use it for the mark.

\markupMap *path* (symbol list or symbol) *markupfun* (markup-function) *music* (music) ⇒ music
: Apply *markupfun* to property *path* in *music*.

  Argument *path* is either of the form *`property`* or *`MusicExpression.property`*. If *`MusicExpression`* is not given, *markupfun* gets applied to all properties called *`property`*, otherwise it is restricted to *`MusicExpression`* events. If *`property`* is not a markup, it is ignored.

  In the following example, both the tempo indication and the bowing instruction are printed in red. If you replace `text` with `TempoChangeEvent.text`, only the tempo indication changes the color.

```
\markupMap
  text
  \markup \with-color #red \etc
  { \tempo "Largo" g'2_"arco" c'' }
```

\measure *music* (music) ⇒ music
: Create a complete measure from *music*.

  This is equivalent to `\measureRemainder` *`music`*, but with a bar check at the beginning as well as the end.

\measureRemainder *music* (duration-or-music) ⇒ music
: End the current measure after *music*.

  The effect is equivalent to this:

```
{
  \initialContextFrom %{ copy of music %}
  \setMeasureLengthFromHere %{ duration of music %}
  music |
  \setDefaultMeasureLength
}
```

  Passing a duration as *music* (e.g., `\measureRemainder 2`) is reserved for future use.

  See `\setMeasureLengthFromHere` (Seite 782) for more information.

  Compare to `\premeasure` (Seite 779).

`\modalInversion` *around* (pitch) *to* (pitch) *scale* (music) *music* (music) ⇒ music
Invert *music* about *around* using *scale* and transpose from *around* to *to*.

`\modalTranspose` *from* (pitch) *to* (pitch) *scale* (music) *music* (music) ⇒ music
Transpose *music* from pitch *from* to pitch *to* using *scale*.

`\musicLength` *music* (music) ⇒ any type
Return the duration of *music* as a `ly:moment`.

Examples:

```
\musicLength 8 ⇒ #(ly:make-moment 1/8)
\musicLength {8. 8 8} ⇒ #(ly:make-moment 7/16)
```

`\musicMap` *proc* (procedure) *mus* (music) ⇒ music
Apply *proc* to *mus* and all of the music it contains.

`\noPageBreak` ⇒ music
Forbid a page break.

May be used at top level (i.e., between scores or markups), or inside a score.

`\noPageTurn` ⇒ music
Forbid a page turn.

May be used at top level (i.e., between scores or markups), or inside a score.

`\noStanza` ⇒ music
Forget the current stanza setting, thereby stopping any stanza reminders derived from it.

`\octaveCheck` *pitch* (pitch) ⇒ music
Do an octave check.

This prints a warning if the interval between the previous note and *pitch* is not within a fourth.

`\offset` *property* (symbol list or symbol) *offsets* (any type) *item* (key list or music) ⇒ music
Offset the default value of *property* of *item* by *offsets*.

If *item* is a string, the result is an override for the specified grob type. If *item* is a music expression, the result is the same music expression with an appropriate tweak applied to it.

`\omit` *item* (symbol list or music) ⇒ music
Omit *item* without taking up space.

If *item* is a symbol list of form *`GrobName`* or *`Context.GrobName`*, the result is an override for the grob name specified by it. If *item* is a music expression, the result is the same music expression with an appropriate tweak applied to it.

This function sets *item*'s `stencil` property to `#f`.

`\once` *music* (music) ⇒ music
Set property `once` to `#t` on all layout instruction events in *music*.

`\ottava` *octave* (integer) ⇒ music
Set the octavation to *octave*.

A positive value *n* indicates *n* octaves higher; a negative value *n* octaves lower, and value 0 means no octavation.

`\overrideProperty` *grob-property-path* (list of indices or symbols) *value* (any type) ⇒ music
Set the grob property specified by *grob-property-path* to *value*.

grob-property-path is a symbol list of the form *`Context.GrobName.property`* or *`GrobName.property`*, possibly with subproperties given as well.

As opposed to `\override`, which overrides the context-dependent defaults with which a grob is created, this command uses `Apply_output_engraver` at the grob acknowledge stage. This may be necessary for overriding values set after the initial grob creation.

`\overrideTimeSignatureSettings` *time-signature* (boolean-or-fraction) *beat-base* (positive exact rational, fraction (as pair), moment, or +inf.0) *structure* (a number list or a list of them) *beam-exceptions* (list) ⇒ music

Override time signature settings.

This function sets the `timeSignatureSettings` entry for *time-signature* to use *beat-base*, *structure*, and *beam-exceptions*.

beat-base and *beam-exceptions* are used as-is for `beatBase` and `beamExceptions`.

structure is used to derive values for `beatStructure` and `submeasureStructure`, as described for `\time` (Seite 784).

`\pageBreak` ⇒ music

Force a page break.

May be used at top-level (i.e., between scores or markups), or inside a score.

`\pageTurn` ⇒ music

Force a page turn.

May be used at top-level (i.e., between scores or markups), or inside a score.

`\palmMute` *note* (music) ⇒ music

Print *note* with a triangle-shaped note head.

`\palmMuteOn` ⇒ music

Set the default note head style to a triangle-shaped style.

`\parallelMusic` *voice-ids* (list) *music* (music) ⇒ void

Define parallel music sequences.

Within *music*, parallel music sequences are separated by '`|`' characters. The sequences are assigned to the LilyPond music identifiers provided in *voice-ids*.

For example, this code

```
\parallelMusic A,B,C {
  c c | d d | e e |
  d d | e e | f f |
}
```

is equivalent to

```
A = { c c | d d }
B = { d d | e e }
C = { e e | f f }
```

The last bar checks in a sequence are not copied to the result in order to facilitate ending the last entry at non-bar boundaries.

`\parenthesize` *arg* (symbol list or music) ⇒ music

Tag *arg* to be parenthesized.

arg may be either a music event or a grob path.

`\partCombine` [*chord-range* (pair of numbers)] *part1* (music) *part2* (music) ⇒ music

Combine two parts into a single staff.

This takes the music in *part1* and *part2* and returns a music expression containing simultaneous `Voice` contexts (called `one` for the upper and `two` for the lower voice).

Where appropriate, *part1* and *part2* are combined into a single voice (called `shared` or `solo`, depending on context).

Optional argument *chord-range* is a pair `(start . stop)` that defines the range in which the two voices are printed as chords (or unison); the default value is `(0 . 8)`, which means that intervals up to and including a ninth are unified.

`\partCombineDown` [*chord-range* (pair of numbers)] *part1* (music) *part2* (music) ⇒ music
Combine two parts into a single staff with all stems downwards.

For details, see `\partCombine` (Seite 777).

`\partCombineForce` [*type* (symbol)] ⇒ music
Override the part-combiner mode with *type*.

The following table gives the possible values for *type*, together with the corresponding shorthand functions.

`apart`	`\partCombineApart`
`chords`	`\partCombineChords`
`unisono`	`\partCombineUnisono`
`solo1`	`\partCombineSoloI`
`solo2`	`\partCombineSoloII`
`\default`	`\partCombineAutomatic`

`\partCombineUp` [*chord-range* (pair of numbers)] *part1* (music) *part2* (music) ⇒ music
Combine two parts into a single staff with all stems upwards.

For details, see `\partCombine` (Seite 777).

`\partial` *dur* (duration) ⇒ music
End the current measure after *dur*.

Adjust `Timing.measurePosition` to end the current measure at *dur* past the point of use. As a special case, when used at the start, create an anacrusis before the first measure.

Notational elements that communicate metric structure (e.g., beam breaks) will be engraved appropriately for the final *dur* part of a measure. Compare to `\setMeasureLengthFromHere` (Seite 782).

The calculation requires `Timing.measureLength` to be finite, so `\partial` cannot be used in a senza-misura passage.

See also `\premeasure` (Seite 779).

`\phrasingSlurDashPattern` *dash-fraction* (number) *dash-period* (number) ⇒ music
Set up a custom dash pattern style for phrasing slurs.

dash-fraction gives the size of one dash relative to *dash-period*; *dash-period* is the length of one dash plus one space. LilyPond adjusts *dash-period* to produce symmetrical output.

More complex patterns can be achieved by directly manipulating the `PhrasingSlur.dash-definition` property.

`\pitchedTrill` *main-note* (music) *secondary-note* (music) ⇒ music
Print a pitched trill.

main-note is the main note of the trill; *secondary-note* gets printed as a stemless note head in parentheses.

`\pointAndClickOff` ⇒ void
Suppress links to LilyPond source code in music output.

\pointAndClickOn ⇒ void
: Generate links to LilyPond source code in music output.

  This enables the creation of code in a PDF or SVG output file to reference the originating LilyPond source code (i.e., file name, line number, and column). This is helpful when developing a score; however, the output file becomes much larger.

\pointAndClickTypes *types* (symbol list or symbol) ⇒ void
: Generate point-and-click info for music of type *types* only.

  types is a single music expression (such as `#'note-event`) or a list of music expressions.

\polymetric [*time-signature-command* (music)] ⇒ music
: Apply *time-signature-command* to the local context rather than to `Timing`.

  Example:

```
\context Staff \polymetric \time 6/8
```

  The local measure must align with the reference measure defined in `Timing`. So that explicit changes to `Timing.measureLength` (e.g., for ad-hoc irregular measures) remain visible, this command does not set the `measureLength` property in the local context.

  A nominally incompatible local measure can be fitted to the controlling `Timing` measure by applying `\scaleDurations` (Seite 781) to this command. That adjusts the value of the internal property `meterScalingFactor` in the local context.

  To unset the local properties and resume using the values from the `Timing` context, use `\polymetric \default`.

\popContextProperty *path* (list of indices or symbols) ⇒ music
: Pop value of context property *path* from stack and set it.

  This is the opposite to function `\pushContextProperty` (Seite 780).

\preBend *mus* (music) ⇒ post-event
: Set `BendSpanner.style` to `'pre-bend` for *mus*.

\preBendHold *mus* (music) ⇒ post-event
: Set `BendSpanner.style` to `'pre-bend-hold` for *mus*.

\premeasure *music* (duration-or-music) ⇒ music
: Create a lead-in partial measure from *music*.

  The effect is equivalent to this:

```
{
  \initialContextFrom %{ copy of music %}
  \partial %{ duration of music %}
  music |
}
```

  Passing a duration as *music* (e.g., `\premeasure 2`) is reserved for future use.

  Compare to `\measureRemainder` (Seite 775).

\propertyOverride *grob-property-path* (list of indices or symbols) *value* (any type) ⇒ music
: Set the grob property specified by *grob-property-path* to *value*.

  grob-property-path is a symbol list of the form *`Context.GrobName.property`* or *`GrobName.property`*, possibly with subproperties given as well. This music function is mostly intended for use from Scheme as a substitute for the built-in `\override` command.

\propertyRevert *grob-property-path* (list of indices or symbols) ⇒ music
: Revert the grob property specified by *grob-property-path* to its previous value.

  grob-property-path is a symbol list of the form *Context.GrobName.property* or *GrobName.property*, possibly with subproperties given as well. This music function is mostly intended for use from Scheme as a substitute for the built-in \revert command.

\propertySet *property-path* (symbol list or symbol) *value* (any type) ⇒ music
: Set the context property specified by *property-path* to *value*.

  This music function is mostly intended for use from Scheme as a substitute for the built-in \set command.

\propertyTweak *prop* (key list or symbol) *value* (any type) *item* (key list or music) ⇒ music
: Add a tweak to *item*, usually music.

  This function sets the value of property *prop* to *value*; it generally behaves like \tweak but will turn into an \override when *item* is a symbol list. In that case, *item* specifies the grob path to override. This is mainly useful when using \propertyTweak as as a component for building other functions like \omit. It is not the default behavior for \tweak since many input strings in \lyricmode can serve equally as music or as symbols, which causes surprising behavior when tweaking lyrics using the less specific semantics of \propertyTweak.

  prop can contain additional elements in which case a nested property (inside of an alist) is tweaked.

\propertyUnset *property-path* (symbol list or symbol) ⇒ music
: Unset the context property specified by *property-path*.

  This music function is mostly intended for use from Scheme as a substitute for the built-in \unset command.

\pushContextProperty *path* (list of indices or symbols) ⇒ music
: Push the current value of context property *path* to a stack.

  The property can later be restored to the saved value with function \popContextProperty (Seite 779).

\pushToTag *tag* (symbol) *more* (music) *music* (music) ⇒ music
: Add *more* to the front of *music* tagged with *tag*.

  A post-event can be added to the articulations of rhythmic events or chords; other expressions may be added to chords, sequential or simultaneous music.

\pushToTagMarkup *tag* (symbol) *more* (markup) *music* (music) ⇒ music
: Prepend *more* to every markup in *music* tagged with *tag*.

\quoteDuring *what* (string) *main-music* (music) ⇒ music
: Indicate a section of music to be quoted.

  what indicates the name of the quoted voice specified with \addQuote (Seite 767). *main-music* is used to indicate the length of music to be quoted; it usually contains spacers or multi-measure rests.

\raiseNote *num* (integer) *music* (music) ⇒ music
: ‚Raise' the *num*-th note in each chord of *music*.

  This function moves the affected notes up (usually by an octave) to be higher than the other notes of the chord. The position in a chord is counted upwards from the bottom.

  The opposite function is \dropNote (Seite 772).

`\reduceChords` *music* (music) ⇒ music
: Reduce chords contained in *music* to single notes.

  This is intended mainly for reusing music in a `RhythmicStaff` context. It does not reduce simultaneous music.

`\relative` [*pitch* (pitch)] *music* (music) ⇒ music
: Make *music* relative to *pitch*.

  If *pitch* is omitted, the first note in *music* is given in absolute pitch.

`\removeFromTagGroup` *tag-group* (symbol list) *tags* (symbol list) ⇒ void
: Remove *tags* from the tag group identified by *tag-group*.

`\removeWithTag` *tags* (symbol list or symbol) *music* (music) ⇒ music
: Remove elements of *music* that are tagged with one of the tags in *tags*.

  tags may be either a single symbol or a list of symbols.

`\replaceWithTag` *tag* (symbol) *replacement* (music) *music* (music) ⇒ music
: Replace tagged elements in *music*.

  Everything tagged with *tag* (including the tagging itself) gets replaced with *replacement*.

`\replaceWithTagMarkup` *tag* (symbol) *replacement* (markup) *music* (music) ⇒ music
: Replace tagged markup in *music*.

  All markup tagged with *tag* (including the tagging itself) gets replaced with *replacement*.

`\resetRelativeOctave` *pitch* (pitch) ⇒ music
: Set the octave inside a `\relative` section to *pitch*.

`\resetTagGroup` *tag-group* (symbol list) ⇒ void
: Reset the tag group equal to *tag-group*.

`\resetTagGroups` ⇒ void
: Reset all tag groups previously created with `\tagGroup`.

`\responsum` *music* (music) ⇒ music
: Prepend character U+211F (RESPONSE) to the lyrics represented by *music*.

`\retrograde` *music* (music) ⇒ music
: Return *music* in reverse order.

`\revertTimeSignatureSettings` *time-signature* (pair) ⇒ music
: Revert `timeSignatureSettings` for time signatures equal to *time-signature*.

`\rightHandFinger` *finger* (index or markup) ⇒ post-event
: Apply *finger* as a right-hand fingering indication.

`\scaleDurations` *fraction* (non-negative rational, fraction, or moment) *music* (music) ⇒ music
: Multiply the duration of events in *music* by *fraction*.

`\sectionLabel` *text* (markup) ⇒ music
: Mark the beginning of a named passage with *text*, e.g., „Coda“.

  This is well suited for use at a section division created with `\section`, but it does not imply `\section` and may be used alone.

`\segnoMark` [*num* (non-negative, exact integer)] ⇒ music
: Create a segno mark (or bar line).

  num may be 1 for the first segno, 2 for the second, etc., or it may be `\default` to use the next number in sequence automatically.

If the segnoStyle context property is 'bar-line, a segno bar line is created instead of a segno mark.

\senzaMisura ⇒ music
Cancel the time signature; begin free time.

This sets Timing.timeSignature to #f.

This sets Timing.measureLength to +inf.0, allowing Timing.measurePosition to advance indefinitely without starting a new measure.

Beyond this point, ending the current measure requires setting Timing.measureLength back to a finite value *before* (not *at*) the transition to the next measure. See \measure (Seite 775) and related functions for assistance.

Compare to \cadenzaOn (Seite 770).

\setDefaultMeasureLength ⇒ music
Set Timing.measureLength from Timing.timeSignature.

\setMeasureLengthFromHere *dur* (duration) ⇒ music
End the current measure after *dur*.

Adjust Timing.measureLength to end at *dur* past the point of use. The resulting measure length remains in effect until it is changed by another command.

Notational elements that communicate metric structure (e.g., beam breaks) will be engraved appropriately to continue the measure. Compare to \partial (Seite 778).

See also \measure (Seite 775) and \measureRemainder (Seite 775).

\shape *offsets* (list) *item* (key list or music) ⇒ music
Offset control points of *item* by *offsets*.

offsets is a list of number pairs (x . y) or a list of such lists. Each pair represents an offset to a control point. The 'y' value of each pair is scaled by staff space.

If *item* is a string, the result is \once\override for the specified grob type. If *item* is a music expression, the result is the same music expression with an appropriate tweak applied.

\shiftDurations *dur* (integer) *dots* (integer) *arg* (music) ⇒ music
Change duration of *arg*.

This function walks over all durations and dot counts in *arg*, adding *dur* to the durations and *dots* to the dot counts.

\single *overrides* (music) *music* (music) ⇒ music
Convert *overrides* to tweaks and apply them to *music*.

This does not convert \revert, \set or \unset.

\skip *arg* (duration-or-music) ⇒ music
Skip over *arg*, which may be music or a duration.

\slashedGrace *music* (music) ⇒ music
Create slashed graces from *music*.

This produces slashes through stems, but no slur.

\slurDashPattern *dash-fraction* (number) *dash-period* (number) ⇒ music
Set up a custom dash pattern style for slurs.

dash-fraction gives the size of one dash relative to *dash-period*; *dash-period* is the length of one dash plus one space. LilyPond adjusts *dash-period* to produce symmetrical output.

More complex patterns can be achieved by directly manipulating the Slur.dash-definition property.

\staffHighlight *color* (color) ⇒ music
: Start a highlight with color *color*.

\stanza *text* (markup) ⇒ music
: Set and display the stanza number *text* in lyrics.

\startGradualTempoChange [*text* (markup)] ⇒ music
: Start an accelerando or ritardando.

  Begin a gradual departure from the tempo most recently set with \tempo. The gradual change extends to the next \stopGradualTempoChange or \tempo command. When \stopGradualTempoChange is used, it defines the target tempo; otherwise, \tempo defines it.

  The *text* argument is a placeholder. Only \default is accepted. It is intended eventually to serve the same purpose as the text argument to \tempo.

\stopGradualTempoChange [*text* (markup)] *tempo-unit* (duration) *metronome-count* (number) ⇒ music
: End an accelerando or ritardando.

  End a gradual tempo change at the tempo defined by *tempo-unit* and *metronome-count*. Unless there is a simultaneous \tempo command setting a new tempo, resume the tempo most recently set with \tempo.

  The *text* argument is a placeholder. Only \default is accepted. It is intended eventually to serve the same purpose as the text argument to \tempo.

\storePredefinedDiagram *fretboard-table* (hash table) *chord* (music) *tuning* (pair) *diagram-definition* (string or pair) ⇒ void
: Add a predefined fret diagram to *fretboard-table*.

  It is defined by *diagram-definition* for the chord pitches *chord* and the string tuning *tuning*.

\stringTuning *chord* (music) ⇒ any type
: Convert *chord* to a string tuning.

  chord must be in absolute pitches and should have the highest string number (generally the lowest pitch) first.

\styledNoteHeads *style* (symbol) *heads* (symbol list or symbol) *music* (music) ⇒ music
: Set *heads* in *music* to *style*.

\tabChordRepeats [*event-types* (list)] *music* (music) ⇒ music
: Extend ‘q’ to also repeat string and fingering information.

  This function walks through *music* putting the notes, fingerings and string numbers of the previous chord into repeat chords, as well as an optional list of *event-types* such as #'(articulation-event).

\tabChordRepetition ⇒ void
: Include the string and fingering information in a chord repetition.

  This function is deprecated; use \tabChordRepeats instead.

\tag *tags* (symbol list or symbol) *music* (music) ⇒ music
: Tag *music* with *tags*.

  This function adds the single symbol or symbol list *tags* to the tags property of *music* and returns the result.

\tagGroup *tags* (symbol list) ⇒ void
: Define a tag group comprising the symbols in the symbol list *tags*.

  Tag groups must not overlap.

\tagGroupRef *tags* (symbol list) ⇒ any type
: Define a tag group comprising the symbols in the symbol list *tags*.

  It returns the created tag group for further reference.

\temporary *music* (music) ⇒ music
: Make \override reversible with \revert.

  This function makes any \override in *music* replace an existing grob property value only temporarily, restoring the old value when a corresponding \revert is executed. This is achieved by clearing the pop-first property normally set on \overrides.

  An \override/\revert sequence created by using \temporary and \undo on the same music containing overrides will cancel out perfectly or cause a warning.

  Non-property-related music is ignored, warnings are generated for any property-changing music that isn't an \override.

\textEndMark *text* (markup) ⇒ music
: Create a right-aligned text mark using *text*.

\textMark *text* (markup) ⇒ music
: Create a (left-aligned) text mark using *text*.

\tieDashPattern *dash-fraction* (number) *dash-period* (number) ⇒ music
: Set up a custom dash pattern style for ties.

  dash-fraction gives the size of one dash relative to *dash-period*; *dash-period* is the length of one dash plus one space. LilyPond adjusts *dash-period* to produce symmetrical output.

  More complex patterns can be achieved by directly manipulating the Tie .dash-definition property.

\time [*structure* (a number list or a list of them)] *time-sig* (time signature) ⇒ music
: Set the time signature to *time-sig*.

  When *structure* is a plain list of numbers, it is used as-is for beatStructure, and submeasureStructure is left at the default.

  When *structure* is a list of lists, beatStructure is derived by discarding the grouping, and submeasureStructure is derived by summing each group. For example, a *structure* of '((1 2) (4 8)) yields a beatStructure of '(1 2 4 8) and a submeasureStructure of '(3 12).

  time-sig may be a fraction, e.g., 3/4.

  time-sig may also describe a complex time signature as a Scheme expression. Fractions are represented as pairs, (*numerator* . *denominator*), where the denominator is always a number. The numerator is one number or a list of two or more numbers. A list represents concatenation.

  For example, a time signature of (3+1)/8 + 2/4 can be created with \time #'(((3 1) . 8) (2 . 4))

\timeAbbrev *time-sig* (pair) ⇒ music
: Set the time signature to *time-sig*.

  This is like \time *time-sig*, except that it allows abbreviating fractions as lists. For example, a time signature of (3+1)/8 + 2/4 can be created with \timeAbbrev #'((3 1 8) (2 4)), and a time signature of (3+2)/8 with either \timeAbbrev #'((3 2 8)) or the shorter version \timeAbbrev 3,2,8.

  This is for backward compatibility. Using \time (Seite 784) instead is recommended.

\times *fraction* (fraction, as pair) *music* (music) ⇒ music
: Scale *music* in time by *fraction*.

`\tocItem` [*label* (symbol list or symbol)] *text* (markup) ⇒ music
: Add *text* as an entry to the table of contents.

  This uses the `tocItemMarkup` paper variable markup for formatting and assigns it to *label* if one is provided. If a hierarchy of labels is given, make the current item a child of the corresponding objects.

`\transpose` *from* (pitch) *to* (pitch) *music* (music) ⇒ music
: Transpose *music* from pitch *from* to pitch *to*.

`\transposedCueDuring` *what* (string) *dir* (direction) *pitch* (pitch) *main-music* (music) ⇒ music
: Create a transposed cue.

  This function inserts notes from the part *what* into a `CueVoice` context called `cue`, using the transposition defined by *pitch*. This happens simultaneously with *main-music*, which is usually a rest. The argument *dir* determines whether the cue notes should be notated as a first or second voice.

`\transposition` *pitch* (pitch) ⇒ music
: Set instrument transposition to *pitch*.

`\tuplet` *ratio* (fraction, as pair) [*tuplet-span* (duration)] *music* (music) ⇒ music
: Scale the given *music* to tuplets.

  ratio is a fraction that specifies how many notes are played in place of the nominal value: it will be 3/2 for triplets, namely three notes being played in place of two.

  If the optional duration *tuplet-span* is specified, it is used instead of `tupletSpannerDuration` for grouping the tuplets. For example,

  ```
  \tuplet 3/2 4 { c8 c c c c c }
  ```

  results in two groups of three tuplets, each group lasting for a quarter note.

`\tupletSpan` [*tuplet-span* (duration)] ⇒ music
: Set `tupletSpannerDuration` to the duration *tuplet-span*.

  This context property is the length into which `\tuplet` without an explicit tuplet span argument of its own will group its tuplets. To revert to the default of not subdividing the contents of a `\tuplet` command without an explicit tuplet span argument, use

  ```
  \tupletSpan \default
  ```

`\tweak` *prop* (key list or symbol) *value* (any type) *music* (music) ⇒ music
: Add a tweak to *music*.

  Layout objects created by *music* get their property *prop* set to *value*. If *prop* has the form *`Grob.property`*, like with

  ```
  \tweak Accidental.color #red cis'
  ```

  an indirectly created grob (`Accidental` is caused by `NoteHead`) can be tweaked; otherwise only directly created grobs are affected.

  prop can contain additional elements in which case a nested property (inside of an alist) is tweaked.

  If *music* is an `event-chord`, every contained `rhythmic-event` is tweaked instead.

`\undo` *music* (music) ⇒ music
: Convert `\override` and `\set` in *music* to `\revert` and `\unset`, respectively.

  Any reverts and unsets already in *music* cause a warning. Non-property-related music is ignored.

\unfolded *music* (music) ⇒ music
: Mask *music* until the innermost enclosing repeat is unfolded.

\unfoldRepeats [*types* (symbol list or symbol)] *music* (music) ⇒ music
: Unfold \repeat.

  This forces \repeat volta, \repeat tremolo or \repeat percent commands in *music* to be interpreted as \repeat unfold, if specified in the optional symbol-list *types*. The default for *types* is an empty list, which forces any of those commands in *music* to be interpreted as \repeat unfold. Possible entries are volta, tremolo or percent. Multiple entries are possible.

\verboseBarNumbers ⇒ music
: Print bar numbers at every bar line and line break.

  This function is a concise way to add detail to documentation examples. The output is subject to change without notice.

  See Abschnitt “Bar_number_engraver” in *Referenz der Interna*.

\versus *music* (music) ⇒ music
: Prepend character U+2123 (VERSICLE) to the lyrics represented by *music*.

\voices *ids* (list of indices or symbols) *music* (music) ⇒ music
: Specify voice order in simultaneous music.

  This takes the key list *ids* of numbers (indicating the use of ‘\voiceOne’...) or symbols (indicating voice names, typically converted from strings by argument list processing) and assign the following \\-separated music in *music* to contexts according to that list. Named rather than numbered contexts can be used for continuing one voice (for the sake of spanners and lyrics), usually requiring a \voiceOne-style override at the beginning of the passage and a \oneVoice override at its end.

  The default

  ```
  << ... \\ ... \\ ... >>
  ```

  construct would correspond to

  ```
  \voices 1,2,3 << ... \\ ... \\ ... >>
  ```

\void *arg* (any type) ⇒ void
: Accept a Scheme argument *arg* and return a void expression.

  Use this if you want to have a Scheme expression evaluated because of its side effects but its return value being ignored.

\volta *volta-numbers* (number list) *music* (music) ⇒ music
: Mark *music* as being limited to the volte given in *volta-numbers*.

  This gets used when the innermost enclosing repeat is unfolded. Volta number begins at 1 and increases by 1 with each repetition.

\vshape *offsets* (list) *item* (key list or music) ⇒ music
: Like \shape, but additionally show control points for ease of tweaking.

\withMusicProperty *sym* (symbol) *val* (any type) *music* (music) ⇒ music
: Set music property *sym* to *val* in *music*.

\withRelativeDir *file-name* (string) ⇒ any type
: Prepend directory of current input file to string *file-name*.

  Use this for markup commands that include files, and where such files should be found relative to the input file. Example:

  ```
  \markup { \image #X #3 \withRelativeDir "test.png" }
  ```
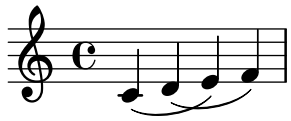
\xNote *note* (music) ⇒ music
: Print *note* with a cross-shaped note head.

\= *id* (index or symbol) *event* (post-event) ⇒ post-event
: Assign an ID to a spanner or an item.

  This sets the `spanner-id` or `id` property of *event* to the given *id*, which is a non-negative integer or a symbol.
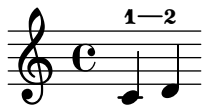
  For spanners this can be used to tell LilyPond how to connect overlapping or parallel slurs or phrasing slurs within a single `Voice` context.

```
\fixed c' { c\=1( d\=2( e\=1) f\=2) }
```



  For items this can be used, for example, to tell LilyPond how to connect a `FingerGlideSpanner` with non-matching fingers.

```
\fixed c' { c\glide \= #'foo -1 d\= #'foo -2 }
```



\% *count* (number) *music* (music) ⇒ music
: This is the same as `\repeat percent` *count music*.

  See `\repeat`.

* *count* (number) *music* (music) ⇒ music
: This is the same as `\repeat unfold` *count music*.

  See `\repeat`.

\~ *music* (music) ⇒ music
: Insert pes or flexa between previous and next note.

A.17 Bezeichner zur Kontextveränderung

Folgende Befehle sind definiert, umd Kontextveränderungen innerhalb von `\layout` oder `\with` vorzunehmen:

\EnableGregorianDivisiones
: Configure division commands such as `\section` to create `Divisio` grobs rather than `BarLine` grobs. This does not affect measure bar lines or the properties of the grobs themselves.

  - Sets translator property `caesuraTypeTransform` to `caesura-to-divisio`.
  - Sets translator property `doubleRepeatBarType` to `'()`.
  - Sets translator property `endRepeatBarType` to `'()`.
  - Sets translator property `fineBarType` to `""`.
  - Sets translator property `sectionBarType` to `""`.
  - Sets translator property `startRepeatBarType` to `'()`.
  - Sets translator property `underlyingRepeatBarType` to `""`.
  - Sets translator property `doubleRepeatSegnoBarType` to `"S-||"`.
  - Sets translator property `endRepeatSegnoBarType` to `"S-||"`.

- Sets translator property `fineSegnoBarType` to `"S-||"`.
- Sets translator property `fineStartRepeatSegnoBarType` to `"S-||"`.
- Sets translator property `segnoBarType` to `"S-||"`.
- Sets translator property `startRepeatSegnoBarType` to `"S-||"`.

`\RemoveAllEmptyStaves`

Remove staves which are considered to be empty according to the list of interfaces set by `keepAliveInterfaces`, including those in the first system.

- Sets grob property `remove-empty` in Abschnitt “VerticalAxisGroup” in *Referenz der Interna* to `#t`.
- Sets grob property `remove-first` in Abschnitt “VerticalAxisGroup” in *Referenz der Interna* to `#t`.

`\RemoveEmptyStaves`

Remove staves which are considered to be empty according to the list of interfaces set by `keepAliveInterfaces`.

- Sets grob property `remove-empty` in Abschnitt “VerticalAxisGroup” in *Referenz der Interna* to `#t`.

A.18 Vordefinierte Typprädikate

Predicates return `#t` (true) if their argument is of the named type and `#f` (false) if it isn't.

A.18.1 R5RS primary predicates

Primary predicates can be applied to any expression. They can be used on their own as predicates for LilyPond functions. The predicates here are part of the Scheme standard R5RS.

Type predicate	**Description**
`boolean?`	boolean
`char?`	character
`complex?`	complex number
`eof-object?`	end-of-file object
`input-port?`	input port
`integer?`	integer
`list?`	list (use `cheap-list?` for faster processing)
`null?`	null
`number?`	number
`output-port?`	output port
`pair?`	pair
`port?`	port
`procedure?`	procedure
`rational?`	rational number
`real?`	real number
`string?`	string
`symbol?`	symbol
`vector?`	vector

A.18.2 R5RS secondary predicates

Secondary predicates are only applicable to specific expressions (for example, to numbers). They will throw a type error when applied to expressions they are not intended for. The predicates here are part of the Scheme standard R5RS.

Type predicate	**Description**
`char-alphabetic?`	alphabetic character

char-lower-case?	lower-case character
char-numeric?	numeric character
char-upper-case?	upper-case character
char-whitespace?	whitespace character
even?	even number
exact?	exact number
inexact?	inexact number
negative?	negative number
odd?	odd number
positive?	positive number
zero?	zero

A.18.3 Guile predicates

These predicates are defined by Guile but are not part of a Scheme standard.

Type predicate	Description
hash-table?	hash table

A.18.4 LilyPond scheme predicates

These predicates are only available within LilyPond and defined in Scheme.

Type predicate	Description
alist?	association list (list of pairs)
boolean-or-symbol?	boolean or symbol
cheap-list?	list (use this instead of list? for faster processing)
color?	color
exact-rational?	an exact rational number
fraction?	fraction, as pair
grob-list?	list of grobs
grouped-number-list?	list of non-empty lists of numbers
index?	non-negative, exact integer
index-or-markup?	index or markup
key?	index or symbol
key-list?	list of indices or symbols
key-list-or-music?	key list or music
key-list-or-symbol?	key list or symbol
ly:skyline-pair?	pair of skylines
markup?	markup
markup-command-list?	markup command list
markup-list?	markup list
moment-pair?	pair of moment objects
musical-length?	non-negative exact rational, fraction (as pair), moment, or +inf.0
musical-length-as-moment?	non-negative moment with no grace part
musical-length-as-number?	non-negative exact rational or +inf.0
non-negative-number?	non-negative number
number-list?	number list
number-or-grob?	number or grob
number-or-number-pair?	number or pair of numbers
number-or-pair?	number or pair
number-or-string?	number or string
number-pair?	pair of numbers

number-pair-list?	list of number pairs
optionally-grouped-beat-structure?	a number list or a list of them
positive-exact-rational?	a positive, exact rational number
positive-fraction?	positive, finite fraction, as pair
positive-musical-length?	positive exact rational, fraction (as pair), moment, or +inf.0
positive-musical-length-as-moment?	positive moment with no grace part
positive-musical-length-as-number?	positive exact rational or +inf.0
positive-number?	positive number
rational-or-procedure?	an exact rational or procedure
rhythmic-location?	rhythmic location
sane-simple-time-signature?	simple time signature
sane-time-signature?	time signature
scale?	non-negative rational, fraction, or moment
scheme?	any type
string-or-music?	string or music
string-or-pair?	string or pair
string-or-symbol?	string or symbol
symbol-key-alist?	alist, with symbols as keys
symbol-list?	symbol list
symbol-list-or-music?	symbol list or music
symbol-list-or-symbol?	symbol list or symbol
time-signature?	time signature
void?	void
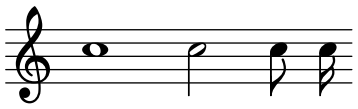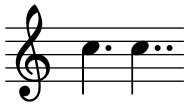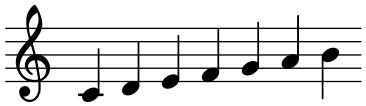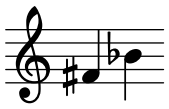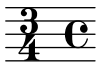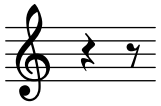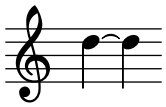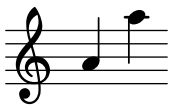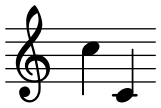
A.18.5 LilyPond exported predicates
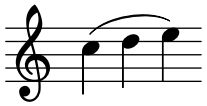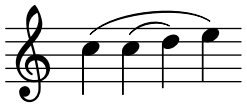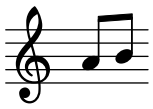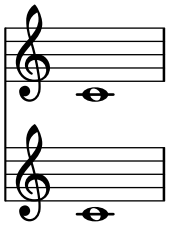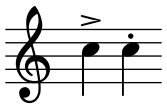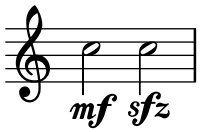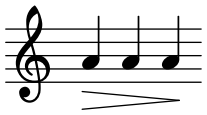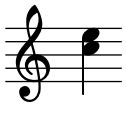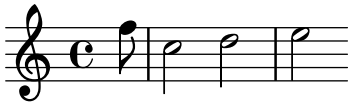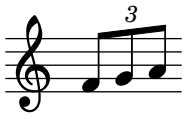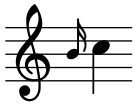
These predicates are only available within LilyPond and usually defined in C++.

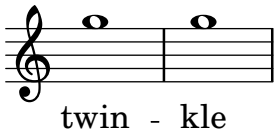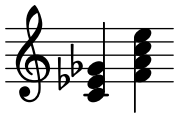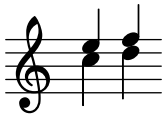
Type predicate	**Description**
ly:book?	book
ly:book-or-bookpart?	book or bookpart
ly:bookpart?	bookpart
ly:context?	context
ly:context-def?	context definition
ly:context-mod?	context modification
ly:dimension?	dimension, in staff space
ly:dir?	direction
ly:dispatcher?	dispatcher
ly:duration?	duration
ly:event?	post-event
ly:font-metric?	font metric
ly:grob?	graphical (layout) object
ly:grob-array?	array of grobs
ly:grob-properties?	grob properties
ly:input-location?	input location
ly:item?	item
ly:iterator?	iterator
ly:lily-lexer?	lily-lexer
ly:lily-parser?	lily-parser
ly:listener?	listener
ly:moment?	moment
ly:music?	music

`ly:music-function?`	music function
`ly:music-list?`	list of music objects
`ly:music-output?`	music output
`ly:note-scale?`	note scale
`ly:otf-font?`	OpenType font
`ly:output-def?`	output definition
`ly:page-marker?`	page marker
`ly:pango-font?`	Pango font
`ly:paper-book?`	paper book
`ly:paper-system?`	paper-system Prob
`ly:pitch?`	pitch
`ly:prob?`	property object
`ly:score?`	score
`ly:skyline?`	skyline
`ly:source-file?`	source file
`ly:spanner?`	spanner
`ly:spring?`	spring
`ly:stencil?`	stencil
`ly:stream-event?`	stream event
`ly:transform?`	coordinate transform
`ly:translator?`	translator
`ly:translator-group?`	translator group
`ly:unpure-pure-container?`	unpure/pure container

Anhang B Befehlsübersicht

Syntax	**Erklärung**	**Beispiel**
`1 2 8 16`	Tondauern	
`c4. c4..`	Punktierung	
`c d e f g a b`	Tonleiter	
`fis bes`	Vorzeichen	
`\clef treble \clef bass`	Notenschlüssel	
`\time 3/4 \time 4/4`	Taktangaben	
`r4 r8`	Pause	
`d ~ d`	Bindebogen	
`\key es \major`	Tonart	
note`'`	Oktavierung	
note`,`	Oktavierung nach unten	

`c( d e)`	Legatobogen	
`c\( c( d) e\)`	Phrasierungsbogen	
`a8[ b]`	Balken	
`<< \new Staff ... >>`	mehr Notensysteme	
`c-> c-.`	Artikulationszeichen	
`c2\mf c\sfz`	Dynamik	
`a\< a a\!`	Crescendo	
`a\> a a\!`	Decrescendo	
`< >`	Noten im Akkord	
`\partial 8`	Auftakt	
`\tuplet 3/2 {f g a}`	Triolen	
`\grace`	Verzierungen	

`\lyricmode { twinkle }`	Texteingabe	twinkle
`\new Lyrics`	Gesangstext	twinkle
`twin -- kle`	Gesangstext-Trennstrich	
`\chordmode { c:dim f:maj7 }`	Akkorde	
`\new ChordNames`	Akkordsymbole drucken	C° F△
`<<{e f} \\ {c d}>>`	Mehrstimmigkeit	
`s4 s8 s16`	unsichtbare Pausen	

Anhang C GNU Free Documentation License

Version 1.3, 3 November 2008

Copyright © 2000, 2001, 2002, 2007, 2008 Free Software Foundation, Inc.
https://fsf.org/

Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document *free* in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

   You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

   You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

   If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both

covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its

Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

   You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

   You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

   A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

   If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

   Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

   If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

   You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

   However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

   Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See `https://www.gnu.org/licenses/`.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (C)  year  your name.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.3
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover
Texts.  A copy of the license is included in the section entitled ``GNU
Free Documentation License''.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with. . . Texts." line with this:

```
with the Invariant Sections being list their titles, with
the Front-Cover Texts being list, and with the Back-Cover Texts
being list.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Anhang D Register

Zusätzlich zu allen LilyPond Befehlen und Schlüsselwörtern listet dieser Index alle relevanten Begriffe auf und verlinkt sie mit den entsprechenden Abschnitten, wo sie erklärt werden. Der erste Teil zeigt auf die genaue Stelle im Handbuch, an der der der Begriff vorkommt, der zweite Teil zeigt auf den gesamten Abschnitt, in dem das Thema behandelt wird.

!

! 8, 123

"

"|" 109

#

#f (false) 788
#t (true) 788

%

\% 787

’

' 3

(

\(..... 133

)

\) 133

*

* 787

,

, ... 3

—

- 120
-+ 761

.

. 45

/

/ 401
/+ 402

:

: 159

<

\< 123, 160
<...> 160

=

= 11, 787

>

\> 123, 160

?

? 8

[

[..... 94

]

] 94

^

^ 401

_

_ 261

|

| 109

~

~ 53
\~ 787

1

15ma 24

8

8va 24
8ve 24

A

a due-Stellen ... 175
Abbildungen im Text ... 241
\abs-fontsize ... 235, 678
\abs-hspace ... 691
\abs-vspace ... 692
Abschnitte definieren, Notenabstände ... 562
Abschnitte markieren ... 110
\absolute ... 767
absolute Lautstärke ... 123
Absolute Spezifikation von Oktaven ... 3
Absoluter Modus: Tonhöhen ... 3
Abstände, absolut ... 613
Abstände, skaliert ... 613
Abstände, vertikal ... 545
Abstand vergrößern, Gesangstext ... 270
Abstand von Hilfslinien ... 191
Abstand zwischen Notensystemen ... 545
Abstand zwischen Systemen in Klaviernoten ... 319
Abstrich ... 120, 760
Accelerando in MIDI ... 509
\accent ... 120, 760
Accentus ... 760
\accentus ... 762
\accepts ... 593, 594, 595
\acciaccatura ... 112, 767
\accidental ... 721
\accidentalStyle ... 26, 767
\addChordShape ... 359, 767
adding white background, to text ... 754
\addInstrumentDefinition ... 204, 212, 767
additionalPitchPrefix ... 406
Additionen in Akkorden ... 400
\addlyrics ... 255, 257, 258
\addQuote ... 205, 767
\addToTagGroup ... 496, 767
\aeolian ... 22
Aeolisch ... 22
\after ... 768
\afterGrace ... 113, 768
Aiken-Notenköpfe ... 39
\aikenHeads ... 39
\aikenHeadsMinor ... 40
Akkolade ... 185
Akkord, eine Noten verändern ... 605
Akkord, gebrochen ... 138
Akkord-Diagramme ... 355
Akkordbezeichnungen ... 397, 403
Akkordbezeichnungen und Bunddiagramme ... 356
Akkorddiagramm ... 346
Akkorddiagramme, automatisch ... 365
Akkorde ... 160, 397
Akkorde über zwei Systeme ... 319
Akkorde und Überbindungen ... 53
Akkorde und relativer Modus ... 6
Akkorde, Entfernen von Tönen ... 401
Akkorde, relative Tonhöhe ... 161
Akkorde, Unterdrückung wiederholt ... 404
Akkorde, Wiederholung ... 162, 332
Akkorde, zwischen Systemen mit \autoChange ... 319
Akkorde: farbige Noten ... 219
Akkorde: Fingersatz ... 215
Akkorde: Versetzungszeichen ... 32
Akkordeigenschaften ... 398
Akkordeon ... 322
Akkordeon, Diskant-Symbole ... 322
Akkordeon, Register ... 322
Akkordformen für bundierte Saiteninstrumente ... 359
Akkordformen für Bundinstrumente ... 359
Akkordglissando ... 334
Akkordmodi ... 398
Akkordmodus ... 397
Akkordstufen, Alteration ... 401
Akkordstufen, Veränderung ... 401
Akkordsymbole ... 403
Akkordsymbole in MIDI ... 509
Akkordsymbole, anpassen ... 405
Akkordtabulatur ... 346
Akzent ... 120, 760
Akzidentien ... 7, 26
al niente ... 126
\alias ... 593
\align-on-other ... 692
alignAboveContext ... 596
alignAboveContext ... 596
alignBelowContext ... 275, 596
alignBelowContext ... 275, 596
alist ... 764
\allowPageTurn ... 540, 768
\allowVoltaHook ... 768
Alte Schlüssel ... 19
\alterBroken ... 768
\alternative ... 144
alternative Endungen und Gesangstext ... 276
alternative Schlüsse in ausgeschriebenen Wiederholungen ... 154
Alternative Schlüsse mit Bindebogen ... 148
alternativer Gesangstext ... 280
alternativer Schluss ... 144
Altschlüssel ... 19
Ambitus ... 33
\ambitusAfter ... 768
Analyse ... 224
andere Stimmen zitieren ... 208
Ändern von Instrumentenbezeichnungen ... 203
Ändern von Schriftarten für das gesamte Dokument ... 249
Anfänger, Notenlernen ... 38
Anführungsstriche im Text ... 233
Anführungsstriche, Gesangstext ... 261
Anführungszeichen, Gesangstext ... 254
Angabe der Oktave: absolut ... 3
Anmerkung, Blase ... 221
annotate moving by spacing, in text ... 744
\annotate-moving ... 744
annotate-spacing ... 573
Anordnung, horizontal ... 561
Anpassen von Akkordsymbolen ... 405
Anpassen von Bunddiagrammen ... 352
Anpassen von staff symbol ... 613
Anpassungen im Gesangstextmodus ... 254
Anstrich ... 120
Anzahl der Notenlinien einstellen ... 191
Anzahl der Wiederholung, ändern ... 152
Äolisch ... 22
\append-to-tag ... 498, 745
\appendToTag ... 491, 768
\appendToTagMarkup ... 499, 768
\applyContext ... 768

\applyMusic ... 768
\applyOutput ... 768
\appoggiatura ... 112, 768
\approximatePitch ... 768
arabische Musik ... 449
arabische Musik, Beispiel ... 453
arabische Notenbezeichnungen ... 449
Arabische Taktarten ... 452
arabische Tonarten ... 450
arabische Vorzeichen ... 450
arabisches Halb-B Versetzungszeichen ... 450
Arpeggio ... 138
\arpeggio ... 138
Arpeggio über Systeme im Klammernstil ... 141
Arpeggio-Symbole, besondere ... 138
\arpeggioArrowDown ... 138, 768
\arpeggioArrowUp ... 138, 768
\arpeggioBracket ... 138, 769
\arpeggioNormal ... 138, 769
\arpeggioParenthesis ... 138, 769
\arpeggioParenthesisDashed ... 138, 769
\arrow-head ... 242, 710
Art der Übungszeichen ... 110
Arten von Notenköpfen ... 678
articulation-event ... 207
Artikulationszeichen ... 120
Artikulationszeichen, greg. Choral ... 434
Artikuliere-Skript ... 509
\ascendens ... 435, 440
\assertBeamQuant ... 769
\assertBeamSlope ... 769
associatedVoice ... 255
assoziative Liste ... 764
Atemzeichen ... 134
\atLeft ... 769
\atRight ... 769
\auctum ... 435, 440
Aufführungsanweisung: Tempo ... 69
Aufklappen von wiederholten Noten ... 154
Auflösungszeichen ... 7
Aufstrich ... 760
Auftakt ... 72
Auftakt in Wiederholung ... 146
Aufteilen von Noten ... 78
Aufteilen von Pausen ... 78
aug ... 399
\augmentum ... 440, 769
Ausdehnen von Noten ... 52
Ausdrück, Text ... 233
Ausgabe von Akkordbezeichnungen ... 403
ausgeschriebene Wiederholungen ... 154
Ausklingen lassen ... 54
Ausklingen lassen, Bögen ... 54
Auslassen von Noten im Gesangstext ... 276
Ausnahmen, Akkordsymbole ... 408
Ausrichten an Kadenz ... 118
Ausrichtung an Objekten ... 630
Ausrichtung von Gesangstext ... 257
Ausrichtung von Taktlinien ... 106
Ausrichtung von Text ... 237
Ausrichtung von Text, Befehle ... 241
Ausrichtung, Papier ... 522
Aussehen von Taktnummern ... 104
Auswahl von Schriftgröße (Notation) ... 214
auto-first-page-number ... 530
\auto-footnote ... 745
auto-knee-gap ... 82
autobeam ... 84
autoBeaming ... 84
\autoBeamOff ... 81, 319
\autoBeamOn ... 81
autoBeamSettings ... 91
\autoChange ... 317, 769
automaticBars ... 624
automatische Ausrichtung von Silben ... 257
automatische Bebalkung ... 81
automatische Bunddiagramme ... 365
automatische Kombination von Stimmen ... 175
Automatische Versetzungszeichen ... 26
Automatischer Systemwechsel ... 317
automatischer Systemwechsel und relativer Modus ... 318
automatisches Aufteilen von Noten ... 78

B

B ... 7
Bézier-Kurven ... 633
Bögen, über Noten ... 130
Bögen, gleichzeitig ... 130
Bögen, gleichzeitige Phrasierung ... 133
Bögen, laissez vibrer ... 54
Bögen, manuelle Platzierung ... 130
Bögen, mehrfach ... 130
Bögen, Phrasierung ... 130
Bögen, unter Noten ... 130
Bögen, verändern ... 633
backslashed digit ... 746
\backslashed-digit ... 746
Balken in Kadenzen ... 73
Balken in polymetrischer Notation ... 75
Balken mit Knie ... 82
Balken und Gesangstext ... 84
Balken und Melismen ... 82
Balken zwischen Systemen ... 315
Balken, automatisch ... 81
Balken, eigene Regeln ... 81
Balken, Einstellungen ... 81
Balken, gespreizt ... 96
Balken, letzter in Partitur ... 92
Balken, letzter in polyphoner Stimme ... 92
Balken, manuell ... 94
Balken, \partCombine und \autoBeamOff ... 83
Balken, Taktartstandard ... 66
Balken, Unterteilung ... 89
Balken, Zeilenumbrüche ... 82
Balkenpausen, mehrtaktig ... 62
Ballon ... 221
Balloon_engraver ... 221
\balloonGrobText ... 221, 769
balloonLengthOff ... 221
balloonLengthOn ... 221
\balloonText ... 221, 769
banjo-c-tuning ... 373
banjo-modal-tuning ... 373
banjo-open-d-tuning ... 373
banjo-open-dm-tuning ... 373
Banjo-Stimmung ... 373
Banjo-Tabulatur ... 327
Banjo-Tabulaturen ... 373

\bar ... 97, 103, 769
bar line, in markup ... 721
\bar-line ... 721
barCheckSynchronize ... 109
Baritonschlüssel ... 19
BarNumber ... 103
\barNumberCheck ... 109, 769
barNumberVisibility ... 103
Barré, anzeigen für bundierte Saiteninstrumente ... 370
Barré, anzeigen für Bundinstrumente ... 370
Barré, Gitarre ... 347
Bartók-Pizzicato ... 326
bartype ... 103
base-shortest-duration ... 561
Bassnote in Akkorden ... 401
Basso continuo ... 412
Bassschlüssel ... 19
\beam ... 710
beamExceptions ... 84, 769
beatBase ... 84
beatStructure ... 84
Bebalken in taktloser Musik ... 73
Bebalkung in Kadenzen ... 73
Bebalkung in polymetrischer Notation ... 75
Bebalkung nach Taktschlag ... 91
Bebalkung, automatisch, Einstellungen ... 84
Bebalkung, Taktartstandard ... 66
Beenden eines Notensystems ... 191
Beenden eines Systems ... 191
Beenden von Notenlinien ... 191
Befehle zur Textausrichtung ... 241
Beginn eines Notensystems ... 183
Beginn von Wiederholung ... 152
Beginnen eines Notensystems ... 191
Beginnen von Notenlinien ... 191
Beispiel der arabischen Musik ... 453
\bendAfter ... 136, 770
\bendHold ... 770
\bendStartLevel ... 770
Beschriftung ... 120
Beschriftung über mehrere Seiten ... 246
Beschriftung über Mehrtaktpausen ... 62
Beschriftung ausrichten ... 237
Beschriftung mit Bedingung ... 474
Beschriftung, Blocksatz ... 240
Beschriftung, mehrzeilig ... 239
Beschriftung, Notation einfügen ... 245
Beschriftung, Notationsobjekte einfügen ... 243
Beschriftung, Sonderzeichen ... 233
Beschriftung, Text ... 233
Beschriftung, Zentrieren auf der Seite ... 239
besondere Arpeggio-Symbole ... 138
besondere Notenköpfe ... 36
besondere Zeichen, Text ... 233
Bezifferter Bass ... 412
Bilder einbinden ... 243
Bindebögen und Akkorde ... 53
Bindebögen wiederholen ... 54
Bindebögen, Aussehen ... 54
Bindebögen, durchgehend ... 54
Bindebögen, gepunktet ... 54
Bindebögen, gestrichelt ... 54
Bindebögen, verändern ... 633
Bindebogen ... 53
Bindebogen in alternativem Schluss ... 148
Bindebogen in Wiederholung ... 148
Bindebogen und Wiederholung ... 151
Bindebogen, Gesangstext ... 261
Binderand ... 528
binding-offset ... 528
Bison ... 766
Blöcke, Text ... 239
blank-after-score-page-penalty ... 529
blank-last-page-penalty ... 529
blank-page-penalty ... 530
Blase ... 221
Blasinstrumente ... 384
Blocksatz, Text ... 240
BNF ... 766
Bogen zur Phrasierung ... 133
Bogen, Anzeige ... 325
Bogen, halb durchgehend, halb gestrichelt ... 133
Bogen, halb gestrichelt, halb durchgehend ... 131
Bogen, Strichelung definieren ... 131
\bold ... 234, 679
\book ... 458, 461
\bookOutputName ... 460, 770
\bookOutputSuffix ... 460, 770
\bookpart ... 459, 461, 538
bookTitleMarkup ... 471
bookTitleMarkup ... 471
bottom-margin ... 524
bounding box, of glyph ... 755
\box ... 241, 679
brace, in markup ... 748
\bracket ... 128, 241, 321, 710
Bratschenschlüssel ... 19
\break ... 538
break-align-symbols ... 630
break-visibility ... 620
break-visibility ... 620
breakable ... 82
\breakAlignInsert ... 770
breakbefore ... 468
\breathe ... 134, 770
\breve ... 44, 57
Brevis-Pause ... 57
Bund ... 331
Bunddiagramm-Beschriftung ... 347
Bunddiagramme ... 346, 355
Bunddiagramme und Akkordbezeichnungen ... 356
Bunddiagramme, anpassen ... 352
Bunddiagramme, ausführlicher Stil ... 350
Bunddiagramme, automatisch ... 365
Bunddiagramme, eigene ... 346
Bunddiagramme, eigene definieren ... 357
Bunddiagramme, Fingersatz ... 367
Bunddiagramme, knapper Stil ... 349
Bunddiagramme, Mandoline ... 355
Bunddiagramme, Transposition ... 356
Bunddiagramme, Ukulele ... 355
bundierte Saiteninstrumente, Akkordformen ... 359
bundierte Saiteninstrumente, Fingersatz der rechten Hand ... 368
bundierte Saiteninstrumente, Flageolett ... 370
bundierte Saiteninstrumente, gedämpfte Noten ... 370
bundierte Saiteninstrumente, Position und Barré anzeigen ... 370
bundierte Saiteninstrumente, Saitenstimmung ... 343
Bundinstrumente, Akkordformen ... 359

Bundinstrumente, Fingersatz der rechten Hand .. 368
Bundinstrumente, Flageolett 370
Bundinstrumente, gedämpfte Noten 370
Bundinstrumente, Position und Barré anzeigen ... 370
Bundinstrumente, Saitenstimmung 343
Bundsteg 528

C

C-Schlüssel 19
\cadenzaOff 73, 770
\cadenzaOn 73, 770
caesura 135
\caesura 433, 770
callback 764
Capo 350
\caps 680
\cavum 435, 440
\center-align 237, 692
\center-column 239, 693
centering column of text 693
\change 315
changing direction of text column 694
chants 301
\char 746
check-consistency 527
Chor-Tenorschlüssel 20
chord-Akkorde 397
chordChanges 404
\chordmode 7, 15, 356
chordNameExceptions 407
chordNameLowercaseMinor 406
ChordNames 356, 403
chordNameSeparator 407
chordNoteNamer 407
chordPrefixSpacer 408
\chordRepeats 332, 770
chordRootNamer 406
\chords 404
Chornoten 289
Chorsystem 185
Christian Harmony-Notenköpfe 39
\circle 241, 711
circling text 711
Circulus 760
\circulus 762
\clef 19, 771
closure 764
Cluster 165, 402
\cm 613
Coda 111, 120, 760
\coda 120, 723, 762
Coda am Taktstrich 229
\codaMark 771
color 218
coloring text 755
\column 239, 693
\column-lines 756
\combine 242, 693
common-shortest-duration 561
Completion_heads_engraver 78
Completion_rest_engraver 78
\compound-meter 723
\compressEmptyMeasures 61, 62
\compressMMRests 771
\concat 694
concatenating text 694
\consists 593
\context 582, 588
\context in \layout-Umgebung 588
\contextPropertyCheck 771
Continuo, Generalbass 412
controlling general text alignment 696
controlpitch 11
Copyright-Zeichen 502
\cr 123
creating a table 757
creating empty text object 749
creating horizontal space, in text 691, 698
creating text fraction 747
creating vertical space, in text 692, 707, 753
\cresc 124
Crescendo 123
crescendo-event 207
Crescendo-Klammer 123
Crescendoklammern, gedreht 625
\crescHairpin 124
\crescTextCresc 124
cross 36
\cross-staff 319
\crossStaff 771
cue notes, removing 212
\cueClef 208, 771
\cueClefUnset 771
\cueDuring 208, 771
\cueDuringWithClef 208, 771
CueVoice 208
currentBarNumber 103, 118
Custodes 423
\customTabClef 725
Custos 423

D

D'al Segno 120
D.S. al Fine 111
Dämpfung, bundierte Saiteninstrumente 370
Dämpfung, Bundinstrumente 370
Dal Segno 111
Dateien einfügen 488
Dateistruktur 461
Dauer 44
Dauer, Standard 45
Dauern skalieren 52
Daumenbezeichnung 120, 760
\deadNote 771
\decr 123
\decresc 124
Decrescendo 123
default 26
default-staff-staff-spacing 545
\defaultTimeSignature 65
\defineBarLine 771
Definieren von eigenen Bunddiagrammen 357
\deminutum 435, 440
\denies 593, 594, 595
\descendens 435, 440
Dicke der Notenlinien einstellen 191
didaktischer Versetzungszeichenstil 32

\dim ... 124, 399
dimension, of bounding box ... 755
\dimHairpin ... 124
Diminuendo ... 123
\dimTextDecr ... 124
\dimTextDecresc ... 124
\dimTextDim ... 124
\dir-column ... 694
\discant ... 740
Diskantsymbole, Akkordeon ... 322
\displayLilyMusic ... 516, 771
\displayMusic ... 771
\displayScheme ... 772
Divisi, Gesangstext ... 280
divisio ... 433
\divisioMaior ... 433
\divisioMaxima ... 433
\divisioMinima ... 433
divisiones ... 433
dodecaphonic ... 31
dodekaphoner Versetzungszeichenstil ... 31
doits ... 136
Doppel-B ... 7
Doppelkreuz ... 7
Doppellinie ... 97
Doppelpraller ... 760
Doppelpunktierung ... 45
Doppelschlag ... 120
doppelte Taktartensymbole ... 75
Doppelter Taktstrich ... 97
\dorian ... 22
Dorisch ... 22
\dotsDown ... 45
\dotsNeutral ... 45
\dotsUp ... 45
\doubleflat ... 725
\doublesharp ... 725
\downbow ... 120, 325, 761
\downmordent ... 120, 760
\downprall ... 120, 760
\draw-circle ... 242, 711
\draw-dashed-line ... 712
\draw-dotted-line ... 712
\draw-hline ... 712
\draw-line ... 242, 713
\draw-squiggle-line ... 713
drawing beam, within text ... 710
drawing box, with rounded corners ... 714
drawing box, with rounded corners, around text . 719
drawing circle, within text ... 711
drawing dashed line, within text ... 712
drawing dotted line, within text ... 712
drawing ellipse, around text ... 714
drawing line, across a page ... 712
drawing line, within text ... 713
drawing oval, around text ... 715
drawing path ... 716
drawing polygon ... 717
drawing solid box, within text ... 714
drawing squiggled line, within text ... 713
drawing triangle, within text ... 720
Drehen von Objekten ... 625
Dreiklänge ... 398
\dropNote ... 772
Drucken von Sonderzeichen ... 233
Druckreihenfolge ... 620
\drummode ... 183
Drums ... 375
DrumStaff ... 183
Dudelsack ... 387
Dur ... 22
durchgehender Legatobogen ... 130
durchgestrichener Hals ... 114
durchsichtig, Objekte ... 620
durchsichtige Noten ... 217
\dynamic ... 128, 680
dynamic-event ... 207
\dynamicDown ... 125
\dynamicNeutral ... 125
\dynamicUp ... 125
Dynamik ... 123
Dynamik, mehrere Zeichen an einer Note ... 124
Dynamik, vertikale Position ... 125
Dynamik, zentriert für Tasteninstrumente ... 315
Dynamikzeichen, Anmerkung ... 128
Dynamikzeichen, eigene ... 128
Dynamikzeichen, Klammer ... 128

E

\easyHeadsOff ... 38
\easyHeadsOn ... 38
Ebenen (layer) ... 620
editorische Dynamikzeichen ... 128
editorische Noten ... 219
eigene Balkenregeln ... 81
eigene Bunddiagramme ... 346, 352
Eigene Bunddiagramme definieren ... 357
eigene Dynamikzeichen ... 128
eigene Kontexte erstellen ... 582
Eigene Saitenstimmung, Tabulatur ... 344
eigene Tabulaturen ... 343
Eigenschaften ... 601
Eigenschaften von Grob ... 603
ein System, Mehrstimmigkeit ... 166
Einbinden von Graphik ... 243
Einfärben von Objekten ... 218, 620
Einfärben von Stimmen ... 169
einfügen von Dateien ... 488
Einfügen von Notationsobjekten ... 243
einfache closure ... 764
Eingabe von Noten parallel ... 179
Eingabedatei, Struktur ... 461
eingebundene Graphik im Text ... 241
Einmal verändern von Kontexten ... 605
Einstellung von Hilfslinien ... 191
einzelnes Notensystem ... 183
Einzug ... 203
\ellipse ... 714
Emmentaler font ... 658
\EnableGregorianDivisiones ... 787
\enablePerStaffTiming ... 772
encapsulated postscript output ... 505
enclosing text, in box ... 679
enclosing text, in box with rounded corners ... 719
Ende von Wiederholung ... 152
\endSpanners ... 772
Engraver, in Kontexte einfügen ... 593
Entfernen eines Stencil ... 619

Entfernen von Stichnoten ... 211
Entfernen von Stufen in Akkorden ... 401
Entfernen von Tönen aus Akkorden ... 400
EPS image ... 715
EPS-Ausgabe ... 505
\epsfile ... 243, 714
Erinnerungsvorzeichen ... 8
Erklärungsblase ... 221
erste Klammer ... 144
erweiterte Akkorde ... 400
Espressivo ... 120, 760
\espressivo ... 120, 124, 760
Espressivo-Artikulation ... 124
\eventChords ... 772
\expandEmptyMeasures ... 61, 62
explicitClefVisibility ... 622
explicitKeySignatureVisibility ... 622
extent, of actual inking ... 755
extent, of bounding box ... 755
extra-offset ... 545
\eyeglasses ... 746

F

\f ... 123
F-Schlüssel ... 19
Fähnchen, Mensuralnotation ... 427
Färben von Objekten ... 620
Färben von Stimmen ... 169
Fülllinie ... 265
Füllung um Text ... 242
falls ... 136
false (#f) ... 788
Farbe ... 218
Farbe, RGB ... 219
Farben, Liste ... 656
farbige Noten ... 218
farbige Noten in Akkorden ... 219
\featherDurations ... 96, 772
feature, OpenType font ... 685
\fermata ... 120, 725, 761
Fermate ... 111, 120, 760
Fermate über Mehrtaktpausen ... 62
Fermate an Taktstrich ... 229
Feta font ... 658
\ff ... 123
\fff ... 123
\ffff ... 123
\fffff ... 123
\figured-bass ... 680
Figuren, Namen ... 293
\fill-line ... 239, 695
\fill-with-pattern ... 695
\filled-box ... 242, 714
finalis ... 433
\finalis ... 433
\finger ... 215, 680, 772
fingeringOrientations ... 216
Fingersatz ... 215, 760
Fingersatz der rechten Hand, bundierte Saiteninstrumente ... 368
Fingersatz in Bunddiagrammen ... 367
Fingersatz und Mehrtaktpausen ... 64
Fingersatz versus Saitenzahl ... 328
Fingersatz: Akkorde ... 215
Fingersatz: Daumen-Zeichen ... 215
Fingerwechsel ... 215
first-page-number ... 530
\first-visible ... 746
\fixed ... 772
Flageolet ... 120, 760
\flageolet ... 120, 761
Flageolett ... 325
Flageolett in Tabulaturen ... 334
Flageolett, bundierte Saiteninstrumente ... 370
Flageolett, Bundinstrumente ... 370
Flageolett, künstliches ... 326
Flageolett-Notenköpfe ... 36
\flat ... 725
\flexa ... 440
Folgen einer Stimmen in anderes System ... 319
followVoice ... 319
font ... 764
font feature, OpenType ... 685
Font, Emmentaler ... 658
Font, Feta ... 658
Font, Größe ändern für Notation ... 214
Font, Parmesan ... 658
font-interface ... 215, 247
\font-select ... 681
font-size ... 214, 215
\fontCaps ... 681
\fontsize ... 235, 681
fontSize ... 214
\footnote ... 746, 772
forget ... 32
forget-Versetzungszeichenstil ... 32
Form-Notenköpfe ... 39
Formatierung von Übungszeichen ... 110
Formatierung von Gesangstext ... 254
Formatierung von Textstreckern ... 228
Formatierung von Triolen ... 47
four-string-banjo ... 373
\fp ... 123
\fraction ... 747
Fragmente, zitieren ... 205, 208
Französischer Violinschlüssel ... 19
\freeBass ... 740
\frenchChords ... 408
fret (Bunddiagramme) ... 347
Fret (Bunddiagramme) ... 346
\fret-diagram ... 347, 734
fret-diagram-interface ... 352
\fret-diagram-terse ... 349, 735
fret-diagram-terse-Markup ... 349
\fret-diagram-verbose ... 350, 736
fret-diagram-verbose-Markup ... 350
FretBoards ... 355
\fromproperty ... 747
Fußbezeichnung ... 760
Fußnoten, automatisch ... 476
Fußnoten, manuell ... 479
Funk-Formnotenköpfe ... 39
\funkHeads ... 39
\funkHeadsMinor ... 40

G

G-Schlüssel ... 19
Ganztaktpausen ... 57, 60
Ganztaktpausen und Fingersatz ... 64
Gebrochene Akkorde ... 138
Gedämpft ... 760
gedämpft ... 120
gedämpfte Noten, bundierte Saiteninstrumente ... 370
gedämpfte Noten, Bundinstrumente ... 370
gedrehte Crescendoklammern ... 625
Geisternoten ... 219
\general-align ... 238, 696
Generalbass ... 412
Generalbass Fortsetzungslinie ... 415
gepunkteter Legatobogen ... 130
gepunkteter Phrasierungsbogen ... 133
\germanChords ... 408
gerundeter Kasten, Graphik ... 241
Gesangsstimmen ... 289
Gesangstext ... 254
Gesangstext und übergebundene Noten ... 277
Gesangstext und Balken ... 84
Gesangstext und Verzierungsnoten ... 283
Gesangstext, alternativ ... 280
Gesangstext, an einer sporadischen Melodie ausrichten ... 584
Gesangstext, an Melodie ausgerichtet ... 255
Gesangstext, Auslassen von Noten ... 276
Gesangstext, Ausrichtung ... 257
Gesangstext, einer Stimme zugewiesen ... 166
Gesangstext, Formatierung ... 254
Gesangstext, innerhalb des Randes behalten ... 227
Gesangstext, Note überspringen ... 59
Gesangstext, Platz zwischen Silben ... 270
Gesangstext, Platzierung ... 267
Gesangstext, Variablen ... 266
Gesangstext, Wiederholungen ... 271
Gesangstext, Wiederholungen mit alternativen Endungen ... 276
geschweifte Klammer ... 185
geschweifte Klammern, Schachteln ... 188
gespreizte Balken ... 96
gestopft ... 120
gestrichelter Legatobogen ... 130
gestrichelter Phrasierungsbogen ... 133
geteilte Stimmen ... 291
getrennter Gesangstext ... 280
Gitarren-Akkordnotation ... 80
Gitarrengriffsymbole ... 346
Gitarrennotenköpfe ... 36
Gitarrenschlagrhythmus, Notation ... 80
Gitarrentabulatur ... 327
Gitterlinien ... 222
gleichzeitige Bögen ... 130
gleichzeitige Noten: Versetzungszeichen ... 32
gleichzeitige Phrasierungsbögen ... 133
Gleiten in Tabulaturen ... 334
Gleiten nach oben/unten ... 136
Glissando ... 136
\glissando ... 136
Glissando, Akkorde ... 334
Glissando, nach oben ... 136
Glissando, nach unten ... 136
Glissando, unbestimmt ... 136
glyph ... 764
glyph, bounding box ... 755
Glyphe ... 764
Größe der Schriftart ... 235
Größe von Notensystem verändern ... 194
Größe, Papier ... 521
Größe, Seite ... 521
\grace ... 112, 773
Grammatik von LilyPond ... 766
graphical objects ... 765
Graphik einbinden ... 243
Graphik, eingebunden ... 241
Graphische Notation ... 242
graphische Objekte ... 765
graphische Objekte, Eigenschaften ... 603
graphische Objekte, Schnittstellen ... 765
Gregorianische quadratische Neumenligaturen ... 435
Gregorianischer Choral, Transkription ... 183
GregorianTranscriptionStaff ... 183
Grid_line_span_engraver ... 222
Grid_point_engraver ... 222
gridInterval ... 222
Griff: Fingersatz ... 215
Griffsymbole, bundierte Saiteninstrumente ... 346
Griffsymbole, Bundinstrumente ... 346
Grob ... 598
grob ... 765
Grob-Eigenschaften ... 603
grob-interface ... 765
\grobdescriptions ... 773
Grobs, Sichtbarkeit ... 619
Grobs, verändern ... 620
grow-direction ... 96
Grundton eines Akkordes ... 400
Grundton eines Akkords ... 398

H

Hälse über zwei Systeme ... 319
Halb-B ... 7, 10
Halb-B-Versetzungszeichen, arabische Musik ... 450
halber Takt ... 72
Halbkreuz ... 7, 10
Halboffen ... 760
\halfopen ... 120, 761
\halign ... 237, 697
Hals ... 220
Hals nach oben ... 220
Hals nach unten ... 220
Hals neutral ... 220
Hals, mit Schrägstrich ... 114
Hals, Richtung ... 220
Hals, Richtung von ... 220
Hals, unsichtbar ... 220
Haltepedal, Stile ... 321
Harfe ... 322
Harfenpedal ... 323
Harmonia Sacra-Notenköpfe ... 39
\harmonic ... 325, 334
\harmonicByFret ... 334, 773
\harmonicByRatio ... 334, 773
\harmonicNote ... 773
\harmonicsOn ... 773
harmonische Obertöne (Flageolett) ... 325
\harp-pedal ... 737

\haydnturn ... 760
\hbracket ... 241, 715
\hcenter-in ... 698
\header ... 461
\heel ... 120, 761
\heelcircle ... 120, 761
\henzelongfermata ... 761
\henzeshortfermata ... 761
\hide ... 773
\hideKeySignature ... 387
\hideNotes ... 217
\hideSplitTiedTabNotes ... 333
\hideStaffSwitch ... 319
Hilfe, Blase ... 221
Hilfslinien, Abstände ... 191
Hilfslinien, Einstellungen ... 191
Hinzufügen von Tönen in Akkorden ... 400
hochgestellt ... 236
hochkant, Papier ... 522
horizontal-shift ... 528
Horizontal_bracket_engraver ... 224
horizontale Abstände ... 563
horizontale Anordnung ... 561
horizontale Ausrichtung von Text ... 237
horizontale Klammer ... 224
horizontale Notenabstände ... 563
horizontale Notenabstände, Abschnitte definierten ... 562
horizontale Platzierung ... 561
horizontale Platzierung, verändern ... 634
horizontally centering text ... 692
\hspace ... 698
Hufnagel ... 420, 421
\huge ... 214, 237, 682
hymns ... 301
hyperlink, as QR code ... 751

I

Ictus ... 760
\ictus ... 762
\if ... 734
Illustrationen im Text ... 241
image ... 715
\image ... 715
immutable-Eigenschaften ... 765
immutable-Objekte ... 765
importing stencil, into text ... 753
Improvisation ... 42
\improvisationOff ... 42, 80
\improvisationOn ... 42, 80
\in ... 613
in markierte Noten einfügen ... 491
\incipit ... 773
\inclinatum ... 435, 440
\include ... 488
include-settings ... 500
indent ... 203, 529, 565
\inherit-acceptability ... 773
\initialContextFrom ... 773
inner-margin ... 528
input-language ... 774
inserting music, into text ... 730
inserting PostScript directly, into text ... 718
inserting URL link, into text ... 721
\inStaffSegno ... 147, 774
Instrumentbezeichnungen ... 507
Instrumente, transponierende ... 13
Instrumentenbezeichnung, komplexe ... 202
Instrumentenbezeichnung, zentriert ... 202
Instrumentenbezeichnungen ... 201
Instrumentenbezeichnungen zu anderen Kontexten hinzufügen ... 203
Instrumentenbezeichnungen, wechseln ... 203
Instrumentengruppe ... 185
Instrumentenwechsel ... 204
\instrumentSwitch ... 204, 774
interface ... 765
Internals Reference ... 579
\inversion ... 15, 774
\invertChords ... 774
\ionian ... 22
Ionisch ... 22
\italianChords ... 408
\italic ... 234, 682

J

\jump ... 774
Justierung von Notensystemen ... 191
\justified-lines ... 246, 756
\justify ... 240, 699
\justify-field ... 699
\justify-line ... 700
\justify-string ... 701
justifying lines of text ... 756
justifying text ... 699

K

künstliches Flageolett ... 326
Kadenz ... 73
Kadenz und Seitenumbrüche ... 75
Kadenz und Seitenumbruch ... 73
Kadenz und Zeilenumbrüche ... 75
Kadenz und Zeilenumbruch ... 73
Kadenz, Ausrichten an ... 118
Kadenzen, Bebalkung ... 73
Kapo ... 350
Kasten, Graphik ... 241
\keep-with-tag ... 498, 748
\keepWithTag ... 491, 774
\key ... 22, 40, 774
KievanStaff ... 441
KievanVoice ... 441
\killCues ... 212, 774
Kirchenpausen ... 62
Kirchentonarten ... 22
Klammer, Crescendo ... 123
Klammer, erste (Wiederholung) ... 144
Klammer, geschweift ... 185
Klammer, vertikal ... 185
Klammer, Wiederholung ... 152
Klammer, Wiederholung mit Text ... 153
Klammer-Arpeggio über Systeme ... 141
Klammern ... 224
Klammern um Noten ... 219
Klammern um Vorzeichen ... 8

Klammern, Analyse ... 224
Klammern, Crescendo, schräg ... 625
Klammern, Graphik ... 241
Klammern, spitze ... 160
Klammern, unterschiedliche Größen ... 247
Klammern, Verschachteln ... 188
Klang ... 506
Klavier, Pedalbezeichnung ... 321
Klavier-Versetzungszeichenstil ... 30
Klavier: Warnungsversetzungszeichen ... 30
Klaviermusik, zentrierte Dynamik ... 315
Klaviersystem ... 185, 315
kleinere Noten ... 208
Knall-Pizzicato ... 326
Kollisionen, vertikal, vermeiden ... 559
Kombinieren von Stimmen ... 175
Komma-Intervalle ... 454
Komprimieren von Noten ... 52
Kontext, Layoutreihenfolge ... 595
Kontexte erstellen ... 582, 583
Kontexte, am Leben erhalten ... 583
Kontexte, einmal verändern ... 605
Kontexte, Lebensdauer ... 583
Kontexte, neue definieren ... 593
Kontexteigenschaften, Einstellungen ändern ... 588
Kontextveränderungen rückgängig machen ... 604
Kontroll-Tonhöhe ... 11
Kontrollpunkte und tweak ... 607
Kontrollpunkte, Bézier-Kurven ... 633
Krebs ... 16
Kreuz ... 7
Kreuznotenköpfe ... 36
kurze Instrumentenbezeichnungen ... 201

L

Länge von Zeilen ... 565
\label ... 485, 774
Laissez vibrer ... 54
\laissezVibrer ... 54
landscape ... 522
\language ... 774
\languageRestore ... 774
\languageSaveAndChange ... 774
\large ... 214, 237, 682
\larger ... 235, 237, 682
last-bottom-spacing ... 526
Lautstärke ... 123
layer (Ebenen) ... 620
\layout ... 461, 532, 588
layout file ... 534
layout objects ... 765
Layout, Partitur ... 532
Layout-Schnittstelle ... 598
Layoutobjekte ... 765
leere Systeme verstecken ... 198
Leerzeichen ... 463
Leerzeichen, Gesangstext ... 254, 261
\left-align ... 237, 701
left-aligning text ... 701
\left-brace ... 748
\left-column ... 702
left-margin ... 527
Legatobögen ... 130
Legatobögen, manuelle Platzierung ... 130
Legatobögen, verändern ... 633
Legatobogen zur Phrasierung ... 133
Legatobogen, gepunktet ... 130
Legatobogen, gestrichelt ... 130
Legatobogen, massiv ... 130
Legatobogen-Stil ... 130
lexer ... 765
\ligature ... 774
ligature, in text ... 694
Ligaturen ... 422
Ligaturen der quadratischen Neumennotation ... 435
Ligaturen, weiße Mensuralnotation ... 429
\line ... 702
line-width ... 527, 565
linea ... 435, 440
\lineprall ... 120, 760
Linien zwischen Systemen ... 222
Linien, Gitter ... 222
Liste der Farben ... 656
Liste der vorhandenen Schriftarten ... 249
\locrian ... 22
Lokrisch ... 22
\longa ... 44, 57
Longa-Pause ... 57
\longfermata ... 120, 761
\lookup ... 748
\lower ... 238, 702
lowering text ... 702
ly:minimal-breaking ... 540
ly:one-line-breaking ... 540
ly:optimal-breaking ... 539
ly:page-turn-breaking ... 539
\lydian ... 22
Lydisch ... 22
\lyricmode ... 254, 255
\lyricsto ... 258
\lyricsto ... 255, 257, 258

M

m ... 399
\magnify ... 235, 682
magnifying text ... 682
\magnifyMusic ... 775
\magnifyStaff ... 775
magstep ... 214, 613
maj ... 399
\major ... 22
major seven symbols ... 408
majorSevenSymbol ... 406
makam ... 454
makamlar ... 454
make-dynamic-script ... 129
make-dynamic-script ... 129
make-pango-font-tree ... 249
\makeClusters ... 165, 775
\makeDefaultStringTuning ... 775
manuelle Balken ... 94
manuelle Balken, Richtung zuweisen ... 94
manuelle Balken, Verzierungen ... 94
manuelle Systemwechsel ... 315
manuelle Taktstriche ... 98
manuelle Wiederholungszeichen ... 152

manuelles Übungszeichen ... 110
Maqam ... 449
Marcato ... 120, 760
\marcato ... 120, 760
\mark ... 110, 229, 775
\markalphabet ... 748
Marke ... 491
Marke, mehrere, in Gruppen ... 494
Markieren von Abschnitten ... 110
markierte Noten behalten ... 491
markierte Noten entfernen ... 491
\markletter ... 749
markup ... 233
\markup ... 229, 231, 232, 233
markup, image ... 715
markup, rhythm ... 729
markup, Syntax ... 233
markup-markup-spacing ... 526
markup-system-spacing ... 526
\markuplist ... 232, 246, 247
\markupMap ... 775
massiver Legatobogen ... 130
Matrize (stencil) ... 767
Matrize, entfernen ... 619
max-systems-per-page ... 529
\maxima ... 44, 57
Maxima-Pause ... 57
\measure ... 775
measureBarType ... 103
measureLength ... 84, 118
measurePosition ... 72, 118
\measureRemainder ... 775
Medicaea, Editio ... 420, 421
mehre Dynamikzeichen an einer Note ... 124
mehrere Phrasierungsbögen ... 133
mehrere Stimmen ... 170
mehrfache Bögen ... 130
mehrnotiger Vorschlag ... 116
mehrseitiger Text ... 246
Mehrstimmigkeit ... 166
Mehrstimmigkeit, ein System ... 166
Mehrtaktpause mit Fermate ... 62
Mehrtaktpausen ... 57, 60
Mehrtaktpausen und Fingersatz ... 64
Mehrtaktpausen, ausschreiben ... 61
Mehrtaktpausen, Beschriftung ... 62
Mehrtaktpausen, komprimieren ... 61
Mehrtaktpausen, Positionierung ... 63
Mehrtaktpausen, Text hinzufügen ... 62
mehrzeiliger Text ... 239
Melisma ... 262, 265
\melisma ... 262
\melismaEnd ... 262
Melismen, Balken ... 82
Melodierhythmus: Anzeige ... 79
Melodietransformation, Krebs ... 16
Melodietransformation, Umkehrung ... 15
Melodietransformationen, modal ... 16
Melodietransposition, modal ... 17
Melodieumkehrung, modal ... 17
Mensur ... 425
Mensuralligaturen ... 429
Mensuralmusik, Transkription ... 187
Mensuralnotation ... 420
MensuralStaff ... 424
MensuralStaff ... 183, 424
Mensuralstil ... 421
MensuralVoice ... 424
MensuralVoice ... 424
Mensurstriche ... 187
\mergeDifferentlyDottedOff ... 170
\mergeDifferentlyDottedOn ... 170
\mergeDifferentlyHeadedOff ... 170
\mergeDifferentlyHeadedOn ... 170
merging text ... 693, 703
Metronombezeichnung ... 69
Metrum ... 65
Metrum, Noten ohne ... 73, 118
Metrum, polymetrisch ... 75
Mezzosopranschlüssel ... 19
\mf ... 123
\midi ... 461
MIDI ... 25, 506
MIDI und Wiederholungen ... 510
MIDI, Akkordsymbole ... 509
MIDI, Artikulationen ... 509
MIDI, Artikuliere-Skript ... 509
MIDI, Mikrotöne ... 509
MIDI, Rhythmen ... 509
MIDI, Tonhöhen ... 509
MIDI, Vierteltöne ... 509
MIDI-Instrumentenbezeichnungen ... 655
MIDI-Kontextdefinitionen ... 508
MIDI-Transposition ... 25
MIDI-Umgebung ... 508
Mikrotöne ... 7, 10
Mikrotöne in MIDI ... 509
min-systems-per-page ... 529
minimum-Y-extent ... 545
minimumFret ... 331, 367
\minor ... 22
minorChordModifier ... 407
mirroring markup ... 720
mixed ... 321
\mixolydian ... 22
Mixolydisch ... 22
\mm ... 613
modale Transformationen ... 16
modale Transposition ... 17
modale Umkehrung ... 17
\modalInversion ... 17, 776
\modalTranspose ... 17, 776
modern ... 29
modern-cautionary ... 29
modern-voice ... 29
modern-voice-cautionary ... 30
modern-Warnung-Versetzungszeichenstil ... 28
moderne Versetzungszeichen ... 29
Moderner Stil, Versetzungszeichen ... 29
moderner Tabulatur-Schlüssel ... 346
moderner Versetzungszeichenstil ... 28, 29
moderner Versetzungszeichenstil mit Warnungen ... 29
moderner Versetzungszeichenstil mit Warnungen für Stimmen ... 30
moderntab ... 346
Modi, in Akkorden ... 398
Modifikatoren, Akkorde ... 398
Modus ... 22
Moll ... 22
Mordent ... 120, 760

\mordent 120, 760
\mp 123
multi-measure rest, within text, by duration 728
multi-measure rest, within text, by number of measures 726
\multi-measure-rest-by-number 726
MultiMeasureRestScript 62
MultiMeasureRestText 62
Musica ficta 429
\musicglyph 111, 726
\musicLength 776
\musicMap 776
Musik komprimieren 52
Musik ohne Metrum, Seitenumbrüche 75
Musik ohne Metrum, Umbrüche 73
Musik ohne Metrum, Zeilenumbrüche 75
Musikanalyse 224
Musikbuchstaben 111
Musikobjekte, Einfügen 243
musikwissenschaftliche Analyse 224
mutable-Objekte 765

N

N-tole, Formatierung 47
N-tolen 47
N.C.-Symbol 403
Nachschlag 113
\name 593
Name von Sänger 282
Namen von Figuren 293
\natural 726
neo-modern 30
neo-modern-cautionary 31
neo-modern-cautionary-Versetzungszeichenstil 31
neo-modern-voice 31
neo-modern-voice-cautionary 31
neo-moderner Versetzungszeichenstil 30
neo-moderner Versetzungszeichenstil pro Stimme 31
neo-moderner Versetzungszeichenstil pro Stimme mit Warnungen 31
Neomensuralstil 421
neue Dynamikzeichen 128
neue Kontexte 582
neues Notensystem 183
\new 582
nicht metrische Musik, Umbrüche 73
Nicht-ASCII-Zeichen 501
Nicht-Textschriftarten in Beschriftung 247
nichtmusikalische Symbole 242
niente, al 126
no-reset 32
\noBeam 94
\noBreak 538
\nonArpeggiato 291, 320
nonstaff-nonstaff-spacing 545
nonstaff-relatedstaff-spacing 545
nonstaff-unrelatedstaff-spacing 545
\noPageBreak 539, 776
\noPageTurn 540, 776
\normal-size-sub 683
\normal-size-super 236, 683
\normal-text 683
\normal-weight 683
normale Wiederholung 144
\normalsize 214, 237, 684
\noStanza 776
Notation für Streicher 324
Notation innerhalb von Beschriftung 245
Notation innerhalb von Text 245
Notation, Aiken 39
Notation, Erklärungen 221
Notation, graphische 242
Notationsobjekte, Einfügen 243
\note 727
note, within text, by duration 727
note, within text, by log and dot-count 727
\note-by-number 727
note-event 207
Note_heads_engraver 78
Noteknöpfe, einfache Notation 38
Noten ausdehnen 52
Noten in Klammern 219
Noten komprimieren 52
Noten ohne Metrum 118
Noten ohne Takt 73, 118
Noten verschmelzen 170
Noten verstecken 217
Noten wiederholt schreiben 154
Noten, Aufteilen 78
Noten, doppelpunktiert 45
Noten, durchsichtig 217
Noten, farbig 218
Noten, farbige in Akkorden 219
Noten, kleiner 208
Noten, parlato 36
Noten, punktiert 45
Noten, Schriftgröße 214
Noten, Stichnoten 208
Noten, transponieren 12
Noten, unsichtbar 217
Noten, Wechsel zwischen Systemen 315
Noten-Schriftzeichen 111
Notenabstände, Abschnitte definieren 562
Notenabstände, horizontal 563
Notenbezeichnungen, arabisch 449
Notenbezeichnungen, Deutsch 7
Notenbezeichnungen, Holländisch 7
Notenbezeichnungen, Standard 7
Notenbezeichungen, andere Sprachen 9
Notencluster 165
Notendauer, Standard 44, 45
Noteneingabe: relative Oktavbestimmung 4
Notengruppenklammer 224
Notenhälse über zwei Systeme 319
Notenhals, durchgestrichen 114
Notenhals, Richtung 220
Notenhals, Richtung von 220
Notenhals, unsichtbar 220
Notenköpfe 214
Notenköpfe für Anfänger 38
Notenköpfe zum Lernen 38
Notenköpfe, Übung 38
Notenköpfe, besondere 36
Notenköpfe, Christian Harmony 39
Notenköpfe, Flageolett 36
Notenköpfe, Formen 39
Notenköpfe, Funk 39
Notenköpfe, Gitarre 36

Notenköpfe, Harmonia Sacra ... 39
Notenköpfe, Improvisation ... 42
Notenköpfe, Kiever Notation ... 442
Notenköpfe, Kreuz ... 36
Notenköpfe, Mensuralnotation ... 426
Notenköpfe, Raute ... 36
Notenköpfe, rautenförmig ... 325
Notenköpfe, sacred harp ... 39
Notenköpfe, Walker ... 39
Notenkopfarten ... 678
Notenlänge ... 44
Notenlinien, Anzahl ... 191
Notenlinien, beenden ... 191
Notenlinien, beginnen ... 191
Notenlinien, Dicke ... 191
Notenlinien, Einstellungen ... 191
Notenlinien, erstellen ... 191
Notenschlüssel ... 19
Notensystem beginnen ... 191
Notensystem stoppen ... 191
Notensystem, anpassen ... 613
Notensystem, beenden ... 191
Notensystem, Größe verändern ... 194
Notensystem, Klavier ... 315
Notensystem, neu ... 183
Notensystem, Tasteninstrumente ... 315
Notensystemabstand ... 545
Notensysteme in Text einfügen ... 245
Notensysteme, gruppieren ... 185
Notensysteme, mehrere ... 185
Notensysteme, Modifikation ... 191
Notensystemgruppe ... 185
Notenzusammenstöße ... 170
\null ... 238, 749
\number ... 684
\numericTimeSignature ... 65
Nummerierung von Saiten ... 328
Nummerierung von Takten ... 103
Nummerierung, Strophen ... 281
nur Text ... 231

O

oberste Ebene, Text ... 231
Objekte verändern ... 620
Objekte, Drehen ... 625
Objekte, einfärben ... 620
Objekte, farbig ... 218
Objekte, Graphik im Text ... 241
Objekte, Sichtbarkeit ... 619
\octaveCheck ... 11, 776
offen ... 120
Offen ... 760
Offene Saite, anzeigen ... 325
\offset ... 776
Oktavbestimmung, relativ ... 4
Oktavenüberprüfung ... 11
Oktavenmodus (relativ) und Akkorde ... 6
oktavierte Schlüssel, Sichtbarkeit ... 624
Oktavierung ... 24
Oktavierungskorrektur ... 11
Oktavtransposition ... 20
Oktavwechsel: Tonhöhe ... 3
\omit ... 776
on-the-fly ... 474
\on-the-fly ... 474, 749
\once ... 603, 605, 776
\oneVoice ... 166
\open ... 120, 325, 761
OpenType, font feature ... 685
Optimieren ... 605
Oratorium ... 289
Orchester, Streicher ... 324
Orgelpedal-Bezeichnung ... 120
Orgelpedalbezeichnung ... 760
\oriscus ... 435, 440
Ornament ... 120
Ornamente ... 112
Osmanische Musik ... 454
ossia ... 199
Ossia ... 194
Ossia-Systeme ... 194
ottava ... 24
\ottava ... 24, 776
outer-margin ... 528
output-def ... 766
outside-staff-horizontal-padding ... 559
outside-staff-padding ... 559
outside-staff-priority ... 559
\oval ... 715
\overlay ... 703
\override ... 603, 749
Override im Gesangstextmodus ... 254
\override in \lyricmode ... 254
override rückgängig machen ... 604
\override, nur einmal ... 605
\override-lines ... 757
\overrideProperty ... 776
\overrideTimeSignatureSettings ... 66, 777
overriding property, within text markup ... 749
\overtie ... 686

P

\p ... 123
pädagogische Notenköpfe ... 38
\pad-around ... 242, 703
\pad-markup ... 242, 703
\pad-to-box ... 242, 704
\pad-x ... 242, 704
\pad-x-left ... 704
\pad-x-right ... 705
padding text ... 703
padding text horizontally ... 704, 705
page-breaking ... 530
page-breaking-system-system-spacing ... 530
page-count ... 530
\page-link ... 749
\page-ref ... 485, 750
page-spacing-weight ... 531
\pageBreak ... 539, 777
\pageTurn ... 540, 777
pageTurnMinimumRepeatLength ... 540
pageTurnMinimumRestLength ... 539
\palmMute ... 777
\palmMuteOn ... 777
Pango ... 247
\paper ... 461, 522
paper-height ... 523
paper-width ... 527

Papier, Ausrichtung ... 522
Papier, quer ... 522
Papierformat ... 522
Papiergröße ... 521
Parallele Notation, Eingabe ... 179
\parallelMusic ... 179, 777
\parenthesize ... 219, 716, 777
Parlato ... 299
Parlato-Notenköpfe ... 36
Parmesan font ... 658
parser ... 766
\partCombine ... 175, 777
\partCombineApart ... 176
\partCombineAutomatic ... 176
\partCombineChords ... 176
\partCombineDown ... 778
\partCombineForce ... 778
\partCombineSoloI ... 176
\partCombineSoloII ... 176
\partCombineUnisono ... 176
\partCombineUp ... 778
\partial ... 72, 144, 146, 778
partieller Takt ... 72
Partitur ... 185
Partitur, Layout ... 532
\path ... 716
path, drawing ... 716
\pattern ... 750
Pausen ... 57
Pausen verschieben, automatisch ... 170
Pausen, Aufteilen ... 78
Pausen, Ganztakt- ... 60
Pausen, ganztaktig ... 57
Pausen, Kirchenstil ... 62
Pausen, mehrere Takte ausschreiben ... 61
Pausen, mehrere Takte komprimieren ... 61
Pausen, Mehrtakt- ... 60
Pausen, mehrtaktig ... 57
Pausen, Mensuralnotation ... 428
Pausen, unsichtbar ... 59
Pausen, vertikale Position festlegen ... 57
Pausen, Zusammenfalten ... 65
Pausen, Zusammenstöße ... 65
Pausendauern ... 57
Pausenzeichen ... 134
Pedal, Harfe ... 323
Pedal, sostenuto ... 321
Pedal-Bezeichnung ... 120
Pedalbezeichnung ... 321
Pedalbezeichnung, Klammer ... 321
Pedalbezeichnung, Stile ... 321
Pedalbezeichnung, Text ... 321
Pedaldiagramme, Harfe ... 323
pedalSustainStyle ... 321
percent ... 156
Percussionsnotensystem ... 183
Perkussion ... 375, 377
Perkussionsnotensystem ... 183
\pes ... 440
Petrucci ... 420
Petrucci-Stil ... 421
Phrasierung, Gesang ... 262
Phrasierungsbögen ... 130, 133
Phrasierungsbögen, gepunktet ... 133
Phrasierungsbögen, gestrichelt ... 133
Phrasierungsbögen, gleichzeitig ... 133
Phrasierungsbögen, mehrfach ... 133
Phrasierungsbogen, halb durchgehend, halb gestrichelt ... 133
Phrasierungsbogen, Strichelmuster definieren ... 134
Phrasierungsklammern ... 224
Phrasierungszeichen ... 133
\phrasingSlurDashed ... 133
\phrasingSlurDashPattern ... 134, 778
\phrasingSlurDotted ... 133
\phrasingSlurDown ... 133
\phrasingSlurHalfDashed ... 133
\phrasingSlurHalfSolid ... 133
\phrasingSlurNeutral ... 133
\phrasingSlurSolid ... 133
\phrasingSlurUp ... 133
\phrygian ... 22
Phrygisch ... 22
piano ... 30
Piano, Pedalbezeichnung ... 321
piano-cautionary ... 30
Piano-System ... 315
Piano-Versetzungszeichenstil ... 30
PianoStaff ... 315, 317
Pitch_squash_engraver ... 80
\pitchedTrill ... 142, 778
pitchnames ... 774
Pizzicato, Bartók ... 326
Pizzicato, Knall- ... 326
place-fret ... 736
placing horizontal brackets, around text ... 715
placing parentheses, around text ... 716
placing vertical brackets, around text ... 710
Platz innerhalb von Systemgruppen ... 545
Platz um Text ... 242
Platz zwischen Notensystemen ... 545
Platzhalternoten ... 59
Platzierung von Gesangstext ... 267
Platzierung, Layouteinstellungen ... 573
PNG image ... 715
PNG-Ausgabe ... 505
\pointAndClickOff ... 778
\pointAndClickOn ... 779
\pointAndClickTypes ... 779
\polygon ... 717
\polymetric ... 779
Polymetrische Notation und Balken ... 75
polymetrische Partitur ... 587
polymetrische Taktarten ... 75
Polyphonie ... 166, 170
Polyphonie, ein System ... 166
\popContextProperty ... 779
Portato ... 120, 760
\portato ... 120, 760
Position und Barré für bundierte Saiteninstrumente ... 370
Position und Barré für Bundinstrumente ... 370
Position von Mehrtaktpausen ... 63
Positionierung, vertikal ... 545
\postscript ... 243, 718
Postscript, Graphik ... 243
Powerakkorde ... 372
Powerchords ... 372
\pp ... 123
\ppp ... 123

\pppp ... 123
\ppppp ... 123
\prall ... 120, 760
\pralldown ... 120, 760
Praller ... 120, 760
Prallermordent ... 760
\prallmordent ... 120, 760
\prallprall ... 120, 760
\prallup ... 120, 760
\preBend ... 779
\preBendHold ... 779
\predefinedFretboardsOff ... 366
\predefinedFretboardsOn ... 366
\premeasure ... 779
Prima volta ... 144
print-all-headers ... 531
print-first-page-number ... 530
print-page-number ... 530
\property-recursive ... 750
\propertyOverride ... 779
\propertyRevert ... 780
\propertySet ... 780
\propertyTweak ... 780
\propertyUnset ... 780
Prozent-Wiederholungen ... 156
psalms ... 301
\pt ... 613
Punktierung ... 45
\push-to-tag ... 498, 750
\pushContextProperty ... 780
\pushToTag ... 491, 780
\pushToTagMarkup ... 499, 780
\put-adjacent ... 705
putting space around text ... 703

Q

\qr-code ... 751
QR code ... 751
Quadratische Neumenligaturen ... 435
Quelldatei, Struktur ... 461
quer, Papier ... 522
\quilisma ... 435, 440
quotedCueEventTypes ... 207
quotedEventTypes ... 207
\quoteDuring ... 205, 208, 780

R

r ... 57
rückgängig machen von Kontextveränderungen ... 604
R ... 60
ragged-bottom ... 524
ragged-last ... 528, 565
ragged-last-bottom ... 524
ragged-right ... 527, 565
Rahmen, Text ... 241
railroad tracks ... 135
\raise ... 238, 705
\raiseNote ... 780
raising text ... 705
Rallantando in MIDI ... 509
Rand um Text ... 242
Rand, überhängender Text ... 227
Ratisbona, Editio ... 421
rautenförmige Notenköpfe ... 325
Rautennotenköpfe ... 36
rechte Hand, Fingersatz für bundierte Saiteninstrumente ... 368
rechte Hand, Fingersatz für Bundinstrumente ... 368
\reduceChords ... 781
referencing page label, in text ... 755
referencing page number, in text ... 749, 750
Referenz der Interna ... 579
regelmäßige Zeilenumbrüche ... 537
reine Container, Scheme ... 634
Relativ ... 4
\relative ... 4, 7, 15, 318, 781
Relative Oktavbestimmung ... 4
relative Tonhöhe, Akkorde ... 161
relativer Modus und Akkorde ... 6
relativer Modus und automatischer Systemwechsel ... 318
Relativer Oktavenmodus und Transposition ... 7
religious music ... 301
\remove-with-tag ... 498, 751
\RemoveAllEmptyStaves ... 788
\RemoveEmptyStaves ... 198, 199, 788
\removeFromTagGroup ... 496, 781
\removeWithTag ... 491, 781
removing cue notes ... 212
Renaissancemusik ... 187
\repeat ... 144
\repeat percent ... 156
\repeat tremolo ... 158
repeatCommands ... 152
\repeatTie ... 54, 148, 277
repetitive Musik ... 154
\replace ... 686
\replace-with-tag ... 498, 752
\replaceWithTag ... 497, 781
\replaceWithTagMarkup ... 499, 781
\resetRelativeOctave ... 781
\resetTagGroup ... 495, 781
\resetTagGroups ... 495, 781
\responsum ... 781
\rest ... 57, 728
rest, within text, by duration ... 728
rest, within text, by log and dot-count ... 729
\rest-by-number ... 729
rest-event ... 207
\retrograde ... 16, 781
\reverseturn ... 120, 760
\revert ... 604
\revertTimeSignatureSettings ... 67, 781
\rfz ... 123
rgb-color ... 219
RGB-Farbe ... 219
\rhythm ... 729
rhythm, in text ... 729
Rhythmen in MIDI ... 509
RhythmicStaff ... 183
Rhythmische Aufteilungen ... 47
rhythmisches Notensystem ... 183
Rhythmus der Melodie anzeigen ... 79
Richtung von Notenhälsen ... 220
\right-align ... 237, 705
right-aligning text ... 705
\right-brace ... 752

\right-column ... 706
right-margin ... 527
\rightHandFinger ... 368, 781
\rotate ... 706
rotating text ... 706
\rounded-box ... 241, 719

S

s ... 59
Sängername ... 282
Sätze, mehrere ... 458
Sackpfeife ... 387
sacred harp-Notenköpfe ... 39
\sacredHarpHeads ... 39
\sacredHarpHeadsMinor ... 40
Saite, offen ... 325
Saitenstimmung für Bundinstrumente ... 343
Saitenzahl ... 328
\sans ... 687
SATB ... 289
Satzzeichen ... 254
scalable vector graphics output ... 505
\scale ... 720
\scaleDurations ... 52, 75, 781
scaling markup ... 720
scaling text ... 707
Schachtelung von Systemen ... 188
Scheme objekt ... 767
Scheme, reine Container ... 634
Scheme, unreine Container ... 634
Schlüssel ... 7, 19
Schlüssel Alter Musik ... 19
Schlüssel, C ... 19
Schlüssel, F ... 19
Schlüssel, G ... 19
Schlüssel, greg. Choral ... 432
Schlüssel, Kiever Notation ... 442
Schlüssel, Mensuralnotation ... 424
Schlüssel, modern, Tabulatur ... 346
Schlüssel, Sichtbarkeit der Oktavierung ... 624
Schlüssel, Sichtbarkeit nach expliziter Änderung ... 622
Schlüssel, transponierend ... 20
Schlaggruppen ... 91
Schlagrhythmus, Gitarre ... 80
Schlagzeug ... 375, 377
schließende Taktstriche ... 97
Schluss, alternativer in Wiederholung ... 144
Schnittstelle von graphischen Objekten ... 765
Schnittstelle, Layout- ... 598
Schottischer Dudelsack ... 387
schräge Crescendoklammern ... 625
schräge Notenköpfe ... 42
Schriftart verändern ... 234
Schriftarten, für das gesamte Dokument ändern ... 249
Schriftarten, Hintergrundinformation ... 247
Schriftarten, Liste zum Auswählen ... 249
Schriftarten, Nicht-Text in Beschriftung ... 247
Schriftarten, vorhandene auflisten ... 249
Schriftartenfamilien, definieren ... 249
Schriftfamilie ... 764
Schriftfamilien ... 236
Schriftgröße ... 235
Schriftgröße (Notation) ändern ... 214
Schriftgröße (Notation), Standard ... 215
Schriftgröße, Einstellung ... 534
Schriftschnitt verändern ... 234
Schriftschnitte ... 236
Schriftzeichen, Notenschrift ... 111
\score ... 457, 461, 730
\score-lines ... 757
score-markup-spacing ... 526
score-system-spacing ... 526
scoreTitleMarkup ... 471
scoreTitleMarkup ... 471
Seconda volta ... 144
\sectionLabel ... 781
Segno ... 111, 120, 760
segno ... 99
\segno ... 120, 731, 762
Segno an Taktstrich ... 229
\segnoMark ... 781
Seitegröße ... 521
Seitenformat ... 522
Seitenrand, überhängender Text ... 227
Seitenumbrüche ... 538, 565
Seitenumbrüche in Kadenzen ... 73, 75
Seitenumbrüche in Musik ohne Metrum ... 75
Seitenumbrüche in nicht metrischer Musik ... 73
Seitenzahlen, automatische Nummerierung ... 530
Seitenzahlen, unterdrücken ... 530
self-alignment-X ... 545
Semai-Form ... 452
Semicirculus ... 760
\semicirculus ... 762
\semiflat ... 731
\semiGermanChords ... 408
\semisharp ... 731
\senzaMisura ... 782
separater Text ... 231
Septakkorde ... 398
\serif ... 687
sesqui-B ... 10
sesqui-Kreuz ... 10
\sesquiflat ... 731
\sesquisharp ... 731
\set ... 84, 601
set-octavation ... 24
\setDefaultMeasureLength ... 782
\setMeasureLengthFromHere ... 782
setting extent of text object ... 755
setting horizontal text alignment ... 697
setting subscript, in standard font size ... 683
setting superscript, in standard font size ... 683
Setzen von Sonderzeichen ... 233
Setzen von Text ... 233
\sf ... 123
\sff ... 123
\sfz ... 123
\shape ... 782
\sharp ... 732
\shiftDurations ... 782
\shiftOff ... 170
\shiftOn ... 170
\shiftOnn ... 170
\shiftOnnn ... 170
short-indent ... 203, 529
\shortfermata ... 120, 761
show-available-fonts ... 249

showFirstLength ... 504
\showKeySignature ... 387
showLastLength ... 504
\showStaffSwitch ... 319
Sichtbarkeit von Objekten ... 619
Sichtbarkeit von oktavierten Schlüsseln ... 624
Sietenzahlen, erste definieren ... 530
signum congruentiae ... 760
\signumcongruentiae ... 120, 762
Silben spreizen ... 270
\simple ... 687
simple text string, with tie characters ... 733
simultane Noten und Versetzungszeichen ... 32
\single ... 782
skalierbare Vektorgraphik-Ausgabe ... 505
Skalieren von Dauern ... 52
Skip ... 59
\skip ... 59, 276, 782
skipTypesetting ... 504
slashChordSeparator ... 407
slashed digit ... 752
\slashed-digit ... 752
\slashedGrace ... 112, 782
\slashturn ... 760
Slide in Tabulaturen ... 334
slur-event ... 207
\slurDashed ... 130
\slurDashPattern ... 131, 782
\slurDotted ... 130
\slurDown ... 130
\slurHalfDashed ... 131
\slurHalfSolid ... 131
\slurNeutral ... 130
\slurSolid ... 130
\slurUp ... 131
\small ... 214, 237, 687
\smallCaps ... 687
\smaller ... 235, 237, 688
smob ... 767
\snappizzicato ... 120, 761
Solesmes ... 421
solo-Stellen ... 175
Sonderzeichen ... 501
Sonderzeichen in Textbeschriftungen ... 233
Sopranschlüssel ... 19
Sopranschlüssel in C ... 19
sos. ... 321
sostenuto-Pedal ... 321
\sostenutoOff ... 321
\sostenutoOn ... 321
Southern-Harmony-Notenköpfe ... 39
\southernHarmonyHeads ... 39
\southernHarmonyHeadsMinor ... 40
\sp ... 123
spacing ... 561
Span_stem_engraver ... 319
spitze Klammern ... 160
\spp ... 123
Sprache, Tonhöhenbezeichnungn in anderer ... 9
Sprechgesang ... 299
Spreizen von Silben ... 270
Springen zwischen Systemen ... 315
Staccatissimo ... 120
\staccatissimo ... 120, 760
Staccato ... 120, 760
\staccato ... 120, 760
stacking text, in a column ... 693
staff brace, in markup ... 748
Staff symbol, erstellen ... 191
staff-affinity ... 545
staff-staff-spacing ... 545
Staff.midiInstrument ... 507
Staff_symbol_engraver ... 198
staffgroup-staff-spacing ... 545
\staffHighlight ... 783
Standard Notendauer ... 44
Standard-Schriftgröße (Notation) ... 215
Standard-Versetzungszeichenstil ... 26, 28
Standardkontexteigenschaften, ändern ... 588
Standardnotenbezeichnungen ... 7
Standardnotendauer ... 45
Standardtakteinstellungen ... 66
Standardtaktstrich, Änderung ... 103
\stanza ... 783
start-repeat ... 152
\startGradualTempoChange ... 783
\startGroup ... 224
\startStaff ... 191, 194
\startTrillSpan ... 141
\stdBass ... 740
\stdBassIV ... 741
\stdBassV ... 742
\stdBassVI ... 743
Stem ... 319
stem-spacing-correction ... 561
\stemDown ... 220
stemLeftBeamCount ... 95
\stemNeutral ... 220
Stempel (stencil), entfernen ... 619
stemRightBeamCount ... 95
\stemUp ... 220
stencil ... 767
\stencil ... 753
stencil, entfernen ... 619
Stichnoten ... 205, 208, 295
Stichnoten innerhalb von rhythmischer Kombination ... 51
Stichnoten, entfernen ... 211
Stichnoten, Formatierung ... 208
Stil von Übungszeichen ... 110
Stil von Legatobögen ... 130
Stil von Taktangaben ... 65
Stile, Notenköpfe ... 36, 678
Stile, Stimmen ... 169
Stimme ... 166
Stimme folgen ... 319
Stimme-Versetzungszeichenstil ... 28
Stimmen kombinieren ... 175
Stimmen verschieben ... 170
Stimmen, \autoBeamOff und \partCombine ... 83
Stimmen, farbige Unterscheidung ... 169
Stimmen, geteilt ... 291
Stimmen, mehrere ... 170
Stimmen, Stile ... 169
Stimmen, Versetzungszeichen für ... 29
Stimmen, Versetzungszeichenstil mit Warnung für Stimmen ... 30
Stimmen, zitieren ... 205
Stimmfolgestriche ... 319
Stimmgruppe ... 185

Stimmkreuzung ... 319
Stimmumfang ... 33
Stimmung, Banjo ... 373
\stopGradualTempoChange ... 783
\stopGroup ... 224
\stopped ... 120, 761
\stopStaff ... 191, 194, 198
\stopTrillSpan ... 141
\storePredefinedDiagram ... 359, 783
Strecker, Text ... 228
Strecker, Text-, Formatierung ... 228
Streicher ... 324
Streicher, Bogenanzeige ... 325
Striche zur Stimmverfolgung ... 319
Striche: Notenköpfe ... 42
Strichnotenköpfe ... 42
\string-lines ... 757
stringNumberOrientations ... 216
\stringTuning ... 344, 783
stringTunings ... 343, 355
strokeFingerOrientations ... 216
\stropha ... 435, 440
Strophennummer ... 281
Struktur, Datei ... 461
\strut ... 753
\styledNoteHeads ... 783
\sub ... 236, 688
Subbassschlüssel ... 19
subdivideBeams ... 89
subscript text ... 688
Subtraktion in Akkorden ... 400
suggestAccidentals ... 429
\super ... 236, 688
superscript text ... 688
sus ... 401
\sustainOff ... 321
\sustainOn ... 321
SVG-Ausgabe ... 505
swing ... 729
Symbole auf der Taktstrich ... 229
Symbole, Akkord- ... 403
Symbole, Akkordeon ... 322
Symbole, nicht musikalische ... 242
Synchronisation von Verzierungen ... 116
System querende Hälse ... 319
System, beenden ... 191
System, Chor ... 185
System, geschachtelt ... 188
System, Größe verändern ... 194
system-count ... 529
system-separator-markup ... 531
system-system-spacing ... 526
System-Trennzeichen ... 190
SystemBeginnBegrenzer, geschachtelt ... 188
Systeme verstecken ... 198
Systeme, leere ... 198
Systeme, mehrere ... 185
Systeme, Tremolo zwischen ... 159
Systeme, Zusammenstöße beim Stimmenwechsel . 316
Systemgröße, Einstellung ... 534
Systemgruppe ... 185
Systemgruppen, Abstände innerhalb ... 545
Systemgruppen, Verschachtelung ... 188
systems-per-page ... 529
Systemwechsel von Stimmen ... 319
Systemwechsel, automatisch ... 317
Systemwechsel, manuell ... 315

T

türkische Notenbezeichnungen ... 454
Türkische Musik ... 454
\tabChordRepeats ... 332, 783
\tabChordRepetition ... 783
\tabFullNotation ... 331
\table ... 757
\table-of-contents ... 487, 758
TabStaff ... 183, 330
Tabulatur ... 183, 327
Tabulatur und Flageolett ... 334
Tabulatur, Banjo ... 343, 373
Tabulatur, Bassgitarre ... 343
Tabulatur, Bratsche ... 343
Tabulatur, Cello ... 343
Tabulatur, eigene Saitenstimmung ... 344
Tabulatur, Geige ... 343
Tabulatur, Gitarre ... 343
Tabulatur, Grundlegendes ... 330
Tabulatur, Kontrabass ... 343
Tabulatur, Mandoline ... 343
Tabulatur, moderner Schlüssel ... 346
Tabulatur, Saitenstimmung ... 343
Tabulatur, Ukulele ... 343
Tabulaturen und Gleiten ... 334
Tabulaturen, eigen ... 343
Tabulatursystem ... 183
TabVoice ... 330
Tag ... 491
\tag ... 491, 753, 783
\tag-list ... 499, 758
\tagGroup ... 494, 783
\tagGroupRef ... 494, 784
Takt unterteilen ... 91
Takt, Noten ohne ... 118
Taktüberprüfung ... 109
Taktangabe ... 65
Taktangabe, Sichtbarkeit ... 65
Taktangaben-Stile ... 65
Taktart, Mensuralnotation ... 425
Taktart, Noten ohne ... 73
Taktart, Standardeigenschaften wiederherstellen .. 67
Taktart, Standardeinstellung ... 66
Taktarten, arabisch ... 452
Taktarten, mehrere in Partitur ... 587
Taktarten, polymetrisch ... 75
Taktarten, unterschiedliche per System ... 587
Taktarten, zusammengesetzt ... 77
Taktartensymbole, doppelt ... 75
Taktartensymbole, unterteilt ... 75
Takte verkürzen ... 72
Taktgruppen ... 91
Taktlinie, manuell ... 98
Taktlinie, Wiederholung ... 152
Taktlinien ... 97
Taktlinien, Ausrichtung ... 106
Taktlinien, unsichtbar ... 98
Taktlose Musik, Bebalkung ... 73
Taktnummer ... 119
Taktnummer, Form ... 104

Taktnummern ... 103
Taktnummern, Zusammenstöße ... 108
Taktposition und Wiederholung ... 151
Taktschläge gruppieren ... 91
Taktstrich, doppelt ... 97
Taktstrich, Symbole anfügen ... 229
Taktstriche ... 97
Taktstriche, Änderung von Standard ... 103
Taktstriche, manuell ... 98
Taktstriche, schließend ... 97
Taktstriche, unsichtbar ... 98
Taktstriche, unterdrücken ... 624
Taktweise Wiederholungen ... 156
Taktzahlen ... 103
Taktzahlen in Wiederholungen ... 105
Taktzahlen mit Buchstaben ... 105
Taktzahlen, gleichmäßige Abstände ... 103
\taor ... 387
taqasim ... 452
Tasteninstrumente, Notensystem ... 315
Tasteninstrumente, zentrierte Dynamik ... 315
teaching ... 32
teaching-Versetzungszeichenstil ... 32
\teeny ... 214, 237, 688
Teiltakt ... 72
Tempo ... 69
\tempo ... 69
tempo, with rhythm ... 729
Tempobezeichnung ... 69
Tempobezeichnungen innerhalb von N-tolen-Klammern ... 52
\temporary ... 784
Tenorschlüssel ... 19
Tenorschlüssel, Chor ... 20
Tenuto ... 120, 760
\tenuto ... 120, 760
text ... 321
Text über mehrere Seiten ... 246
Text über Mehrtaktpausen ... 62
Text alleine ... 231
Text am Taktstrich ... 229
Text außerhalb des Randes ... 227
Text auf der Seite zentrieren ... 239
text column, left-aligned ... 702
text column, right-aligned ... 706
Text einrahmen ... 241
Text in Voltaklammer ... 153
Text mit Sonderzeichen ... 233
Text und Balken ... 84
Text verzieren ... 241
Text, an Melodie ausgerichtet ... 255
Text, andere Sprachen ... 226
Text, Ausrichtung ... 237
Text, Blocksatz ... 240
Text, horizontale Ausrichtung ... 237
Text, innerhalb des Randes behalten ... 227
Text, mehrere Zeilen ... 239
Text, Notation innerhalb ... 245
Text, oberste Ebene ... 231
Text, Rand außen ... 242
Text, Syntax ... 233
Text, vertikale Ausrichtung ... 238
text, with a tie above ... 686, 689
text, with a tie below ... 689, 690
\text-accidental ... 732
\text-common-time ... 732
\text-cut-time ... 732
\text-doubleflat ... 732
\text-doublesharp ... 733
\text-flat ... 733
\text-natural ... 733
\text-sharp ... 733
Textarten ... 226
textartige Zeichen ... 229
Textausrichtung, Befehle ... 241
Textausrichtungsbefehle ... 241
Textbeschriftung ... 233
Textbeschriftung über mehrere Seiten ... 246
Textbeschriftung ausrichten ... 237
Textbeschriftung, Blocksatz ... 240
Textbeschriftung, mehrzeilig ... 239
Textbeschriftung, Notationsobjekte einfügen ... 243
Textbeschriftung, Sonderzeichen ... 233
Textbeschriftungs-Ausdrücke ... 233
Textblöcke ... 239
Textblasen ... 221
Textelemente, nicht leer ... 226
\textEndMark ... 784
Textgröße ... 235
\textLengthOff ... 62, 227
\textLengthOn ... 62, 227
\textMark ... 784
\textSpannerDown ... 228
\textSpannerNeutral ... 228
\textSpannerUp ... 228
Textstrecker ... 228
Textstrecker, Formatierung ... 228
Textzeichen ... 229
\thumb ... 120, 215, 761
thumb-script ... 215
\tie ... 689
tie, above text ... 686, 689
tie, below text ... 689, 690
\tied-lyric ... 733
\tieDashed ... 54
\tieDashPattern ... 784
\tieDotted ... 54
\tieDown ... 54
tiefergestellt ... 236
\tieNeutral ... 54
ties, placement ... 54
\tieSolid ... 54
\tieUp ... 54
\time ... 65, 84, 784
\timeAbbrev ... 77, 784
\times ... 47, 75, 784
timeSignature ... 75
\tiny ... 214, 237, 689
\tocItem ... 487, 785
\toe ... 120, 761
Tonart ... 7, 22
Tonart, Mensuralnotation ... 428
Tonart, Sichtbarkeit nach expliziter Änderung ... 622
Tonarten, greg. Choral ... 432
Tonhöhe: Wechsel der Oktave ... 3
Tonhöhen in MIDI ... 509
Tonhöhen, transponieren ... 12
Tonhöhenbezeichnungen ... 3
Tonhöhenbezeichnungen, andere Sprachen ... 9
top-margin ... 524

top-markup-spacing ... 526
top-system-spacing ... 526
Transformationen, modal ... 16
Transkription von Mensuralmusik ... 187
\translate ... 238, 706
\translate-scaled ... 238, 707
translating text ... 706, 707
\transparent ... 753
transparent, Noten ... 217
transparent, Objekte ... 620
Transponieren ... 12
transponierende Instrumente ... 13
transponierende Schlüssel ... 20
Transponierendes Instrument ... 25
\transpose ... 7, 12, 15, 785
\transposedCueDuring ... 211, 785
Transposition ... 12
\transposition ... 25, 205, 785
Transposition und relativer Modus ... 7
Transposition von Bunddiagrammen ... 356
Transposition, Instrumente ... 25
Transposition, MIDI ... 25
Transposition, modal ... 17
tre corde ... 321
\treCorde ... 321
Tremolo ... 158
tremolo ... 158
Tremolo über Systeme ... 159
Tremolobalken ... 158
tremoloFlags ... 159
Tremolozeichen ... 159
Trennstriche, Gesangstext ... 265
Trennzeichen ... 190
\triangle ... 242, 720
\trill ... 120, 141, 760
Triller ... 120, 141, 760
Triller in MIDI ... 509
Triller mit Tonhöhe ... 142
Triller mit Tonhöhe und Versetzungszeichen ... 142
Triole, Formatierung ... 47
Triolen ... 47
Triolenklammer, Platzierung ... 47
Triolennummer, Änderung ... 48
true (#t) ... 788
\tuplet ... 785
\tupletDown ... 47
\tupletNeutral ... 47
TupletNumber ... 48
\tupletSpan ... 47, 785
tupletSpannerDuration ... 47
\tupletUp ... 47
\turn ... 120, 760
\tweak ... 605, 785
tweak und Kontrollpunkte ... 607
two-sided ... 528
\type ... 593
typeface ... 764
\typewriter ... 689

U

U.C. ... 321
Überbindung ... 53
Überbindung in Wiederholung ... 148
Überbindung und Wiederholungen ... 54
Überbindung, Versetzungszeichen ... 8
Überbindungen und Akkorde ... 53
Überspringen von Noten im Gesangstext ... 276
Überspringen von Zeichen ... 59
Übungszeichen ... 110
Übungszeichen formatieren ... 110
Übungszeichenstil ... 110
Übungszwecke, Notenköpfe ... 38
Ukulele ... 347
Umbrüche in Kadenzen ... 73
Umbrüche in nicht metrischer Musik ... 73
Umbrüche von Zeilen ... 536
Umbrüche, Seite ... 538
Umbrechen von Seiten ... 565
Umbruch von Text ... 240
Umkehrung ... 15
Umkehrung, modal ... 17
Umkehrungen ... 398, 401
una corda ... 321
\unaCorda ... 321
\underline ... 234, 690
underlining text ... 690
\undertie ... 690
\undo ... 785
unfold ... 154
\unfolded ... 786
\unfoldRepeats ... 510, 786
\unHideNotes ... 217
Unicode ... 501
\unless ... 734
unreine Container, Scheme ... 634
\unset ... 602
unsichtbar, Objekte ... 620
unsichtbare Noten ... 217
Unsichtbare Pausen ... 59
unsichtbare Taktstriche ... 98
unsichtbarer Notenhals ... 220
unterschiedlicher Gesangstext ... 280
Unterteilen von Takten ... 91
unterteilte Taktarten ... 75
\upbow ... 120, 325, 761
\upmordent ... 120, 760
\upprall ... 120, 760
\upright ... 691
URL link, as QR code ... 751
UTF-8 ... 501

V

Varcoda ... 120, 760
\varcoda ... 120, 733, 762
\varheel ... 120, 761
Variablen ... 462
Variablen, Benutzung ... 489
Variablen, Gesangstext ... 266
\vartoe ... 120, 761
Vaticana, Editio ... 420, 421
VaticanaStaff ... 431
VaticanaStaff ... 183, 431

VaticanaVoice 431
VaticanaVoice 431
\vcenter 707
veränderbare (mutable) Objekte 765
Verändern der Schriftart 234
Verändern der Schriftgröße 534
Verändern der Systemgröße 534
Verändern von automatischer Bebalkung 84
Verändern von Eigenschaften 601
verändern von Objekten 620
veränderte Akkorde 400
Veränderung des Notensystems 613
Veränderung von Kontexten nur einmal 605
Veränderung von Verzierungsnoten 114
\verbatim-file 754
\verboseBarNumbers 786
Vermeidung von vertikalen Zusammenstößen 559
Verschachtelte Musik 179
verschachtelte Systemklammern 188
verschachtelte Wiederholung 151
Verschachtelung von Systemen 188
Verschieben von Noten 170
Verschieben von Pausen, automatisch 170
Verschiebung 599
Verschiebung von Gesangstext 267
Verschmelzen von Noten 170
Verschwinden von leeren Systemen 198
Versetzungszeichen 7
Versetzungszeichen an übergebundener Note 8
Versetzungszeichen für Klavier 30
Versetzungszeichen in Akkorden 32
Versetzungszeichen pro Stimme 29
Versetzungszeichen und gleichzeitige Noten 32
Versetzungszeichen, automatisch 26
Versetzungszeichen, Deutsch 7
Versetzungszeichen, Erinnerung 8
Versetzungszeichen, für Triller 142
Versetzungszeichen, greg. Choral 432
Versetzungszeichen, Kiever Notation 442
Versetzungszeichen, Mensuralnotation 428
Versetzungszeichen, moderne Stile 29
Versetzungszeichen, moderner Stil mit Warnungen 29
Versetzungszeichen, musica ficta 429
Versetzungszeichen, piano cautionary 30
Versetzungszeichen, Standard 26
Versetzungszeichen, Vierteltöne 7
Versetzungszeichen, Viertelton 9
Versetzungszeichen, Warnung 8
Versetzungszeichenstil 26
Versetzungszeichenstil forget 32
Versetzungszeichenstil Klavier mit Warnungen 30
Versetzungszeichenstil modern 28
Versetzungszeichenstil neo-modern mit Warnungen 31
Versetzungszeichenstil teaching 32
Versetzungszeichenstil Vergessen 32
Versetzungszeichenstil, modern 29
Versetzungszeichenstil, modern mit Warnung für Stimmen 30
Versetzungszeichenstil, modern-cautionary 28
Versetzungszeichenstil, neo-modern 30
Versetzungszeichenstil, neo-modern-voice 31
Versetzungszeichenstil, neo-modern-voice-cautionary 31
Versetzungszeichenstil, no reset 32
Versetzungszeichenstil, piano 30
Versetzungszeichenstil, Standard 28
Versetzungszeichenstil, Stimme 28
Versetzungszeichenstil, Zwölftonmusik 31
Versetzungszeichenstil: nicht zurücksetzen 32
Verstecken von Noten 217
Verstecken von Rhythmus-Systemen 199
Verstecken von Systemen 198
Verstecken von Systemen der Alten Musik 199
versteckte Notensysteme 194
\versus 786
VerticalAxisGroup 545
vertically centering text 707
vertikale Ausrichtung von Text 238
vertikale Linien zwischen Systemen 222
vertikale Position von Dynamik 125
vertikale Positionierung 545
vertikale Zusammenstöße, vermeiden 559
Verwaltung der Zeiteinheiten 118
verwendbare Schriftarten auflisten 249
\verylongfermata 120, 761
\veryshortfermata 761
Verzierung innerhalb von rhythmischer Kombination 51
Verzierung innerhalb von Triole 51
Verzierung, danach 113
Verzierungen 112
Verzierungen in MIDI 509
Verzierungen verändern 114
Verzierungen, Aussehen verändern 114
Verzierungen, manuelle Bebalkung 94
Verzierungen, Synchronisation 116
Verzierungsnoten und Gesangstext 283
viele Stimmen 170
Vierteltöne 7
Vierteltöne in MIDI 509
Vierteltonversetzungszeichen 9
Violinschlüssel 19
\virga 435, 440
\virgula 433
Voice 166
Voice 166
voice 26, 28
Voice-Stile 169
Voice-Versetzungszeichenstil 28
\voiceFourStyle 169
\voiceNeutralStyle 169
\voiceOne 166
\voiceOne ... \voiceFour 166
\voiceOneStyle 169
\voices 786
\voiceThreeStyle 169
\voiceTwoStyle 169
\void 516, 786
Volta 144
\volta 786
Volta und Überbindung 54
Volta-Klammer mit Text 153
Volta-Klammern und Wiederholungen 54
\volta-number 691
Voltaklammer, ändern 152
Vorhalt 112
vorhandene Schriftarten auflisten 249
Vorlage, arabische Musik 453
Vorschlag 112
Vorschlag, mehrere Noten 116

Vorzeichen ... 22
Vorzeichen in Klammern ... 8
Vorzeichen, Erinnerung ... 8
Vorzeichen, greg. Choral ... 432
Vorzeichen, Mensuralnotation ... 428
Vorzeichen, Vierteltöne ... 7
\vshape ... 786
\vspace ... 707

W

Walker-Formnotenköpfe ... 39
\walkerHeads ... 39
\walkerHeadsMinor ... 40
Warnungsversetzungszeichen für Klavier ... 30
Warnungsversetzungszeichen, neo-modern ... 31
Warnungsvorzeichen ... 8
Wechsel der Oktave ... 3
Wechsel des Systems, automatisch ... 317
Wechsel des Systems, manuell ... 315
Wechsel von Instrument ... 204
Wechsel zwischen Systemen ... 319
Wechseln von Instrumentenbezeichnungen ... 203
Weiße Mensuralligaturen ... 429
weit außeinander liegende Balken ... 82
whichBar ... 103
\whiteout ... 754
Wiederherstellen von
Taktart-Standardeigenschaften ... 67
wiederholte Musik ... 154
Wiederholung mit alternativem Schluss ... 144
Wiederholung mit Auftakt ... 146
Wiederholung mit q ... 162, 332
Wiederholung und Bindebögen ... 54
Wiederholung und Bindebogen ... 151
Wiederholung und Zählzeit ... 151
Wiederholung von Gesangstext bei alternativen
Endungen ... 276
Wiederholung, alternative Schlüsse ... 152
Wiederholung, aufklappen ... 154
Wiederholung, Beginn ... 152
Wiederholung, Ende ... 152
Wiederholung, kurz ... 156
Wiederholung, manuell ... 152
Wiederholung, mehrdeutig ... 151
Wiederholung, Prozent ... 156
Wiederholung, taktweise ... 156
Wiederholung, Tremolo ... 158
Wiederholung, verschachtelt ... 151
Wiederholung, Voltaklammer ... 152
Wiederholungen ... 100, 144
Wiederholungen in MIDI ... 510
Wiederholungen mit Überbindung ... 148
Wiederholungen und Gesangstext ... 271
Wiederholungen, alternative Taktnummerierung ... 150
Wiederholungen, ausgeschrieben ... 154
Wiederholungen, Taktnummer mit Buchstaben ... 150
Wiederholungsklammer mit Text ... 153
Wiederholungstaktlinie ... 152
Wiederholungszeichen ... 97
wirkliche Tonhöhe ... 7
\with ... 591
\with ... 586, 591
with-color ... 218
\with-color ... 218, 755
\with-dimension ... 755
\with-dimension-from ... 755
\with-dimensions ... 755
\with-dimensions-from ... 755
\with-link ... 755
\with-outline ... 755
\with-string-transformer ... 691
\with-true-dimension ... 755
\with-true-dimensions ... 756
\with-url ... 721
\withMusicProperty ... 786
\withRelativeDir ... 786
\woodwind-diagram ... 738
\wordwrap ... 240, 708
\wordwrap-field ... 708
\wordwrap-lines ... 246, 759
\wordwrap-string ... 709

X

x11-color ... 218, 219
x11-Farbe ... 219
X-offset ... 545
X11-Farben ... 218
\xNote ... 787

Z

Zählzeit und Wiederholung ... 151
Zahl der Notenlinien einstellen ... 191
Zahl eines Taktes ... 103
Zahl von Saiten ... 328
Zeichen ... 120
Zeichen, Übung: Formatierung ... 110
Zeichen, textartige ... 229
Zeichnen im Text ... 241
Zeilenlänge ... 565
Zeilenumbrüche ... 98, 536
Zeilenumbrüche in Intervallen ... 537
Zeilenumbrüche in Kadenzen ... 73, 75
Zeilenumbrüche in Musik ohne Metrum ... 75
Zeilenumbrüche in nicht metrischer Musik ... 73
Zeilenumbruch, Balken ... 82
Zeit (in der Partitur) ... 118
Zentrieren von Text auf der Seite ... 239
zentrierte Dynamik für Klaviermusik ... 315
Ziernoten ... 112
Zitieren von anderen Stimmen ... 205, 208
zitierter Text ... 226
Zurücksetzen von Taktart-Standardeigenschaften ... 67
Zusammenfalten von Pausen ... 65
Zusammengesetzte Taktarten ... 77
Zusammenstöße ... 170
Zusammenstöße zwischen Systemen ... 316
Zusammenstöße, ignorieren ... 165
Zusammenstöße, kollidierende Notenkolumnen ... 165
Zusammenstöße, Taktnummern ... 108
Zusammenstöße, vertikal, vermeiden ... 559
Zwölftonmusik, Versetzungszeichenstil ... 31
zweite Klammer ... 144
Zwischensystem-Tremolo ... 159
Zwischensysteme-Klammer-Arpeggio ... 141