LilyPond

Das Notensatzprogramm

Benutzerhandbuch

Das LilyPond-Entwicklerteam

Copyright © 1999–2008 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, 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 2.12.1

Inhaltsverzeichnis

1 Musikalische Notation ... 1

1.1 Tonhöhen ... 1
1.1.1 Tonhöhen setzen ... 1
Absolute Oktavenbezeichnung ... 1
Relative Oktavenbezeichnung ... 2
Versetzungszeichen ... 4
Notenbezeichnungen in anderen Sprachen ... 7
1.1.2 Viele Tonhöhen gleichzeitig verändern ... 8
Oktavenüberprüfung ... 8
Transposition ... 9
1.1.3 Tonhöhen anzeigen lassen ... 12
Notenschlüssel ... 12
Tonartbezeichnung ... 15
Oktavierungsklammern ... 17
Transposition von Instrumenten ... 17
Automatische Versetzungszeichen ... 19
Tonumfang ... 25
1.1.4 Notenköpfe ... 27
Besondere Notenköpfe ... 27
Easy-Notation-Notenköpfe ... 28
Notenköpfe mit besonderen Formen ... 28
Improvisation ... 30
1.2 Rhythmus ... 31
1.2.1 Rhythmen eingeben ... 31
Tondauern ... 31
Andere rhythmische Aufteilungen ... 32
Tondauern skalieren ... 35
Bindebögen ... 36
1.2.2 Pausen eingeben ... 38
Pausen ... 39
Unsichtbare Pausen ... 40
Ganztaktige Pausen ... 41
1.2.3 Rhythmen anzeigen lassen ... 45
Taktangabe ... 45
Auftakte ... 47
Musik ohne Metrum ... 48
Polymetrische Notation ... 49
Automatische Aufteilung von Noten ... 52
Melodierhythmus anzeigen ... 53
1.2.4 Balken ... 55
Automatische Balken ... 55
Einstellung von automatischen Balken ... 57
Manuelle Balken ... 66
Gespreizte Balken ... 68
1.2.5 Takte ... 69
Taktlinien ... 69
Taktzahlen ... 71
Takt- und Taktzahlüberprüfung ... 74

Übungszeichen ... 75
1.2.6 Besondere rhythmische Fragen ... 77
Verzierungen ... 77
An Kadenzen ausrichten ... 81
Verwaltung der Zeiteinheiten ... 82
1.3 Ausdrucksbezeichnungen ... 83
1.3.1 An Noten angehängt ... 83
Artikulationszeichen und Verzierungen ... 83
Dynamik ... 85
Neue Lautstärkezeichen ... 89
1.3.2 Bögen ... 90
Legatobögen ... 90
Phrasierungsbögen ... 92
Atemzeichen ... 93
Glissando zu unbestimmter Tonhöhe ... 94
1.3.3 Linien ... 95
Glissando ... 95
Arpeggio ... 96
Triller ... 98
1.4 Wiederholungszeichen ... 100
1.4.1 Lange Wiederholungen ... 100
Normale Wiederholungen ... 100
Manuelle Wiederholungszeichen ... 103
Ausgeschriebene Wiederholungen ... 106
1.4.2 Kurze Wiederholungen ... 106
Prozent-Wiederholungen ... 106
Tremolo-Wiederholung ... 108
1.5 Gleichzeitig erscheinende Noten ... 109
1.5.1 Eine einzelne Stimme ... 109
Noten mit Akkorden ... 109
Gleichzeitige Ausdrücke ... 110
Cluster ... 111
1.5.2 Mehrere Stimmen ... 111
Mehrstimmigkeit in einem System ... 111
Stimmenstile ... 114
Auflösung von Zusammenstößen ... 114
Automatische Kombination von Stimmen ... 118
Musik parallel notieren ... 121
1.6 Notation auf Systemen ... 124
1.6.1 Systeme anzeigen lassen ... 124
Neue Notensysteme erstellen ... 124
Systeme gruppieren ... 125
Verschachtelte Notensysteme ... 129
1.6.2 Einzelne Systeme verändern ... 131
Das Notensystem ... 131
Ossia-Systeme ... 133
Systeme verstecken ... 137
1.6.3 Orchesterstimmen erstellen ... 140
Metronomangabe ... 140
Instrumentenbezeichnungn ... 142
Stichnoten ... 146
Stichnoten formatieren ... 149
1.7 Anmerkungen ... 152
1.7.1 Innerhalb des Systems ... 152

Auswahl der Notations-Schriftgröße ... 152
Fingersatzanweisungen ... 153
Unsichtbare Noten ... 155
Farbige Objekte ... 156
Klammern ... 157
Hälse ... 158
1.7.2 Außerhalb des Notensystems ... 159
Erklärungen in Ballonform ... 159
Gitternetzlinien ... 160
Analyseklammern ... 162
1.8 Text ... 163
1.8.1 Text eingeben ... 163
Textarten ... 163
Text mit Verbindungslinien ... 164
Textartige Zeichen ... 165
Separater Text ... 169
1.8.2 Text formatieren ... 170
Textbeschriftung (Einleitung) ... 170
Überblick über die wichtigsten Textbeschriftungsbefehle ... 172
Textausrichtung ... 174
Graphische Notation innerhalb einer Textbeschriftung ... 177
Musikalische Notation innerhalb einer Textbeschriftung ... 180
Textbeschriftung über mehrere Seiten ... 182
1.8.3 Schriftarten ... 183
Was sind Schriftarten ... 183
Schriftarten für einen Eintrag ... 185
Schriftart des gesamten Dokuments ... 186

2 Spezielle Notation ... 187

2.1 Notation von Gesang ... 187
2.1.1 Übliche Notation für Vokalmusik ... 187
Referenz für Vokalmusik und Gesangstext ... 187
Oper ... 187
Liederhefte ... 187
Gesprochene Musik ... 188
Hymnen ... 188
Alte Vokalmusik ... 188
2.1.2 Eingabe von Text ... 188
Was ist Gesangtext ... 188
Einfache Lieder setzen ... 190
Mit Gesangtexten und Bezeichnern arbeiten ... 191
2.1.3 Text an einer Melodie ausrichten ... 191
Automatische Silbendauer ... 191
Manuelle Silbendauer ... 192
Mehrere Silben zu einer Note ... 193
Mehrere Noten zu einer Silbe ... 193
Noten überspringen ... 195
Fülllinien und Trennstriche ... 195
Gesangtext und Wiederholungen ... 195
2.1.4 Besonderheiten der Gesangtextnotation ... 195
Getrennte Texte ... 196
Text unabhängig von den Noten ... 197
Silben platzieren ... 197
Gesangtext zwischen Systemen zentrieren ... 199

2.1.5 Strophen ... 199
Strophennummern hinzufügen ... 199
Lautstärkebezeichnung zu Strophen hinzufügen ... 199
Sängernamen zu Strophen hinzufügen ... 200
Strophen mit unterschiedlichem Rhythmus ... 200
Die Strophen am Ende ausdrucken ... 201
Die Strophen am Ende in mehreren Spalten drucken ... 202
2.2 Tasteninstrumente und andere Instrumente mit mehreren Systemen ... 204
2.2.1 Übliche Notation für Tasteninstrumente ... 204
Referenz für Tasteninstrumente ... 205
Notensysteme manuell verändern ... 206
Automatischer Systemwechsel ... 206
Stimmführungslinien ... 208
Hälse über beide Systeme ... 209
2.2.2 Klavier ... 210
Klavierpedal ... 210
2.2.3 Akkordion ... 211
Diskant-Symbole ... 211
2.2.4 Harfe ... 215
Referenzen für Harfe ... 215
Harfenpedal ... 215
2.3 Bundlose Saiteninstrumente ... 216
2.3.1 Übliche Notation für nichtgebundene Saiteninstrumente ... 217
Hinweise für ungebundene Saiteninstrumente ... 217
Bezeichnung des Bogens ... 217
Flageolett ... 218
Bartók-Pizzicato ... 218
2.4 Saiteninstrumente mit Bünden ... 219
2.4.1 Übliche Notation für gebundene Saiteninstrumente ... 220
Referenz für gebundene Saiteninstrumente ... 220
Seitennummerbezeichnung ... 220
Standardtabulaturen ... 222
Angepasste Tabulaturen ... 225
Bund-Diagramm-Beschriftung ... 226
Vordefinierte Bund-Diagramme ... 234
Automatische Bund-Diagramme ... 242
Fingersatz der rechten Hand ... 245
2.4.2 Gitarre ... 246
Position und Barret anzeigen ... 246
Flageolett und gedämpfte Noten ... 247
2.4.3 Banjo ... 247
Banjo-Tabulaturen ... 247
2.5 Schlagzeug ... 248
2.5.1 Übliche Notation für Schlagzeug ... 248
Referenz für Schlagzeug ... 248
Grundlagen der Schlagzeugnotation ... 248
Trommelwirbel ... 249
Schlagzeug mit Tonhöhe ... 249
Schlagzeugsysteme ... 250
Eigene Schlagzeugsysteme ... 252
Geisternoten ... 256
2.6 Blasinstrumente ... 256
2.6.1 Übliche Notation für Bläser ... 256
Referenz für Blasinstrumente ... 257

Fingersatz ... 258
2.6.2 Dudelsack ... 258
Dudelsack-Defintionen ... 258
Dudelsack-Beispiele ... 259
2.7 Notation von Akkorden ... 260
2.7.1 Akkord-Modus ... 260
Überblick über den Akkord-Modus ... 260
Übliche Akkorde ... 261
Erweiterte und modifizierte Akkorde ... 263
2.7.2 Akkorde anzeigen ... 266
Akkordbezeichnungen drucken ... 266
Akkordbezeichnungen anpassen ... 268
2.7.3 Generalbass ... 272
Grundlagen des Bezifferten Basses ... 272
Eingabe des Generalbass' ... 273
Generalbass anzeigen ... 276
2.8 Notation von alter Musik ... 279
2.8.1 Einführung in die Notation Alter Musik ... 279
Notation von Alter Musik, die unterstützt wird ... 279
2.8.2 Alternative Notenzeichen ... 279
Notenköpfe Alter Musik ... 279
Versetzungszeichen Alter Musik ... 280
Pausen Alter Musik ... 281
Schlüssel Alter Musik ... 281
Fähnchen Alter Musik ... 283
Taktangaben Alter Musik ... 284
2.8.3 Zusätzliche Notenzeichen ... 285
Artikulationszeichen Alter Musik ... 285
Custodes ... 286
Divisiones ... 286
Ligaturen ... 287
Weiße Mensuralligaturen ... 288
Ligaturen der gregorianischen Quadratnotation ... 289
2.8.4 Vordefinierte Umgebungen ... 294
Gregorianische Gesangs-Kontexte ... 295
Mensural-Kontexte ... 295
2.8.5 Transkription Alter Musik ... 296
Alte und moderne Edition aus einer Quelldatei ... 296
Incipite ... 296
Mensurstriche ... 296
Gregorianischen Choral transkribieren ... 296
2.8.6 Herausgeberische Anmerkungen ... 296
Vorgeschlagene Versetzungszeichen ... 296
Notation barocker Rhythmen ... 297
2.9 Weltmusik ... 297
2.9.1 Arabische Musik ... 297
References for Arabic music ... 297
Arabic note names ... 297
Arabic key signatures ... 298
Arabic time signatures ... 300
Arabic music example ... 301
Weitere Literatur ... 301

3 Allgemeine Eingabe und Ausgabe ... 303

3.1 Eingabestruktur ... 303
3.1.1 Struktur einer Partitur ... 303
3.1.2 Mehrere Partituren in einem Buch ... 304
3.1.3 Die Dateistruktur ... 305
3.2 Titel ... 307
3.2.1 Titel erstellen ... 307
3.2.2 Eigene Titel ... 310
3.2.3 Verweis auf die Seitenzahlen ... 311
3.2.4 Inhaltsverzeichnis ... 312
3.3 Arbeiten an Eingabe-Dateien ... 314
3.3.1 LilyPond-Dateien einfügen ... 314
3.3.2 Verschiedene Editionen aus einer Quelldatei ... 315
Variablen benutzen ... 315
Marken benutzen ... 317
3.3.3 Zeichenkodierung ... 320
3.3.4 LilyPond-Notation anzeigen ... 321
3.4 Ausgabe kontrollieren ... 321
3.4.1 Notationsfragmente extrahieren ... 321
3.4.2 Korrigierte Musik überspringen ... 322
3.5 MIDI-Ausgabe ... 322
3.5.1 MIDI-Dateien erstellen ... 323
Instrumentenbezeichnungn ... 323
3.5.2 Der MIDI-Block ... 325
3.5.3 Was geht in die MIDI-Ausgabe ... 325
In MIDI unterstützt ... 326
In MIDI nicht unterstützt ... 326
3.5.4 Wiederholungen im MIDI ... 326
3.5.5 MIDI-Lautstärke kontrollieren ... 327
Dynamik-Zeichen ... 327
MIDI-Lautstärke ... 328
Verschiedene Instrumente angleichen (i) ... 329
Verschidene Instrumente angleichen (ii) ... 330
3.5.6 Schlagzeug in MIDI ... 331

4 Abstände ... 332

4.1 Papier und Seiten ... 332
4.1.1 Papierformat ... 332
4.1.2 Seitenformatierung ... 332
Vertikale Dimensionen ... 332
Horizontale Dimensionen ... 332
Weitere Layout-Variablen ... 332
4.2 Notenlayout ... 332
4.2.1 Die Notensystemgröße einstellen ... 332
4.2.2 Partiturlayout ... 332
4.3 Umbrüche ... 332
4.3.1 Zeilenumbrüche ... 332
4.3.2 Seitenumbrüche ... 332
4.3.3 Optimale Seitenumbrüche ... 332
4.3.4 Optimale Umbrüche zum Blättern ... 332
4.3.5 Minimale Seitenumbrüche ... 332
4.3.6 Ausdrückliche Umbrüche ... 332
4.3.7 Eine zusätzliche Stimme für Umbrüche benutzen ... 332

4.4 Vertikale Abstände ... 332
4.4.1 Vertikale Abstände innerhalb eines Systems ... 332
4.4.2 Vertikale Abstände zwischen Systemen ... 332
4.4.3 Explizite Positionierung von Systemen ... 332
4.4.4 Vertikale Abstände mit zwei Durchgängen ... 332
4.4.5 Vermeidung von vertikalen Zusammenstößen ... 332
4.5 Horizontale Abstände ... 332
4.5.1 Überblick über horizontale Abstände ... 332
4.5.2 Eine neuer Bereich mit anderen Abständen ... 333
4.5.3 Horizontale Abstände verändern ... 333
4.5.4 Zeilenlänge ... 333
4.5.5 Proportionale Notation ... 333
4.6 Die Musik auf weniger Seiten zwingen ... 333
4.6.1 Abstände anzeigen lassen ... 333
4.6.2 Abstände verändern ... 333

5 Standardeinstellungen verändern ... 335

5.1 Interpretationsumgebungen ... 335
5.1.1 Was sind Umgebungen? ... 335
Score - Meister aller Kontexte ... 335
Oberste Kontexte – Container für Systeme ... 335
Mittlere Kontexte – Systeme ... 335
Unterste Kontexte – Stimmen ... 335
5.1.2 Umgebungen erstellen ... 335
5.1.3 Umgebungs-Plugins verändern ... 335
5.1.4 Die Standardeinstellungen von Umgebungen ändern ... 335
5.1.5 Neue Umgebungen definieren ... 335
5.1.6 Umgebungen aneinander ausrichten ... 335
5.2 Die Referenz der Programminterna erklärt ... 335
5.2.1 Zurechtfinden in der Programmreferenz ... 335
5.2.2 Layout-Schnittstellen ... 335
5.2.3 Die Grob-Eigenschaften ... 335
5.2.4 Benennungsübereinkommen ... 335
5.3 Eignschaften verändern ... 335
5.3.1 Überblick über verändernde Eigenschaften ... 335
5.3.2 Der `\set`-Befehl ... 335
5.3.3 Der \override-Befehl ... 335
5.3.4 Der `\tweak`-Befehl ... 335
5.3.5 `\set` versus `\override` ... 335
5.4 Nützliche Konzepte und Eigenschaften ... 335
5.4.1 Eingabe-Modi ... 335
5.4.2 Richtung und Platzierung ... 335
5.4.3 Abstände und Maße ... 336
5.4.4 Eigenschaften von Staff (Notensystem)-Symbol ... 336
5.4.5 Strecker ... 336
Das `spanner-interface` benutzen ... 336
Das `line-spanner-interface` benutzen ... 336
5.4.6 Sichtbarkeit von Objekten ... 336
Einen stencil entfernen ... 336
Objekten unsichtbar machen ... 336
Objekte weißmalen ... 336
break-visibility (unsichtbar machen) benutzen ... 336
Besonderheiten ... 336
5.4.7 Zeilenstile ... 336

5.4.8 Drehen von Objekten ... 336
Drehen von Objekten ... 336
Textbeschriftung drehen ... 336
5.5 Fortgeschrittene Optimierungen ... 336
5.5.1 Umgebungen ausrichten ... 336
`X-offset` und `Y-offset` direkt setzen ... 336
Das `side-position-interface` benutzen ... 336
Das `self-alignment-interface` benutzen ... 336
Die `aligned-on-parent`-Prozeduren benutzen ... 336
Die `centered-on-parent`-Prozeduren benutzen ... 336
Das `break-aligned-interface` benutzen ... 336
5.5.2 Vertikale Gruppierung der grafischen Objekte („grob“s) ... 336
5.5.3 stencils verändern ... 336
5.5.4 Formen verändern ... 336
Bögen verändern ... 336

6 Schnittstellen für Programmierer ... 337
6.1 Musikalische Funktionen ... 337
6.1.1 Überblick über musikalische Funktionen ... 337
6.1.2 Einfache Ersetzungsfunktionen ... 337
6.1.3 Paarige Ersetzungsfunktionen ... 337
6.1.4 Mathematik in Funktionen ... 337
6.1.5 Leere Funktionen ... 337
6.1.6 Funktionen ohne Argumente ... 337
6.1.7 Überblick über vorhandene musikalische Funktionen ... 337
6.2 Schnittstelle für Programmierer ... 340
6.2.1 Eingabevariablen und Scheme ... 340
6.2.2 Interne Repräsentation der Musik ... 340
6.3 Komplizierte Funktionen erstellen ... 340
6.3.1 Musikalische Funktionen darstellen ... 340
6.3.2 Eigenschaften von Musikobjekten ... 340
6.3.3 Verdoppelung einer Note mit Bindebögen (Beispiel) ... 340
6.3.4 Artikulationszeichen zu Noten hinzufügen (Beispiel) ... 340
6.4 Programmierungsschnittstelle für Textbeschriftungen ... 341
6.4.1 Beschriftungskonstruktionen in Scheme ... 341
6.4.2 Wie Beschriftungen intern funktionieren ... 341
6.4.3 Neue Definitionen von Beschriftungsbefehlen ... 341
6.4.4 Neue Definitionen von Beschriftungsbefehlen für Listen ... 341
6.5 Kontexte für Programmierer ... 341
6.5.1 Kontextauswertung ... 341
6.5.2 Eine Funktion auf alle Layout-Objekte anwenden ... 341
6.6 Scheme-Vorgänge als Eigenschaften ... 341
6.7 Scheme-Code anstelle von `weak` verwenden ... 341
6.8 Schwierige Korrekturen ... 341

Anhang A Literatur ... 342

Anhang B Notationsübersicht ... 343
B.1 Liste der Akkordbezeichnungen ... 343
B.2 Übliche Akkord-Variablen ... 344
B.3 Vordefinierte Bund-Diagramme ... 347
B.4 MIDI-Instrumente ... 350
B.5 Liste der Farben ... 351
B.6 Die Feta-Schriftart ... 352
B.7 Notenkopfstile ... 366
B.8 Text markup commands ... 367
B.8.1 Font ... 367
B.8.2 Align ... 376
B.8.3 Graphic ... 389
B.8.4 Music ... 393
B.8.5 Instrument Specific Markup ... 396
B.8.6 Other ... 399
B.9 Text markup list commands ... 403
B.10 Liste der Arktikulationszeichen ... 404
B.11 Schlagzeugnoten ... 405
B.12 Alle Kontexteigenschaften ... 406
B.13 Eigenschaften des Layouts ... 416
B.14 Bezeichner ... 429
B.15 Scheme-Funktionen ... 433

Anhang C Befehlsübersicht ... 452

Anhang D GNU Free Documentation License ... 456
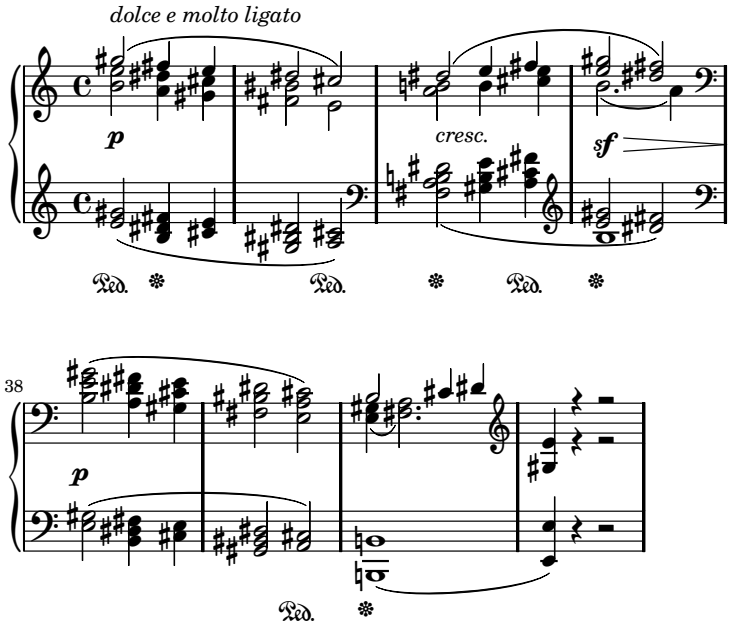
Anhang E Index der LilyPond-Befehle ... 462

Anhang F LilyPond-Index ... 469

1 Musikalische Notation

Dieses Kapitel erklärt, wie die Notation von Musik erstellt wird.

1.1 Tonhöhen



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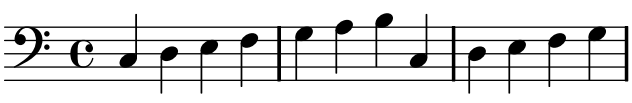
1.1.1 Tonhöhen setzen

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.

Absolute Oktavenbezeichnung
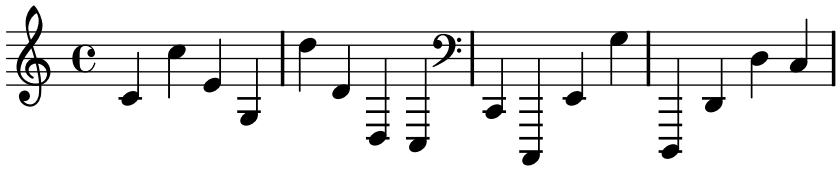
Tonhöhenbezeichnungen werden durch Kleinbuchstaben von `a` bis `g` angegeben. Dabei wird ein aus dem Englischen entlehntes Modell benutzt, das sich vom Deutschen dadurch unterscheidet, dass `b` für die Note „H“ steht. Die Benutzung deutscher Notenbezeichnungen mit der Unterscheidung von b und h ist auch möglich, siehe [Notenbezeichnungen in anderen Sprachen], Seite 7. Die Notenbezeichnungen `c` bis `b` werden in der Oktave unter dem zweigestrichenen C gesetzt.

```
\clef bass
c d e f
g a b c
d 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' c'' e' g
d'' d' d c
\clef bass
c, c,, e, g
d,, d, d c
```



Siehe auch

Glossar: Abschnitt “Tonhöhenbezeichnungen” in *Glossar*.

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

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.
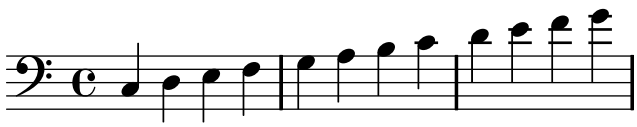
`\relative` ***`Anfangstonhöhe musikalischer Ausdruck`***

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.
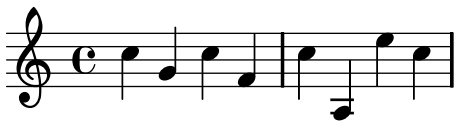
So funktioniert der relative Modus:

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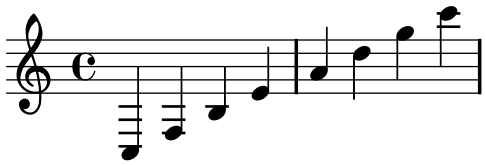


Oktavversetzungen müssen für alle Intervalle angezeigt werden, die größer als eine Quarte sind.

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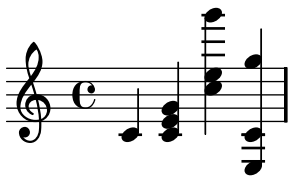


Eine Sequenz ohne ein einziges Oktaväänderungszeichen kann aber trotzdem weite Intervalle umfassen:

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



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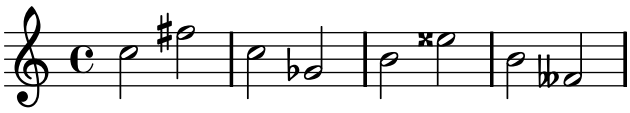
```
\relative c' {
  c
  <c e g>
  <c' e g'>
  <c, e, g''>
}
```



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'' {
  c2 fis
  c2 ges
  b2 eisis
  b2 feses
```

```
}
```



Siehe auch

Musickgossar: Abschnitt “Quinte” in *Glossar*, Abschnitt “Intervalle” in *Glossar*, Abschnitt “Tonhöhenbezeichnungen” in *Glossar*.

Notationsreferenz: [Oktavenüberprüfung], Seite 8.

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

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

Bekannte Probleme und Warnungen

Die relative Veränderung wirkt sich nicht auf Transposition (`\transpose`), Akkordnotation (`\chordmode`) oder `\relative`-Abschnitte aus. Um den relativen Modus innerhalb von transponierter Musik zu verwenden, muss ein zusätzliches `\relative` innerhalb der Klammern des `\transpose`-Befehls gesetzt werden.
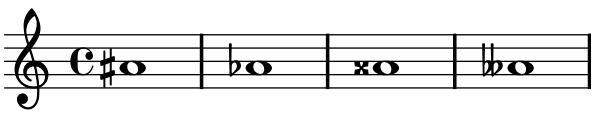
Wenn keine *Anfangstonhöhe* für `\relative` angegeben wird, wird `c'` angenommen. Das ist aber eine veraltete Option, die in späteren Programmversionen verschwinden kann. Darum wird von der Benutzung abgeraten.

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 “Accidentals and key signatures” in *Handbuch zum Lernen*.
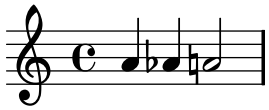
Ein *Kreuz* wird eingegeben, indem man `-is` an die Notenbezeichnung hängt, ein b durch `-es`. Doppelkreuze und Doppel-Bs werden durch Hinzufügen von `-isis` und `-eses` hinter die Notenbezeichnung erzeugt. Diese Syntax leitet sich von den holländischen Notenbezeichnungen ab. Um andere Bezeichnungen für Versetzungszeichen zu benutzung, siehe [Notenbezeichnungen in anderen Sprachen], Seite 7.

```
ais1 aes aisis aeses
```
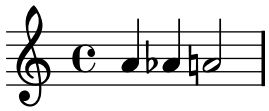


Auch die deutschen Varianten `as` für `aes` und `es` für `ees` sind erlaubt. Im Unterschied zum Deutschen ist aber `bes` die einzige Version für den Ton B, während his als `bis` geschrieben werden muss. Das kann aber auch verändert werden, siehe [Notenbezeichnungen in anderen Sprachen], Seite 7.

```
a4 aes a2
```



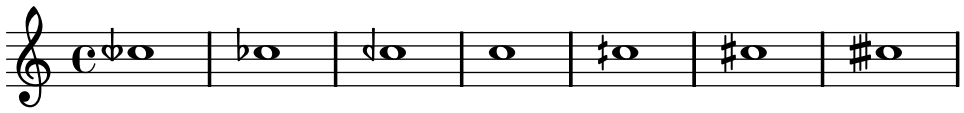
Ein Auflösungszeichen macht die Wirkung eines Kreuzes oder Bs rückgängig. Diese Auflösungszeichen werden jedoch nicht als Suffix einer Tonhöhenbezeichnung eingegeben, sondern sie ergeben sich (automatisch) aus dem Kontext, wenn die nicht alterierte Notenbezeichnung eingegeben wird.

```
a4 aes a2
```
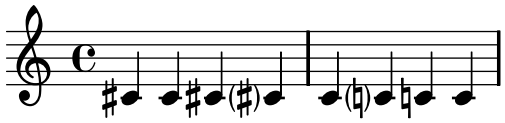


Versetzungszeichen für Vierteltöne werden durch Anhängen der Endungen `-eh` (Erniedrigung) und `-ih` (Erhöhung) an den Tonhöhenbuchstaben erstellt. Das Beispiel zeigt eine in Vierteltönen aufsteigende Serie vom eingestrichenen C.

```
ceseh1 ces ceh c cih cis cisih
```



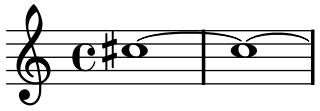
Normalerweise werden Versetzungszeichen automatisch gesetzt, aber sie können auch manuell hinzugefügt werden. Ein erinnerndes Versetzungszeichen kann erzwungen werden, indem man ein Ausrufungszeichen (`!`) hinter die Notenbezeichnung schreibt. Ein warnendes Versetzungszeichen (also ein Vorzeichen in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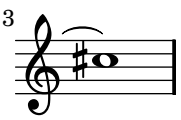
```
cis cis cis! cis? c c? c! c
```



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

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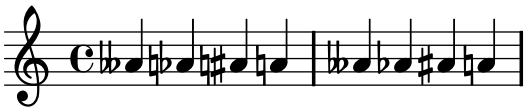




Ausgewählte Schnipsel

*Verhindern, dass zusätzliche Auflösungszeichen automatisch*hinzugefügt werden Den tranditionellen 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.
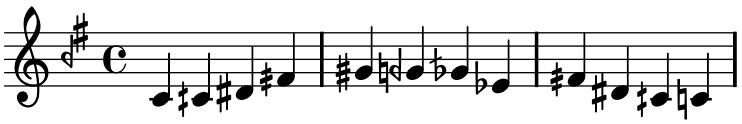
```
\relative c'' {
  aeses4 aes ais a
  \set Staff.extraNatural = ##f
  aeses4 aes ais a
}
```



*Makam example*Makam is a type of melody from Turkey using 1/9th-tone microtonal alterations. Consult the initialization file `makam.ly` (see the 'Learning Manual 2.12.1, 4.6.3 Other sources of information' for the location of this file) for details of pitch names and alterations.

```
% Makam-Einstellungen initialisieren
\include "makam.ly"

\relative c' {
  \set Staff.keySignature = #`((3 . ,BAKIYE) (6 . ,(- KOMA)))
  c4 cc db fk
  gbm4 gfc gfb efk
  fk4 db cc c
}
```



Siehe auch

Glossar: Abschnitt "Kreuz" in *Glossar*, Abschnitt "B" in *Glossar*, Abschnitt "Doppelkreuz" in *Glossar*, Abschnitt "Doppel-B" in *Glossar*, Abschnitt "Tonhöhenbezeichnungen" in *Glossar*, Abschnitt "Viertelton" in *Glossar*.

Handbuch zum Lernen: Abschnitt "Accidentals and key signatures" in *Handbuch zum Lernen*.

Notationsreferenz: [Automatische Versetzungszeichen], Seite 19, [Vorgeschlagene Versetzungszeichen], Seite 296, [Notenbezeichnungen in anderen Sprachen], Seite 7.

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.

Notenbezeichnungen in anderen Sprachen

Es gibt vordefinierte Bezeichnungen für die Notenbezeichnungen in anderen Sprachen als Englisch. Um sie zu benutzen, muss nur die entsprechende Datei für die jeweilige Sprache eingefügt werden. Zum Beispiel fügt man mit `\include "deutsch.ly"` die Notendefinitionen für die deutsche Sprache am Anfang der Datei hinzu. In der Tabelle sind die existierenden Sprachdefinitionen mit den dazugehörigen Notenbezeichnungen dargestellt
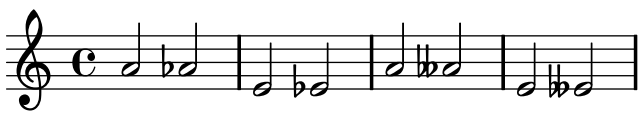
Sprachdatei	**Notenbezeichnung**
‘`nederlands.ly`’	c d e f g a bes b
‘`arabic.ly`’	do re mi fa sol la sib si
‘`catalan.ly`’	do re mi fa sol la sib si
‘`deutsch.ly`’	c d e f g a b h
‘`english.ly`’	c d e f g a bf b
‘`espanol.ly`’	do re mi fa sol la sib si
‘`italiano.ly`’	do re mi fa sol la sib si
‘`norsk.ly`’	c d e f g a b h
‘`portugues.ly`’	do re mi fa sol la sib si
‘`suomi.ly`’	c d e f g a b h
‘`svenska.ly`’	c d e f g a b h
‘`vlaams.ly`’	do re mi fa sol la sib si

und die dazugehörigen Versetzungszeichen-Endungen:

Sprachdatei	**Kreuz**	**B**	**Doppelkreuz**	**Doppel-B**
‘`nederlands.ly`’	-is	-es	-isis	-eses
‘`arabic.ly`’	-d	-b	-dd	-bb
‘`catalan.ly`’	-d/-s	-b	-dd/-ss	-bb
‘`deutsch.ly`’	-is	-es	-isis	-eses
‘`english.ly`’	-s/-sharp	-f/-flat	-ss/-x/-sharpsharp	-ff/-flatflat
‘`espanol.ly`’	-s	-b	-ss	-bb
‘`italiano.ly`’	-d	-b	-dd	-bb
‘`norsk.ly`’	-iss/-is	-ess/-es	-ississ/-isis	-essess/-eses
‘`portugues.ly`’	-s	-b	-ss	-bb
‘`suomi.ly`’	-is	-es	-isis	-eses
‘`svenska.ly`’	-iss	-ess	-ississ	-essess
‘`vlaams.ly`’	-k	-b	-kk	-bb

Auf Holländisch, Deutsch, Norwegisch und Schwedisch (u. a.) werden die Erniedrigungen von ‚a‘ wie `aes` und `aeses` zu `as` und `ases` (oder auch `asas`) zusammengezogen. In manchen Sprachen sind nur diese Kurzformen definiert.

```
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. „einein-halb“. Für alle anderen Sprachen sind noch keine eigenen Namen definiert.

Sprachdatei	**Vierteltonkreuz**	**Viertelton-B**	**3/4-tonkreuz**	**3/4-ton-B**

‘nederlands.ly’	-ih	-eh	-isih	-eseh
‘arabic.ly’	-sd	-sb	-dsd	-bsb
‘deutsch.ly’	-ih	-eh	-isih	-eseh
‘english.ly’	-qs	-qf	-tqs	-tqf
‘italiano.ly’	-sd	-sb	-dsd	-bsb
‘portugues.ly’	-sqt	-bqt	-stqt	-btqt

Siehe auch

Glossar: Abschnitt “Tonhöhenbezeichnungen” in *Glossar*.

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

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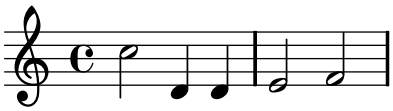
Dieser Abschnitt zeigt, wie man Tonhöhen beeinflusst.

Oktavenüberprüfung

Im relativen Modus geschieht es recht häufig, dass ein Oktavänderungszeichen 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.
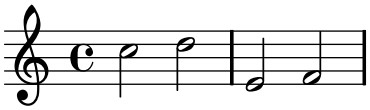
Um die Oktave einer Note zu überprfüfen, muss die absolute Oktave nach dem `=`-Symbol angegeben werden. Im folgenden Beispiel wird eine Warnung (und eine Tonhöhenänderung) generiert, weil die zweite Note als absolute Oktave ein `d''` anstelle von `d'` notiert ist, wie es die Oktavierungskorrektur markiert.

```
\relative c'' {
  c2 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*.
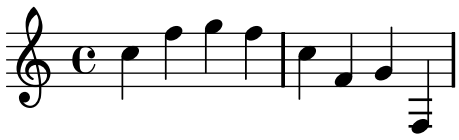
```
\relative c'' {
  c2 d
  \octaveCheck c'
  e2 f
}
```



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'' {
  c4 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*.

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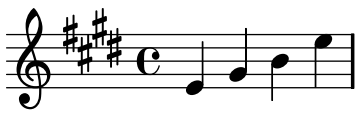
Ein musikalischer Ausdruck kann mit dem Befehl `\transpose` transponiert werden. Die Syntax lautet:

```
\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.

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 c' {
    \key d \major
    d4 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 entpsrechende Stimme wie folgt erstellt werden:

```
\transpose a c' {
  \relative c' {
    \key c \major
    c4 d e g
  }
```
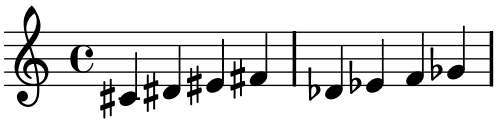
```
}
```



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.

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



`\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 [Transposition von Instrumenten], Seite 17.

Ausgewählte Schnipsel

*Noten mit minimaler Anzahl an Versetzungszeichen transponieren.*Dieses Beispiel benutzt Scheme-Code, um enharmonische Verwechslungen für Noten zu erzwingen, damit nur eine minimale Anzahl an Versetzungszeichen ausgegeben wird. In diesem Fall gelten die folgenden Regeln:

- Doppelte Versetzungszeichen sollen entfernt werden
- His -> C
- Eis -> F
- Ces -> B
- Fes -> E

Auf diese Art werden am meisten natürliche Tonhöhen als enharmonische Variante gewählt.

```
#(define  (naturalize-pitch p)
  (let* ((o (ly:pitch-octave p))
         (a (* 4 (ly:pitch-alteration p)))
    ; alteration, a, in quarter tone steps, for historical reasons
         (n (ly:pitch-notename p)))
    (cond
     ((and (> a 1) (or (eq? n 6) (eq? n 2)))
      (set! a (- a 2))
      (set! n (+ n 1)))
     ((and (< a -1) (or (eq? n 0) (eq? 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))))
    (if (< n 0) (begin (set! o (- o 1)) (set! n (+ n 7))))
    (if (> n 6) (begin (set! o (+ o 1)) (set! n (- n 7))))
    (ly:make-pitch o n (/ a 4))))
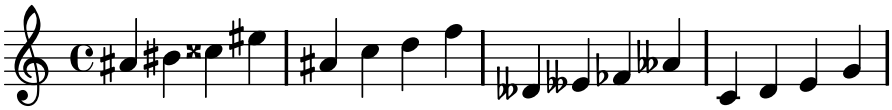
#(define (naturalize Noten)
  (let* ((es (ly:music-property Noten 'elements))
         (e (ly:music-property Noten 'element))
         (p (ly:music-property Noten 'pitch)))
    (if (pair? es)
        (ly:music-set-property!
         Noten 'elements
         (map (lambda (x) (naturalize x)) es)))
    (if (ly:music? e)
        (ly:music-set-property!
         Noten 'element
         (naturalize e)))
    (if (ly:pitch? p)
        (begin
          (set! p (naturalize-pitch p))
          (ly:music-set-property! Noten 'pitch p)))
    Noten))

AuflösungszeichenNoten =
#(define-music-function (parser location m)
                                        (ly:music?)
                        (naturalize m))

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

\score {
  \new Staff {
    \transpose c ais { \Noten }
    \AuflösungszeichenNoten \transpose c ais { \Noten }
    \transpose c deses { \Noten }
    \AuflösungszeichenNoten \transpose c deses { \Noten }
  }
  \layout { }
```

```
}
```



Siehe auch

Notationsreferenz: [Transposition von Instrumenten], Seite 17.

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 `\tranpose` zusätzlich gesetzt werden.
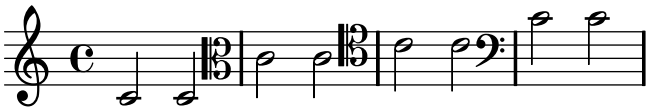
1.1.3 Tonhöhen anzeigen lassen

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

Notenschlüssel

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

```
\clef treble
c2 c
\clef alto
c2 c
\clef tenor
c2 c
\clef bass
c2 c
```
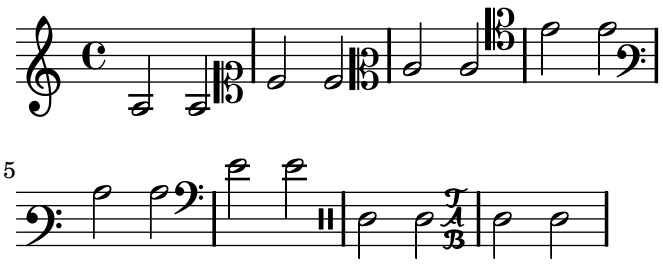


Andere Schlüssel sind u. A.:

```
\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
```
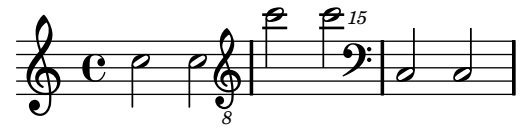
```
\clef percussion
c2 c
\clef tab
c2 c
```



Weitere unterstützte Schlüssel sind beschrieben in [Schlüssel Alter Musik], Seite 281.

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. Die Schlüsselbezeichnung muss in Anführungszeichen gesetzt werden, wenn sie Unterstriche oder Zahlen enthält, siehe Beispiel:

```
\clef treble
c2 c
\clef "treble_8"
c2 c
\clef "bass^15"
c2 c
```



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 `clefOctavation`. 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 `clefOctavation`-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.

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

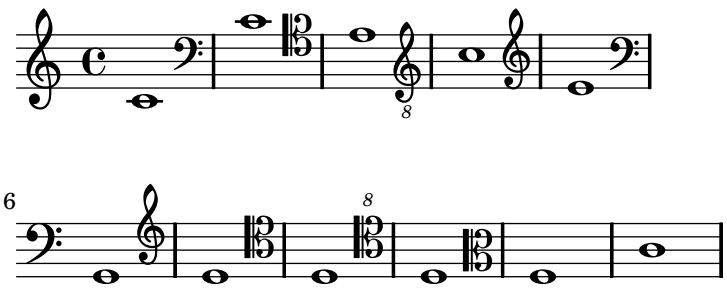
{
  % Standard-Sopranschlüssel
  c'1
  % Standard-Bassschlüssel
  \set Staff.clefGlyph = #"clefs.F"
  \set Staff.clefPosition = #2
  \set Staff.middleCPosition = #6
  c'1
  % Baritonschlüssel
  \set Staff.clefGlyph = #"clefs.C"
  \set Staff.clefPosition = #4
  \set Staff.middleCPosition = #4
  c'1
  % Standard-Chortenorschlüssel
  \set Staff.clefGlyph = #"clefs.G"
  \set Staff.clefPosition = #-2
  \set Staff.clefOctavation = #-7
  \set Staff.middleCPosition = #1
  c'1
  % Nicht-Standard-Schlüssel
  \set Staff.clefPosition = #0
  \set Staff.clefOctavation = #0
  \set Staff.middleCPosition = #-4
  c'1 \break

  % Folgende Schlüsselwechsel erhalten nicht
  % das normale Verhältnis zwischen Noten und Schlüsseln

  \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.clefOctavation = #7
  c'1
  \set Staff.clefOctavation = #0
  \set Staff.clefPosition = #0
  c'1

  % Wieder der normale Schlüssel:

  \set Staff.middleCPosition = #0
  c'1
}
```



Siehe auch

Notationsreferenz: [Schlüssel Alter Musik], Seite 281.

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

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

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 “Accidentals and key signatures” in *Handbuch zum Lernen*.
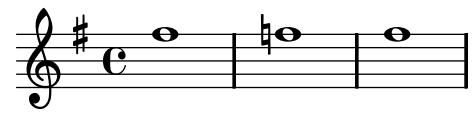
Die Vorzeichen zeigen die Tonart an, in welcher ein Stück notiert ist. Es handelt sich um eine Anzahl von Alterationszeichen (Kreuzen oder Bs) am Beginn jedes Notensystems.

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` (Locrisch), `\aeolian` (Aeolisch), `\mixolydian` (Mixolydisch), `\lydian` (Lydisch), `\phrygian` (Phrygisch) und `\dorian` (Dorisch).

```
\key g \major
fis1
f
fis
```
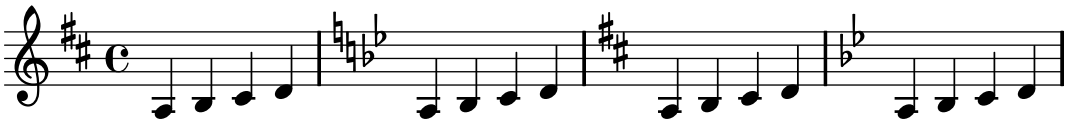


Ausgewählte Schnipsel

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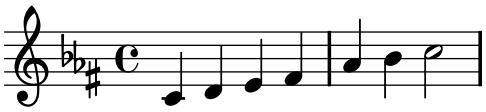
Der üblicherweise benutzte `\key`-Befehl setzt die `keySignature`-Eigenschaft im `Staff`-Kontext.

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:

```
\relative c' {
  \set Staff.keySignature = #`(((0 . 3) . ,SHARP)
                              ((0 . 5) . ,FLAT)
                              ((0 . 6) . ,FLAT))
  c4 d e fis
  aes4 bes c2
}
```



Siehe auch

Glossar: Abschnitt “Kirchentonart” in *Glossar*, Abschnitt “Skordatur” in *Glossar*.

Handbuch zum Lernen: Abschnitt “Accidentals and key signatures” in *Handbuch zum Lernen*.
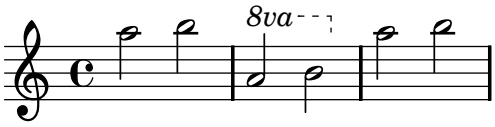
Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

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*.

Oktavierungsklammern

Oktavierungsklammern zeigen eine zusätzliche Transposition von einer Oktave an:

```
a'2 b
\ottava #1
a b
\ottava #0
a b
```



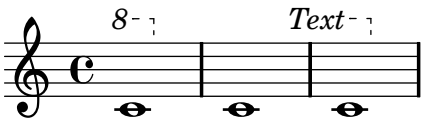
Die `ottava`-(Oktavierungs)-Funktion kann auch die Werte -1 (für 8va bassa), 2 (für 15ma), und -2 (für 15ma bassa) als Argumente haben.

Ausgewählte Schnipsel

Ottava-Text

Intern setzt die `set-octavation`-Funktion die Eigenschaften `ottavation` (etwa auf den Wert `"8va"` oder `"8vb"`) und `middleCPosition`. Um den Text der Oktavierungsklammer zu ändern, kann `ottavation` manuell gesetzt werden, nachdem `set-octavation` benützt wurde.

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



Siehe auch

Glossar: Abschnitt “Oktavierung” 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*.

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 [Stichnoten], Seite 146.

`\transposition` *Tonhöhe*
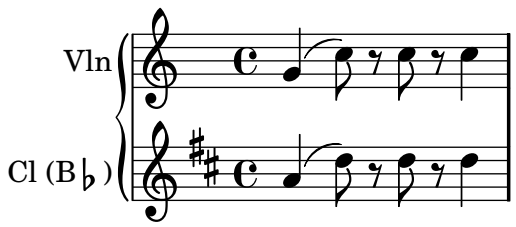
Die Tonhöhe, die für `\transposition` benutzt wird, muss mit dem wirklichen Ton übereinstimmen, der erklingt, wenn das Instrument ein `c'` in seiner Stimme spielt. Die Tonhöhe wird im absoluten Modus angegeben, ein Instrument also, dass einen Ton höher erklingt als es notiert wird, muss folgenden Befehl 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 = "Geige" {
    \relative c'' {
      \set Staff.instrumentName = #"Vln"
      \set Staff.midiInstrument = #"Geige"
      % nicht unbdingt nötig, aber gut zu erinnern
      \transposition c'

      \key c \major
      g4( c8) r c r c4
    }
  }
  \new Staff = "Klarinette" {
    \relative c'' {
      \set Staff.instrumentName = \markup { Cl (B\flat) }
      \set Staff.midiInstrument = #"Klarinette"
      \transposition bes

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

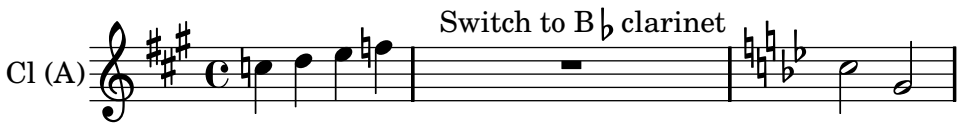


Die `\transposition` kann während eines Stückes geändert werden. Ein Klarinetist zum Beispiel kann zwischen B- und A-Klarinette wechseln.

```
\set Staff.instrumentName = #"Cl (A)"
\key a \major
\transposition a
c d e f
\textLengthOn
s1*0^\markup { Switch to B\flat clarinet }
R1

\key bes \major
\transposition bes
```

```
c2 g
```



Siehe auch

Glossar: Abschnitt “Kammerton” in *Glossar*, Abschnitt “Transponierende Instrumente” in *Glossar*.

Notationsreferenz: [Stichnoten], Seite 146, [Transposition], Seite 9.

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

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 <<
  #(set-accidental-style '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 Stimmbruppe (`StaffGroup`) zu benutzen, müsste der Befehl so aussehen:

```
#(set-accidental-style 'voice 'StaffGroup)
```

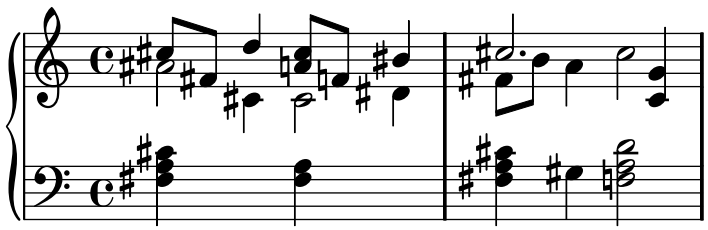
Folgende Versetzungszeichenstile sind unterstützt. Um jeden Stil zu erklären, wird folgendes Beispiel benützt:

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

NotenB = {
  \clef bass
  \new Voice {
    \voiceTwo \relative c' {
      <fis, a cis>4
      \change Staff = oben
      cis'
```

```
        \change Staff = unten
        <fis, a>
        \change Staff = oben
        dis' |
        \change Staff = unten
        <fis, a cis>4 gis <f a d>2 |
      }
    }
  }

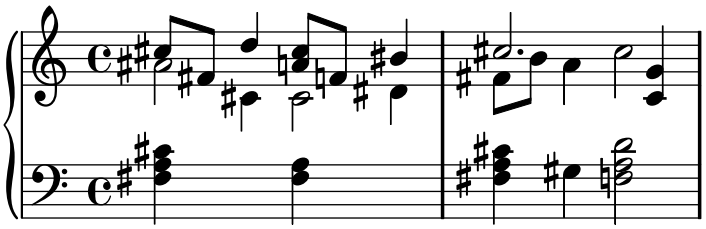
  \new PianoStaff {
    <<
      \context Staff = "oben" {
        #(set-accidental-style 'default)
        \NotenA
      }
      \context Staff = "unten" {
        #(set-accidental-style 'default)
        \NotenB
      }
    >>
  }
```



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" {
        %%% change the next line as desired:
        #(set-accidental-style 'default 'Score)
        \musicA
      }
      \context Staff = "down" {
        \musicB
      }
    >>
  }
```

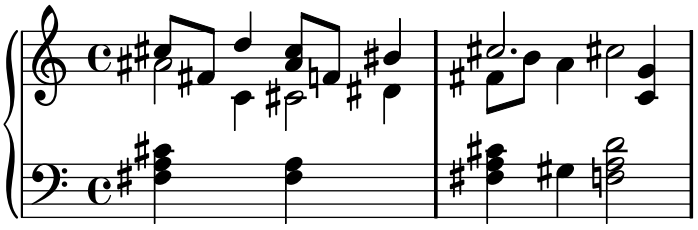
`default (Standard)`
: Das ist das Standardverhalten. Es entspricht der Konvention für Notation von Musik des 18. Jahrhunderts: Versetzungszeichen werden bis zum Taktende erinnert, in dem sie gesetzt wurden, und nur in ihrer eigenen Oktave. Im nächsten Beispiel wird also kein Auslösungszeichen vor dem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



`voice (Stimme)`

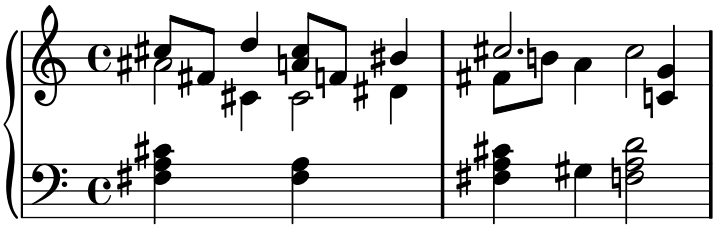
Das normale Verhalten ist es, die Versetzungszeichen auf der Notensystemebene zu erinnern. In diesem Stil aber werden Versetzungszeichen individuell für jede Stimme errechnet. Abgesehen davon gelten die Regeln des Standardstiles (`default`).

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.



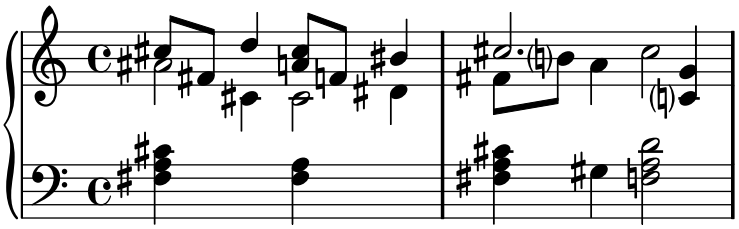
`modern (Modern)`

Dieser Stil orientiert sich an den üblichen Regeln für das 20. Jahrhundert. Die gleichen Versetzungszeichen wie im Standardstil werden gesetzt, allerdings mit zwei Ausnahmen, 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:



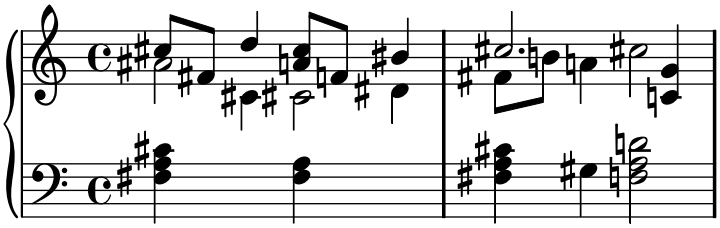
`modern-cautionary (Modern mit Warnungen)`

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.
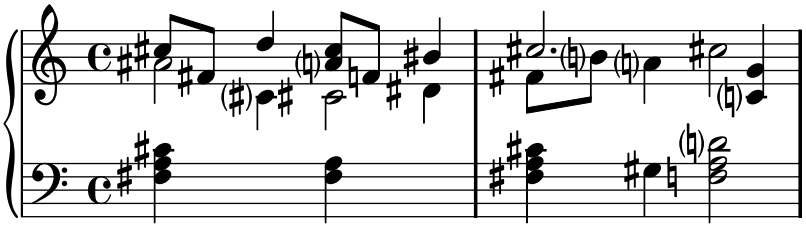


`modern-voice (Modern für Stimmeen)`

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 Versetzunszeichens in einer anderen Stimme im vorigen Takt:



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

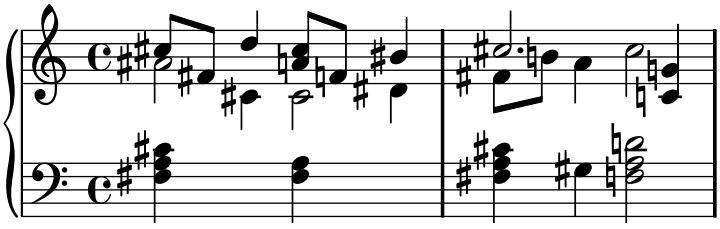
Dieser Stil ist der gleiche wie `modern-voice`, nur dass hier die zusätzlichen Versetzungszeichen (die nicht vom `voice`-Stil gesetzt werden) als Warnungsversetzungszeichen gesetzt werden. Obwohl alle Versetzungszeichen, die mit `default` gesetzt werden, auch mit diesem Stil gesetzt werden, sind manche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



`piano (Klavier)`

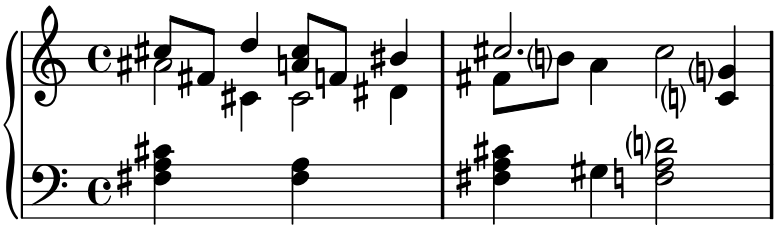
Dieser Stil orientiert sich an den Regeln im 20. Jahrhundert für die Notation von Klaviermusik. Er ist sehr ähnlich mit dem modernen Stil, aber Versetzungszeichen werden auch über Notensysteme hinweg für die selbe Akkolade (`GrandStaff` oder `PianoStaff`) aufgelöst.

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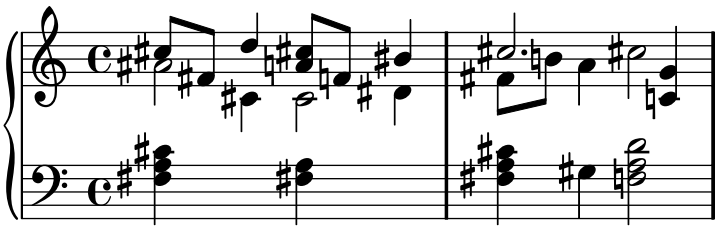


`piano-cautionary (Klavier mit Warnungen)`

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



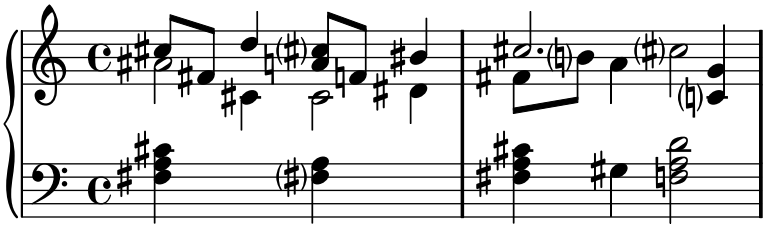
`neo-modern`

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.



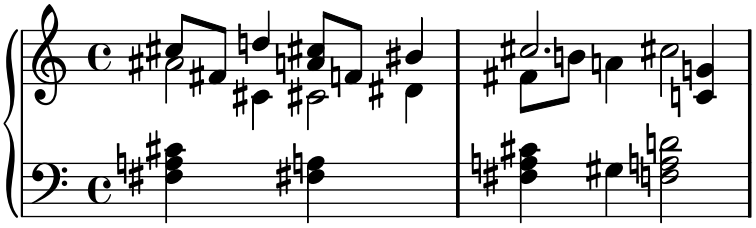
`neo-modern-cautionary (neo-modern mit Warnungen)`

Dieser Stil ähnelt `neo-modern`, aber die zusätzlichen Versetzungszeichen werden als Warnungen gesetzt.
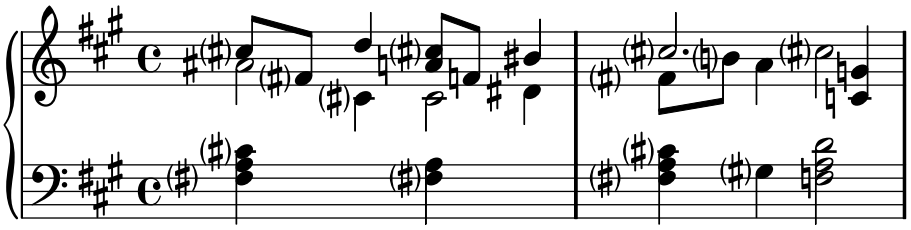


`dodecaphonic (Zwölftonmusik)`

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.
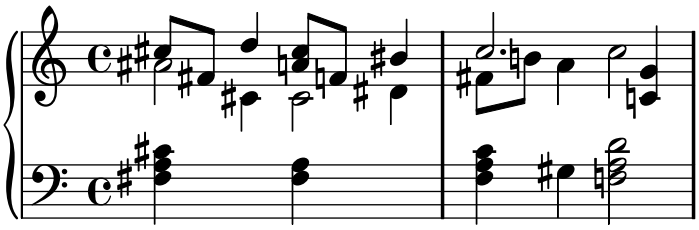


`teaching (didaktisch)`
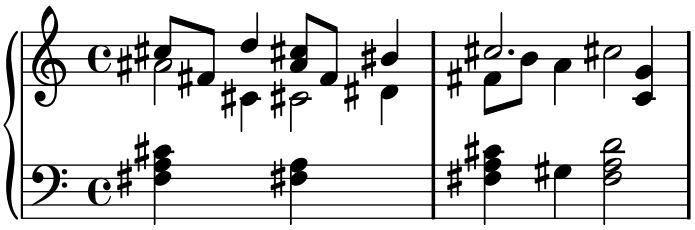
Dieser Stil ist für Lernende bestimmt: der Stil orientiert sich am `modern`-Stil, aber die Alterationen, die sich durch die Tonart ergeben, werden zusätzlich als Warnungsversetzungszeichen gesetzt. Eine Ausnahme sind direkt wiederholte Noten.



`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. Anders als `dodecaphonic` werden nie Auflösungszeichen gesetzt:



Ausgewählte Schnipsel
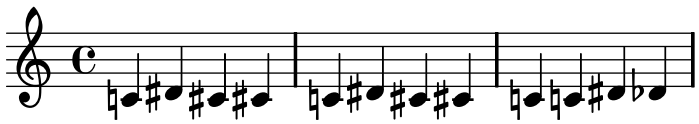
Versetzungszeichen für jede Note im Stil der Zwölftonmusik

In Werken des fürhen 20. Jahrhundert, angefangen mit Schönberg, Berg und Webern (die zweite Wiener Schule), wird jeder Ton der Zwölftonleiter als gleichwertig erachtet, ohne hierarchische Ordnung. Deshalb wird in dieser Musik für jede Note ein Versetzungszeichen ausgegeben, auch für unalterierte Tonhöhen, um das neue Verständnis der Musiktheorie und Musiksprache zu verdeutlichen.

Dieser Schnipsel zeigt, wie derartige Notationsregeln zu erstellen sind.

```
\score {
  \new Staff {
    #(set-accidental-style 'dodecaphonic)
    c'4 dis' cis' cis'
    c'4 dis' cis' cis'
    c'4 c' dis' des'
  }
  \layout {
    \context {
    \Staff
    \remove "Key_engraver"
```

```
    }
  }
}
```



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* and 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 müssen im sequenziellen Modus eingegeben werden. Das bedeutet, dass die Versetzungszeichen von Noten in Akkorden so gesetzt werden, als ob die Noten nacheinander auftreten, in der Reihenfolge, in der sie im Quelltext erscheinen. Das ist ein Problem, wenn Versetzungzeichen in einem AKkord voneinander abhängen, was im Standard-Stil nicht vorkommt. Das Problem kann gelöst werden, indem man manuell `!` oder `?` für die problematischen Noten schreibt.
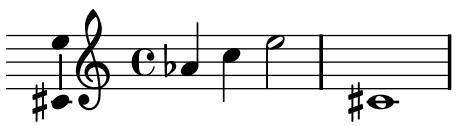
Tonumfang

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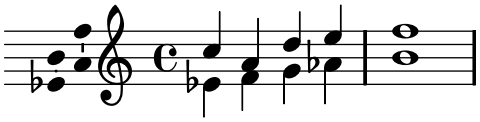


Ausgewählte Schnipsel

Ambitus pro Stimme hinzufügen

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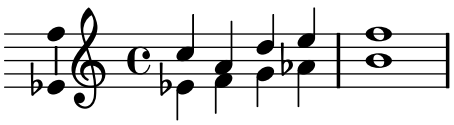
```
\new Staff <<
  \new Voice \with {
    \consists "Ambitus_engraver"
  } \relative c'' {
    \override Ambitus #'X-offset = #2.0
    \voiceOne
    c4 a d e
    f1
  }
  \new Voice \with {
    \consists "Ambitus_engraver"
  } \relative c' {
    \voiceTwo
    es4 f g as
    b1
  }
>>
```



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
  }
>>
```



Siehe auch

Glossar: Abschnitt “Tonumfang” 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.
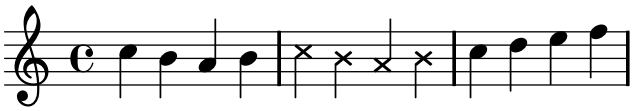
1.1.4 Notenköpfe

Dieser Abschnitt zeigt, wie man Notenköpfe ändern kann.

Besondere Notenköpfe
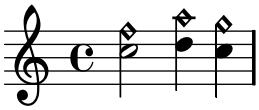
Notenköpfe können verändert werden:

```
c4 b a b
\override NoteHead #'style = #'cross
c4 b a b
\revert NoteHead #'style
c4 d e f
```



Es gibt einen definierten Befehl für die Raute, der nur innerhalb von Akkorden benutzt werden kann:

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



Alle möglichen Notenkopf-Stile finden sich in Abschnitt B.7 [Notenkopfstile], Seite 366.

Siehe auch

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

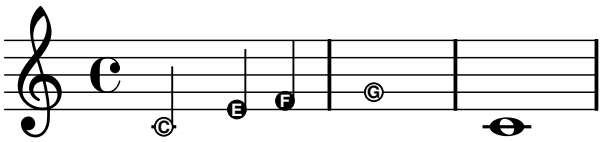
Notationsreferenz: Abschnitt B.7 [Notenkopfstile], Seite 366, [Noten mit Akkorden], Seite 109.

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*.

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 4.2.1 [Die Notensystemgröße einstellen], Seite 332.

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



Vordefinierte Befehle

`\easyHeadsOn`, `\easyHeadsOff`.

Siehe auch

Notationsreferenz: Abschnitt 4.2.1 [Die Notensystemgröße einstellen], Seite 332.
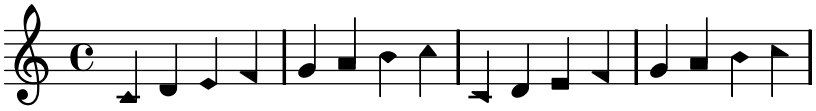
Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

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*.

Notenköpfe mit besonderen Formen

In dieser Notation haben die Notenköpfe eine Form, die ihrer harmonischen Funktion innherhalb der Tonleiter entspricht. Die Notation war sehr beliebt in amerikanischen Liederbüchern des 19. Jahrhunderts. Auf diese Weise können die Formen benutzt werden:

```
\aikenHeads
c, d e f g a b c
\sacredHarpHeads
c, d e f g a b c
```



Die unterschiedlichen Formen richten sich nach der Stufe in der Skala, wobei der Grundton der Skala aus dem `\key`-Befehl entnommen wird.

Vordefinierte Befehle

`\aikenHeads`, `\sacredHarpHeads`.

Ausgewählte Schnipsel

Notenkopfstile besierend auf der Tonleiterstufe erstellen

Die `shapeNoteStyles`-(NotenFormenStile)-Eigenschaft kann benutzt werden, um verschiedene Notenstile für jeden Schritt der Tonleiter zudefinieren (vorgegeben von der Tonart oder der ‚tonic' (Tonika)-Eigneschaft. 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.
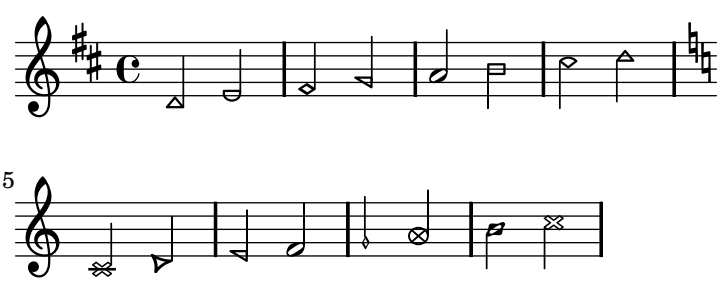
```
\layout { ragged-right = ##t }

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

\score {
  \new Staff {
    \transpose c d
    \relative c' {
      \set shapeNoteStyles = #'#(do re mi fa
                                 #f la ti)
      \Frament
    }

    \break

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



Alle Notenkopfstile finden sich in Abschnitt B.7 [Notenkopfstile], Seite 366.

Siehe auch

Schnipsel: Abschnitt “Pitches” in *Schnipsel*.

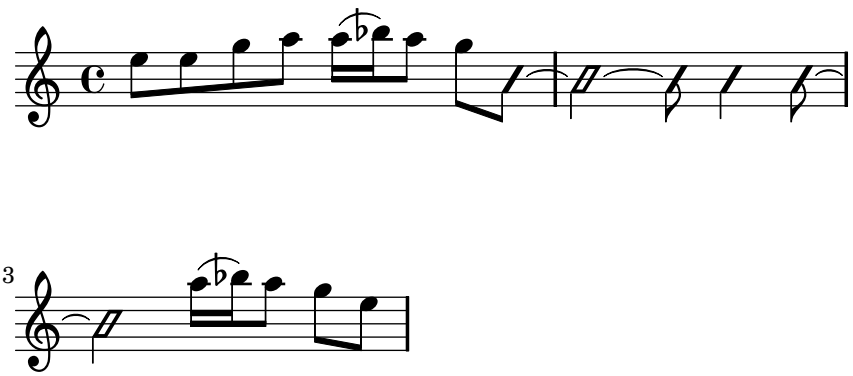
Notationsreferenz: Abschnitt B.7 [Notenkopfstile], Seite 366.

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*.

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"
} {
  e8 e g a a16( bes) a8 g
  \improvisationOn
  e8 ~
  e2 ~ e8 f4 f8 ~
  f2
  \improvisationOff
  a16( bes) a8 g e
}
```



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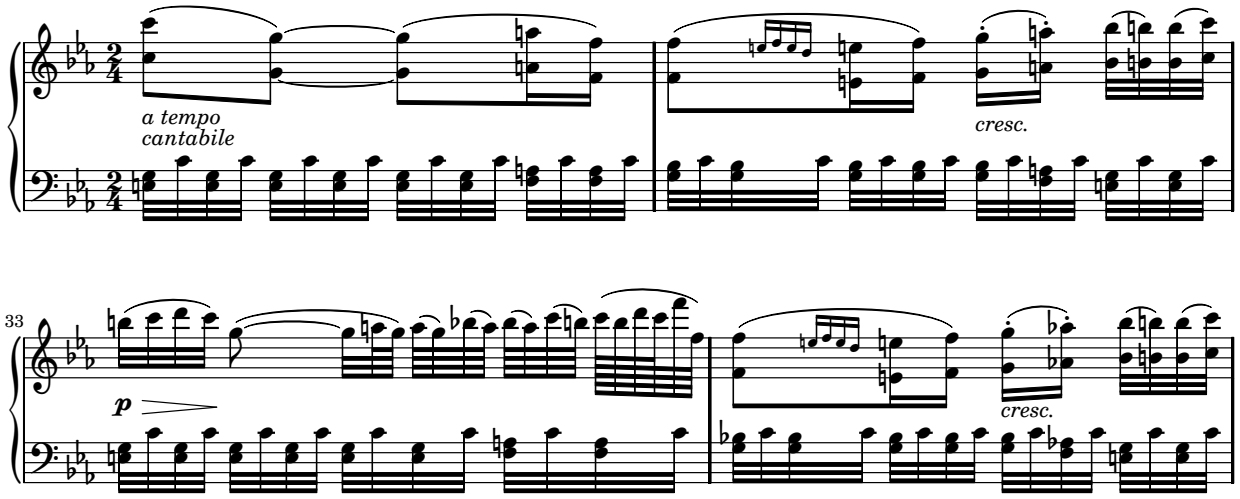
`\improvisationOn`, `\improvisationOff`.

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*.

1.2 Rhythmus



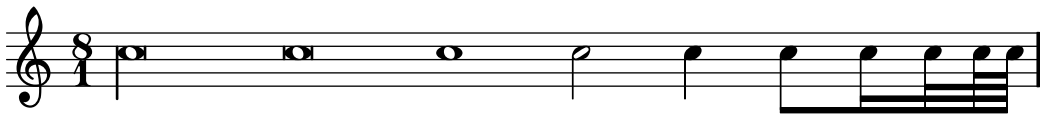
Dieser Abschnitt erklärt die Eingabe von Rhythmen, Pausen, Dauern, Bebalkung und Takten.

1.2.1 Rhythmen eingeben

Tondauern
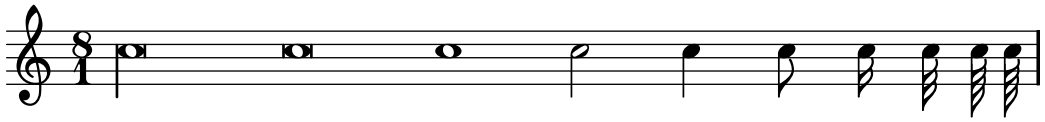
Notenlängen (Dauern) werden durch Zahlen und Punkte notiert: Dauern werden als reziproke Werte geschrieben. Zum Beispiel wird eine Viertelnote mit `4` notiert (weil sie eine 1/4-Note ist), eine halbe Note mit `2` (weil sie eine 1/2-Note ist). Noten, die länger als eine Ganze sind, müssen mit `\longa` (für die Longa, also vier Ganze) und `\breve` (für die Brevis, auch Doppelganze genannt) notiert werden.

```
\time 8/1
c\longa c\breve c1 c2
c4 c8 c16 c32 c64 c64
```



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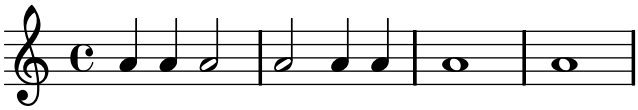
```
\time 8/1
\autoBeamOff
c\longa c\breve c1 c2
c4 c8 c16 c32 c64 c64
```



Eine Note mit der vierfacheen Dauer einer Brevis kann mit dem Befehl `\maxima` eingegeben werden, aber ihre Darstellung ist nur für die Alte Musiknotation unterstützt. Zu Einzelheiten siehe Abschnitt 2.8 [Notation von alter Musik], Seite 279.
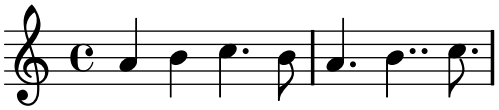
Wenn die Dauer hinter einer Notenbezeichnung nicht angegeben ist, wird die Dauer der vorhergehenden Note eingesetzt. Der Standardwert für die erste Note ist eine Viertel.

```
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.

```
a4 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 [Bindebögen], Seite 36.

Wie den Silben von Gesangtext eigene Dauern zu gewiesen werden können und wie man sie an den Noten ausrichtet ist erklärt in Abschnitt 2.1 [Notation von Gesang], Seite 187.

Optional können Noten streng proportional nach ihrer exakten Dauer gesetzt werden. Zu Einzelheiten hierzu und weiteren Einstellungen für proportionale Notation siehe Abschnitt 4.5.5 [Proportionale Notation], Seite 333.

Punkte werden normalerweise nach oben verschoben, damit sie die Notenlinien nicht berühren. Fertige Befehle können eingesetzt werden, um eine bestimmte Richtung manuell zu erzwingen, zu Einzelheiten siehe Abschnitt 5.4.2 [Richtung und Platzierung], Seite 335.

Vordefinierte Befehle

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

Siehe auch

Glossar: Abschnitt “Brevis” in *Glossar*, Abschnitt “Longa” in *Glossar*, Abschnitt “Notenwert” in *Glossar*, Abschnitt “Dauerbezeichnungen für Noten und Pausen” in *Glossar*.

Notationsreferenz: [Automatische Balken], Seite 55, [Bindebögen], Seite 36, Abschnitt 1.2.1 [Rhythmen eingeben], Seite 31, Abschnitt 1.2.2 [Pausen eingeben], Seite 38, Abschnitt 2.1 [Notation von Gesang], Seite 187, Abschnitt 2.8 [Notation von alter Musik], Seite 279, Abschnitt 4.5.5 [Proportionale Notation], Seite 333.

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.
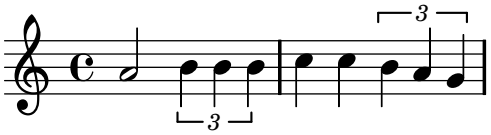
Andere rhythmische Aufteilungen

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.
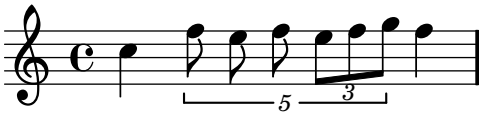
```
a2 \times 2/3 { b4 b b }
c4 c \times 2/3 { b4 a g }
```



Die automatische Platzierung der Triolenklammer über oder unter den Noten kann manuell geändert werden mit definierten Befehlen, siehe Abschnitt 5.4.2 [Richtung und Platzierung], Seite 335.

N-tolen können ineinander geschachtelt werden:

```
\autoBeamOff
c4 \times 4/5 { f8 e f \times 2/3 { 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 [Tondauern skalieren], Seite 35.

Vordefinierte Befehle

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

Ausgewählte Schnipsel
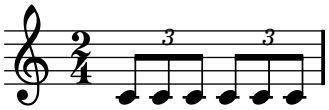
Mehrere Triolen notieren, aber nur einmal `\times` *benutzen*

Die Eigenschaft `tupletSpannerDuration` bestimmt, wie lange jede der N-tolen innerhalb der Klammern nach dem `\times`-Befehl dauert. Auf diese Art können etwa viele Triolen nacheinander mit nur einem `\times`-Befehl geschrieben werden.

Im Beispiel sind zwei Triolen zu sehen, obwohl `\times` nur einmal geschrieben wurde.

Mehr Information über `make-moment` gibt es in "Verwaltung der Zeiteinheiten".

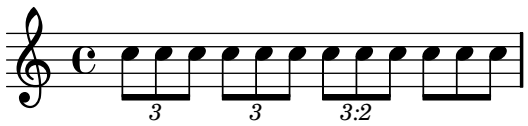
```
\relative c' {
  \time 2/4
  \set tupletSpannerDuration = #(ly:make-moment 1 4)
  \times 2/3 { c8 c c c c c }
}
```



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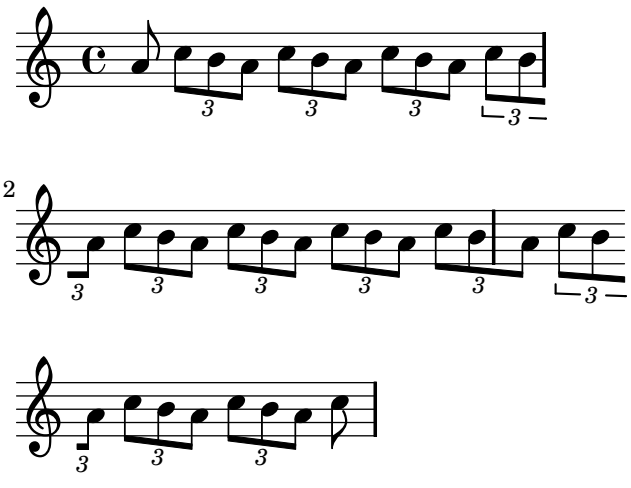
```
\relative c'' {
  \times 2/3 { c8 c c } \times 2/3 { c8 c c }
  \override TupletNumber #'text = #tuplet-number::calc-fraction-text
  \times 2/3 { c8 c c }
  \override TupletNumber #'stencil = ##f
  \times 2/3 { c8 c c }
}
```



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
    % Zeilenumbrüche innerhalb von N-tolen zulassen
    \remove "Forbid_line_break_engraver"
    % Balken können an Umbrüchen gebrochen werden
    \override Beam #'breakable = ##t
  }
}
\relative c'' {
  a8
  \repeat unfold 8 { \times 2/3 { c[ b a] } }
  % Manuellen Zeilenumbruch in N-tole
  \times 2/3 { c[ b \bar "" \break a] }
  \repeat unfold 2 { \times 2/3 { c[ b a] } }
  c8
}
```



Siehe auch

Glossar: Abschnitt “Triole” in *Glossar*, Abschnitt “N-tole” in *Glossar*, Abschnitt “Polymetrie” in *Glossar*.

Handbuch zum Lernen: Abschnitt “Tweaking methods” in *Handbuch zum Lernen*.

Notationrreferenz: [Verwaltung der Zeiteinheiten], Seite 82, [Tondauern skalieren], Seite 35, Abschnitt 5.3.4 [Der tweak-Befehl], Seite 335, [Polymetrische Notation], Seite 49.

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
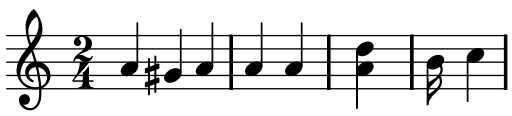
Wenn die erste Noten eines Systems ein Vorschlag (eine Verzierung) ist, die von einer N-tole gefolgt ist, muss der Vorschlag vor den `\times`-Befehl gesetzt werden um Fehler zu vermeiden. Überall sonst können Vorschläge innerhalb von N-tolen gesetzt werden.

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“.

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

```
\time 2/4
% Dauern zu Triolen verändern
a4*2/3 gis4*2/3 a4*2/3
% Normale Tondauern
a4 a4
% Dauer der Akkorde verdoppeln
<a d>4*2
% Dauer einer Vietel, erscheint vie eine Sechszehntel
b16*4 c4
```
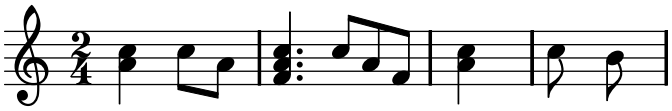


Die Dauer von unsichtbaren Noten kann auch mit einem Faktor beeinflusst werden. Das ist sinnvoll, wenn man viele Takte überspringen muss, etwa `s1*23`.

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. Die Leerzeichen um den Punkt im Beispiel sind notwendig. Hier ein Beispiel, das zeigt, wie Noten komprimiert und ausgedehnt werden kann:

```
\time 2/4
% Normale Tondauern
<c a>4 c8 a
% Noten mit 2/3 skalieren
```

```
\scaleDurations #'(2 . 3) {
  <c a f>4. c8 a f
}
% Noten *2 skalieren
\scaleDurations #'(2 . 1) {
  <c' a>4 c8 b
}
```



Eine Anwendung für diesen Befehl ist polymetrische Notation, siehe [Polymetrische Notation], Seite 49.

Siehe auch

Notationsreferenz: [Andere rhythmische Aufteilungen], Seite 32, [Unsichtbare Pausen], Seite 40, [Polymetrische Notation], Seite 49.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Bindebögen

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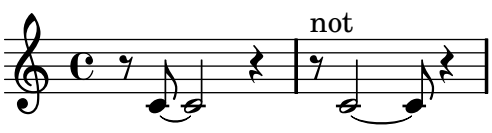
> **Achtung:** Bindebögen dürfen nicht mit Legatobögen verwechselt werden, durch die die Vortragsart bezeichnet wird, noch mit Phrasierungsbögen, die musikalische Phrasen anzeigen. Ein Bindebogen ist nur eine Art, die Tondauer zu verlängern, ähnlich etwa wie die Punktierung.

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

```
a2 ~ a
```



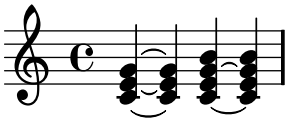
Bindebögen werden eingesetzt, wenn die Note entweder über eine Taktlinie hinüberreicht, oder wenn die entsprechende Dauer der Note nicht mit Punktierung erreicht werden kann. Bindebögen sollten auch benutzt werden, wenn Notenwerte über die inneren Unterteilungen von Takten hinüberreichen:



Wenn viele Noten über Taktlinien gebunden werden müssen, kann es einfacher sein, automatische Notenaufteilung einzustellen, wie beschrieben in [Automatische Aufteilung von Noten], Seite 52. Mit diesem Mechanismus werden lange Noten automatisch aufgeteilt, wenn sie über Taktgrenzen reichen.
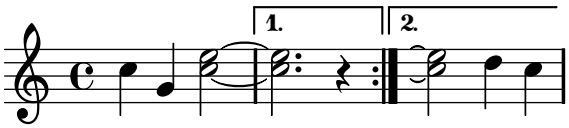
Wenn ein Bindebogen an einen Akkord gehängt wird, werden alle Noten dieses Akkordes übergebunden. Wenn kein Notenkopf passt, wird auch kein Bogen erzeugt. Noten in Akkorden können auch einzeln übergebunden werden, indem sie innerhalb des Akkordes hinter die entsprechende Note geschrieben werden.
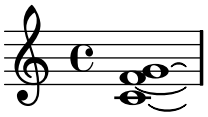
```
<c e g> ~ <c e g>
<c~ e g~ b> <c e g b>
```



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 {
  % Erste Weiche: nächste Note ist normal gebunden
  { <c e>2. r4 }
  % Zweite Weiche: folgende Note hat einen Wiederholten Bindebogen
  { <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
```



Die vertikale Position von Bindebögen kann kontrolliert werden, siehe die vordefinierten Befehle unten oder für Einzelheiten Abschnitt 5.4.2 [Richtung und Platzierung], Seite 335.

Durchgehende, gepunktete oder gestrichelte Bindebögen können spezifiziert werden, siehe die vordefinieren Befehle.

Vordefinierte Befehle
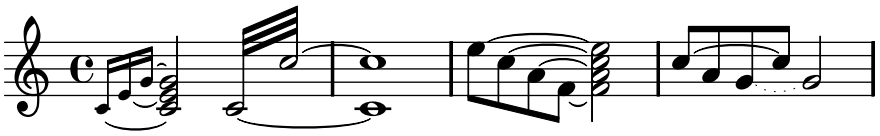
`\tieUp`, `\tieDown`, `\tieNeutral`, `\tieDotted`, `\tieDashed`, `\tieSolid`.

Ausgewählte Schnipsel

Überbingungen für Arpeggio genutzen

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' {
  <c e g>2 ~ <c e g>
  \override TieColumn #'tie-configuration =
    #'((0.0 . 1) (-2.0 . 1) (-4.0 . 1))
  <c e g> ~ <c e g>
}
```



Siehe auch

Glossar: Abschnitt “Bindebogen” in *Glossar*, Abschnitt “Laissez-vibrer-Bögen” in *Glossar*.

Notationsreferenz: [Automatische Aufteilung von Noten], Seite 52.

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.

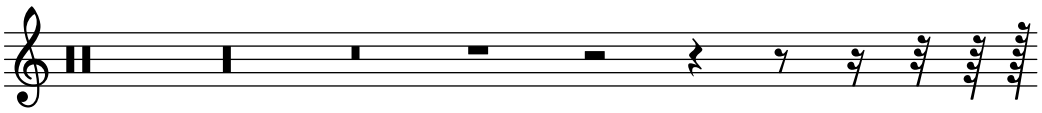
1.2.2 Pausen eingeben

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

Pausen

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

```
\new Staff {
  % Die zwei Linien sind nur zur Verschönerung
  \time 16/1
  \override Staff.TimeSignature #'stencil = ##f
  % Setze Maxima-Pause, gleich vier Breven
  r\maxima
  % Setze Longa-Pause, gleich zwei Breven
  r\longa
  % Setze Breve-Pause
  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 [Ganztaktige Pausen], Seite 41.

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.

```
a4\rest d4\rest
```



Pausenstile

Pausen können in verschiedenen Stilen dargestellt werden.

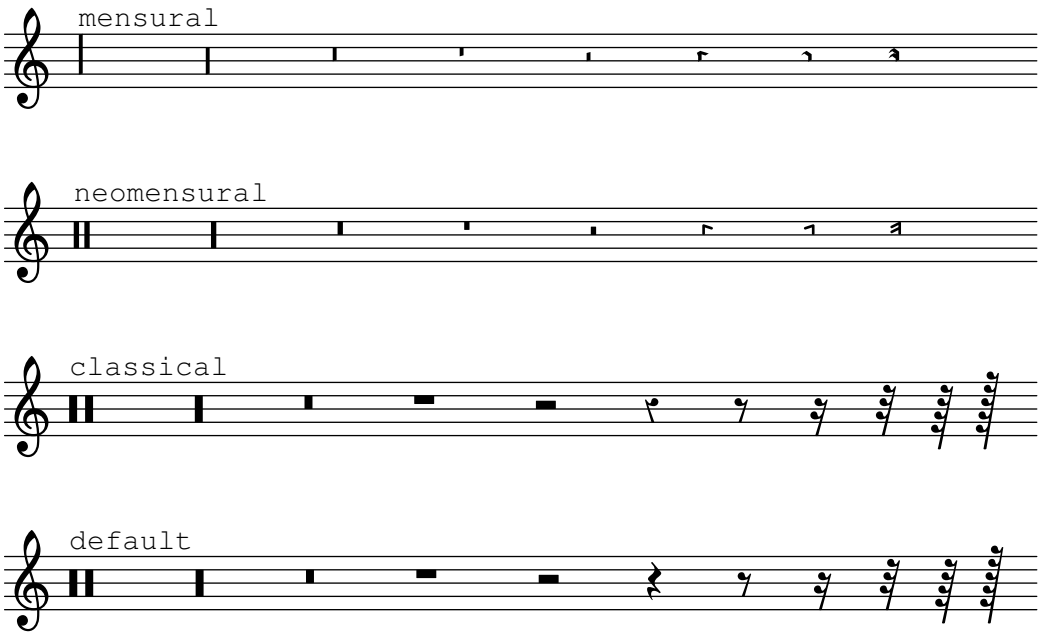
```
\layout {
  indent = 0.0
  \context {
    \Staff
    \remove "Time_signature_engraver"
  }
}

\new Staff \relative c {
  \cadenzaOn
  \override Staff.Rest #'style = #'mensural
  r\maxima^\markup \typewriter { mensural }
  r\longa r\breve r1 r2 r4 r8 r16 s32 s64 s128 s128
  \bar ""
```

```
  \override Staff.Rest #'style = #'neomensural
  r\maxima^\markup \typewriter { neomensural }
  r\longa r\breve r1 r2 r4 r8 r16 s32 s64 s128 s128
  \bar ""

  \override Staff.Rest #'style = #'classical
  r\maxima^\markup \typewriter { classical }
  r\longa r\breve r1 r2 r4 r8 r16 r32 r64 r128 s128
  \bar ""

  \override Staff.Rest  #'style = #'default
  r\maxima^\markup \typewriter { default }
  r\longa r\breve r1 r2 r4 r8 r16 r32 r64 r128 s128
}
```



Siehe auch

Notationsreferenz: [Ganztaktige Pausen], Seite 41.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

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

Bekannte Probleme und Warnungen

Es gibt keine gundlegende 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).

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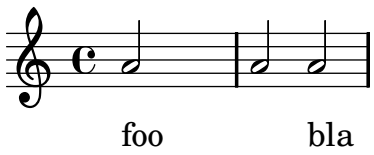
Eine unsichtbare Pause (auch als „skip“ oder Überspringung bezeichnet) kann wie eine Note eingegeben werden, die Notationsbezeichnung ist `s`.

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



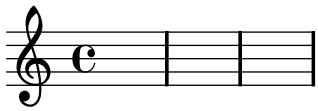
Die s-Syntax steht nur im Noten- oder Akkordmodus zur Verfügung. In anderen Situationen, z. B. innerhalb eines Liedtextes, muss `\skip` benutzt werden. `\skip` benötigt eine explizite Dauerangabe.

```
<<
  {
    a2 \skip2 a2 a2
  }
  \new Lyrics {
    \lyricmode {
      foo2 \skip 1 bla2
    }
  }
>>
```



Die Überspringung 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.

```
% Gültiger Eingabecode, aber tut nichts
\skip 1 \skip1 \skip 1
```

Siehe auch
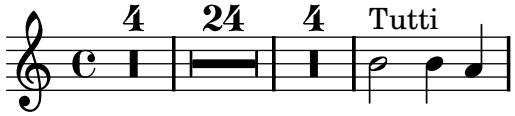
Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

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

Ganztaktige Pausen

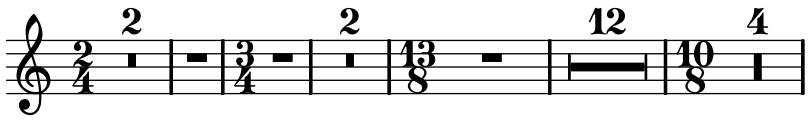
Pausen für einen oder mehrere ganze Takte werden wie Noten eingegeben, wobei die Bezeichnung ein Großbuchstabe `R` ist:

```
% Pausentakte zu einem einzelnen Takt zusammegezogen
\compressFullBarRests
R1*4
R1*24
R1*4
b2^"Tutti" b4 a4
```
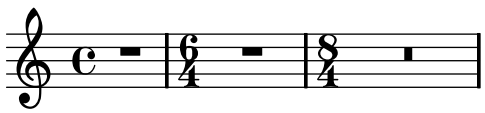


Die Dauer von Ganztaktpausen wird genauso angegeben wie die Dauer von Noten. Die Dauer einer Ganztaktpause muss immer eine ganze Anzahl an Taktlängen sein, weshalb Punktierungen und Brüche recht häufig eingesetzt werden müssen.

```
\compressFullBarRests
\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 |
```



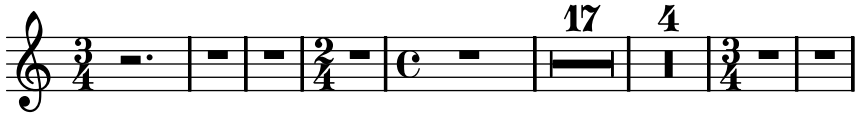
Eine Ganztaktpause wird abhängig von der Taktart entweder als Ganze oder Brevis-Pause gesetzt, zentriert im Takt.

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



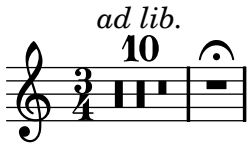
In den Standardeinstellungen werden mehrtaktige Pausen ausgeschrieben gesetzt, sodass sie die entsprechende Anzahl von Takten einnehmen. Alternativ kann die mehrtaktige Pause aber auch nur in einem Takt angezeigt werden, der ein Mehrtaktpausensymbol geinhaltet, wobei die Anzahl der Takte der Pausendauer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

```
% Standart
\time 3/4 r2. | R2.*2 |
\time 2/4 R2 |
\time 4/4
% Pausentakte zu einem einzelnen Takt zusammegezogen
\compressFullBarRests
r1 | R1*17 | R1*4 |
% Pausentakte   ausklappen
\expandFullBarRests
\time 3/4
R2.*2 |
```



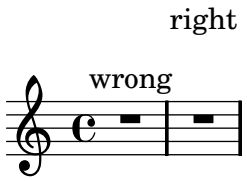
Textbeschriftung kann Mehrtaktpausen mit `\markup` hinzugefügt werden. Ein vordefinierte Befehl `\fermataMarkup` fügt eine Fermate ein.

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



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

```
% Das funktioniert nicht, weil die falsche Objektbezeichnung angegeben ist
\override TextScript #'padding = #5
R1^"wrong"
% Das stimmt und funktioniert
\override MultiMeasureRestText #'padding = #5
R1^"right"
```



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

Vordefinierte Befehle

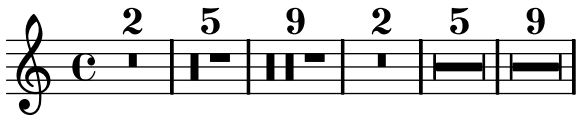
`\textLengthOn`, `\textLengthOff`, `\fermataMarkup`, `\compressFullBarRests`, `\expandFullBarRests`.

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:

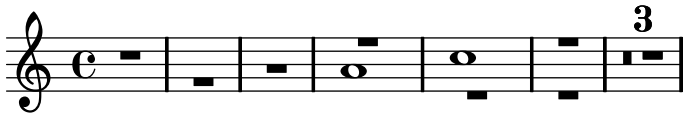
```
\relative c'' {
  \compressFullBarRests
  R1*2 | R1*5 | R1*9
  \override MultiMeasureRest #'expand-limit = #3
  R1*2 | R1*5 | R1*9
}
```



Positionierung von Ganztaktpausen

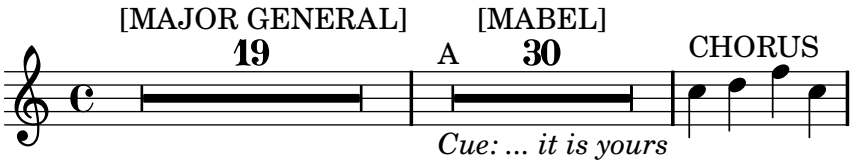
Anders als bei normalen Pausen gibt es keinen direkten Befehl, um die vertikale Position von Ganztaktpausen zu beeinflussen, indem man sie an eine Tonhöhe anhängt. In polyphoner Notation wird aber dennoch die Position der Pausen von geraden und ungeraden Stimmen voneinander unterschieden. Die Position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

```
\relative c'' {
  % Ganztaktpausen werden unter die zweite Linie gesetzt
  R1
  % Sie können mit override verschoben werden
  \override MultiMeasureRest #'staff-position = #-2
  R1
  % Ein Wert von 0 ist die Standardposition
  % hiermit wird die Pause zur Mittellinie verschoben
  \override MultiMeasureRest #'staff-position = #-0.01
  R1
  % Ganztaktpausen in Stimmen mit ungerader Zahl sind unter der obersten Linie
  << { R1 } \\ { a1 } >>
  % Ganztaktpausen in geraden Stimmen sind unter der untersten Linie
  << { c1 } \\ { R1 } >>
  % auch in leeren Takten bleichen sie getrennt
  << { R1 } \\ { R1 } >>
  % Hiermit werden sie zusammegefügt, auch wenn sie zwei Stimmen sind
  \compressFullBarRests
  <<
    \revert MultiMeasureRest #'staff-position
    { R1*3 }
    \\
    \revert MultiMeasureRest #'staff-position
    { R1*3 }
  >>
}
```



Markups attached to a multi-measure rest will be centered above or below it. Long markups attached to multi-measure rests do not cause the measure to expand. To expand a multi-measure rest to fit the markup, use a spacer rest with an attached markup before the multi-measure rest:

```
\compressFullBarRests
\textLengthOn
s1*0^\markup { [MAJOR GENERAL] }
R1*19
s1*0_\markup { \italic { Cue: ... it is yours } }
s1*0^\markup { A }
R1*30^\markup { [MABEL] }
\textLengthOff
c4^\markup { CHORUS } d f c
```



Note that the spacer rest causes a bar to be inserted. Text attached to a spacer rest in this way is left-aligned to the position where the note would be placed in the measure, but if the measure length is determined by the length of the text, the text will appear to be centered.

Siehe auch

Glossar: Abschnitt “Pausen über mehrere Takte” in *Glossar*.

Notationsreferenz: [Tondauern], Seite 31, Abschnitt 1.8 [Text], Seite 163, Abschnitt 1.8.2 [Text formatieren], Seite 170, [Textarten], Seite 163.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

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

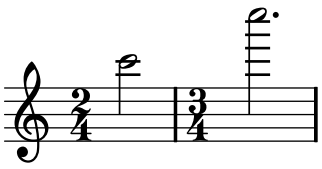
Ganztaktpausen werden bei der Vermeidung von Zusammenstößen nicht berücksichtigt.

1.2.3 Rhythmen anzeigen lassen

Taktangabe
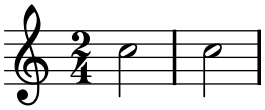
Taktangaben könne wie folgt erstellt werden.
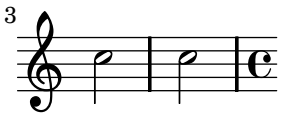
```
\time 2/4 c'2
\time 3/4 c'2.
```
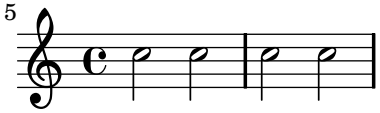


Taktangaben werden zu Beginn eines Stückes gesetzt und immer dann, wenn sich die Taktart ändert. Wenn eine Änderung am Ende einer Zeile geschieht, wird eine warnende Taktangabe am Ende der Zeile ausgegeben. Dieses Verhalten kann verändert werden, siehe Abschnitt 5.4.6 [Sichtbarkeit von Objekten], Seite 336.

```
\time 2/4
c2 c
\break
c c
\break
\time 4/4
c c c c
```
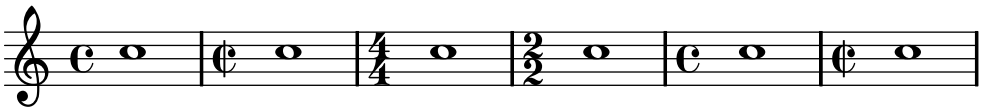






Das Symbol für die Taktarten 2/2 und 4/4 kann in ein Zahlensymbol umgewandelt werden:

```
% Standardstil
\time 4/4 c1
\time 2/2 c1
% Wechsel zum nummerierten Stil
\numericTimeSignature
\time 4/4 c1
\time 2/2 c1
% Zurück zum Standard
\defaultTimeSignature
\time 4/4 c1
\time 2/2 c1
```



Symbole für Modus und Proprietas der mensuralen Notation werden behandelt unter [Taktangaben Alter Musik], Seite 284.

Vordefinierte Befehle

`\numericTimeSignature`, `\defaultTimeSignature`.

Ausgewählte Schnipsel
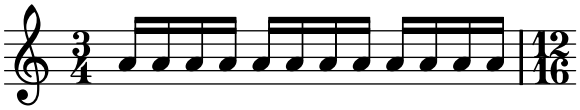
Changing the time signature without affecting the beaming

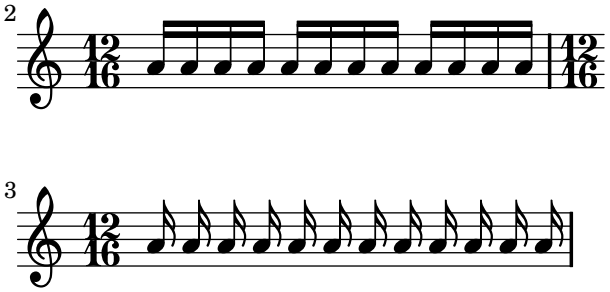
The `\time` command sets the properties `timeSignatureFraction`, `beatLength`, `beatGrouping` and `measureLength` in the `Timing` context, which is normally aliased to `Score`. Changing the value of `timeSignatureFraction` causes the new time signature symbol to be printed without changing any of the other properties:

```
\relative c'' {
  \time 3/4
  a16 a a a a a a a a a a a

  % Taktartsymbol ändern, aber 3/4-Balken
  % wegen unveränderter Taktartaufteilung behalten
  \set Score.timeSignatureFraction = #'(12 . 16)
  a16 a a a a a a a a a a a

  \time 12/16
  % 3/4-Balken lösen, da \time geändert wurde
  a16 a a a a a a a a a a a
}
```
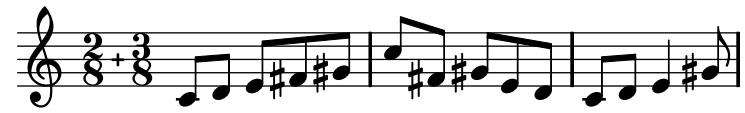




Zusammengesetzte Taktarten

Ungerade Taktarten werden (wie etwa "5/8") werden oft als zusammengesetzte Taktarten interpretiert (bspw. "3/8 + 2/8"), in welchen zwei oder mehr Teiltakte unterschieden werden. LilyPond kann derartige Noten produzieren, indem entsprechende Taktarten gesetzt werden und die automatische Bebalkung angepasst wird.

```
#(define ((compound-time one two num) grob)
  (grob-interpret-markup grob
    (markup #:override '(baseline-skip . 0) #:number
      (#:line (
          (#:column (one num))
          #:vcenter "+"
          (#:column (two num))))
      )))

\relative c' {
  \override Staff.TimeSignature #'stencil = #(compound-time "2" "3" "8")
  \time 5/8
  #(override-auto-beam-setting '(end 1 8 5 8) 1 4)
  c8 d e fis gis
  c8 fis, gis e d
  c8 d e4 gis8
}
```



Siehe auch

Glossar: Abschnitt “Taktangabe” in *Glossar*

Notationsreferenz: [Taktangaben Alter Musik], Seite 284, [Verwaltung der Zeiteinheiten], Seite 82.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

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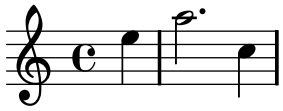
Auftakte

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

`\partial` *Dauer*

wobei *Dauer* die rhythmische Länger der Noten darstellt, die vor dem ersten vollständigen Takt gesetzt werden sollen:

```
\partial 4 e4 |
a2. c,4 |
```



Das wird intern übersetzt nach:

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

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

Siehe auch

Glossar: Abschnitt “Auftakt” in *Glossar*.

Notationsreferenz: [Verzierungen], Seite 77.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

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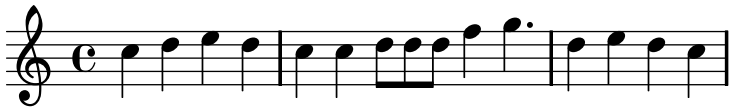
Bekannte Probleme und Warnungen

`\partial` ist nur für den Anfang eines Stückes vorgesehen. Wenn der Befehl innerhalb eines Stückes verwendet wird, können seltsame Warnungen auftreten.

Musik ohne Metrum
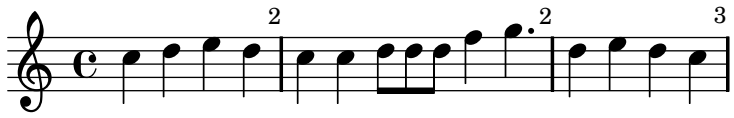
Taktlinien und Taktzahlen werden automatisch erzeugt. Für Musik ohne Metrum hingegen (etwa Kadenzen) ist das jedoch nicht erwünscht. Mit den Befehlen `\cadenzaOn` und `\cadenzaOff` kann dieses Verhalten ausgeschaltet und wieder angeschaltet werden.

```
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, als ob es die Kadenz nicht gegeben hätte:

```
% Alle Taknummern anzeigen
\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
```



Vordefinierte Befehle

`\cadenzaOn`, `\cadenzaOff`.

Siehe auch

Glossar: Abschnitt “Kadenz” in *Glossar*.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Bekannte Probleme und Warnungen

LilyPond fügt Zeilen- und Seitenumbrüche nur an einer Taktlinie ein. Wenn die Kadenz nicht vor einem Umbruch endet, müssen Sie selber unsichtbare Taktlinien mit

```
\bar ""
```

einfügen, um anzuzeigen, wo umgebrochen werden darf.

Polymetrische Notation

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

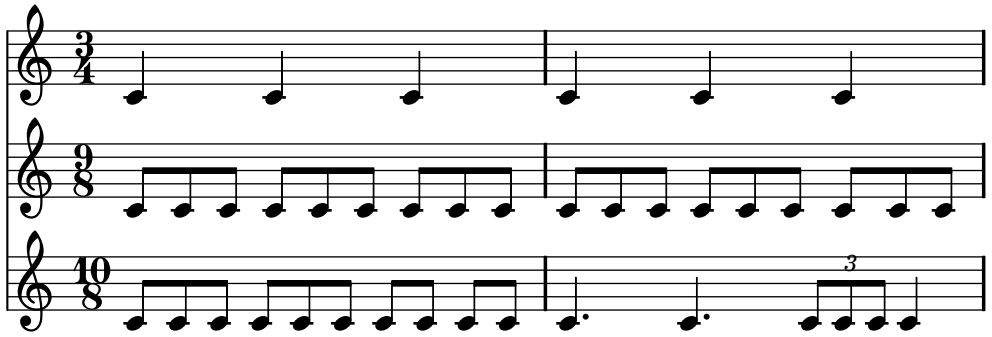
Systeme mit unterschiedlichen Taktarten, gleiche Taktlänge

Diese Art der Notation kann erstellt werden, indem für jedes System eine identische Taktart eingestellt wird, aber manuell für jeden Takt durch Einstellung von `timeSignatureFraction` auf den gewünschten Bruch geändert und dann die Länge der Noten entsprechenden skaliert wird, siehe auch [Taktangabe], Seite 45. Die Skalierung geschieht mit dem Befehl `\scaleDurations`, der auf ähnliche Weise wie `\times` benutzt wird, aber keine Klammer über den Noten ausgibt. Siehe auch [Tondauern skalieren], Seite 35.

In diesem Beispiel werden Noten mit den Taktarten 3/4, 9/8 und 10/8 parallel benutzt. Im zweiten System werden die gezeigten Dauern mit 2/3 multipliziert, da 2/3x9/8=3/4, und im dritten System werden die gezeigten Dauern mit 3/5 multipliziert, da 3/5x10/8=3/4. Oft wird es nötig sein, Balken manuell zu setzen, weil die Skalierung sich auch auf die automatische Bebalkung auswirkt.

```
\relative c' <<
  \new Staff {
    \time 3/4
    c4 c c |
    c c c |
  }
  \new Staff {
    \time 3/4
    \set Staff.timeSignatureFraction = #'(9 . 8)
    \scaleDurations #'(2 . 3)
    \repeat unfold 6 { c8[ c c] }
  }
  \new Staff {
    \time 3/4
    \set Staff.timeSignatureFraction = #'(10 . 8)
    \scaleDurations #'(3 . 5) {
      \repeat unfold 2 { c8[ c c] }
      \repeat unfold 2 { c8[ c] } |
      c4. c4. \times 2/3 { c8[ c c] } c4
```

```
    }
  }
>>
```



Systeme mit unterschiedlichen Taktarten, unterschiedliche Taktlänge

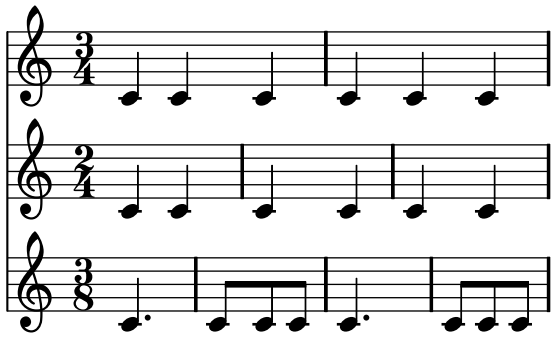
Jedes System kann auch eine eigene unabhängige Taktart erhalten. Dazu muss der `Timing_translator` und der `Default_bar_line_engraver` in den `Staff`-Kontext verschoben werden.

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

% Jetzt hat jedes System eine eigene Taktart

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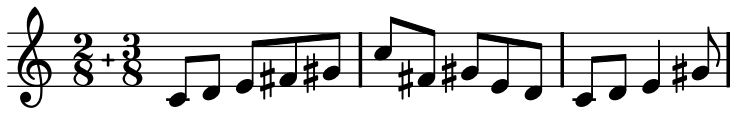
>>



Ausgewählte Schnipsel

Zusammengesetzte Taktarten

Ungerade Taktarten werden (wie etwa "5/8") werden oft als zusammengesetzte Taktarten interpretiert (bspw. "3/8 + 2/8"), in welchen zwei oder mehr Teiltakte unterschieden werden. LilyPond kann derartige Noten produzieren, indem entsprechende Taktarten gesetzt werden und die automatische Bebalkung angepasst wird.

```
#(define ((compound-time one two num) grob)
  (grob-interpret-markup grob
    (markup #:override '(baseline-skip . 0) #:number
      (#:line (
          (#:column (one num))
          #:vcenter "+"
          (#:column (two num))))
      )))

\relative c' {
  \override Staff.TimeSignature #'stencil = #(compound-time "2" "3" "8")
  \time 5/8
  #(override-auto-beam-setting '(end 1 8 5 8) 1 4)
  c8 d e fis gis
  c8 fis, gis e d
  c8 d e4 gis8
}
```



Siehe auch

Glossar: Abschnitt “Polymetrie” in *Glossar*, Abschnitt “Polymetrische Taktangabe” in *Glossar*, Abschnitt “Metrum” in *Glossar*.

Notationreferenz: [Taktangabe], Seite 45, [Tondauern skalieren], Seite 35.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

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

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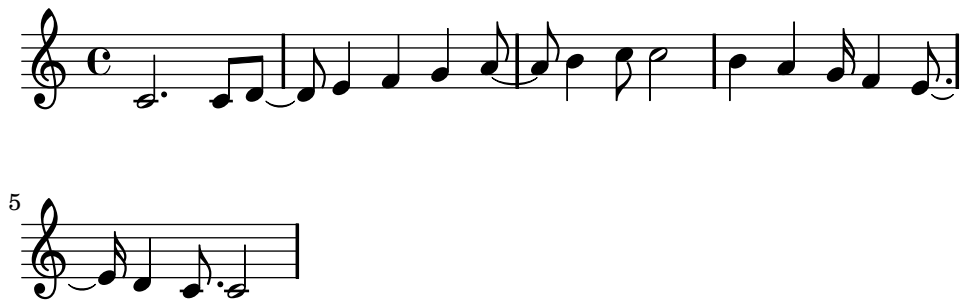
Wenn unterschiedliche Taktarten paralell benutzt werden, werden Noten auf demselben musikalischen Moment horizontal auf die gleiche Position gesetzt. Die unterschiedlichen Taktlinien führen allerdings dazu, dass die Noten nicht ganz so regelmäßig gesetzt werden, wie es ohne unterschiedliche Taktarten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

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. Im nächsten Beispiel werden Noten, die über die Taktlinie dauern, aufgeteilt und übergebunden.

```
\new Voice \with {
  \remove "Note_heads_engraver"
  \consists "Completion_heads_engraver"
}

{ c2. c8 d4 e f g a b c8 c2 b4 a g16 f4 e d c8. c2 }
```



Dieser Engraver teilt alle Noten auf, die über eine Taktlinie dauern und fügt 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 "Bindebogen" in *Glossar*

Handbuch zum Lernen: Abschnitt "Engravers explained" in *Handbuch zum Lernen*, Abschnitt "Adding and removing engravers" 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 "Forbid_line_break_engraver" in *Referenz der Interna*.

Bekannte Probleme und Warnungen

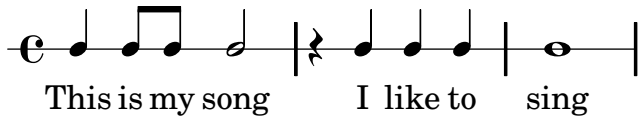
Nicht alle Notenwerte (besonders wenn sie andere rhythmische Aufteilungen beinhalten) können exakt durch normale Noten und Punktierungen wiedergegeben werden. Der Engraver setzt aber trotzdem keine Triolen etc.

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

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.
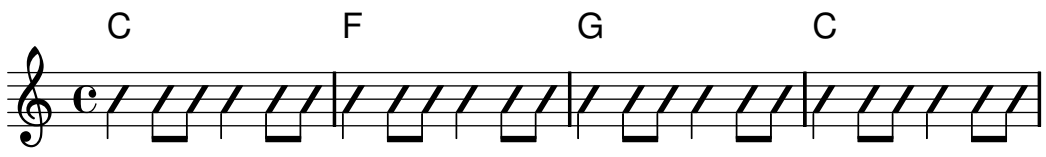
```
<<
  \new RhythmicStaff {
    \new Voice = "meinRhythmus" {
      \time 4/4
      c4 e8 f g2
      r4 g g f
      g1
    }
  }
  \new Lyrics {
    \lyricsto "meinRhythmus" {
      This is my song
      I like to sing
    }
  }
>>
```



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

Guitar strum rhythms

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

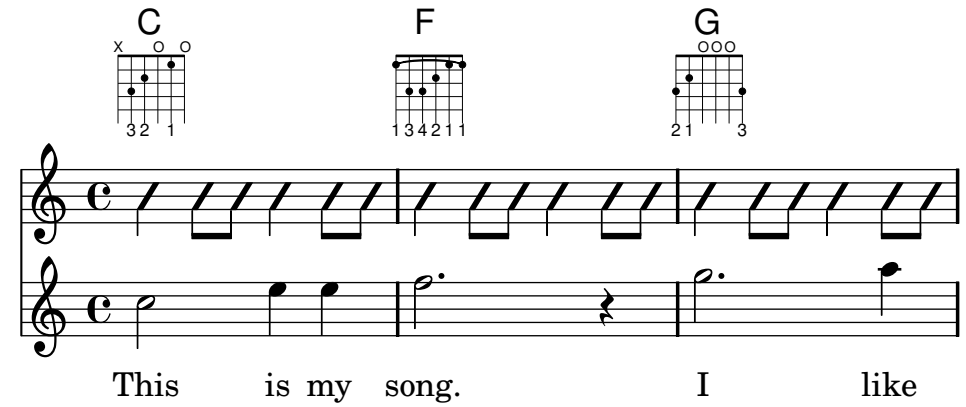
```
\include "predefined-guitar-fretboards.ly"
<<
  \new ChordNames {
    \chordmode {
      c1 f g c
    }
  }

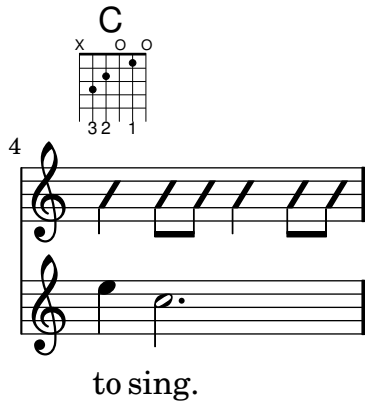
  \new FretBoards {
    \chordmode {
      c1 f g c
    }
  }


  \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 = "Melodie" {
    \relative c'' {
      \improvisationOff
      c2 e4 e4
      f2. r4
      g2. a4
      e4 c2.
    }
  }

  \new Lyrics {
    \lyricsto "Melodie" {
      This is my song.
      I like to sing.
    }
  }
>>
```





Siehe auch

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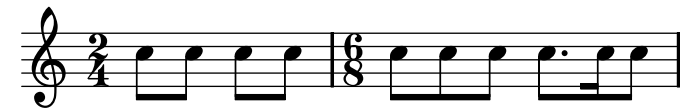
Referenz der Interna: Abschnitt “RhythmicStaff” in *Referenz der Interna*, Abschnitt “Pitch_squash_engraver” in *Referenz der Interna*.

1.2.4 Balken

Automatische Balken

LilyPond setzt Balken (engl. beam) automatisch.

```
\time 2/4 c8 c c c
\time 6/8 c c c c8. c16 c8
```
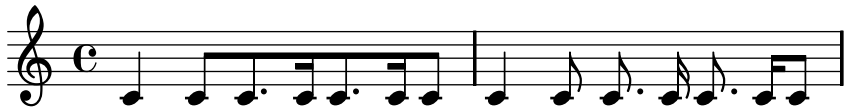


Wenn diese automatischen Entscheidungen nicht gut genug sind, können die Balken auch explizit eingegeben werden, siehe [Manuelle Balken], Seite 66. Es können auch bestimmte Balkenmuster, die sich vom Standard unterscheiden, definiert werden. Die Standard-Regeln für die gebräuchlichsten Taktarten sind in der Datei ‘`scm/auto-beam.scm`’ definiert. Wenn dort keine Balkenregeln für die bestimmte Balkendauer in der aktuellen Taktart vorhanden sind, wird die Bebalkung geregelt von drei Kontexteigenschaften: `measureLength`, `beatLength` und `beatGrouping`. Sowohl die Balkenregeln als auch diese Kontexteigenschaften können geändert werden, siehe [Einstellung von automatischen Balken], Seite 57.

Achtung: Wenn Balken eingesetzt werden, um Melismen in Gesang anzuzeigen, müssen die automatische Bebalkung ausgeschaltet werden mit dem Befehl `\autoBeamOff` und die Balken mit der Hand eingegeben werden.

Automatische Bebalkung kann mit dem Befehl `\autoBeamOff` aufgehoben werden und mit dem Befehl `\autoBeamOn` wieder eingeschaltet werden.

```
c4 c8 c8. c16 c8. c16 c8
\autoBeamOff
c4 c8 c8. c16 c8.
\autoBeamOn
c16 c8
```



Vordefinierte Befehle
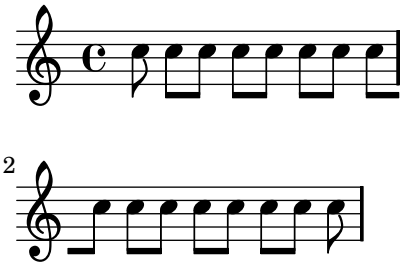
`\autoBeamOff`, `\autoBeamOn`.

Ausgewählte Schnipsel

Balken über Zeilenumbrüche

Zeilenumbrüche sind normalerweise während Balken verboten. Das kann geändert werden.
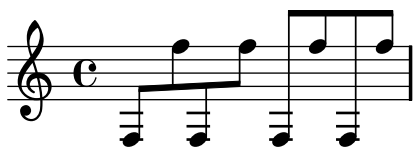
```
\relative c'' {
  \override Beam #'breakable = ##t
  c8 c[ c] c[ c] c[ c] c[ \break
  c8] c[ c] c[ c] c[ c] c
}
```



Balken für weit außeinander 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
}
```



Siehe auch

Notationsreferenz: [Manuelle Balken], Seite 66, [Einstellung von automatischen Balken], Seite 57.

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

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

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

Bekannte Probleme und Warnungen

Automatische Bebalkung von weit außeinander liegenden Noten (wobei der Balken als Knie erscheint) kann nicht mit versteckten Systemen benutzt werden. Siehe [Systeme verstecken], Seite 137.

Balken können mit Notenköpfen und Versetzungszeichen in anderen Stimmen zusammenstoßen.

Einstellung von automatischen Balken

Die Position und Länge der automatischen Balken wird bestimmt von bestimmten Regeln, wie beschrieben in [Automatische Balken], Seite 55. Es gibt zwei einander ausschließende Arten, diese Regeln zu verändern. Mit der ersten Art verändert man Gruppen von Noten, die einer bestimmten Taktart entsprechend mit Balken versehen werden. Das muss manuell getan werden für die Taktarten, für welche keine fertigen Regeln definiert worden sind. Die zweite Methode verändert die Definitionen für die Endpunkte der Balken und kann für jede Taktart eingesetzt werden. Diese zweite Methode **muss** eingesetzt werden für die Taktarten, für welche schon Regeln definiert worden sind, es sei denn, alle Regeln wurden mit `revert` rückgängig gemacht. Regeln sind definiert für die Taktarten 3/2, 3/4, 4/4, 2/4, 4/8, 4/16, 6/8, 9/8 und 12/8.

Die Gruppierung von Noten verändern

Wenn keine Regeln für die Beendung von Balken für die aktuelle Balkendauer in der benutzen Taktart vorhanden sind, wird die Bebalkung von drei Kontext-Eigenschaften kontrolliert: `measureLength`, `beatLength` und `beatGrouping`. Diese Eigenschaften können in den `Score`, `Staff` oder `Voice`-Kontexten gesetzt werden, um ihre Wirkungsweise zu begrenzen.

Durch sie werden die Balken wie folgt bestimmt:

Balken können überall beginnen (außer ein Balken ist schon aktiv). Balken enden zu den Werten, die `beatGrouping` und `beatLength` bestimmten, nach folgenden Regeln:

- Wenn `beatGrouping` und `beatLength` den gleichen Wert wie `measureLength` haben, wird `beatGrouping` benutzt, um die Endpunkte der Balken zu bestimmen.
- Wenn `beatGrouping` und `beatLength` nicht mit `measureLength` übereinstimmen, wird `beatLength` benutzt, um die Endpunkte der Balken zu bestimmen.

Achtung: Diese drei Eigenschaften werden für einen bestimmten Balken **nur dann** aktiv, wenn für diese Balkendauer keine Beendungsregeln für die benutzte Taktart definiert sind, oder wenn alle diese Regeln mit `revert` rückgängig gemacht wurden.

Standardmäßig werden `measureLength` (Taktlänge) und `beatLength` von der Taktart entnommen, die mit `\time` gesetzt wurde. `measureLength` hat standardmäßig genau die gleiche Länge wie die Taktlänge und der Standardwert für `beatLength` (Taktzeit-Länge) wird durch den Nenner des Taktart-Bruches bestimmt.

Der Standardwert von `beatGrouping` wird aus einer Tabelle in der Datei ‘`scm/music-functions.scm`’ entnommen. Um sie zu finden, siehe Abschnitt “Other sources of information” in *Handbuch zum Lernen*. Hier werden Taktzeiten-Gruppen für die Taktarten 5/8, 6/8, 8/8, 9/8 und 12/8 definiert.

Sowohl `measureLength` als auch `beatLength` sind *Momente*, Einheiten musikalischer Dauer. Eine Größe der Art *Moment* wird durch die Scheme-Funktion `ly:make-moment` erstellt. Für mehr Information zu dieser Funktion siehe [Verwaltung der Zeiteinheiten], Seite 82.
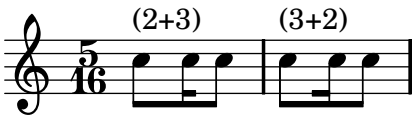
`beatGrouping` ist eine Liste an Integren, die die Anzahl von Zählzeiten für jede Gruppe darstellen.

Ausgewählte Schnipsel

Notengruppen

Balkengruppen können mit der `beatGrouping`-Eigenschaft geändert werden:

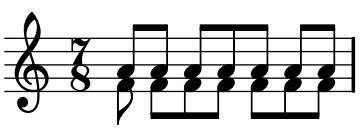
```
\relative c'' {
  \time 5/16
  \set beatGrouping = #'(2 3)
  c8[^"(2+3)" c16 c8]
  \set beatGrouping = #'(3 2)
  c8[^"(3+2)" c16 c8]
}
```



Specifying context with beatGrouping

By specifying the context, the effect of `beatGrouping` can be limited to the context specified, and the values which may have been set in higher-level contexts can be overridden. The `\set` commands must be placed *after* all `ime` commands:

```
\score {
  \new Staff <<
    \time 7/8
    \new Voice {
      \relative c'' {
        \set Staff.beatGrouping = #'(2 3 2)
        a8 a a a a a a
      }
    }
    \new Voice {
      \relative c' {
        \voiceTwo
        \set Voice.beatGrouping = #'(1 3 3)
        f8 f f f f f f
      }
    }
  >>
}
```



Using beatLength and beatGrouping

The property `measureLength` determines where bar lines should be inserted and, with `beatLength` and `beatGrouping`, how automatic beams should be generated for beam durations and time signatures for which no beam-ending rules are defined. This example shows several ways of controlling beaming by setting these properties. The explanations are shown as comments in the code.

```
\relative c'' {
  \time 3/4
  % Standard in 3/4-Takt sind drei bebalkte Gruppen
  % jede mit der Dauer einer Viertel
  a16 a a a a a a a a a a

  \time 12/16
  % Keine autom. Bebalkung definiert für 12/16
  a16 a a a a a a a a a a
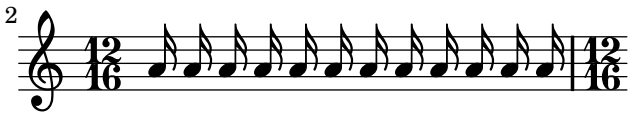
  \time 3/4
  % Taktart-Symbol ändern, aber 3/4-Bebalkung beibehalten
  \set Score.timeSignatureFraction = #'(12 . 16)
  a16 a a a a a a a a a a

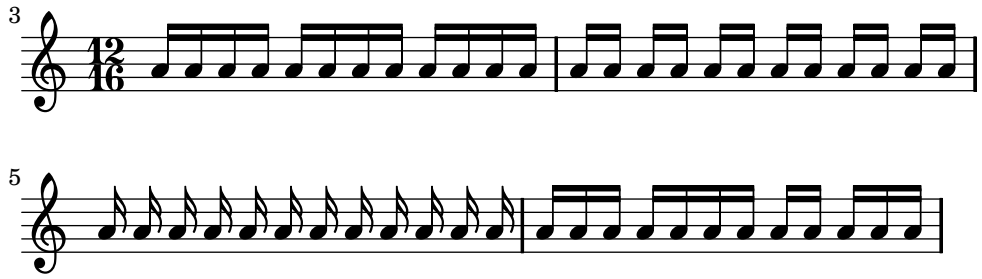
  % 3/4-Taktart Standardgruppen (1 1 1) und beatLenght von 1/8
  % stimme nicht überein mit measureLenght von 3/4, sodass die Balken
  % anhand von beatLength-Intervallen gruppiert werden
  \set Score.beatLength = #(ly:make-moment 1 8)
  a16 a a a a a a a a a a

  % Definition von Balken in Gruppen von (3 3 2 3) 1/16-Noten
  % 3+3+2+3=11, und 11*16<>3/4, also trifft beatGrouping nicht zu,
  % und Balken werden gruppiert anhand von beatLength (1/16)-Intervallen
  \set Score.beatLength = #(ly:make-moment 1 16)
  \set Score.beatGrouping = #'(3 3 2 3)
  a16 a a a a a a a a a a

  % Definition von Balkengruppen für (3 4 2 3) 1/16-Noten
  % 3+4+2+3=12, und 12*1/16=3/4, also beatGrouping wird verwendet
  \set Score.beatLength = #(ly:make-moment 1 16)
  \set Score.beatGrouping = #'(3 4 2 3)
  a16 a a a a a a a a a a
}
```
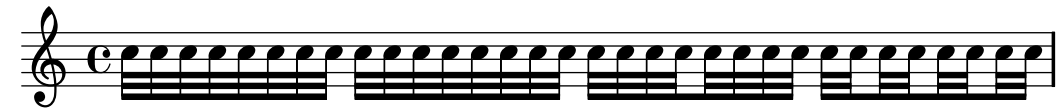






Sub-dividing beams

The beams of consecutive 16th (or shorter) notes are, by default, not sub-divided. That is, the three (or more) beams stretch unbroken over entire groups of notes. This behavior can be modified to sub-divide the beams into sub-groups by setting the property `subdivideBeams`. When set, multiple beams will be sub-divided at intervals defined by the current value of `beatLength` by reducing the multiple beams to just one beam between the sub-groups. Note that `beatLength` defaults to one over the denominator of the current time signature if not set explicitly. It must be set to a fraction giving the duration of the beam sub-group using the `make-moment` function, as shown here:

```
\relative c'' {
  c32[ c c c c c c]
  \set subdivideBeams = ##t
  c32[ c c c c c c]

  % Balkenuntergruppenlänge als Achtel definieren
  \set beatLength = #(ly:make-moment 1 8)
  c32[ c c c c c c]

  % Balkenuntergruppenlänge als Sechszehntel definieren
  \set beatLength = #(ly:make-moment 1 16)
  c32[ c c c c c c]
}
```

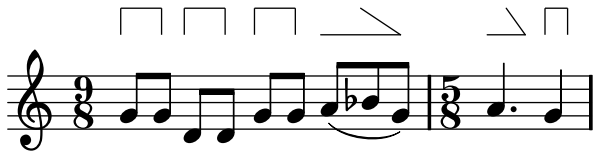


Conducting signs, measure grouping signs

Options to group beats within a bar are available through the Scheme function `set-time-signature`, which takes three arguments: the number of beats, the beat length, and the internal grouping of beats in the measure. If the `Measure_grouping_engraver` is included, the function will also create `MeasureGrouping` signs. Such signs ease reading rhythmically complex modern music. In the example, the 9/8 measure is subdivided in 2, 2, 2 and 3. This is passed to `set-time-signature` as the third argument: `'(2 2 2 3)`:

```
\score {
  \relative c'' {
    #(set-time-signature 9 8 '(2 2 2 3))
    #(revert-auto-beam-setting '(end * * 9 8) 3 8)
    #(override-auto-beam-setting '(end 1 8 9 8) 1 4)
    #(override-auto-beam-setting '(end 1 8 9 8) 2 4)
    #(override-auto-beam-setting '(end 1 8 9 8) 3 4)
    g8 g d d g g a( bes g) |
    #(set-time-signature 5 8 '(3 2))
    a4. g4
  }
```

```
  \layout {
    \context {
      \Staff
      \consists "Measure_grouping_engraver"
    }
  }
}
```



Die Endpunkte von Balken bestimmen

In üblichen Taktarten können automatisch gesetzte Balken an jeder Note beginnen, aber nur an einigen bestimmten Positionen innerhalb des Taktes beendet werden. Diese Positionen werden durch die Eigenschaften in `autoBeamSettings` bestimmt. Sie bestehen aus einer Liste an Regeln, die bestimmen, wo Balken enden können. Die Standardeinstellungen dieser automatischen Einstellungen befinden sich in der Datei '`scm/auto-beam.scm`'. Um diese Datei zu finden, siehe Abschnitt "Other sources of information" in *Handbuch zum Lernen*.

Diese Methode **muss** benutzt werden, wenn die Einstellungen für die Balken in Taktarten verändert werden sollen, für welche schon Regeln existieren, es sei denn, alle diese Regeln wurden rückgängig gemacht. Die Methode ist auch in vielen anderen Taktarten gut anzuwenden, wenn die Taktart sich oft ändert, oder wenn die Balken unterschiedlich für unterschiedliche Balkendauern gesetzt werden sollen.

Um eine neue Regel zu der Liste hinzuzufügen, muss folgende Syntax verwendet werden:

```
#(override-auto-beam-setting
  '(Balken-Limit
    Balken-Zähler Balken-Nenner
    Taktart-Zähler Taktart-Nenner)
  Moment-Zähler Moment-Nenner [Kontext])
```

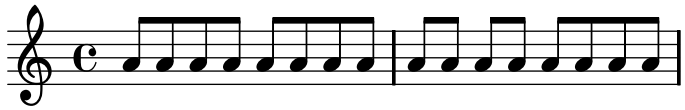
wobei

- `Balken-Limit` die Art der Balkenbegrenzung ist. Das kann entweder der Wert `begin` (Anfang) oder `end` (Ende) sein, aber nur `end` hat eine Auswirkung.
- `Balken-Zähler/Balken-Nenner` ist die Balken-Dauer, auf die die Regel sich bezieht. Ein Balken hat per Definition die Dauer seiner kürzesten Note. Wenn sowohl `Balken-Zähler` als auch `Balken-Nenner` auf `'*'` gesetzt werden, gilt die Regel für alle Dauern.
- `Taktart-Zähler/Taktart-Nenner` bestimmen die Taktart, auf die die Regel sich bezieht. Wenn `Taktart-Zähler` und `Taktart-Nenner` auf `'*'` gesetzt werden, gilt die Regel für alle Taktarten.
- `Moment-Zähler/Moment-Nenner` ist die Position im Takt, an welcher der Balken aufhören soll.
- `Kontext` ist optional und bestimmt den Kontext, in welchem die Änderungen vorgenommen werden sollen. Der Standard ist `'Voice`.

  `#(score-override-auto-beam-setting '(A B C D) E F)` ist gleichbedeutend mit `#(override-auto-beam-setting '(A B C D) E F 'Score)`.

Wenn beispielsweise die automatischen Balken immer auf der ersten Viertel enden sollen, unabhängig von der Taktart oder Dauer des Balkens, kann

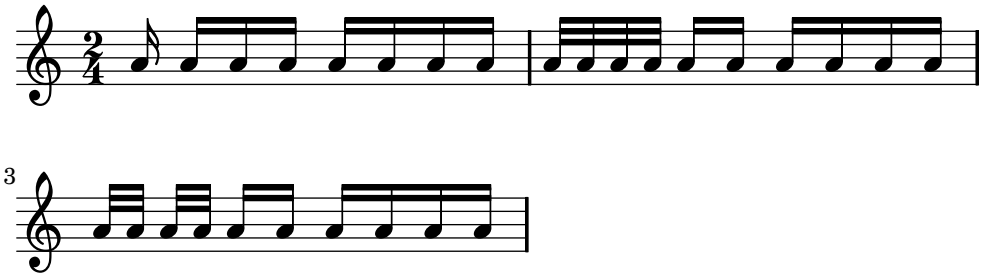
```
a8 a a a a a a a
#(override-auto-beam-setting '(end * * * *) 1 4)
a8 a a a a a a a
```



benutzt werden.
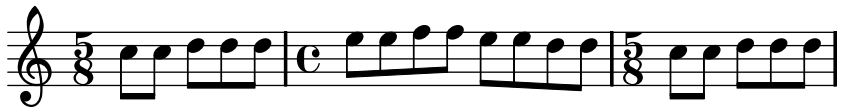
Balkenregeln können aber auch auf Notengruppen beschränkt werden, deren kürzeste Note einen bestimmten Wert hat:

```
\time 2/4
% 1/16-Balken für alle Taktarten beim 1/16-Moment beenden
#(override-auto-beam-setting '(end 1 16 * *) 1 16)
a16 a a a a a a a |
a32 a a a a16 a a a a |
% 1/23-Balken für alle Taktarten beim 1/16-Moment beenden
#(override-auto-beam-setting '(end 1 32 * *) 1 16)
a32 a a a a16 a a a a |
```



Balkenregeln können so definiert werden, dass sie nur in bestimmten Taktarten angewendet werden:

```
\time 5/8
% Balken aller Längen im 5/8-Takt am 2/8-Moment beenden
#(override-auto-beam-setting '(end * * 5 8) 2 8)
c8 c d d d
\time 4/4
e8 e f f e e d d
\time 5/8
c8 c d d d
```



Wenn mehrfache Stimmen benutzt werden, muss der `Staff`-Kontext angegeben werden, wenn die Balkenregeln auf alle Stimmen im System angewendet werden sollen:

```
\time 7/8
% Rhythmus 3-1-1-2
% Kontext nicht angegeben - funktioniert nicht richtig
#(override-auto-beam-setting '(end * * 7 8) 3 8)
#(override-auto-beam-setting '(end * * 7 8) 4 8)
```

```
#(override-auto-beam-setting '(end * * 7 8) 5 8)
<< {a8 a a a16 a a a a8 a} \\ {f4. f8 f f f} >>

% Funktioniert richtig mit angegebenem Kontext
#(override-auto-beam-setting '(end * * 7 8) 3 8 'Staff)
#(override-auto-beam-setting '(end * * 7 8) 4 8 'Staff)
#(override-auto-beam-setting '(end * * 7 8) 5 8 'Staff)
<< {a8 a a a16 a a a a8 a} \\ {f4. f8 f f f} >>
```



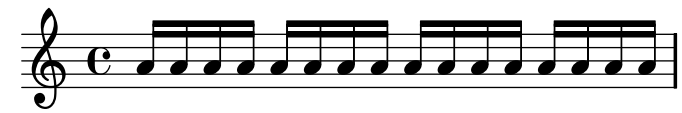
Achtung: Wenn eine unerwartete Bebalkung auftritt, schauen Sie zuerste die Balkeneinstellungen in '`scm/auto-beam.scm`' nach, ob sich hier Überschneidungen ergeben, weil die Balkenenden, die dort definiert sind, auch noch weiterhin wirksam sind zusätzlich zu den von Ihnen definierten.

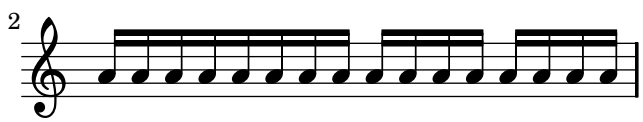
Jede ungewollte oder störende Balkenbeendigung aus den Standard-Eingesllungen muss für Ihre Taktart rückgängig gemacht werden. Existierende automatische Bebalkungsregeln werden entfernt mit folgender Syntax:

```
#(revert-auto-beam-setting
  '(Balken-Limit
    Balken-Zähler Balken-Nenner
    Taktart-Zähler Taktart-Nenner)
  Moment-Zähler Moment-Nenner [Kontext])
```

wobei `Balken-Limit`, `Balken-Zähler`, `Balken-Nenner`, `Taktart-Zähler`, `Taktart-Nenner`, `Moment-Zähler`, `Moment-Nenner` sowie `Kontext` die gleichen sind wie oben erklärt.

```
\time 4/4
a16 a a a a a a a a a a a a a a a a
% Regel annullieren, dass 1/16-Balken im 4/4-Takt am 1/4-Moment enden
#(revert-auto-beam-setting '(end 1 16 4 4) 1 4)
a16 a a a a a a a a a a a a a a a a
```
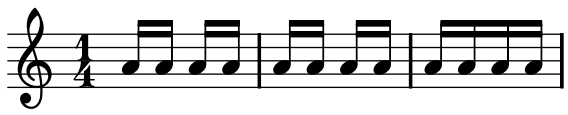




Die Regel in einer `revert-auto-beam-setting`-Definition muss exakt der ursprünglichen Regel entsprechen. Dabei werden keine Platzhalter akzeptiert.

```
\time 1/4
#(override-auto-beam-setting '(end 1 16 1 4) 1 8)
a16 a a a
#(revert-auto-beam-setting '(end 1 16 * *) 1 8) % Das macht es nicth rückgängig
a a a a
#(revert-auto-beam-setting '(end 1 16 1 4) 1 8) % Das schon
```

```
a a a a
```



Ausgewählte Schnipsel

Balkengruppen für 7/8-Takte

Es gibt keine automatischen Balkengruppen für 7/8-Takte. Wenn diese Taktart benötigt wird, müssen die Gruppierungen definiert werden. Um beispielsweise alle Noten in 2/8-3/8-2/8 aufzuteilen, müssen Balkenenden für 2/8 und 5/8 definiert werden:

```
\relative c'' {
  \time 7/8
  % Rhythmus 2-3-2
  a8 a a a a a a
  #(override-auto-beam-setting '(end * * 7 8) 2 8)
  #(override-auto-beam-setting '(end * * 7 8) 5 8)
  a8 a a a a a a
}
```



Reverting default beam endings
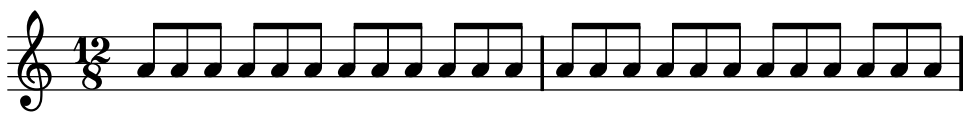
To typeset beams grouped `3-4-3-2` in 12/8 it is necessary first to override the default beam endings in 12/8, and then to set up the new beaming endings:

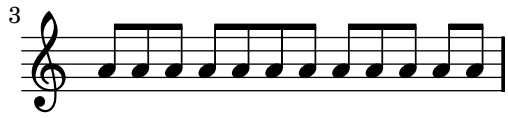
```
\relative c'' {
  \time 12/8

  % Standartbebalkung
  a8 a a a a a a a a a a a

  % Standardwerte von scm/auto-beam.scm für 12/8-Takt herstellen
  #(revert-auto-beam-setting '(end * * 12 8) 3 8)
  #(revert-auto-beam-setting '(end * * 12 8) 3 4)
  #(revert-auto-beam-setting '(end * * 12 8) 9 8)
  a8 a a a a a a a a a a a

  % Neue Werte für Balkenenden
  #(override-auto-beam-setting '(end * * 12 8) 3 8)
  #(override-auto-beam-setting '(end * * 12 8) 7 8)
  #(override-auto-beam-setting '(end * * 12 8) 10 8)
  a8 a a a a a a a a a a a
}
```

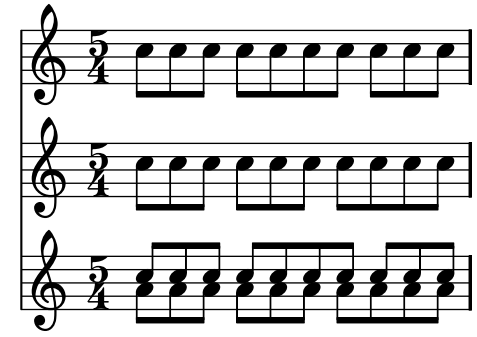




Beam endings in Score context

Beam-ending rules specified in the `Score` context apply to all staves, but can be modified at both `Staff` and `Voice` levels:

```
\relative c'' {
  \time 5/4
  % Automatische Bebalkung für alle Systeme
  #(score-override-auto-beam-setting '(end * * 5 4) 3 8)
  #(score-override-auto-beam-setting '(end * * 5 4) 7 8)
  <<
    \new Staff {
      c8 c c c c c c c c c
    }
    \new Staff {
      % Balken nur für dieses System verändern
      #(override-auto-beam-setting '(end * * 5 4) 6 8 'Staff)
      #(revert-auto-beam-setting '(end * * 5 4) 7 8 'Staff)
      c8 c c c c c c c c c
    }
    \new Staff {
      % Bebalkung vom Score-Kontext geerbt
      <<
        {
          \voiceOne
          c8 c c c c c c c c c
        }
        % Balken nur für diese Stimme ändern
        \new Voice {
          \voiceTwo
          #(override-auto-beam-setting '(end * * 5 4) 6 8)
          #(revert-auto-beam-setting '(end * * 5 4) 7 8)
          a8 a a a a a a a a a
        }
      >>
    }
  >>
}
```



Vordefinierte Befehle

`\autoBeamOff`, `\autoBeamOn`.

Bekannte Probleme und Warnungen

Wenn eine Partitur aufhört, während ein automatischer Balken noch nicht geschlossen ist und noch Noten erwartet, wird dieser letzte Balken überhaupt nicht ausgegeben. Das gilt auch für polyphone Stimmen, die mit `<< ... \\ ... >>` gesetzt wurden. Wenn eine polyphone Stimme endet, während ein Balken noch Noten erwartet, wird dieser Balken nicht gesetzt.
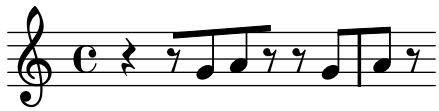
Siehe auch

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

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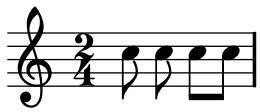
In einigen Fällen kann es nötig sein, den automatischen Algorithmus für die Balken zu überschreiben. Die automatischen Balken werden beispielsweise nicht über Pausen oder Taktlinien hinweg gesetzt, und in Gesang werden die Balken oft nach dem Rhythmus des Textes und nicht dem der Musik gesetzt. Manuell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

```
{
  r4 r8[ g' a r8] r8 g[ | a] r8
}
```
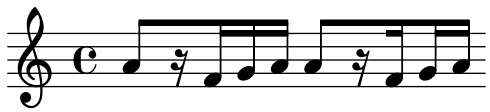


Einzelne Noten können mit dem Befehl `\noBeam` markiert werden, damit sie nicht mit einem Balken versehen werden.

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



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).

```
a8[ r16 f g a]
a8[ r16
\set stemLeftBeamCount = #2
\set stemRightBeamCount = #1
f
\set stemLeftBeamCount = #1
g a]
```



Ausgewählte Schnipsel

Flat flags and beam nibs

Flat flags on lone notes and beam nibs at the ends of beamed figures are both possible with a combination of `stemLeftBeamCount`, `stemRightBeamCount` and paired `[ ]` beam indicators.

For right-pointing flat flags on lone notes, use paired `[ ]` beam indicators and set `stemLeftBeamCount` to zero (see Example 1).

For left-pointing flat flags, set `stemRightBeamCount` instead (Example 2).

For right-pointing nibs at the end of a run of beamed notes, set `stemRightBeamCount` to a positive value. And for left-pointing nibs at the start of a run of beamed notes, set `stemLeftBeamCount` instead (Example 3).

Sometimes it may make sense for a lone note surrounded by rests to carry both a left- and right-pointing flat flag. Do this with paired `[ ]` beam indicators alone (Example 4).

(Note that `\set stemLeftBeamCount`is always equivalent to `\once \set`. In other words, the beam count settings aren't "sticky", so the pair of flat flags attached to the lone `c'16 [ ]` in the last example have nothing to do with the `\set` two notes prior.)

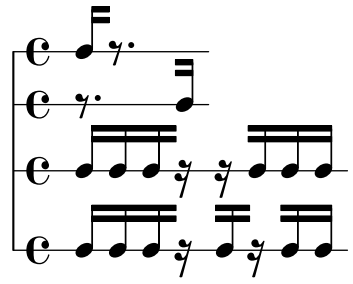
```
\score {
<<
% Beispiel 1
\new RhythmicStaff {
  \set stemLeftBeamCount = #0
  c'16 [ ]
  r8.
}

% Beispiel 2
\new RhythmicStaff {
  r8.
  \set stemRightBeamCount = #0
  c'16 [ ]
}

% Beispiel 3
\new RhythmicStaff {
  c'16
  c'16
  \set stemRightBeamCount = #2
  c'16
  r16
  r16
  \set stemLeftBeamCount = #2
  c'16
  c'16
  c'16
}

% Beispiel 4
\new RhythmicStaff {
   c'16
   c'16
   \set stemRightBeamCount = #2
```

```
    c'16
    r16
    c'16 [ ]
    r16
    \set stemLeftBeamCount = #2
    c'16
    c'16
}
>>
}
```



Gespreizte Balken
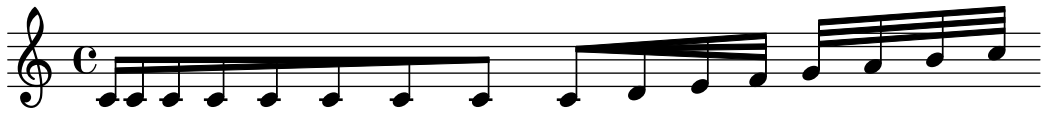
Gespreizte Balken werden teilweise eingesetzt um anzuzeigen, dass kleine Notengruppen in beschleunigendem oder verlangsamendem Tempo gespielt werden sollen, ohne dass sich das Tempo des Stückes verändert. Die Reichweite der gespreizten Balken muss manuell mit `[` und `]` angegeben werden und die Spreizung 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 Verhaltnis 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.

```
\override Beam #'grow-direction = #LEFT
\featherDurations #(ly:make-moment 2 1)
{ c16[ c c c c c c c] }
\override Beam #'grow-direction = #RIGHT
\featherDurations #(ly:make-moment 2 3)
{ c32[ d e f] }
% in nichtgespreizte Balken zurückverwandeln
\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.

Bekannte Probleme und Warnungen

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

Siehe auch

Snippets: Abschnitt “Rhythms” in *Schnipsel*.
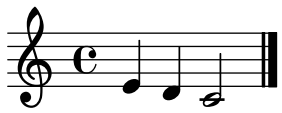
1.2.5 Takte

Taktlinien

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

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

```
e4 d c2 \bar "|."
```



Es ist kein Fehler, wenn die letzte Note in einem Takt nicht zur automatische eingefügten Taktlinie 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 eine Taktlinie manuell eingefügt wird, auch, wenn es sich um keinen vollständigen Takt handelt. Um einen Zeilenumbruch zu erlauben, ohne die Taktlinie auszugeben, kann
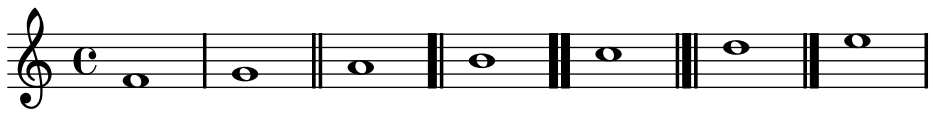
```
\bar ""
```

benutzt werden. Damit wird eine unsichtbare Taktlinie 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 4.3.1 [Zeilenumbrüche], Seite 332.

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

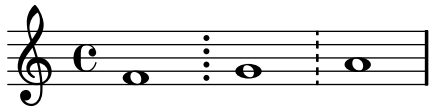
Manuell können die einfache Taktlinie und zusätzlich fünf Arten einer doppelten Taktlinie gesetzt werden:

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



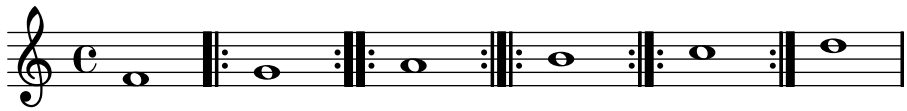
Zusätzlich gibt es noch punktierte und gestrichelte Taktlinien:

```
f1 \bar ":" g \bar "dashed" a
```



und fünf unterschiedliche Wiederholungstaktlinien:
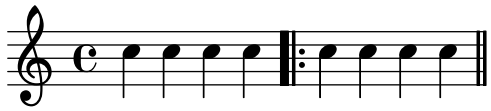
```
f1 \bar "|:" g \bar ":|:" a \bar ":|.|:" b \bar ":|.:" c \bar ":|" d
```
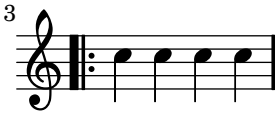


Auch wenn die Taktlinien, die Wiederholungen angeben, manuell eingefügt werden können, wird die Wiederholung dadurch nicht von LilyPond erkannt. Wiederholte Stellen werden besser notiert, indem man die Wiederholguns-Befehle einsetzt, die automatische die richtigen Taktlinien setzen. Das ist beschrieben inAbschnitt 1.4 [Wiederholungszeichen], Seite 100.

Zusätzlich kann noch `"||:"` verwendet werden, dass sich genauso wie `"|:"` verhält, außer bei Zeilenumbrüchen, wo eine doppelte Taktlinie am Ende der Zeile ausgegeben wird und eine öffnende Wiederholungslinie am Anfang der nächsten Zeile.

```
\override Score.RehearsalMark #'padding = #3
c c c c
\bar "||:"
c c c c \break
\bar "||:"
c c c c
```
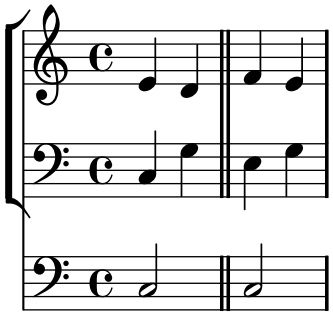




In Partituren mit vielen Systemen wird ein `\bar`-Befehl in einem System automatisch auf alle anderen Systeme angewendet. Die resultierenden Taktlinien sind miteinander verbunden innerhalb einer Gruppe (`StaffGroup`) oder einem Klaviersystem (`PianoStaff` bzw. (`GrandStaff`).

```
<<
  \new StaffGroup <<
    \new Staff {
      e'4 d'
      \bar "||"
```

```
      f' e'
    }
    \new Staff { \clef bass c4 g e g }
  >>
  \new Staff { \clef bass c2 c2 }
>>
```



Ausgewählte Schnipsel

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

Die automatische erzeugte Taktlinie ist `"|"`. Das kann jederzeit durch den Befehl `\set Timing.defaultBarType =` *Takstrichart* geändert werden.

Siehe auch

Notationsreferenz: Abschnitt 4.3.1 [Zeilenumbrüche], Seite 332, Abschnitt 1.4 [Wiederholungszeichen], Seite 100, [Systeme gruppieren], Seite 125.

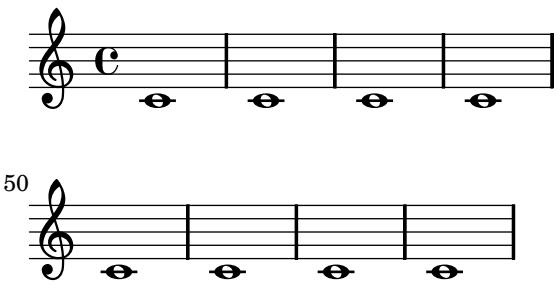
Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Referenz der Interna: Abschnitt “BarLine” in *Referenz der Interna* (created at Abschnitt “Staff” in *Referenz der Interna* level), Abschnitt “SpanBar” in *Referenz der Interna* (across staves), Abschnitt “Timing_translator” in *Referenz der Interna* (for Timing properties).

Taktzahlen

Taktnummern 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:

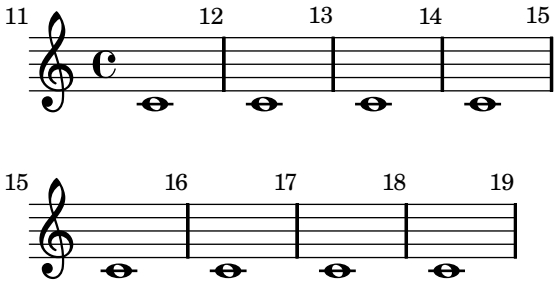
```
c1 c c c
\break
\set Score.currentBarNumber = #50
c1 c c c
```



Ausgewählte Schnipsel

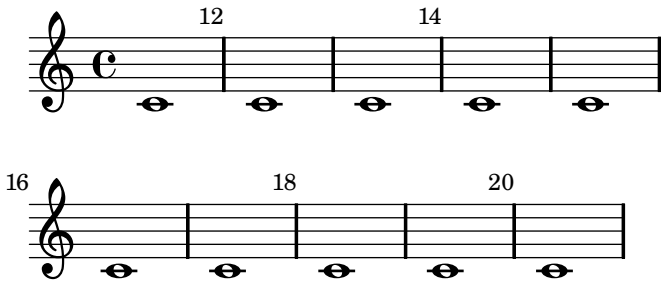
Taktnummern können in regelmäßigem Abstand ausgegeben werden, anstatt dass sie nur am Beginn des Systems erscheinen. Um das zu erreichen, muss die Standardeinstellung verändert werden, um zu erlauben, dass Taktnummern an anderen Stellen als dem Beginn von Systemen ausgegeben werden. Das wird mit der 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 entprechenden 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
\bar ""  % Erste Taktnummer soll auch gesetzt werden
c1 c c c
\break
c c c c
```



Im nächsten Beispiel werden die Taktnummern nur für jeden zweiten Takt gesetzt, außer am Ende der Zeile:

```
\override Score.BarNumber #'break-visibility = #'#(#f #t #t)
\set Score.currentBarNumber = #11
\bar ""  % Erste Taktnummer soll auch gesetzt werden
% Jeden zweiten Takt eine Taktnummer setzen
\set Score.barNumberVisibility = #(every-nth-bar-number-visible 2)
c1 c c c c
\break
c c c c c
```



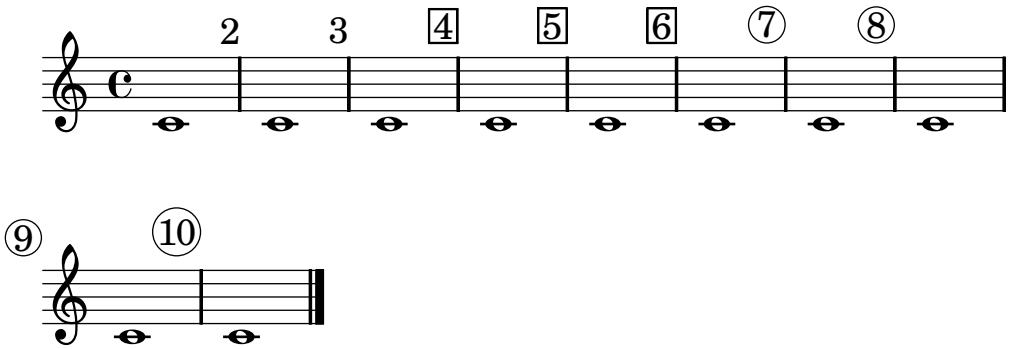
Die Größe der Taktnummer kann geändert werden. Das wird im folgenden Beispiel gezeigt, welches auch illustriert, wie man Taktnummern in Kästen oder Kreise einfasst und eine Alternative zeigt, wie man `#(#f #t #t)` für `break-visibility` definieren kann.

```
% Taktnummern am Zeilenende verbieten und woanders erlauben
\override Score.BarNumber #'break-visibility
  = #end-of-line-invisible
```

```
% Taktnummer um Faktor zwei vergrößern
\override Score.BarNumber #'font-size = #2
\repeat unfold 3 { c1 } \bar "|"

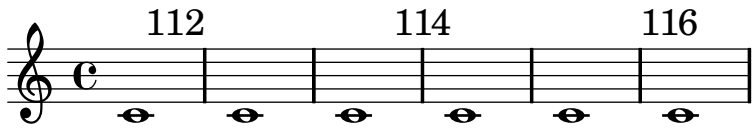
% Einen Kasten um die folgenden Taktnummern zeichnen
\override Score.BarNumber  #'stencil
  = #(make-stencil-boxer 0.1 0.25 ly:text-interface::print)
\repeat unfold 3 { c1 } \bar "|"

% Einen Kreis um die folgenden Taktnummern zeichnen
\override Score.BarNumber  #'stencil
  = #(make-stencil-circler 0.1 0.25 ly:text-interface::print)
\repeat unfold 4 { c1 } \bar "|."
```



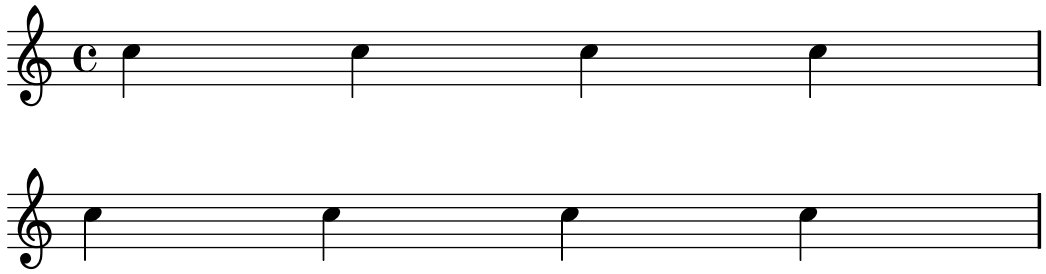
Taktnummern sind standardmäßig links von ihrem Anker angeordnet. Der Anker ist meistens das linke Ende einer Zeile, oder, wenn die Zahlen innerhalb der Zeile gesetzt werden, die linke Taktlinie eines Taktes. Die Nummern können auch direkt auf der Taktlinie positioniert oder rechts davon gesetzt werden:

```
\set Score.currentBarNumber = #111
\override Score.BarNumber #'break-visibility = #'#(#t #t #t)
% Taktnummer um Faktor zwei vergrößern
\override Score.BarNumber #'font-size = #2
% Jeden zweiten Takt eine Taktnummer setzen
\set Score.barNumberVisibility = #(every-nth-bar-number-visible 2)
c1 c1
% Taktnummern zentrieren
\override Score.BarNumber #'self-alignment-X = #0
c1 c1
% Taktnummern rechtsbündig
\override Score.BarNumber #'self-alignment-X = #-1
c1 c1
```



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

```
\layout {
  \context {
    \Score
    \remove "Bar_number_engraver"
  }
}
\relative c''{
  c4 c c c \break
  c4 c c c
}
```



Siehe auch

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

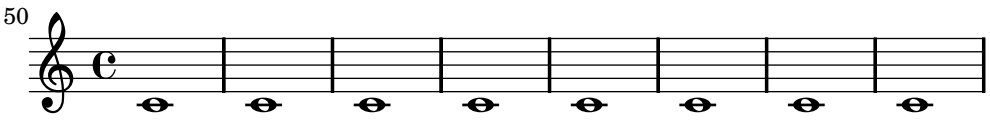
Referenz der Interna: Abschnitt “BarNumber” in *Referenz der Interna*.

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 `BarNumer` verwendet werden, um die Zahl zu verschieben.

Taktnummern werden nur an Taktzahlen ausgegeben. Um eine Taktzahl auch am Anfang eines Stückes zu erhalten, muss eine leere Taktlinie manuell eingefügt werden, und für die aktuelle Taktzahl (`currentBarNumber` muss ein anderer Wert als `1` angegeben werden:

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



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:
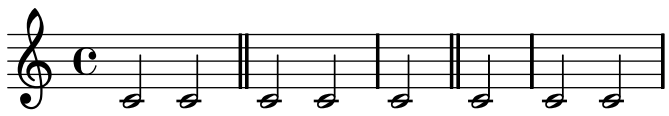
```
\lyricmode {
  \time 2/4
  Twin -- kle | Twin -- kle
}
```

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 | umzudefinieren, so dass hiermit eine andere Aktion als eine Taktüberprüfung erreicht wird. Das geschieht, indem man der Pipe (`pipeSymbol` einen musikalischen Ausdruck zuweist. Im nächsten Beispiel wird | dazu verwendet, eine doppelte Taktlinie auszugeben, woimmer man das Zeichen auchsetzt. Gleichzeitig hört das Zeichen auf, als Taktüberprüfung zu funktionieren.

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



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:

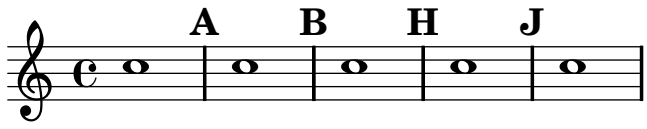
```
\barNumberCheck #123
```

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

Übungszeichen

Übungszeichen können mit dem `\mark`-Befehl ausgegeben werden:

```
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

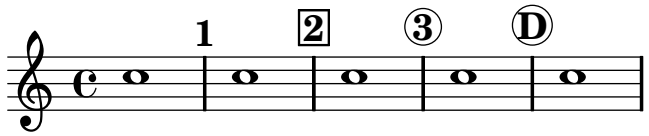
```
\set Score.markFormatter = #format-mark-alphabet
```

benutzen.

Das Zeichen wird automatisch erhöht, wenn Sie `\mark \default` schreiben, aber Sie können auch eine beliebige Ganzzahl als Argument angeben. Der Wert, der gesetzt werden soll, wird in der Eigenschaft `rehearsalMark` gespeichert.

Der Stil der Übungszeichen wird von der Eigenschaft `markFormatter` 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 `markFormatter` so definiert, dass eine Zahl ausgegeben wird. In den folgenden Takten werden dann andere mögliche Einstellungen gezeigt.

```
\set Score.markFormatter = #format-mark-numbers
c1 \mark \default
c1 \mark \default
\set Score.markFormatter = #format-mark-box-numbers
c1 \mark \default
\set Score.markFormatter = #format-mark-circle-numbers
c1 \mark \default
\set Score.markFormatter = #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-barnumbers`und `format-mark-circle-barnumbers` könenn eingesetzt werden, um Taktnummern anstelle der fortlaufenden Zahlen bzw. Buchstaben zu erhalten.

Andere Übungszeichenstile können auch manuell gesetzt werden:

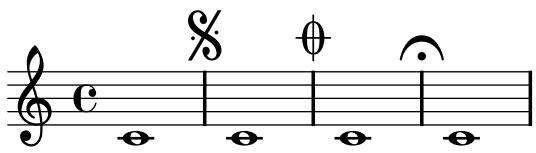
```
\mark "A1"
```

`Score.markFormatter` 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:

```
c1 \mark \markup { \musicglyph #"scripts.segno" }
c1 \mark \markup { \musicglyph #"scripts.coda" }
c1 \mark \markup { \musicglyph #"scripts.ufermata" }
c1
```



Siehe Abschnitt B.6 [Die Feta-Schriftart], Seite 352, wo alle Symbole gezeigt sind, die mit dem Befehl `\musicglyph` ausgegeben werden können.

Übliche Veränderungen der Positionierung von Übungszeichen finden sich in Abschnitt 1.8.2 [Text formatieren], Seite 170.

Siehe auch

Notationsreferenz: Abschnitt B.6 [Die Feta-Schriftart], Seite 352, Abschnitt 1.8.2 [Text formatieren], Seite 170.

Installierte Dateien: '`scm/translation-functions.scm`' beinhaltet die Definition von `format-mark-numbers` und `format-mark-letters`. Sie können als Anleitung für eigene Funktionen benutzt werden.

Schnipsel: Abschnitt "Rhythms" in *Schnipsel*.
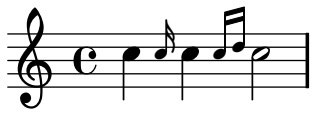
Referenz der Interna: Abschnitt "RehearsalMark" in *Referenz der Interna*.

1.2.6 Besondere rhythmische Fragen

Verzierungen
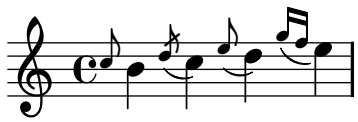
Verzierungen sind ausgeschriebene Verzierungen. Sie werden in einer kleineren Schriftgröße gesetzt und nehmen keine logische Zeit im Takt ein.

```
c4 \grace c16 c4
\grace { c16[ d16] } c2
```
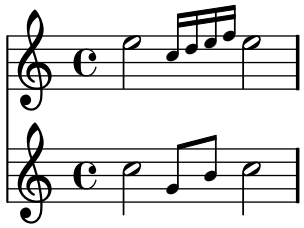


LilyPond hat auch Unterstützung für zwei besondere Verzierungen, den Vorschlag und den Vorhalt. Der Vorschlag wird durch eine verkleinerte Note mit Schrägstrich und Bogen notiert. Der Vorhalt dagegen ist eine Verzierung, die einen bestimmten Notenwert der Hauptnote für sich beansprucht. Er wird als verkleinerte Note ohne Schrägstrich notiert.

```
\grace c8 b4
\acciaccatura d8 c4
\appoggiatura e8 d4
\acciaccatura { g16[ f] } e4
```
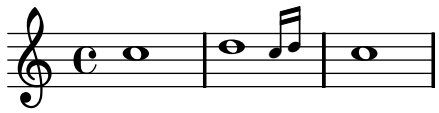


Die Position von Verzierungen ist zwischen Notensystemen synchronisiert. Im nächsten Beispiel stehen im einen System zwei 16-Noten für jede 8-Note des zweiten Systems:

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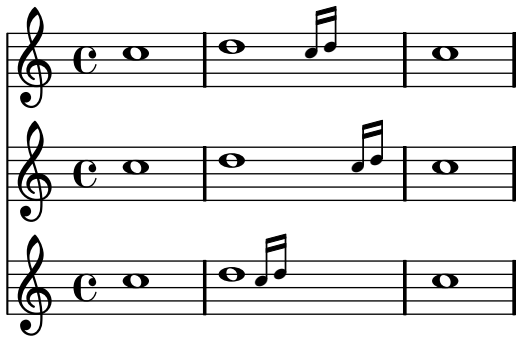


Wenn Sie eine Note mit einer Verzierung abschließen wollen, müssen Sie den `\afterGrace`-Befehl benutzen. Er benötigt zwei Argumente: die Hauptnote und die Verzierung, die nach der Hauptnote folgen soll:

```
c1 \afterGrace d1 { c16[ d] } c1
```
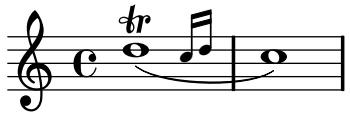


Damit wird die Verzierung mit einem Abstand von der Hauptnote gesetzt, der 3/4 der Dauer der Hauptnote entspricht. Dieser Standard kann durch Definition von `afterGraceFraction` verändert werden. Das nächste Beispiel zeigt, wie sich der Abstand verändert, wenn der Wert 3/4, 15/16 und 1/2 der Hauptnote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

```
<<
  \new Staff {
    c1 \afterGrace d1 { c16[ d] } c1
  }
  \new Staff {
    #(define afterGraceFraction (cons 15 16))
    c1 \afterGrace d1 { c16[ d] } c1
  }
  \new Staff {
    #(define afterGraceFraction (cons 1 2))
    c1 \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:
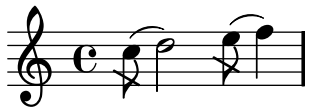
```
\new Voice {
  \acciaccatura {
    \stemDown
    f16->
    \stemNeutral
  }
  g4 e c2
}
```



Ausgewählte Schnipsel
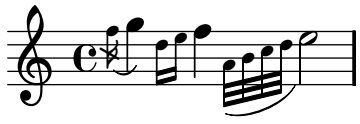
The slash through the stem found in *acciaccatura*s can be applied in other situations:

```
\relative c'' {
  \override Stem #'stroke-style = #"grace"
  c8( d2) e8( f4)
}
```



The layout of grace expressions can be changed throughout the music using the function add-grace-property. The following example undefines the Stem direction for this grace, so that stems do not always point up.

```
\relative c'' {
  \new Staff {
    #(add-grace-property 'Voice 'Stem 'direction ly:stem::calc-direction)
    #(remove-grace-property 'Voice 'Stem 'direction)
    \new Voice {
       \acciaccatura { f16 } g4
       \grace { d16[ e] } f4
       \appoggiatura { a,32[ b c d] } e2
    }
  }
}
```

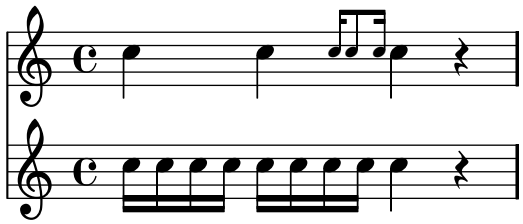


Another option is to change the variables startGraceMusic, stopGraceMusic, startAcciaccaturaMusic, stopAcciaccaturaMusic, startAppoggiaturaMusic,

`stopAppoggiaturaMusic`. The default values of these can be seen in the file `ly/grace-init.ly`. By redefining them other effects may be obtained.

Grace notes may be forced to align with regular notes in other staves:

```
\relative c'' {
  <<
    \override Score.SpacingSpanner #'strict-grace-spacing = ##t
    \new Staff {
      c4
      \afterGrace c4 { c16[ c8 c16] }
      c4 r
    }
    \new Staff {
      c16 c c c c c c c c4 r
    }
  >>
}
```



Siehe auch

Glossar: Abschnitt “Verzierungen” in *Glossar*, Abschnitt “Vorschlag” in *Glossar*, Abschnitt “Vorhalt” in *Glossar*.

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

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

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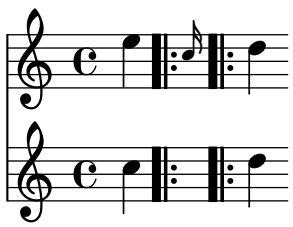
Bekannte Probleme und Warnungen

Eine Partitur, die mit einem `\grace`-Ausdruck beginnt, benötigt eine explizit gesetzte neue Stimme (`\new Voice`), sonst werden Hauptnote und Verzierung auf verschiedenen Systemen gesetzt.

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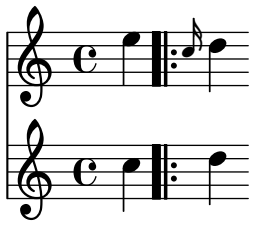
Die Synchronisation von Verzierungen kann auch zu Überraschungen führen. Auch andere Symbole der Systeme, wie Vorzeichen, Taktlinien usw., werden synchronisiert. Vorsicht ist geboten, wenn nur in bestimmten Systemen Verzierungen vorkommen:

```
<< \new Staff { e4 \bar "|:" \grace c16 d4 }
   \new Staff { c4 \bar "|:" d4 } >>
```



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

```
<< \new Staff { e4 \bar "|:" \grace c16 d4 }
   \new Staff { c4 \bar "|:" \grace s16 d4 } >>
```



gesetzt werden.

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

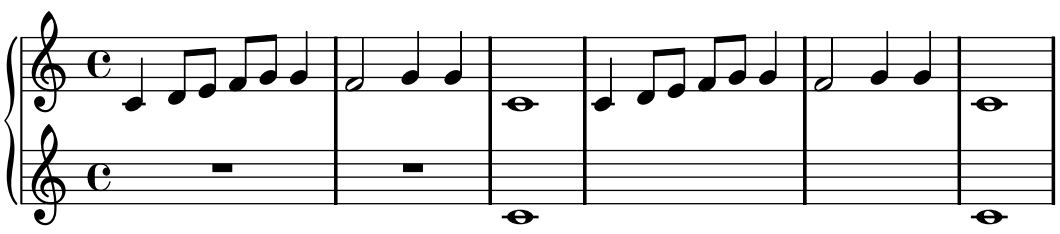
An Kadenzen ausrichten

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.

```
MeineKadenz = \relative c' {
  c4 d8 e f g g4
  f2 g4 g
}

\new GrandStaff <<
  \new Staff {
    \MeineKadenz c'1
    \MeineKadenz c'1
  }
  \new Staff {
    #(ly:export (mmrest-of-length MeineKadenz))
    c'1
    #(ly:export (skip-of-length MeineKadenz))
    c'1
  }
>>
```



Siehe auch

Glossar: Abschnitt “Kadenz” in *Glossar*.

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

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.

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 [Taktzahlen], Seite 71.
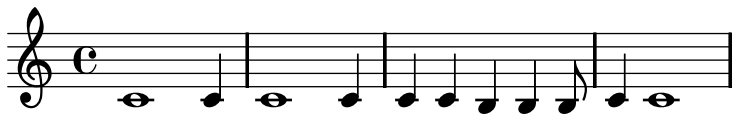
`measureLength (Taktlänge)`
: Die Länge der Takte mit der aktuellen Taktart. In einem 4/4-Takt ist sie 1, in einem 6/8-Takt 3/4. Dieser Wert bestimmt, wann eine Taktlinie gezogen wird und wie automatische Balken erstellt werden sollen.

`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 = #(ly:make-moment 5 4)
c1 c4
c1 c4
c4 c4
\set Score.measurePosition = #(ly:make-moment 5 8)
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.
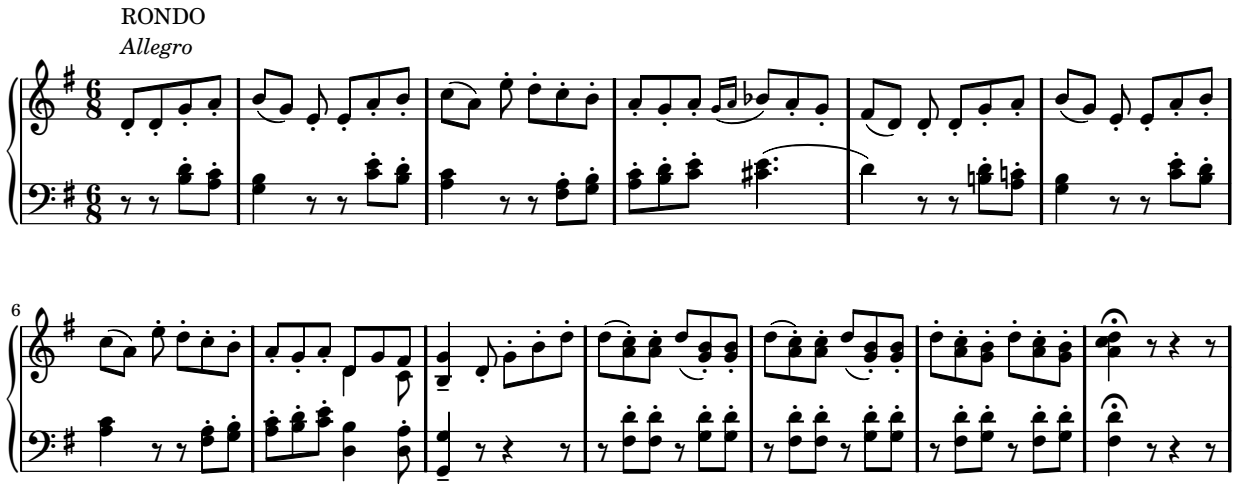
Siehe auch

Notationsreferenz: [Taktzahlen], Seite 71, [Musik ohne Metrum], Seite 48

Schnipsel: Abschnitt “Rhythms” in *Schnipsel*.

Referenz der Interna: Abschnitt “Timing_translator” in *Referenz der Interna*, Abschnitt “Score” in *Referenz der Interna*

1.3 Ausdrucksbezeichnungen



Dieser Abschnitt zeigt verschiedene Ausdrucksbezeichnungen, die zur Partitur hinzugefügt werden können.

1.3.1 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.
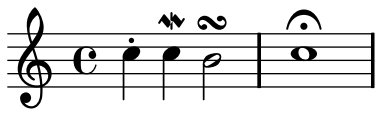
Artikulationszeichen und Verzierungen

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
```
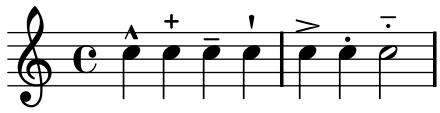
Die möglichen Werte für *Bezeichnung* sind aufgelistet in Abschnitt B.10 [Liste der Arktikulationszeichen], Seite 404. Ein Beispiel:

```
c4\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 Artikultationszeichen zugeordnet ist. Es gibt diese Abkürzungen für *marcato*, *stopped* (gedämpft), *tenuto*, *staccatissimo*, *accent*, *staccato*, and *portato*. Die ihnen entsprchenden Symbole werden also folgendermaßen notiert:

```
c4-^  c-+  c--  c-|
c4->  c-.  c2-_
```



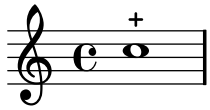
Die Regeln für die standardmäßige Platzierung von Artikulationszeichen werden in der Datei ‘scm/script.scm’ definiert. Artikulationszeichen und Onrnamente können manuell über oder unter dem System gesetzt werden, siehe Abschnitt 5.4.2 [Richtung und Platzierung], Seite 335.
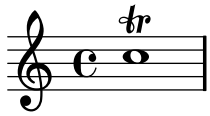
Ausgewählte Schnipsel

Die Standardwerte für Arkkikulationsabkürzungen 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-+ }
```
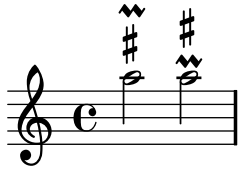




Die vertikale Anordnung von Beschriftungen kontrollieren

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 }
}
```



Siehe auch

Glossar: Abschnitt “Tenuto” in *Glossar*, Abschnitt “Akzent” in *Glossar*, Abschnitt “Staccato” in *Glossar*, Abschnitt “Portato” in *Glossar*.

Notationsreferenz: Abschnitt 5.4.2 [Richtung und Platzierung], Seite 335, Abschnitt B.10 [Liste der Arktikulationszeichen], Seite 404, [Triller], Seite 98.

Installierte Dateien: ‘`scm/script.scm`’.

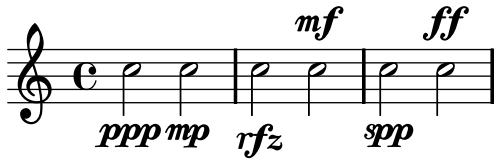
Schnipsel: Abschnitt “Expressive marks” in *Schnipsel*.

Referenz der Interna: Abschnitt “Script” in *Referenz der Interna*, Abschnitt “TextScript” in *Referenz der Interna*.

Dynamik

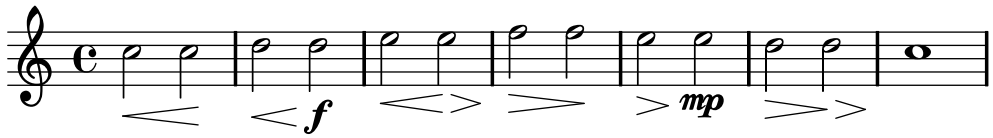
Absolute Dynamikbezeichnung wird mit Befehlen nach den Noten angezeigt, etwa `c4\ff`. Die vordefinierten Befehle lauten: `\ppppp`, `\pppp`, `\ppp`, `\pp`, `\p`, `\mp`, `\mf`, `\f`, `\ff`, `\fff`, `\ffff`, `\fp`, `\sf`, `\sff`, `\sp`, `\spp`, `\sfz`, and `\rfz`. Die Dynamikzeichen können manuell unter- oder oberhalb des Systems platziert werden, siehe Abschnitt 5.4.2 [Richtung und Platzierung], Seite 335.

```
c2\ppp c\mp
c2\rfz c^\mf
c2_\spp c^\ff
```



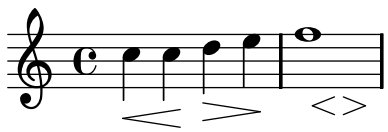
Eine *Crescendo*-Klammer wird mit dem Befehl `\<` begonnen und mit `\!`, einem absoluten Dynamikbefehl oder einer weiteren Crescento- oder Decrescendo-Klammer beendet. Ein *Decrescendo* beginnt mit `\>` und wird auch beendet mit `\!`, einem absoluten Dynamikbefehl oder einem weiteren Crescendo oder Decrescendo. `\cr` und `\decr` können anstelle von `\<` und `\>` benutzt werden. Die Befehle ergeben standardmäßig Crescendo-Klammern.

```
c2\< c\!
d2\< d\f
e2\< e\>
f2\> f\!
e2\> e\mp
d2\> d\>
c1\!
```
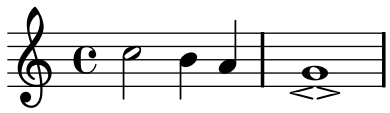


Unsichtbare Pausen werden benötigt, um mehrere Zeichen einer Note zuzuweisen.

```
c4\< c\! d\> e\!
<< f1 { s4 s4\< s4\> s4\! } >>
```
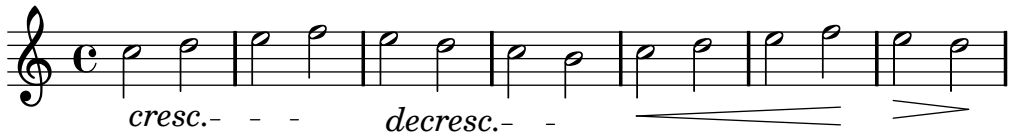


In manchen Situationen kann auch der `\espressivo`-Befehl geeignet sein, ein An- und Abschwellen einer Note anzuzeigen.

```
c2 b4 a
g1\espressivo
```



Crescendo und Decrescendo kann auch mit Text anstelle der Klammern angezeigt werden. Gestrichelte Linien werden ausgegeben, um die Dauer des Crescendos oder Decrescendos anzuzeigen. Die vorgegebenen Befehle, mit denen dieses Verhalten erreicht werden kann, sind `\crescTextCresc`, `\dimTextDecresc`, `\dimTextDecr` und `\dimTextDim`. Die entsprechenden Befehle `\crescHairpin` und `\dimHairpin` stellen wieder die spitzen Klammern ein:

```
\crescTextCresc
c2\< d | e f\!
\dimTextDecresc
e2\> d | c b\!
\crescHairpin
c2\< d | e f\!
\dimHairpin
e2\> d\!
```



Um neue absolute Dynamikzeichen oder Text, der mit ihnen angeordnet wird, zu erstellen, siehe [Neue Lautstärkezeichen], Seite 89.

Vertikale Position der Zeichen wird von der Funktion Abschnitt “DynamicLineSpanner” in *Referenz der Interna* verwaltet.

Vordefinierte Befehle

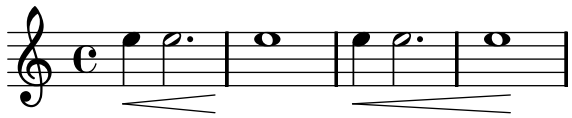
`\dynamicUp`, `\dynamicDown`, `\dynamicNeutral`, `\crescTextCresc`, `\dimTextDim`, `\dimTextDecr`, `\dimTextDecresc`, `\crescHairpin`, `\dimHairpin`.

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 Tatklinie beendet. Dieses Verhalten kann auch mit der Eigenschaft `'to-barline` geändert werden:
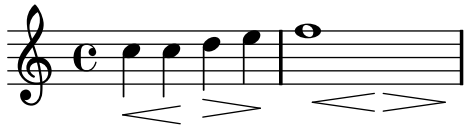
```
\relative c'' {
  e4\< e2.
  e1\!
  \override Hairpin #'to-barline = ##f
  e4\< e2.
  e1\!
}
```



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.
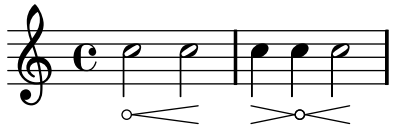
```
\relative c'' {
  c4\< c\! d\> e\!
  \override Hairpin #'minimum-length = #5
  << f1 { s4 s\< s\> s\! } >>
}
```



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\!
}
```



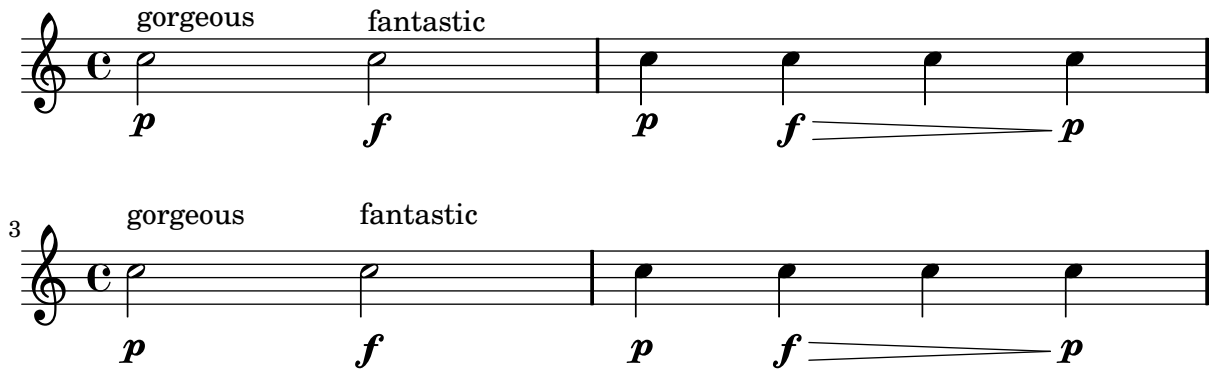
Vertikale Ausrichtung von Dynamik und Textbeschriftung beeinflussen

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.

```
Noten = \relative c'' {
  c2\p^\markup { gorgeous } c\f^\markup { fantastic }
  c4\p c\f\> c c\!\p
}

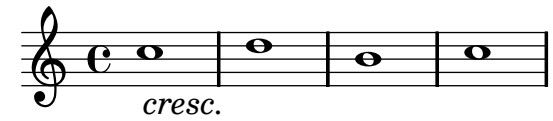
{
  \Noten \break
  \override DynamicLineSpanner #'staff-padding = #2.0
  \override DynamicLineSpanner #'Y-extent = #'(-1.5 . 1.5)
  \override TextScript #'Y-extent = #'(-1.5 . 1.5)
  \Noten
}
```



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 foldenge Weise unterdrückt werden:
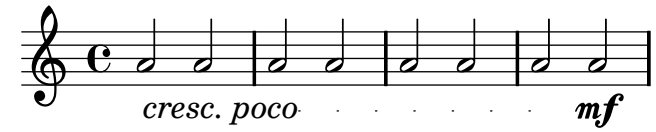
```
\relative c'' {
  \override DynamicTextSpanner #'dash-period = #-1.0
  \crescTextCresc
  c1\< | d | b | c\!
}
```



Text und Strecker-Stile für Dynamik-Texte ändern

Der Text, der für Crescendo und Decrescendo gestzt 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`, ander 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 “Gabel” in *Glossar*. Handbuch zum Lernen: Abschnitt “Articulation and dynamics” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 5.4.2 [Richtung und Platzierung], Seite 335, [Neue Lautstärkezeichen], Seite 89, Abschnitt 3.5.3 [Was geht in die MIDI-Ausgabe], Seite 325, Abschnitt 3.5.5 [MIDI-Lautstärke kontrollieren], Seite 327.

Schnipsel: Abschnitt “Expressive marks” in *Schnipsel*.
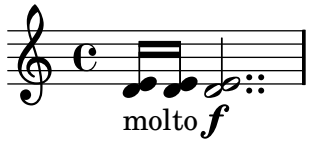
Referenz der Interna: Abschnitt “DynamicText” in *Referenz der Interna*, Abschnitt “Hairpin” in *Referenz der Interna*, Abschnitt “DynamicLineSpanner” in *Referenz der Interna*.

Neue Lautstärkezeichen

Die einfachste ARt, eigene Dynamikbezeichnungen zu erstellen, ist die Benutzung von `\markup`-(Textbeschriftungs)-Objekten.

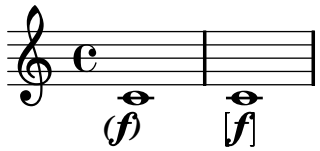
```
moltoF = \markup { molto \dynamic f }

\relative c' {
  <d e>16_\moltoF <d e>
  <d e>2..
}
```
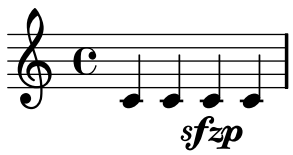


Mit einer Textbeschriftung können editorische Dynamikzeichen (in runden oder eckigen Klammern) erstellt werden. Die Syntax für den Textbeschriftungsmodus wird erklärt in Abschnitt 1.8.2 [Text formatieren], Seite 170.

```
rundF = \markup { \center-align \concat { \bold { \italic ( }
           \dynamic f \bold { \italic ) } } }
KastenF = \markup { \bracket { \dynamic f } }
\relative c' {
  c1_\rundF
  c1_\KastenF
}
```



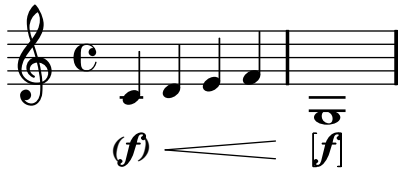
Einfache, mittig gesetzte Dynamikzeichen können schnell mit der `make-dynamic-script`-Funktion erstellt werden. Die Schriftart für Dynamikzeichen enthält nur die Buchstaben `f,m,p,r,s` sowie `z`.

```
sfzp = #(make-dynamic-script "sfzp")
\relative c' {
  c4 c c\sfzp c
}
```



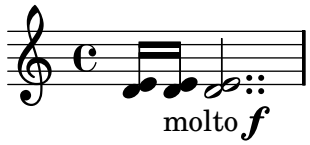
Allgemein gesagt kann `make-dynamic-script` jegliches Textbeschriftungsobjekt als Argument haben. Im nächsten Beispiel wird die vertikale Ausrichtung von den Beschriftungen (engl. markup) und den spitzen Klammern an der selben Linie durch `make-dynamic-script` gewährleistet, wenn beide an die selbe Note angehängt werden.

```
rundF = \markup { \center-align \concat {
           \normal-text { \bold { \italic ( } }
           \dynamic f
           \normal-text { \bold { \italic ) } } } }
KastenF = \markup { \bracket { \dynamic f } }
rundFDynamik = #(make-dynamic-script rundF)
KastenFDynamik = #(make-dynamic-script KastenF)
\relative c' {
  c4_\rundFDynamik\< d e f
  g,1_\KastenFDynamik
}
```



Anstelle dessen kann auch die Scheme-Form des Beschriftungs-Modus verwendet werden. Seine Syntax ist erklärt in Abschnitt 6.4.1 [Beschriftungskonstruktionen in Scheme], Seite 341.

```
moltoF = #(make-dynamic-script
            (markup #:normal-text "molto"
                    #:dynamic "f"))
\relative c' {
  <d e>16 <d e>
  <d e>2..\moltoF
}
```



Die Auswahl von Schriftarten in Textbeschriftungen ist erklärt in [Überblick über die wichtigsten Textbeschriftungsbefehle], Seite 172.

Siehe auch

Notationsreferenz: Abschnitt 1.8.2 [Text formatieren], Seite 170, [Überblick über die wichtigsten Textbeschriftungsbefehle], Seite 172, Abschnitt 6.4.1 [Beschriftungskonstruktionen in Scheme], Seite 341, Abschnitt 3.5.3 [Was geht in die MIDI-Ausgabe], Seite 325, Abschnitt 3.5.5 [MIDI-Lautstärke kontrollieren], Seite 327.

Schnipsel: Abschnitt “Expressive marks” in *Schnipsel*.
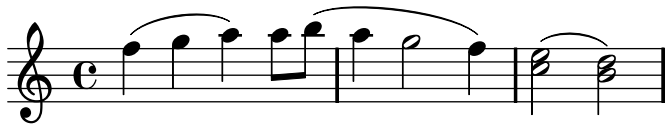
1.3.2 Bögen

Dieser Abschnitt erklärt, wie man verschiedene gebogene Ausdrucksbezeichnungen erstellt: Legato- und Phrasierungsbögen, Atemzeichen und Glissandos zu unbestimmten Tonhöhen.

Legatobögen
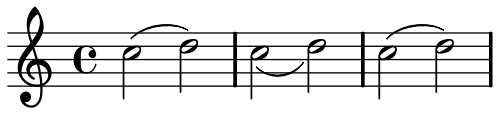
Ein Legatobogen (engl. slur) zeigt an, dass die Noten *legato* gespielt werden sollen. Er wird mit Klammern hinter den Notenwerten notiert.

```
f4( 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 5.4.2 [Richtung und Platzierung], Seite 335.
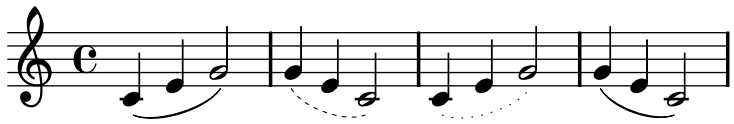
```
c2( d)
\slurDown
c2( d)
\slurNeutral
c2( d)
```



Gleichzeitige, überlappende Legatobögen sind nicht erlaubt, aber ein Phrasierungsbogen kann einen Legatobogen überlappen. Damit können zwei Bögen gleichzeitig ausgegeben werden. Siehe auch [Phrasierungsbögen], Seite 92.

Legatobögen können durchgehend, gepunket oder gestrichelt dargestellt werden. Standard ist der durchgehende Bogen:

```
c4( e g2)
\slurDashed
g4( e c2)
\slurDotted
c4( e g2)
\slurSolid
g4( e c2)
```



Vordefinierte Befehle

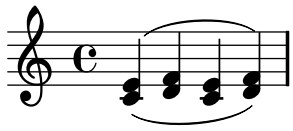
`\slurUp`, `\slurDown`, `\slurNeutral`, `\slurDashed`, `\slurDotted`, `\slurSolid`.

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>)
}
```



Siehe auch

Glossar: Abschnitt “Legatobogen” in *Glossar*.

Handbuch zum Lernen: Abschnitt “On the un-nestedness of brackets and ties” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 5.4.2 [Richtung und Platzierung], Seite 335, [Phrasierungsbögen], Seite 92.
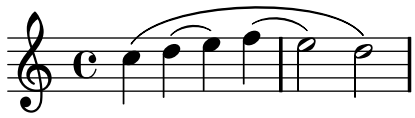
Schnipsel: Abschnitt “Expressive marks” in *Schnipsel*.

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

Phrasierungsbögen

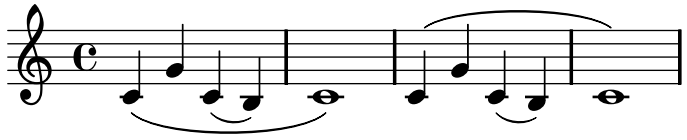
Ein Phrasierungsbogen verbindet Noten und wird verwendet, um einen musikalischen Ausdruck anzuzeigen. Er wird mit den Befehlen `\(` und `\)` eingegeben.

```
c4\( d( e) f(
e2) d\)
```



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 5.4.2 [Richtung und Platzierung], Seite 335.
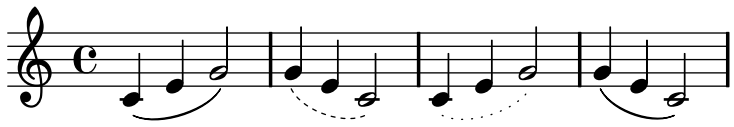
```
c4\( g' c,( b) | c1\)
\phrasingSlurUp
c4\( g' c,( b) | c1\)
```



Simultane oder überlappende Phrasierungsbögen sind nicht erlaubt.

Phrasierungsbögen können durchgehend, gepunket oder gestrichelt dargestellt werden. Standard ist der durchgehende Bogen:

```
c4\( e g2\)
\phrasingSlurDashed
g4\( e c2\)
\phrasingSlurDotted
c4\( e g2\)
\phrasingSlurSolid
g4\( e c2\)
```



Vordefinierte Befehle

`\phrasingSlurUp`, `\phrasingSlurDown`, `\phrasingSlurNeutral`, `\phrasingSlurDashed`, `\phrasingSlurDotted`, `\phrasingSlurSolid`.

Siehe auch

Handbuch zum Lernen: Abschnitt “On the un-nestedness of brackets and ties” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 5.4.2 [Richtung und Platzierung], Seite 335.
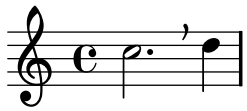
Schnipsel: Abschnitt “Expressive marks” in *Schnipsel*.

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

Atemzeichen

Atemzeichen werden mit dem Befehl `\breathe` eingegeben.

```
c2. \breathe d4
```



Musikalische Zeichen für Atemzeichen in Alter Notation, auch Divisiones genannt, sind unterstützt. Für Einzelheiten siehe [Divisiones], Seite 286.

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
}
```
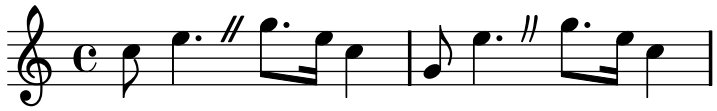


Eine Zäsur einfügen

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 “Zäsur” in *Glossar*.

Notationsreferenz: [Divisiones], Seite 286.

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

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

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.

```
c2-\bendAfter #+4
c2-\bendAfter #-4
c2-\bendAfter #+8
c2-\bendAfter #-8
```



Das Minuszeichen (`-`) direkt vor dem `\bendAfter`-Befehl ist *notwendig* um unbestimmte Glissandos zu notieren.

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 #-3
  c2-\bendAfter #+8
  c2-\bendAfter #-6
}
```



Siehe auch

Glossar: Abschnitt “Glissando nach unten” in *Glossar*, Abschnitt “doit” in *Glossar*.

Schnipsel: Abschnitt “Expressive marks” in *Schnipsel*.
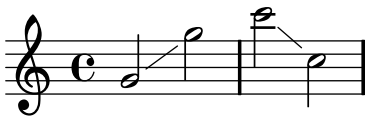
1.3.3 Linien

Dieser Abschnitt zeigt, wie man verschiedene Ausdrucksbezeichnungen erstellt, die sich linear erstrecken: Glissando, Arppegio und Triller.

Glissando

Ein *Glissando* wird mit dem Befehl `\glissando` auf eine Note folgend notiert:

```
g2\glissando g'
c2\glissando c,
```
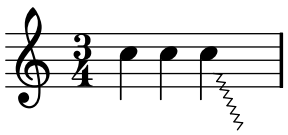


Verschiedene Glissando-Stile sind möglich. Für Einzelheiten siehe Abschnitt 5.4.7 [Zeilenstile], Seite 336.

Ausgewählte Schnipsel

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 5.4.7 [Zeilenstile], Seite 336.

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

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

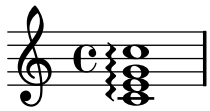
Bekannte Probleme und Warnungen

Printing text over the line (such as *gliss.*) is not supported.

Arpeggio
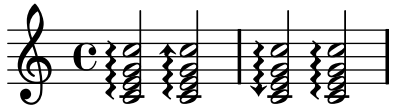
Ein *Arpeggio* als Zeichen, dass ein Akkord gebrochen gespielt werden soll, kann mit dem Befehl `\arpeggio` hinter der Akkord-Konstruktion erzeugt werden.

```
<c e g c>1\arpeggio
```
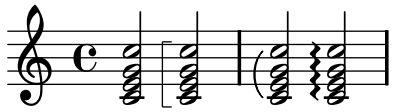


Unterschiedliche Arpeggio-Typen können benutzt werden. `\arpeggioNormal` stellt wieder das normale Verhalten her:

```
<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:

```
<c e g c>2
\arpeggioBracket
<c e g c>2\arpeggio
\arpeggioParenthesis
<c e g c>2\arpeggio
\arpeggioNormal
<c e g c>2\arpeggio
```



Ein Arpeggio kann auch explizit ausgeschrieben werden, indem Überbindungsbögen benutzt werden. Für mehr Information siehe [Bindebögen], Seite 36.

Vordefinierte Befehle

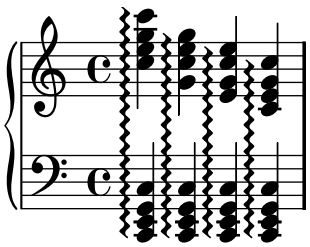
`\arpeggio`, `\arpeggioArrowUp`, `\arpeggioArrowDown`, `\arpeggioNormal`, `\arpeggioBracket`, `\arpeggioParenthesis`.

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.

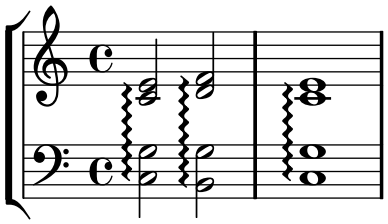
```
\new PianoStaff \relative c'' <<
  \set PianoStaff.connectArpeggios = ##t
  \new Staff {
    <c e g c>4\arpeggio
    <g c e g>4\arpeggio
    <e g c e>4\arpeggio
    <c e g c>4\arpeggio
  }
  \new Staff {
    \clef bass
    \repeat unfold 4 {
      <c,, e g c>4\arpeggio
    }
  }
>>
```



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.

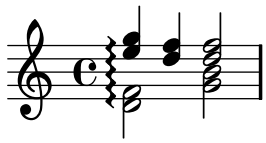
```
\score {
  \new StaffGroup {
    \set Score.connectArpeggios = ##t
    <<
      \new Voice \relative c' {
        <c e>2\arpeggio
        <d f>2\arpeggio
        <c e>1\arpeggio
      }
      \new Voice  \relative c {
        \clef bass
         <c g'>2\arpeggio
         <b g'>2\arpeggio
         <c g'>1\arpeggio
      }
    >>
  }
  \layout {
    \context {
      \Score
      \consists "Span_arpeggio_engraver"
    }
  }
}
```



Arpeggios zwischen unterschiedlichen Stimmen erzeugen

Ein Arpeggio kann zwischen Noten aus unterschidlichen 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: [Bindebögen], Seite 36.

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

Referenz der Interna: Abschnitt “Arpeggio” 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.

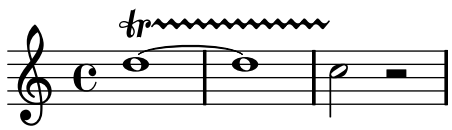
Die Arpeggios im Klammer-Stil funktionieren nicht über mehrere Notensysteme.

Triller

Kurze *Triller* ohne eine Dauer werden nmit dem Befehl `\trill` notiert, siehe auch [Artikulationszeichen und Verzierungen], Seite 83.
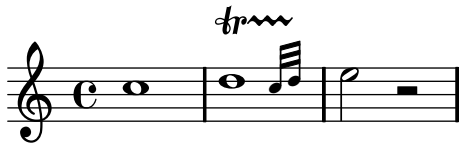
Längere Triller mit einer Dauer werden mit den Befehlen `\startTrillSpan` zu Beginn und `\stopTrillSpan` am Ende erstellt.

```
d1~\startTrillSpan
d1
c2\stopTrillSpan r2
```



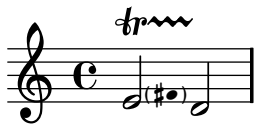
Das nächste Beispiel zeigt Triller in Kombination mit einem Vorschlag. Die Syntax dieser Konstruktion und die Methode, wie man die Vorschläge genau positioniert, ist beschrieben in [Verzierungen], Seite 77.

```
c1 \afterGrace
d1\startTrillSpan { c32[ d]\stopTrillSpan }
e2 r2
```
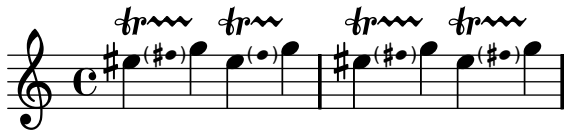


Triller, die auf einer bestimmten Note ausgeführt werden sollen, können mit dem Befehl `pitchedTrill` notiert werden. Das erste Argument ist die Hauptnote, das zweite die Note, auf der getrillert wird. Sie wird als Note ohne Hals in Klammern ausgegeben.

```
\pitchedTrill e2\startTrillSpan fis
d\stopTrillSpan
```



Im nächsten Beispiel ist der zweite Triller nicht eindeutig notiert, denn das Versetzungszeichen der Trillernote ist nicht ausgegeben. Man kann das Versetzungszeichen erzwingen. Der zweite Takt zeigt diese Methode:

```
\pitchedTrill eis4\startTrillSpan fis
g\stopTrillSpan
\pitchedTrill eis4\startTrillSpan fis
g\stopTrillSpan
\pitchedTrill eis4\startTrillSpan fis
g\stopTrillSpan
\pitchedTrill eis4\startTrillSpan fis!
g\stopTrillSpan
```



Vordefinierte Befehle

`\startTrillSpan`, `\stopTrillSpan`.

Siehe auch
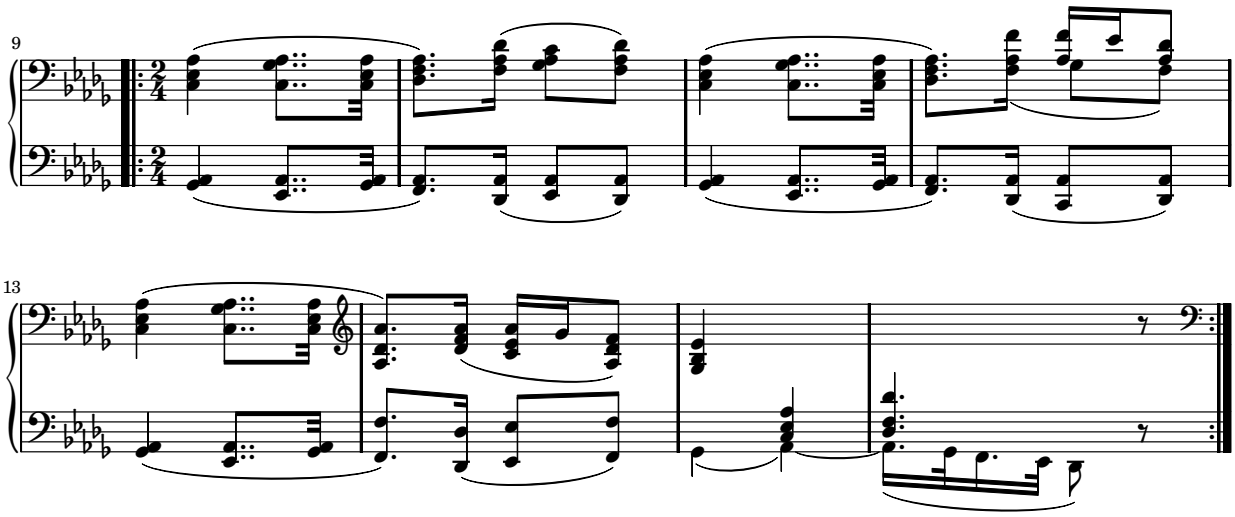
Glossar: Abschnitt “Triller” in *Glossar*.

Notationsreferenz: [Artikulationszeichen und Verzierungen], Seite 83, [Verzierungen], Seite 77.

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

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

1.4 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 geschreiben, sondern zwischen zwei Wiederholungstaktlinien 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 repititive 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.

1.4.1 Lange Wiederholungen

Normale Wiederholungen
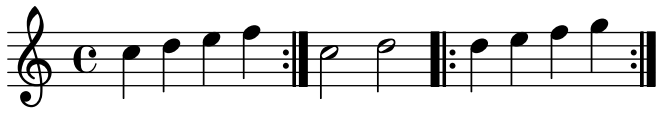
Die Syntax für normale Wiederholungen ist

`\repeat` *`Typ Wiederholungszähler musikAusdr`*

wobei *musikAusdr* ein musikalischer Ausdruck ist. Alternative Schlüsse können mit `\alternative` gesetzt werden. Damit die alternativen Schlüsse von den wiederholten Noten abgegrenzt werden, müssen sie in geschweiften Klammern zusammengefasst werden. Wenn es mehr Wiederholungen gibt, als Alternativen angegeben sind, erhalten die ersten Wiederholungen den ersten Schluss.
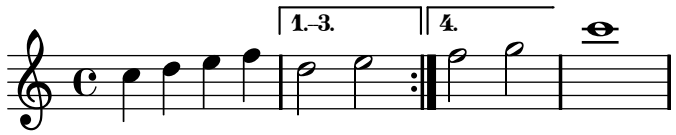
Normale Wiederholungen ohne alternative Schlüsse:

```
\repeat volta 2 { c4 d e f }
c2 d
\repeat volta 2 { d4 e f g }
```
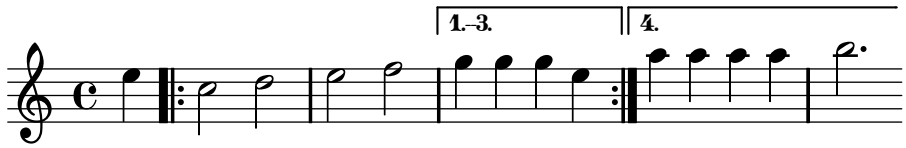


Normale Wiederholungen mit alternativen Schlüssen:

```
\repeat volta 4 { c4 d e f }
\alternative {
  { d2 e }
  { f2 g }
}
c1
```



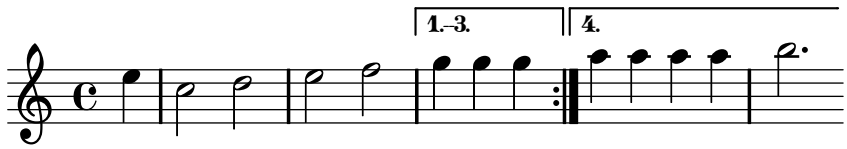
Normale Wiederholungen mit Auftakt können auf zwei Arten notiert werden:

```
\partial 4
e |
\repeat volta 4 { c2 d | e2 f | }
\alternative {
  { g4 g g e }
  { a4 a a a | b2. }
}
```



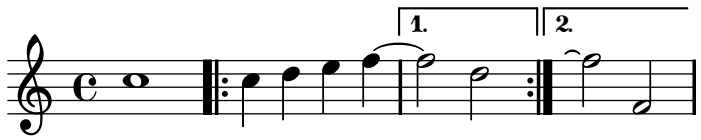
oder

```
\partial 4
\repeat volta 4 { e4 | c2 d | e2 f | }
\alternative {
  { \partial 4*3 g4 g g }
  { a4 a a a | b2. }
}
```



Bindebögen können auch an eine zweite Klammer angefügt werden:

```
c1
\repeat volta 2 { c4 d e f ~ }
\alternative {
  { f2 d }
  { f2\repeatTie f, }
}
```
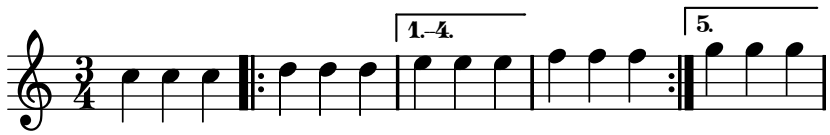


Ausgewählte Schnipsel

Shortening volta brackets

By default, the volta brackets will be drawn over all of the alternative music, but it is possible to shorten them by setting `voltaSpannerDuration`. In the next example, the bracket only lasts one measure, which is a duration of 3/4.

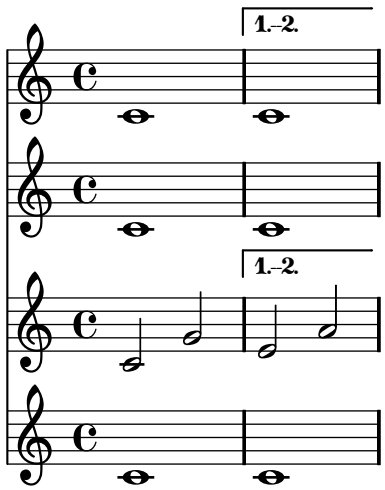
```
\relative c'' {
  \time 3/4
  c4 c c
  \set Score.voltaSpannerDuration = #(ly:make-moment 3 4)
  \repeat volta 5 { d4 d d }
  \alternative {
    {
      e4 e e
      f4 f f
    }
    { g4 g g }
  }
}
```



Adding volta brackets to additional staves

The `Volta_engraver` by default resides in the `Score` context, and brackets for the repeat are thus normally only printed over the topmost staff. This can be adjusted by adding the `Volta_engraver` to the `Staff` context where the brackets should appear; see also the "Volta multi staff" snippet.

```
<<
  \new Staff { \repeat volta 2 { c'1 } \alternative { c' } }
  \new Staff { \repeat volta 2 { c'1 } \alternative { c' } }
  \new Staff \with { \consists "Volta_engraver" } { c'2 g' e' a' }
  \new Staff { \repeat volta 2 { c'1 } \alternative { c' } }
>>
```



Siehe auch

Glossar: Abschnitt “Wiederholungen” in *Glossar*, Abschnitt “Volta-Klammer” in *Glossar*.

Notationsreferenz: [Taktlinien], Seite 69, Abschnitt 5.1.3 [Umgebungs-Plugins verändern], Seite 335.

Schnipsel: Abschnitt “Repeats” in *Schnipsel*.

Referenz der Interna: Abschnitt “VoltaBracket” in *Referenz der Interna*, Abschnitt “RepeatedMusic” in *Referenz der Interna*, Abschnitt “VoltaRepeatedMusic” in *Referenz der Interna*, Abschnitt “UnfoldedRepeatedMusic” in *Referenz der Interna*.

Bekannte Probleme und Warnungen

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.

Die Taktposition wird bei einer alternativen Endung nicht mitgeteilt, so dass nach einer Wiederholung diese Information manuell angegeben werden muss, entweder durch setzen von `Score.measurePosition` oder indem der Befehl `\partial` benutzt wird. Gleichermaßen werden auch Bindebögen nicht wiederholt.

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 Taktlinien eingegeben werden. Mehr Information in [Taktlinien], Seite 69.
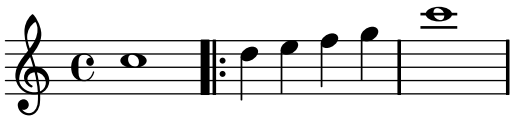
Die Eigenschaft `repeatCommands` kann verwendet werden, um das Aussehen der Wiederholungen zu beeinflussen. Ihr Argument ist eine Scheme-Liste an Wiederholungsbefehlen.

`start-repeat`

Setzt eine `|:` Taktlinie.

```
c1
\set Score.repeatCommands = #'(start-repeat)
```

```
d4 e f g
c1
```
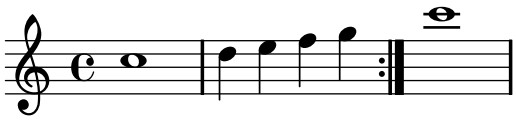


Der Notensatzpraxis folgend werden Wiederholungstaktlinien nicht zu Beginn eines Stückes gesetzt.

`end-repeat`

Setzt eine `:|` Taktlinie.

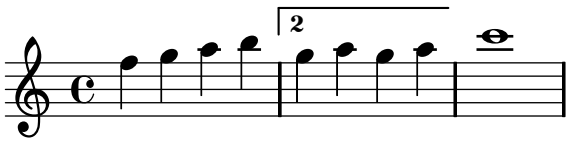
```
c1
d4 e f g
\set Score.repeatCommands = #'(end-repeat)
c1
```



`(volta` *`Zahl`*`) ... (volta #f)`

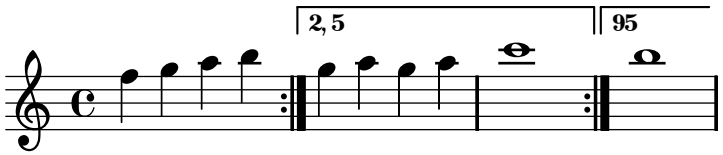
Setzt eine Volta-Klammer mit der Beschriftung *Nummer*. Die Volta-Klammer muss explizit beendet werden, sonst wird sie nicht ausgegeben.

```
f4 g a b
\set Score.repeatCommands = #'((volta "2"))
g4 a g a
\set Score.repeatCommands = #'((volta #f))
c1
```
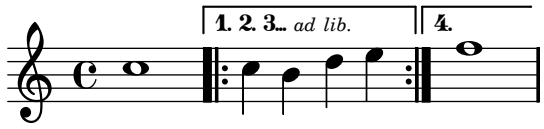


Mehrfache Wiederholungszeichen können an der selben Stelle vorkommen:

```
f4 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 1.8.2 [Text formatieren], Seite 170. Die einfachste Art Text zu benutzen ist, die Beschriftung zuerst zu definieren und dann die Beschriftung in einer Scheme-Liste einzufügen.

```
VoltaAdLib = \markup { 1. 2. 3... \text \italic { ad lib. } }
\relative c'' {
  c1
  \set Score.repeatCommands = #(list(list 'volta VoltaAdLib) 'start-repeat)
  c4 b d e
  \set Score.repeatCommands = #'((volta #f) (volta "4.") end-repeat)
  f1
  \set Score.repeatCommands = #'((volta #f))
}
```

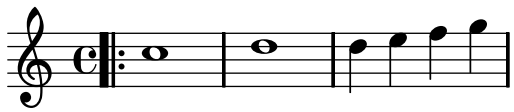


Ausgewählte Schnipsel

Printing a repeat sign at the beginning of a piece

A `|:` bar line can be printed at the beginning of a piece, by overriding the relevant property:

```
\relative c'' {
  \once \override Score.BreakAlignment #'break-align-orders =
        #(make-vector 3 '(instrument-name
                          left-edge
                          ambitus
                          span-bar
                          breathing-sign
                          clef
                          key-signature
                          time-signature
                          staff-bar
                          custos
                          span-bar))
  \bar "|:"
  c1
  d1
  d4 e f g
}
```



Siehe auch

Notationsreferenz: [Taktlinien], Seite 69, Abschnitt 1.8.2 [Text formatieren], Seite 170.

Schnipsel: Abschnitt “Repeats” in *Schnipsel*.

Referenz der Interna: Abschnitt “VoltaBracket” in *Referenz der Interna*, Abschnitt “RepeatedMusic” in *Referenz der Interna*, Abschnitt “VoltaRepeatedMusic” in *Referenz der Interna*.
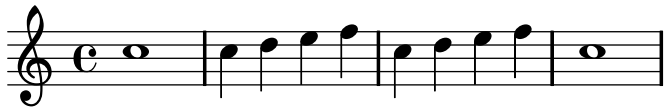
Ausgeschriebene Wiederholungen

Mit dem `unfold`-Befehl können Wiederholungen eingesetzt werden, um repitetive Musik zu notieren. Die Syntax ist

```
\repeat unfold Wiederholgunszähler musikAusdr
```
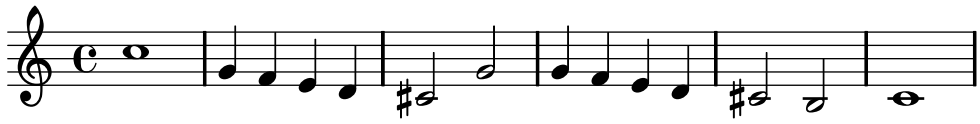
wobei *musikAusdr* ein musikalischer Ausdruck ist und *Wiederholungszähler* die Anzahl bezeichnet, mit der *musikAusdr* wiederholt wird.

```
c1
\repeat unfold 2 { c4 d e f }
c1
```



Ausgeschriebene Wiederholungen können auch alternative Schlüsse haben. Wenn mehr Wiederholungen als alternative Schlüsse notiert werden, wird der erste Schluss für die ersten Wiederholungen benutzt.

```
c1
\repeat unfold 2 { g4 f e d }
  \alternative {
    { cis2 g' }
    { cis,2 b }
  }
c1
```



Siehe auch

Schnipsel: Abschnitt “Repeats” in *Schnipsel*.

Referenz der Interna: Abschnitt “RepeatedMusic” in *Referenz der Interna*, Abschnitt “UnfoldedRepeatedMusic” in *Referenz der Interna*.

1.4.2 Kurze Wiederholungen

Dieser Abschnitt zeigt, wie man kurze Wiederholungen notiert. Kurze Wiederholungen haben zwei grundlegende Formen: Wiederholungen von einer Note bis zu zwei Takten, die mit Schrägstrichen oder Prozentzeichen dargestellt werden, und Tremolos.

Prozent-Wiederholungen
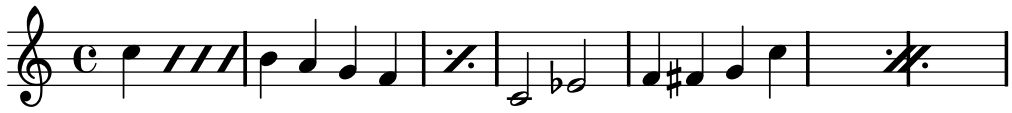
Kurze wiederholte Musikphrasen werden unterstützt. Dabei werden die Noten einmal gedruckt und dann durch ein spezielles Zeichen ersetzt. Phrasen, die kürzer als ein Takt sind, durch einen Schrägstrich dargestellt, Phrasen von ein oder zwei Takten Dauer werden durch ein dem Prozentzeichen ähnlichen Zeichen markiert. Die Syntax lautet

```
\repeat percent Wiederholungszahl musikAusdr
```

wobei *musikAusdr* ein musikalischer Ausdruck ist.

```
\repeat percent 4 { c4 }
\repeat percent 2 { b4 a g f }
```

```
\repeat percent 2 { c2 es | f4 fis g c | }
```

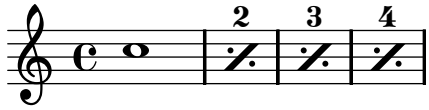


Ausgewählte Schnipsel

Percent repeat counter

Measure repeats of more than two repeats can get a counter when the convenient property is switched, as shown in this example:

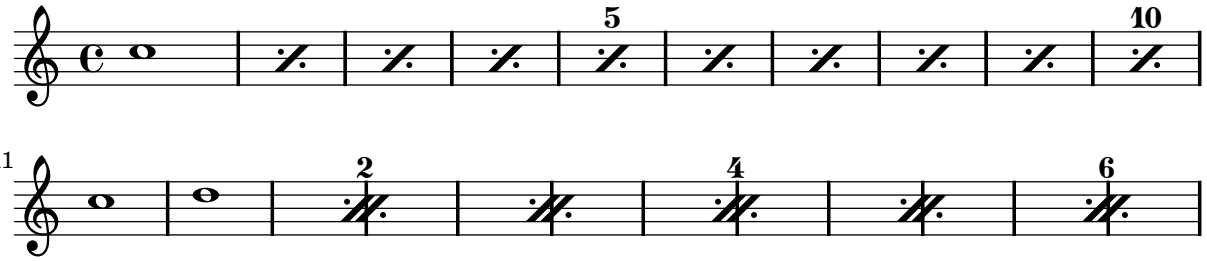
```
\relative c'' {
  \set countPercentRepeats = ##t
  \repeat percent 4 { c1 }
}
```



Percent repeat count visibility

Percent repeat counters can be shown at regular intervals by setting the context property `repeatCountVisibility`.
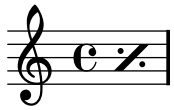
```
\relative c'' {
  \set countPercentRepeats = ##t
  \set repeatCountVisibility = #(every-nth-repeat-count-visible 5)
  \repeat percent 10 { c1 } \break
  \set repeatCountVisibility = #(every-nth-repeat-count-visible 2)
  \repeat percent 6 { c1 d1 }
}
```



Isolated percent repeats

Isolated percents can also be printed. This is done by entering a multi-measure rest with a different print function:

```
\relative c'' {
  \override MultiMeasureRest #'stencil
    = #ly:multi-measure-rest::percent
  \override MultiMeasureRest #'thickness = #0.48
  R1
}
```



Siehe auch

Glossar: Abschnitt “Prozent-Wiederholung” in *Glossar*, Abschnitt “gleichartig” in *Glossar*.

Schnipsel: Abschnitt “Repeats” in *Schnipsel*.

Referenz der Interna: Abschnitt “RepeatSlash” in *Referenz der Interna*, Abschnitt “PercentRepeat” in *Referenz der Interna*, Abschnitt “DoublePercentRepeat” in *Referenz der Interna*, Abschnitt “DoublePercentRepeatCounter” in *Referenz der Interna*, Abschnitt “PercentRepeatCounter” in *Referenz der Interna*, Abschnitt “PercentRepeatedMusic” in *Referenz der Interna*.

Bekannte Probleme und Warnungen
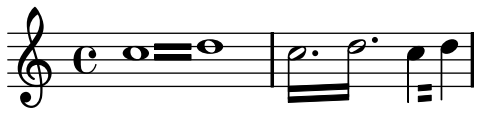
Nur drei Arten von Prozent-Wiederholungen sind unterstützt: ein einfacher Schrägstrich, der einen Taktschlag darstellt (unabhängig von der wirklichen Dauer der wiederholten Noten), ein einfacher Schrägstrich mit Punkten, der einen ganzen wiederholten Takt darstellt und zwei Schrägstriche mit Punkten über eine Taktlinie gedruckt, der zwei ganze Takte darstellt. Weder können mehrere Schrägstriche für Taktwiederholungen von Sechzehntelnoten dargestellt werden, noch zwei Striche mit Punkten für nur einen Takt, der aus unterschiedlichen Notenwerten besteht.

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:

```
\repeat tremolo 8 { c16 d }
\repeat tremolo 6 { c16 d }
\repeat tremolo 2 { c16 d }
```
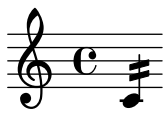


Die `\repeat tremolo`-Syntax braucht genau zwei Noten innerhalb der geschweiften Klammern, und die Anzahl der Wiederholungen muss einem Wert entsprechen, der mit einfachen oder punktierten Noten ausgedrückt werden kann. `\repeat tremolo 7` funktioniert und setzt Tremolo für die Dauer einer Doppeltpunktierten, 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:
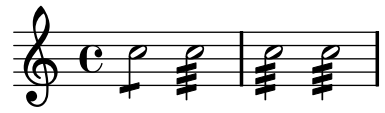
```
\repeat tremolo 4 c'16
```



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`):

```
c2:8 c:32
c: c:
```
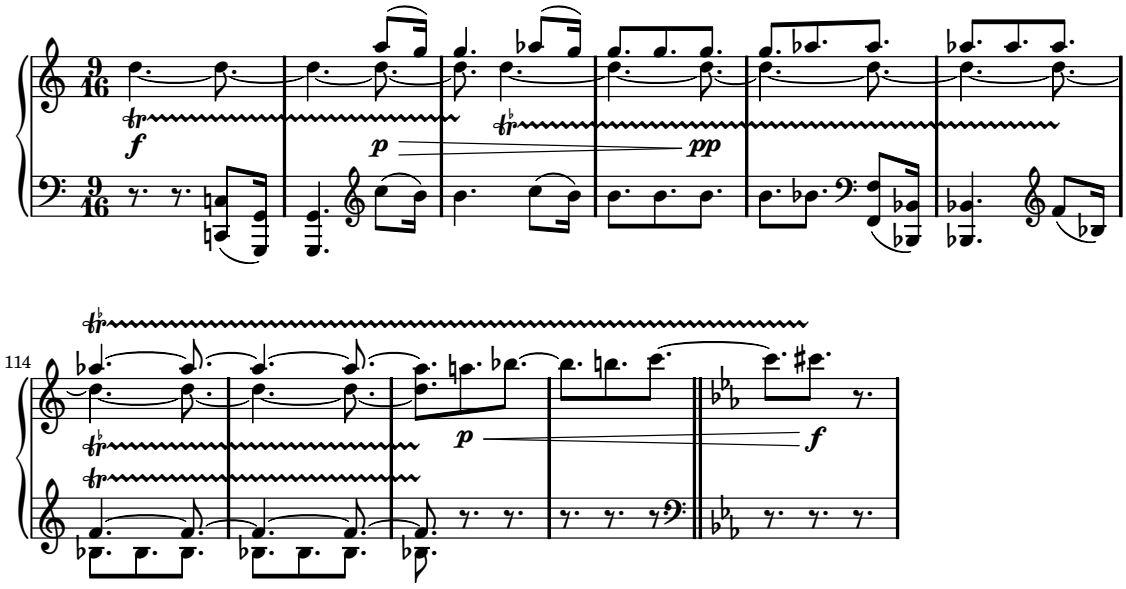


Siehe auch

Schnipsel: Abschnitt “Repeats” in *Schnipsel*.

Bekannte Probleme und Warnungen

Tremolo über Notensysteme hinweg funktioniert nicht gut.

1.5 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.

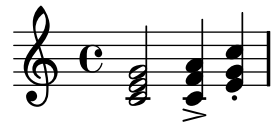
1.5.1 Eine einzelne Stimme

Dieser Abschnitt behandelt gleichzeitige Noten innerhalb derselben Stimme.

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 und/oder eine Anzahl an Artikulationsbezeichnungen folgen, genauso wie bei einfachen Noten.

```
<c e g>2 <c f a>4-> <e g c>-.
```



Der relative Modus kann auch für Tonhöhen in Akkorden benutzt werden. Die Oktave jeder Tonhöhe wird relativ zur vorhergehenden Tonhöhe bestimmt. Eine Ausnahme bildet die erste Tonhöhe in einem Akkord: ihre Oktave wird bestimmt relativ zur *ersten* Tonhöhe des vorherigen Akkords.

Mehr Information über Akkorden findet sich in Abschnitt 2.7 [Notation von Akkorden], Seite 260.

Siehe auch

Musikglossar: Abschnitt “Akkord” in *Glossar*.

Handbuch zum Lernen: Abschnitt “Combining notes into chords” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt 2.7 [Notation von Akkorden], Seite 260.
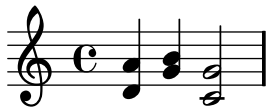
Schnipsel: Abschnitt “Simultaneous notes” in *Schnipsel*.

Gleichzeitige Ausdrücke
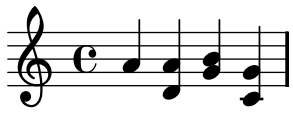
Eine oder mehrere musikalische Ausdrücke, die in doppelte spitze Klammern eingeschlossen werden, werden gleichzeitig gesetzt. Wenn der erste Ausdruck mit einer einzelnen Note beginnt oder die gesamte Konstruktion explizit in einer einzelnen Stimme erstellt wird, wird auch nur ein Notensystem erstellt. In anderem Falle werden die Elemente der simultanen Konstruktion auf unterschiedlichen Systemen gesetzt.

Das nächste Beispiel zeigt simultane Konstruktionen auf einem System:

```
\new Voice {  % explizit eine Stimme
  << {a4 b g2} {d4 g c,2} >>
}
```
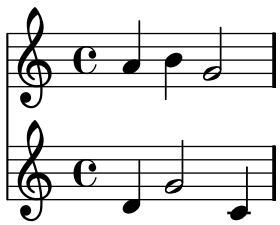


```
% einzelne erste Note
a << {a4 b g}  {d4 g c,} >>
```



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.

Das nächste Beispiel zeigt, wie ein simultaner Ausdruck implizit mehrere Systeme erstellt:

```
% keine einzelne erste Note
<< {a4 b g2}  {d4 g2 c,4} >>
```

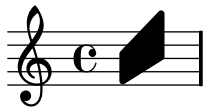


In diesem Fall stellt der unterschiedliche Rhythmus kein Problem dar.

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:

```
\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 – anderfalls 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.

1.5.2 Mehrere Stimmen
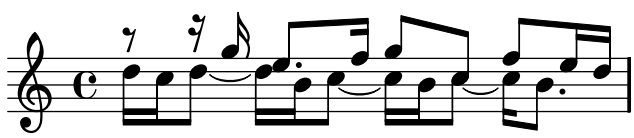
Dieser Abschnitt behandelt gleichzeitige Noten in mehreren Stimmen oder mehreren Systemen.

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"
    { \voiceOne r8 r16 g e8. f16 g8[ c,] f e16 d }
  \new Voice= "second"
    { \voiceTwo d16 c d8~ d16 b c8~ c16 b c8~ c16 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.
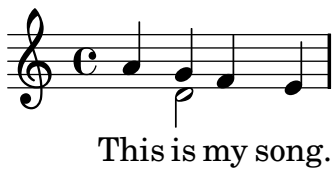
Vorrübergehende polyphone Passagen

Ein vorrübergehender 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:

```
<<
  \new Voice = "Melodie" {
    a4
    <<
      {
        \voiceOne
        g f
      }
      \new Voice {
        \voiceTwo
        d2
      }
    >>
    \oneVoice
    e4
  }
  \new Lyrics \lyricsto "Melodie" {
  This is my song.
  }
>>
```
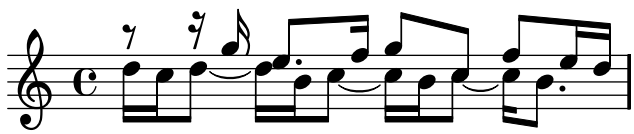


Hierbei sind die Befehle `\voiceOne` und `\voiceTwo` notwendig, um die Einstellungen für jede Stimme zu initialisieren.

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:
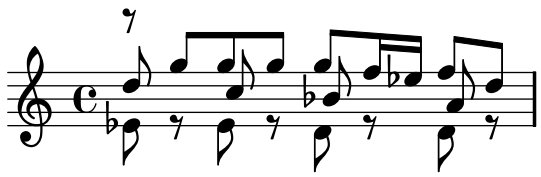
```
<<
  { r8 r16 g e8. f16 g8[ c,] f e16 d }
  \\
  { d16 c d8~ d16 b c8~ c16 b c8~ c16 b8. }
>>
```



Diese Syntax kann benutzt werden, wenn es keine Rolle spielt, ob vorrübergehend 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` zugewiesen bekommt. Unsichtbare Pause werden eingesetzt, damit keine doppelten Pausen ausgegeben werden.

```
<<
  { r8 g g  g g f16 ees f8 d }
  \\
  { ees,8 r ees r d r d r }
  \\
  { d'8 s c s bes s a s }
>>
```
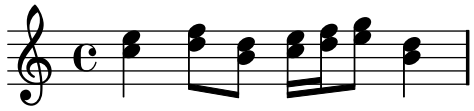


Es wird sehr empfohlen, in allen außer den allereinfachsten Stücken explizite Stimmenkontexte zu erstellen, wie erklärt inAbschnitt “Contexts and engravers” in *Handbuch zum Lernen* und Abschnitt “Explicitly instantiating voices” in *Handbuch zum Lernen*.

Identische Rhythmen

Wenn paralelle 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 <<
  { e4 f8 d e16 f g8 d4 }
  { c4 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 “Voices contain music” in *Handbuch zum Lernen*, Abschnitt “Explicitly instantiating voices” in *Handbuch zum Lernen*.

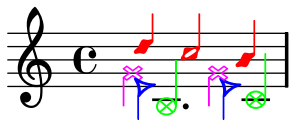
Notationsreferenz: [Schlagzeugsysteme], Seite 250, [Unsichtbare Pausen], Seite 40.

Schnipsel: Abschnitt “Simultaneous notes” in *Schnipsel*.

Stimmenstile

Stimmen können unterschiedliche Farben erhalten, um einfach erkennbar zu sein:

```
<<
  { \voiceOneStyle d4 c2 b4 }
  \\
  { \voiceTwoStyle e,2 e }
  \\
  { \voiceThreeStyle b2. c4 }
  \\
  { \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 “I’m hearing Voices” in *Handbuch zum Lernen*, Abschnitt “Other sources of information” in *Handbuch zum Lernen*.
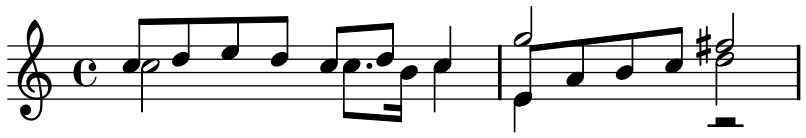
Schnipsel: Abschnitt “Simultaneous notes” in *Schnipsel*.

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.
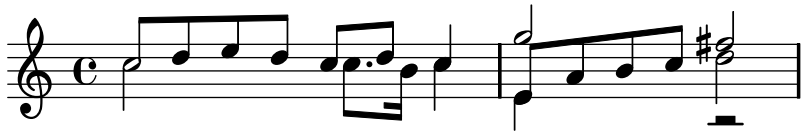
```
<<
  {
    c8 d e d c d c4
    g'2 fis
  } \\ {
    c2 c8. b16 c4
```

```
    e,2 r
  } \\ {
    \oneVoice
    s1
    e8 a b c d2
  }
>>
```



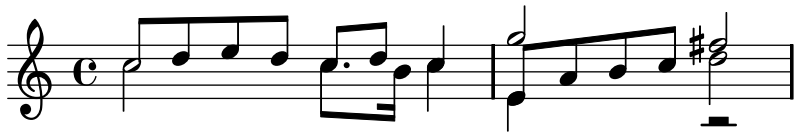
Noten mit unterschiedlichen Notenköpfen können verschmolzen werden, mit der Ausnahme von Halben- und Viertelnotenköpfen:

```
<<
  {
    \mergeDifferentlyHeadedOn
    c8 d e d c d c4
    g'2 fis
  } \\ {
    c2 c8. b16 c4
    e,2 r
  } \\ {
    \oneVoice
    s1
    e8 a b c d2
  }
>>
```



Auch Köpfe mit unterschiedlichen Punktierungen können verschmolzen werden:
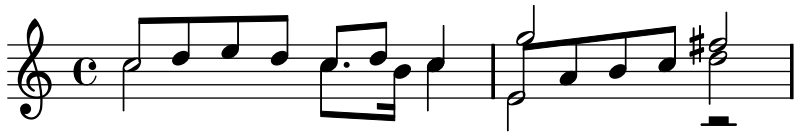
```
<<
  {
    \mergeDifferentlyHeadedOn
    \mergeDifferentlyDottedOn
    c8 d e d c d c4
    g'2 fis
  } \\ {
    c2 c8. b16 c4
    e,2 r
  } \\ {
    \oneVoice
    s1
    e8 a b c d2
  }
>>
```



Die Halbe und die Achtel am Anfang des zweiten Taktes werden fehlerhaft verschmolzen, weil `\mergeDifferentlyHeadedOn` (Unterschiedliche Köpfe Verschmelzen An) nicht richtig arbeiten kann, wenn drei oder mehr Noten zur gleichen Zeit auftreten – in diesem Fall wird eine Warnung ausgegeben. Damit die Verschmlezung richtig funktionieren kann, muss ein `\shift` (Verschiebung) der Note hinzugefügt werden, die nicht mit verschmolzen werden soll. In diesem Fall wurde `\shiftOn` gesetzt, um das oberste *g* aus der Kolumne zu entfernen. Jetzt funktioniert `\mergeDifferentlyHeadedOn` so wie es soll.

```
<<
  {
    \mergeDifferentlyHeadedOn
    \mergeDifferentlyDottedOn
    c8 d e d c d c4
    \shiftOn
    g'2 fis
  } \\ {
    c2 c8. b16 c4
    e,2 r
  } \\ {
    \oneVoice
    s1
    e8 a b c d2
  }

>>
```



Die Befehle `\shiftOn`, `\shiftOnn` und `\shiftOnnn` bezeichnen den Grad, mit dem Noten der aktuellen Stimme verschoben werden sollen. Die äußeren Stimmen (normalerweise Stimme eins und zwei) haben diese Funktion standardmäßig ausgeschaltet (`\shiftOff`), während die inneren Stimmen (drei und vier) ein `\shiftOn` eingestellt haben (Verschiebung an). Die Befehle `\shiftOnn` und `\shiftOnnn` stellen weitere Verschiebungesebenen dar.

Noten werden nur verschmolzen, wenn ihre Hälse in gegengesetzte Richtungen zeigen (also etwa wie `Voice` 1 und 2).

Vordefinierte Befehle

`\mergeDifferentlyDottedOn`, `\mergeDifferentlyDottedOff`, `\mergeDifferentlyHeadedOn`, `\mergeDifferentlyHeadedOff`, `\shiftOn`, `\shiftOnn`, `\shiftOnnn`, `\shiftOff`.
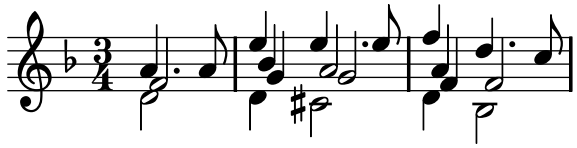
Ausgewählte Schnipsel

Additional voices to avoid collisions

In some instances of complex polyphonic music, additional voices are necessary to prevent collisions between notes. If more than four parallel voices are needed, additional voices can be added by defining a variable using the Scheme function `context-spec-music`.

```
StimmeFünf = #(context-spec-music (make-voice-props-set 4) 'Voice)
\relative c'' {
```

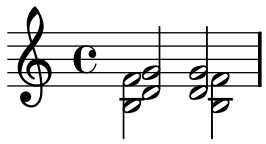
```
  \time 3/4 \key d \minor \partial 2
  <<
    { \voiceOne
      a4. a8
      e'4 e4. e8
      f4 d4. c8
    } \\ {
      \voiceThree
      f,2
      bes4 a2
      a4 s2
    } \\ {
      \StimmeFünf
      s2
      g4 g2
      f4 f2
    } \\ {
      \voiceTwo
      d2
      d4 cis2
      d4 bes2
    }
  >>
}
```



Forcing horizontal shift of notes

When the typesetting engine cannot cope, the following syntax can be used to override typesetting decisions. The units of measure used here are staff spaces.

```
\relative c' <<
  {
    <d g>2 <d g>
  }
  \\
  {
    <b f'>2
    \once \override NoteColumn #'force-hshift = #1.7
    <b f'>2
  }
>>
```



Siehe auch

Musikglossar: Abschnitt “Polyphonie” in *Glossar*.

Handbuch zum Lernen: Abschnitt “Multiple notes at once” in *Handbuch zum Lernen*, Abschnitt “Voices contain music” in *Handbuch zum Lernen*, Abschnitt “Collisions of objects” 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

Wenn `\mergeDifferentlyHeadedOn` mit einer Achtel- oder kürzeren Note benutzt wird, deren Hals nach oben zeigt, und einer Halben Note mit Hals nach unten, erhält der Hals der Achtelnote eine geringe Verschiebung, weil der schwarze und weiße Notenkopf eine unterschiedliche Breite beistzen.

Es gibt keine Unterstützung für Akkorde, in denen die gleiche Note mit unterschiedlichen Versetzungszeichen im selben Akkord auftaucht. In diesem Fall wird empfohlen, enharmonische Töne zu verwenden, oder die besondere Cluster-Notation (siehe [Cluster], Seite 111).

Automatische Kombination von Stimmen

Automatische Kombination von Stimmen wird verwendet, um zwei Stimmen auf einem Notensystem zu setzen. Es wird vor allem in Orchesterpartituren eingesetzt. Wenn beide Stimmen für einige Noten identisch sind, wird nur eine dargestellt. An den Stellen, an denen die beiden Stimmen sich unterscheiden, werden sie als unterschiedliche Stimmen gesetzt, und die Richtung der Hälse wird automatisch bestimmt. Zusätzlich werden *solo* und *a due*-Stellen erkannt und bezeichnet.

Die Syntax zur Stimmenkombination lautet:

`\partcombine` *`musikAusdr1 musikAusdr2`*
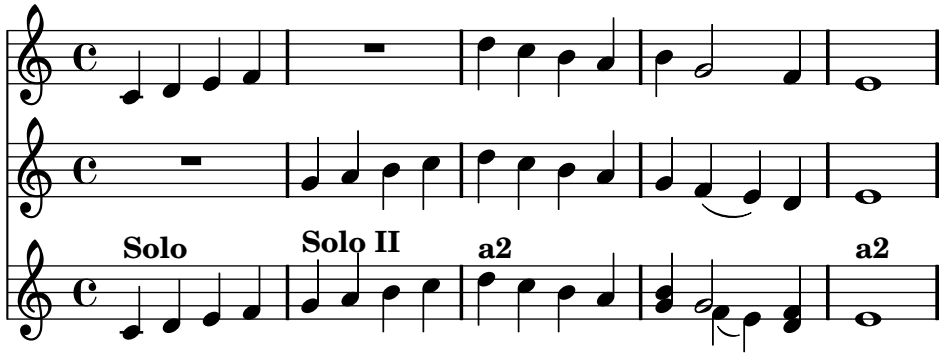
Das nächste Beispiel zeigt, wie die Kombination funktioniert. Hier werden die Stimmen erst auf einem jeweils eigenen System und dann kombiniert gesetzt, beachten Sie, wie sich die Einstellungen für Polyphonie ändern.

```
InstrumentEins = \relative c' {
  c4 d e f
  R1
  d'4 c b a
  b4 g2 f4
  e1
}

InstrumentZwei = \relative g' {
  R1
  g4 a b c
  d c b a
  g f( e) d
  e1
}

<<
  \new Staff \InstrumentEins
  \new Staff \InstrumentZwei
```

```
  \new Staff \partcombine \InstrumentEins \InstrumentZwei
>>
```



Die Noten des dritten Taktes werden nur einfach ausgegeben, obwohl sie ein beiden Stimmen definiert sind. 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 immer Hälse nach oben, die zweite Stimme 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 `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 sich außerhalb von`\partcombine` befindet, hat keinen Einfluss auf die Tonhöhen von *musikAusdr1* oder *musikAusdr2*.

Ausgewählte Schnipsel

Combining two parts on the same staff
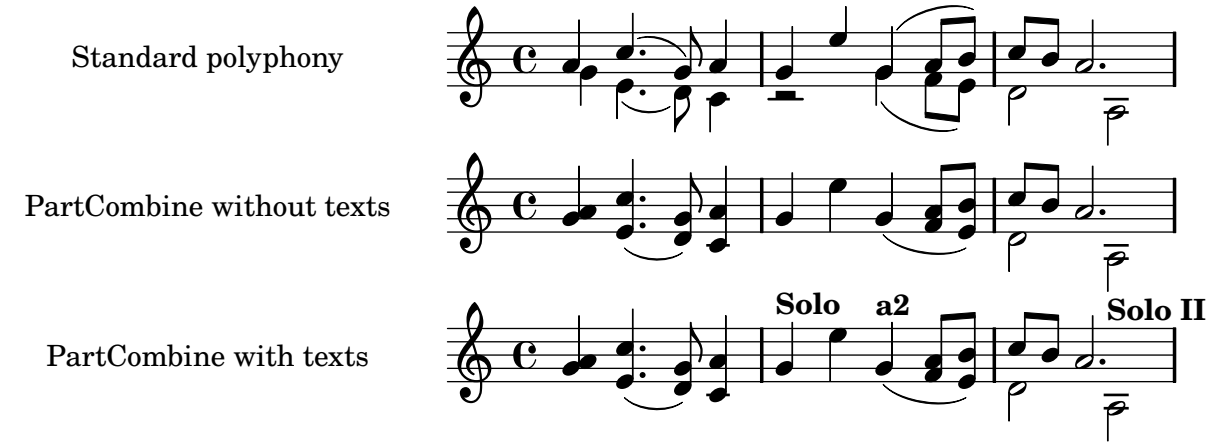
The part combiner tool (`\partcombine` command) allows the combination of several different parts on the same staff. Text directions such as "solo" or "a2" are added by default; to remove them, simply set the property `printPartCombineTexts` to "false". For vocal scores (hymns), there is no need to add "solo"/"a2" texts, so they should be switched off. However, it might be better not to use it if there are any solos, as they won't be indicated. In such cases, standard polyphonic notation may be preferable.

This snippet presents the three ways two parts can be printed on a same staff: standard polyphony, `\partcombine` without texts, and `\partcombine` with texts.

```
NotenHoch = \relative c'' {
  \time 4/4
  a4 c4.( g8) a4 |
  g4 e' g,( a8 b) |
  c b a2.
}

NotenRunter = \relative c'' {
  g4 e4.( d8) c4 |
  r2 g'4( f8 e) |
  d2 \stemDown a
}
```

```
\score {
  <<
    <<
    \new Staff {
      \set Staff.instrumentName = "Standard polyphony  "
      << \NotenHoch \\ \NotenRunter >>
    }
    \new Staff \with { printPartCombineTexts = ##f } {
      \set Staff.instrumentName = "PartCombine without texts  "
      \partcombine \NotenHoch \NotenRunter
    }
    \new Staff {
      \set Staff.instrumentName = "PartCombine with texts  "
      \partcombine \NotenHoch \NotenRunter
    }
    >>
  >>
  \layout {
    indent = 6.0\cm
    \context {
      \Score
      \override SystemStartBar #'collapse-height = #30
    }
  }
}
```
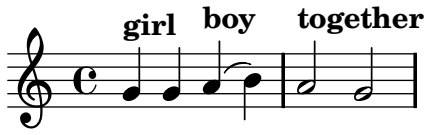


Changing partcombine texts

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 “Stimme” in *Glossar*.

Notationsreferenz: Abschnitt 1.6.3 [Orchesterstimmen erstellen], Seite 140.

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

`\partcombine` kann nur zwei Stimmen bearbeiten.

Wenn `printPartCombineTexts` (drucke Stimmenkombinationstext) gesetzt ist und die Stimmen die gleichen Noten wiederholgt spielen, kann `a2` in einem Takt mehrmals gesetzt werden.

`\partcombine` kann nicht innerhalb von `\times` benutzt werden.

`\partcombine` kann nicht innerhalb von `\relative` benutzt werden.

Intern werden beide Argumente von `\partcombine` als Stimmen (`Voice`) interpretiert und entschieden, wann die Stimmen kombiniert werden können. Wenn sie unterschiedliche Dauern haben, können sie nicht kombiniert werden und erhalten die Bezeichnung `one` und `two` Darum werden Wechsel zu einem `Voice`-Kontext, der eine andere Bezeichnung hat, ignoriert. Genausowenig ist die Stimmenkombination dazu ausgelegt, Gesangtext zu verarbeiten: wenn eine der Stimmen eine explizite Bezeichnung erhält, damit Text damit verknüpft werden kann, hört die Stimmenkombination auf zu arbeiten.

`\partcombine` findet nur den Beginn von Noten. Es kann nicht bestimmen, ob eine vorher begonnene Noten weiterklingt, was zu verschiedenen Problemen führen kann.

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.

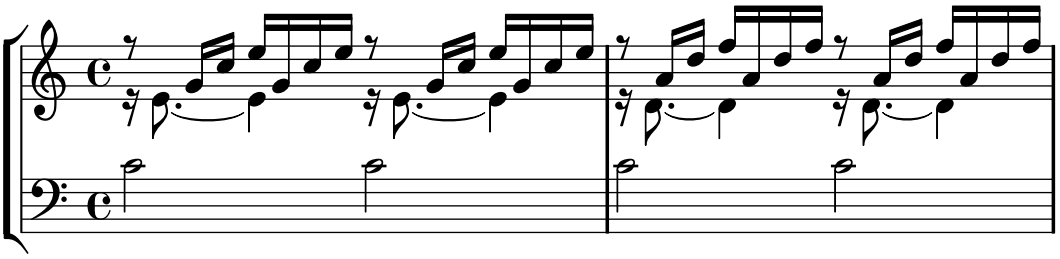
Achtung: Taktüberprüfungen `|` müssen benutzt werden, und die Takte müssen die gleiche Länge haben.

```
\parallelMusic #'(voiceA voiceB voiceC) {
  % Takt 1
  r8 g'16 c'' e'' g' c'' e'' r8 g'16 c'' e'' g' c'' e'' |
  r16 e'8.~   e'4           r16 e'8.~   e'4           |
```

```
  c'2                         c'2                         |

  % Takt 2
  r8 a'16 d'' f'' a' d'' f'' r8 a'16 d'' f'' a' d'' f'' |
  r16 d'8.~   d'4             r16 d'8.~   d'4             |
  c'2                         c'2                         |

}
\new StaffGroup <<
  \new Staff << \voiceA \\ \voiceB >>
  \new Staff { \clef bass \voiceC }
>>
```

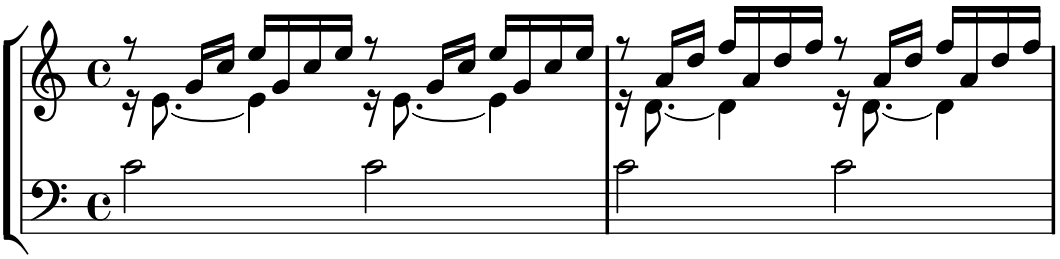


Der relative Modus kann auch benutzt werden. Beachten Sie, dass der `\relative`-Befehl nicht innerhalb von `\parallelMusic` benutzt wird. Die Noten sind paralell zu der vorherigen Note der gleichen Stimme , nicht zu der vorherigen Note in der Quelldatei. Anders gesagt ignorieren realtive Noten von `voiceA` die Noten von `voiceB`.

```
\parallelMusic #'(voiceA voiceB voiceC) {
  % Takt 1
  r8 g16 c e g, c e r8 g,16 c e g, c e  |
  r16 e8.~ e4       r16 e8.~  e4        |
  c2                c                   |

  % Takt 2
  r8 a,16 d f a, d f r8 a,16 d f a, d f |
  r16 d8.~  d4       r16 d8.~  d4       |
  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:
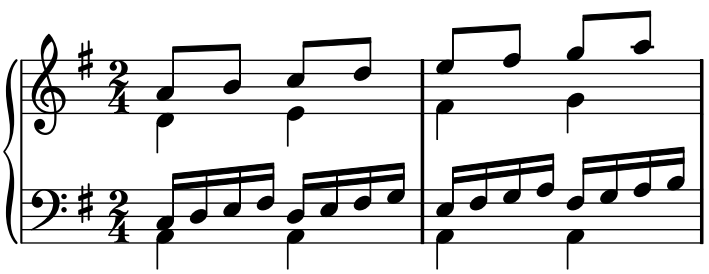
```
global = {
  \key g \major
  \time 2/4
}

\parallelMusic #'(voiceA voiceB voiceC voiceD) {
  % Takt 1
  a8    b     c   d     |
  d4          e         |
  c16 d e fis d e fis g |
  a4          a         |

  % Takt 2
  e8      fis  g     a   |
  fis4         g         |
  e16 fis g  a fis g a b |
  a4           a         |

  % Takt 3 ...
}

\score {
  \new PianoStaff <<
     \new Staff {
       \global
       <<
         \relative c'' \voiceA
         \\
         \relative c'  \voiceB
       >>
     }
     \new Staff {
       \global \clef bass
       <<
         \relative c \voiceC
         \\
         \relative c \voiceD
       >>
     }
  >>
}
```
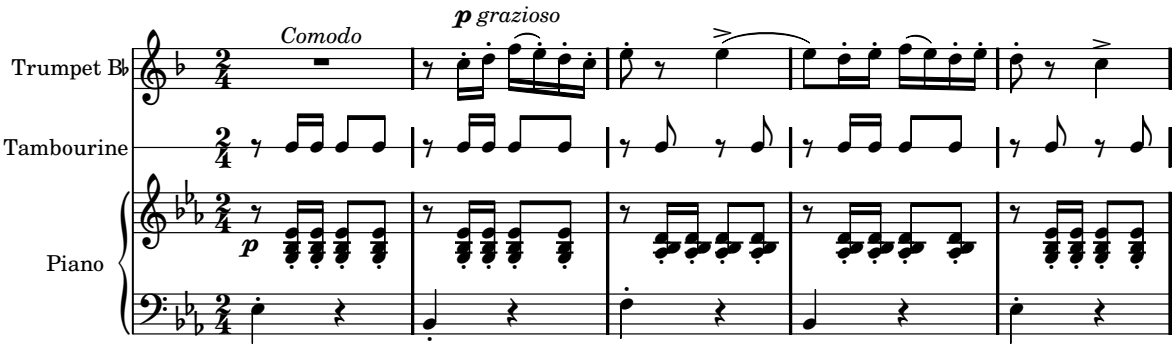


Siehe auch

Handbuch zum Lernen: Abschnitt “Organizing pieces with variables” in *Handbuch zum Lernen*.

Schnipsel: Abschnitt “Simultaneous notes” in *Schnipsel*.

1.6 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.

1.6.1 Systeme anzeigen lassen
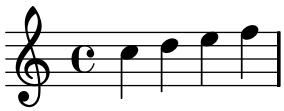
Dieser Abschnitt zeigt unterschiedliche Methoden, Notensysteme und Gruppen von Systemen zu erstellen.

Neue Notensysteme erstellen

Notensysteme (engl. *staff*, Pl. *staves*) werden mit dem `\new` oder `\context`-Befehl erstellt. Zu Einzelheiten siehe Abschnitt 5.1.2 [Umgebungen erstellen], Seite 335.

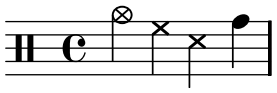
Der einfachste Notensystem-Kontext ist `Staff`:

```
\new Staff { c4 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 [Schlagzeugsysteme], Seite 250.

```
\new DrumStaff {
  \drummode { cymc hh ss tomh }
}
```



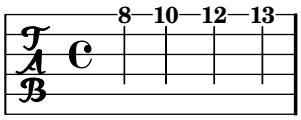
`RhythmicStaff` (Rhythmus-System) erstellt ein Notensystem mit nur einer Notenlinie, auf welcher nur die rhythmischen Werte der eingegebenen Noten dargestellt werden. Die wirklichen Längen bleiben erhalten. Zu Einzelheiten, siehe [Melodierhythmus anzeigen], Seite 53.

```
\new RhythmicStaff { c4 d e f }
```



`TabStaff` (Tabulatursystem) erstellt eine Tabulatur mit sechs Saiten in der üblichen Gitarrenstimmung. Zu Einzelheiten siehe [Standardtabulaturen], Seite 222.
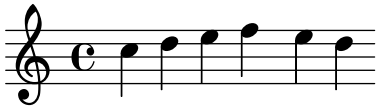
```
\new TabStaff { c4 d e f }
```



Es gibt zwei Notensysteme, die zur Notation von Alter Musik eingesetzt werden: `MensuralStaff` and `VaticanaStaff`. Sie sind erklärt in Abschnitt 2.8.4 [Vordefinierte Umgebungen], Seite 294.

Das `GregorianTranscriptionStaff` (System zur Transkription des Gregorianischen Chorals) erstellt ein Notensystem, um modernen Gregorianischen Choral zu notieren. Es hat keine Notenlinien.

```
\new GregorianTranscriptionStaff { c4 d e f e d }
```



Neue Notensystem-Kontexte können selber definiert werden. Zu Einzelheiten, siehe Abschnitt 5.1.5 [Neue Umgebungen definieren], Seite 335.

Siehe auch

Glossar: Abschnitt “System” in *Glossar*, Abschnitt “Systeme” in *Glossar*.

Notationsreferenz: Abschnitt 5.1.2 [Umgebungen erstellen], Seite 335, [Schlagzeugsysteme], Seite 250, [Melodierhythmus anzeigen], Seite 53, [Standardtabulaturen], Seite 222, Abschnitt 2.8.4 [Vordefinierte Umgebungen], Seite 294, [Das Notensystem], Seite 131, [Gregorianische Gesangs-Kontexte], Seite 295, [Mensural-Kontexte], Seite 295, Abschnitt 5.1.5 [Neue Umgebungen definieren], Seite 335.

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

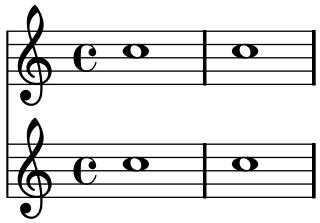
Referenz der Interna: Abschnitt “Staff” in *Referenz der Interna*, Abschnitt “DrumStaff” in *Referenz der Interna*, Abschnitt “GregorianTranscriptionStaff” in *Referenz der Interna*, Abschnitt “RhythmicStaff” in *Referenz der Interna*, Abschnitt “TabStaff” in *Referenz der Interna*, Abschnitt “MensuralStaff” in *Referenz der Interna*, Abschnitt “VaticanaStaff” in *Referenz der Interna*, Abschnitt “StaffSymbol” in *Referenz der Interna*.

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.

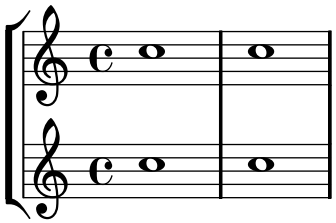
Wenn kein Kontext angegeben ist, wird die Standardeinstellung eingesetzt: die Gruppe beginnt mit einer vertikalen Linie und die Taktlinien sind nicht verbunden.

```
<<
  \new Staff { c1 c }
  \new Staff { c1 c }
>>
```



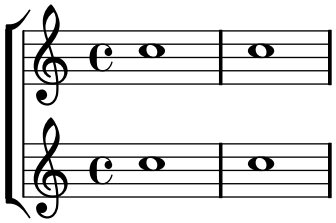
Im `StaffGroup`-Kontext die Gruppe mit einer eckigen Klammer begonnen und die Taktlinien durch alle Systeme gezogen.

```
\new StaffGroup <<
  \new Staff { c1 c }
  \new Staff { c1 c }
>>
```



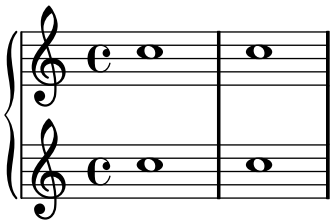
In einem `ChoirStaff` (Chorsystem) beginnt die Gruppe mit einer eckigen Klammer, aber die Taktlinien sind nicht verbunden.

```
\new ChoirStaff <<
  \new Staff { c1 c }
  \new Staff { c1 c }
>>
```



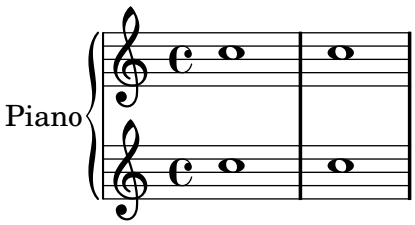
In einem `GrandStaff` (Akkolade) beginnt die Gruppe mit einer geschweiften Klammer und die Taktlinien sind durchgezogen.

```
\new GrandStaff <<
  \new Staff { c1 c }
  \new Staff { c1 c }
>>
```



Der `PianoStaff`-(Klaviersystem)-Kontext ist identisch mit dem `GrandStaff`-Kontext, aber es ermöglicht zusätzlich direkt die Angabe einer Instrumentbezeichnung. Zu Einzelheiten siehe [Instrumentenbezeichnungn], Seite 142.

```
\new PianoStaff <<
  \set PianoStaff.instrumentName = #"Piano"
  \new Staff { c1 c }
  \new Staff { 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.

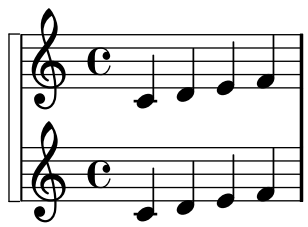
Neue Systemgruppen können definiert werden. Zu Einzelheiten siehe Abschnitt 5.1.5 [Neue Umgebungen definieren], Seite 335.

Ausgewählte Schnipsel

Use square bracket at the start of a staff group

The system start delimiter `SystemStartSquare` can be used by setting it explicitly in a `StaffGroup` or `ChoirStaffGroup` context.

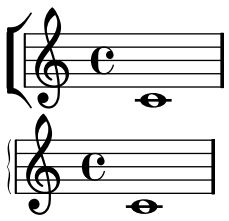
```
\score {
  \new StaffGroup { <<
  \set StaffGroup.systemStartDelimiter = #'SystemStartSquare
    \new Staff { c'4 d' e' f' }
    \new Staff { c'4 d' e' f' }
  >> }
}
```



*Display bracket with only one staff in a system*If there is only one staff in one of the staff types `ChoirStaff` or `StaffGroup`, the bracket and the starting bar line will not be displayed as standard behavior. This can be changed by overriding the relevant properties.

Note that in contexts such as `PianoStaff` and `GrandStaff` where the systems begin with a brace instead of a bracket, another property has to be set, as shown on the second system in the example.

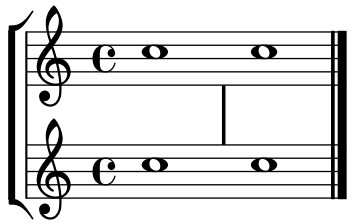
```
\markup \left-column {
  \score {
    \new StaffGroup <<
      % Muss geringer sein als die aktuelle Zahl der Notenlinien
      \override StaffGroup.SystemStartBracket #'collapse-height = #1
      \override Score.SystemStartBar #'collapse-height = #1
      \new Staff {
        c'1
      }
    >>
    \layout { }
  }
  \score {
    \new PianoStaff <<
      \override PianoStaff.SystemStartBrace #'collapse-height = #1
      \override Score.SystemStartBar #'collapse-height = #1
      \new Staff {
        c'1
      }
    >>
    \layout { }
  }
}
```



Mensurstriche layout (bar lines between the staves)

The mensurstriche-layout where the bar lines do not show on the staves but between staves can be achieved with a `StaffGroup` instead of a `ChoirStaff`. The bar line on staves is blanked out by setting the `transparent` property.

```
global = {
  \override Staff.BarLine #'transparent = ##t
  s1 s
  % Letze Taktlinie ist nicht unterbrochen
  \revert Staff.BarLine #'transparent
  \bar "|."
}
\new StaffGroup \relative c'' {
  <<
    \new Staff { << \global { c1 c } >> }
    \new Staff { << \global { c c } >> }
  >>
}
```



Siehe auch

Glossar: Abschnitt “Klammer” in *Glossar*, Abschnitt “Klammer” in *Glossar*, Abschnitt “Akkolade” in *Glossar*.

Notationsreferenz: [Instrumentenbezeichnungn], Seite 142, Abschnitt 5.1.5 [Neue Umgebungen definieren], Seite 335.

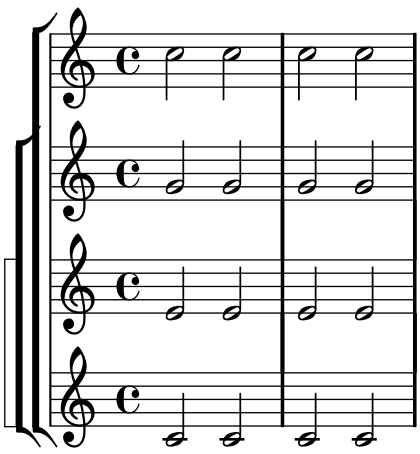
Schnipsel: Abschnitt “Staff notation” in *Schnipsel*.

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*.

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 { c2 c | c2 c }
  \new StaffGroup <<
    \new Staff { g2 g | g2 g }
    \new StaffGroup \with {
      systemStartDelimiter = #'SystemStartSquare
    }
    <<
      \new Staff { e2 e | e2 e }
      \new Staff { c2 c | c2 c }
    >>
  >>
>>
```



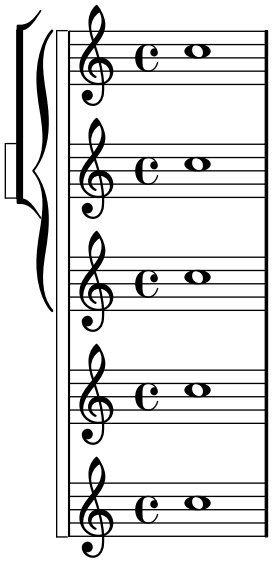
Neue geschachtelte Systemgruppen können definiert werden. Zu Einzelheiten siehe Abschnitt 5.1.5 [Neue Umgebungen definieren], Seite 335.

Ausgewählte Schnipsel

Nesting staves

The property `systemStartDelimiterHierarchy` can be used to make more complex nested staff groups. The command `\set StaffGroup.systemStartDelimiterHierarchy` takes an alphabetical list of the number of staves produced. Before each staff a system start delimiter can be given. It has to be enclosed in brackets and takes as much staves as the brackets enclose. Elements in the list can be omitted, but the first bracket takes always the complete number of staves. The possibilities are `SystemStartBar`, `SystemStartBracket`, `SystemStartBrace`, and `SystemStartSquare`.

```
\new StaffGroup
\relative c'' <<
  \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: [Systeme gruppieren], Seite 125, [Instrumentenbezeichnungn], Seite 142, Abschnitt 5.1.5 [Neue Umgebungen definieren], Seite 335.

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*.

1.6.2 Einzelne Systeme verändern
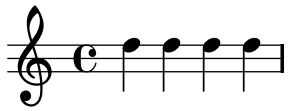
Dieser Abschnitt zeigt, wie man bestimmte Eigenschaften eines Systems ändert – etwa die Anzahld der Notenlinien oder die Größe des Systems. Es werden auch Methoden dargestellt, ein System zu beginnen und zu beenden sowie eine Methode, Ossia-Systeme zu erstellen.

Das Notensystem

Die Linien eines Notensystems gehören zu dem `StaffSymbol`-(NotensystemSymbol)-Grob. `StaffSymbol`-Eigenschaften können verändert werden, um die Erscheinung des Notensystems zu beeinflussen, aber sie müssen gesetzt werden, bevor das System erstellt wird.
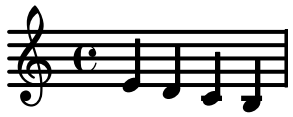
Die Anzahl der Notenlinien kann verändert werden. Die Position des Notenschlüssels und die Position von c' können geändert werden, um dem neuen System zu entsprechen. Eine Erklärung findet sich im Schnipselabschnit in [Notenschlüssel], Seite 12.

```
\new Staff \with {
  \override StaffSymbol #'line-count = #3
}
{ d4 d d d }
```
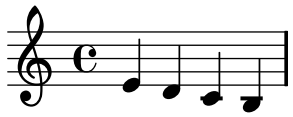


Die Liniendicke der Notenlinien kann verändert werden. Die Dicke der Hilfslinien und Notenhälse wird auch beeinflusst, weil sie von der Notenliniendicke abhängen.

```
\new Staff \with {
  \override StaffSymbol #'thickness = #3
}
{ e4 d c b }
```



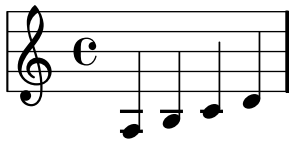
Die Dicke der Hilfslinien kann auch unabhängig von der Notenliniendicke verändert werden. Die zwei Zahlen in dem Beispiel sind Faktoren, mit denen die Notenlinien-Dicke und der Notenlinienabstand multipliziert werden. Die Addition beider Werte ergibt die Dicke der Hilfslinien.

```
\new Staff \with {
  \override StaffSymbol #'ledger-line-thickness = #'(1 . 0.2)
}
{ e4 d c b }
```



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
}
{ a4 b c d }
```



Weitere Einzelheiten zu den Eigenschaften von `StaffSymbol` findet sich in Abschnitt “staff-symbol-interface” in *Referenz der Interna*.

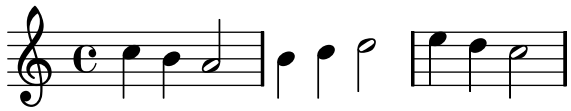
Veränderungen der Eigenschaften eines Notensystems mitten in einer Partitur können zwischen die Befehle `\stopStaff` und `\startStaff` gesetzt werden:

```
c2 c
\stopStaff
\override Staff.StaffSymbol #'line-count = #2
\startStaff
b2 b
\stopStaff
\revert Staff.StaffSymbol #'line-count
\startStaff
a2 a
```



Die Befehle `\startStaff` und `\stopStaff` können benutzt werden, um ein Notensystem irgendwo zu beenden oder zu beginnen.

```
c4 b a2
\stopStaff
b4 c d2
\startStaff
e4 d c2
```



Vordefinierte Befehle
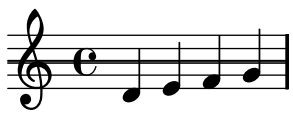
`\startStaff`, `\stopStaff`.

Ausgewählte Schnipsel

Making some staff lines thicker than the others

For pedagogical purposes, a staff line can be thickened (e.g., the middle line, or to emphasize the line of the G clef). This can be achieved by adding extra lines very close to the line that should be emphasized, using the `line-positions` property of the `StaffSymbol` object.

```
{
  \override Staff.StaffSymbol #'line-positions = #'(-4 -2 -0.2 0 0.2 2 4)
  d'4 e' f' g'
}
```



Siehe auch

Glossar: Abschnitt “Linie” in *Glossar*, Abschnitt “Hilfslinie” in *Glossar*, Abschnitt “System” in *Glossar*.

Notationsreferenz: [Notenschlüssel], Seite 12.

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

Referenz der Interna: Abschnitt “StaffSymbol” in *Referenz der Interna*, Abschnitt “staff-symbol-interface” in *Referenz der Interna*.
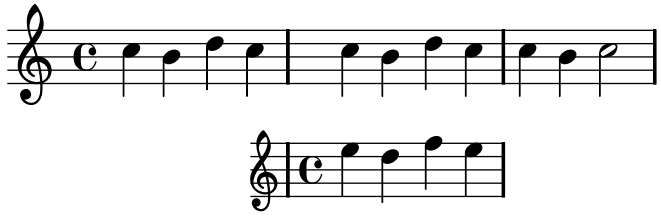
Bekannte Probleme und Warnungen

Wenn die vertikale Position der Notenlinien manuell verändert wird, werden Taktlinien immer auf der Position 0 zentriert. Somit muss die Distanz der äußeren Notenlinien vom Zentrum des Systems gleichgroß sein.

Ossia-Systeme

Ossia-Systeme können gesetzt werden, indem zwei gleichzeitige Notensysteme an der entsprechenden Position erstellt werden:

```
\new Staff \relative c'' {
  c4 b d c
  <<
    { c4 b d c }
    \new Staff { e4 d f e }
  >>
  c4 b c2
}
```



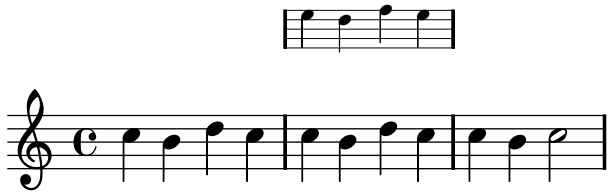
Dieses Beispiel ist aber normalerweise nicht erwünscht. Um Ossia-Systeme zu setzen, die sich über dem eigentlichen System befinden, keine Takt- und Schlüsselangaben haben und kleiner gesetzt sind, müssen einige Optimierungen angewendet werden. Im Handbuch zum Lernen wird eine Technik vorgestellt, mit der das gewünschte Ergebnis erreicht werden kann, beginnend in Abschnitt “Nesting music expressions” 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 = haupt \relative c'' {
  c4 b d c
  <<
    { c4 b d c }

    \new Staff \with {
      \remove "Time_signature_engraver"
      alignAboveContext = #"haupt"
      fontSize = #-3
      \override StaffSymbol #'staff-space = #(magstep -3)
```
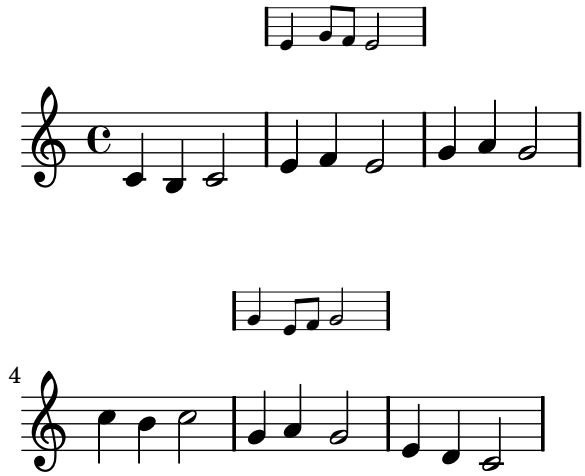
```
      \override StaffSymbol #'thickness = #(magstep -3)
      firstClef = ##f
    }
    { e4 d f e }
  >>
  c4 b c2
}
```



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"
    \override Clef #'transparent = ##t
    fontSize = #-3
    \override StaffSymbol #'staff-space = #(magstep -3)
    \override StaffSymbol #'thickness = #(magstep -3)
  }
  { \stopStaff s1*6 }
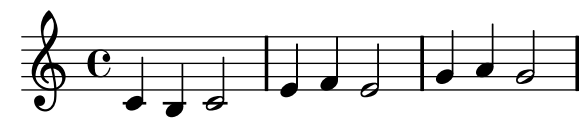
  \new Staff \relative c' {
    c4 b c2
    <<
      { e4 f e2 }
      \context Staff = Ossia {
        \startStaff e4 g8 f e2 \stopStaff
      }
    >>
    g4 a g2 \break
    c4 b c2
    <<
      { g4 a g2 }
      \context Staff = Ossia {
        \startStaff g4 e8 f g2 \stopStaff
      }
    >>
    e4 d c2
  }
>>
```
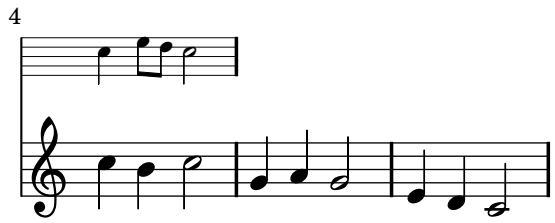


Man kann auch den `\RemoveEmptyStaffContext`-Befehl einsetzen, um Ossia-Systeme zu erstellen. Diese Methode eignet sich am besten, wenn nach dem Ossia sofort ein Zeilenumbruch erfolgt. In diesem Fall müssen auch keine unsichtbaren Pausen eingesetzt werden; es reicht, `\startStaff` und `\stopStaff` einzusetzen. Mehr Information zu `\RemoveEmptyStaffContext` findet sich in [Systeme verstecken], Seite 137.

```
<<
  \new Staff = Ossia \with {
    \remove "Time_signature_engraver"
    \override Clef #'transparent = ##t
    fontSize = #-3
    \override StaffSymbol #'staff-space = #(magstep -3)
    \override StaffSymbol #'thickness = #(magstep -3)
  }
  \new Staff \relative c' {
    c4 b c2
    e4 f e2
    g4 a g2 \break
    <<
      { c4 b c2 }
      \context Staff = Ossia {
        c4 e8 d c2 \stopStaff
      }
    >>
    g4 a g2
    e4 d c2
  }
>>

\layout {
  \context {
    \RemoveEmptyStaffContext
    \override VerticalAxisGroup #'remove-first = ##t
  }
}
```





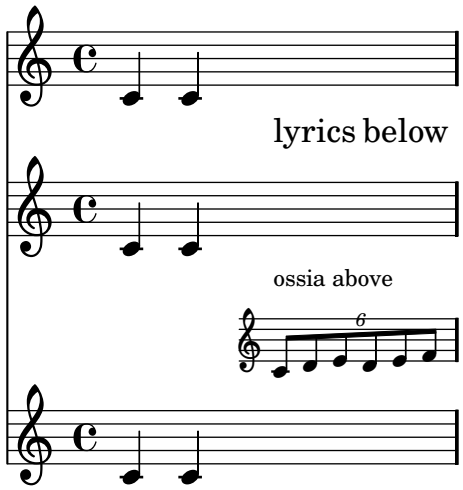
Ausgewählte Schnipsel

Vertically aligning ossias and lyrics

This snippet demonstrates the use of the context properties `alignBelowContext` and `alignAboveContext` to control the positioning of lyrics and ossias.

```
\paper {
  ragged-right = ##t
}

\relative c' <<
  \new Staff = "1" { c4 c s2 }
  \new Staff = "2" { c4 c s2 }
  \new Staff = "3" { c4 c s2 }
  { \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"
      } {
        \times 4/6 {
          \override TextScript #'padding = #3
          c8[^"ossia above" d e d e f]
        }
      }
    >>
  }
>>
```



Siehe auch

Glossar: Abschnitt “Ossia” in *Glossar*, Abschnitt “System” in *Glossar*, Abschnitt “Verkleinerte Systeme” in *Glossar*.

Handbuch zum Lernen: Abschnitt “Nesting music expressions” in *Handbuch zum Lernen*, Abschnitt “Size of objects” in *Handbuch zum Lernen*, Abschnitt “Length and thickness of objects” in *Handbuch zum Lernen*.

Notationsreferenz: [Systeme verstecken], Seite 137.

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

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

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 c''' { a8 f e16 d c b a2 }
```



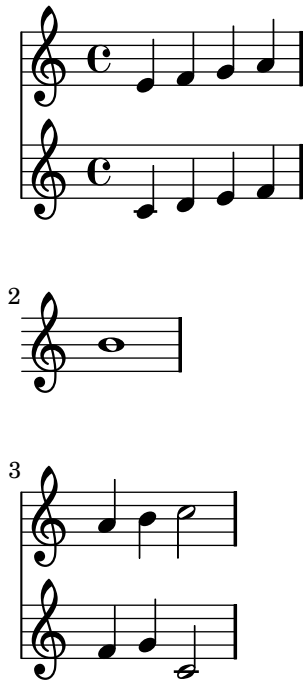
Leere Systeme können versteckt werden, wenn der `\RemoveEmptyStaffContext`-Befehl im `\layout`-Abschnitt benutzt wird. In großen Orchesterpartituren wird dies oft verwendet, um die leeren Systeme von gerade nicht spielenden Instrumenten zu verstecken. In der Standardeinstellung werden alle leeren Notenzeilen außer die des ersten Systems entfernt.

Achtung: Eine Notenzeile gilt als leer, wenn sie nur Ganztaktpausen, unsichtbare Noten, `\skip`-Befehle oder eine Kombination der drei enthält.

```
\layout {
  \context {
    \RemoveEmptyStaffContext
  }
}

\relative c' <<
```

```
  \new Staff {
    e4 f g a \break
    b1 \break
    a4 b c2
  }
  \new Staff {
    c,4 d e f \break
    R1 \break
    f4 g c,2
  }
>>
```



`\RemoveEmptyStaffContext` kann auch eingesetzt werden, um Ossiaabschnitte zu erstellen. Zu Einzelheiten, siehe [Ossia-Systeme], Seite 133.

Der `\AncientRemoveEmptyStaffContext`-Befehl kann benutzt werden, um leere Takte in Notation der Alten Musik zu entfernen. Gleichermaßen kann `\RemoveEmptyRhythmicStaffContext` eingesetzt werden, um leere Takte in einem `RhythmicStaff`-Kontext zu entfernen.

Vordefinierte Befehle

`\RemoveEmptyStaffContext`, `\AncientRemoveEmptyStaffContext`, `\RemoveEmptyRhythmicStaffContext`.

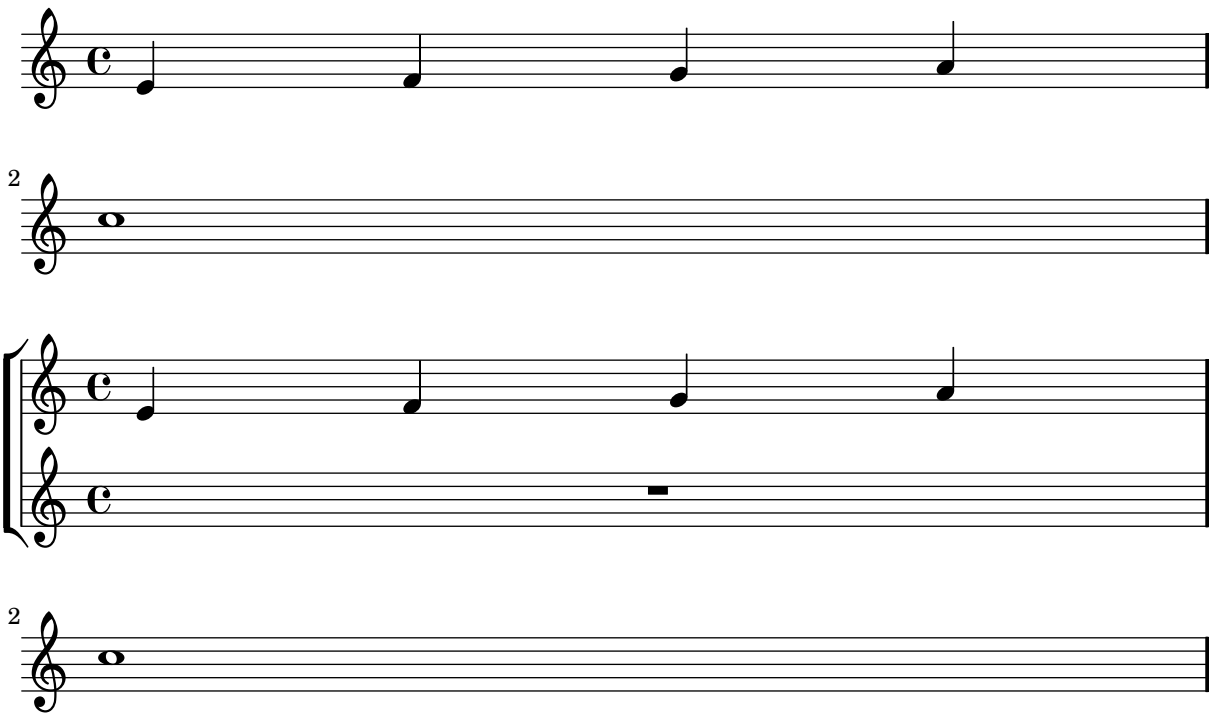
Ausgewählte Schnipsel

Removing the first empty line

The first empty staff can also be removed from the score by setting the `VerticalAxisGroup` property `remove-first`. This can be done globally inside the `\layout` block, or locally inside the specific staff that should be removed. In the latter case, you have to specify the context (`Staff` applies only to the current staff) in front of the property.

The lower staff of the second staff group is not removed, because the setting applies only to the specific staff inside of which it is written.

```
\layout {
  \context {
    \RemoveEmptyStaffContext
    % Um die Einstellung global zu benutzen, diese Zeile benutzen:
    % \overrideVerticalAxisGroup#'remove-first=##t
  }
}
\new StaffGroup <<
  \new Staff \relative c' {
    e4 f g a \break
    c1
  }
  \new Staff {
    % Um Einstellung global zu benutzen, folgende Zeile auskommentieren
    % und die Zeile im \layout-Block oben benutzen
    \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 “Verkleinerte Systeme” in *Glossar*.

Notationsreferenz: [Das Notensystem], Seite 131, [Ossia-Systeme], Seite 133.

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:

```
\override StaffSymbol #'stencil = ##f
\override NoteHead #'no-ledgers = ##t
```
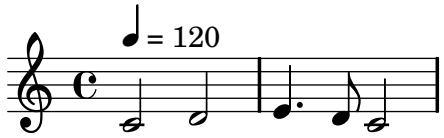
1.6.3 Orchesterstimmen erstellen

Dieser Abschnitt zeigt, wie man Tempo-Anweisungen und Instrumentenbezeichnungen einfügt. Es werden auch Möglichkeiten vorgestellt, andere Stimmen zu zitieren und Stichnoten zu formatieren.

Metronomangabe
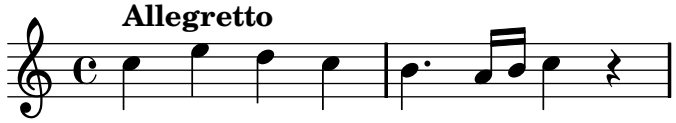
Eine Metronomanweisung wird wie folgt erstellt:

```
\tempo 4 = 120
c2 d
e4. d8 c2
```
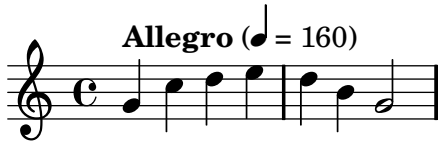


Anstelle dessen kann auch Text als Argument angegeben werden:

```
\tempo "Allegretto"
c4 e d c
b4. a16 b c4 r4
```
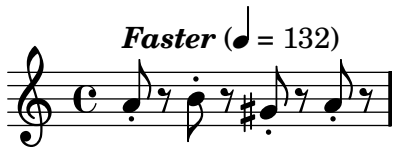


Wenn eine Metronombezeichnung und Text kombiniert wird, wird die Metronombezeichnung automatisch in Klammern gesetzt:

```
\tempo "Allegro" 4 = 160
g4 c d e
d4 b g2
```
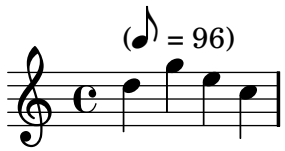


Der Text kann ein beliebiges Textbeschriftungsobjekt sein:

```
\tempo \markup { \italic Faster } 4 = 132
a8-. r8 b-. r gis-. r a-. r
```



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

```
\tempo "" 8 = 96
d4 g e c
```



Ausgewählte Schnipsel

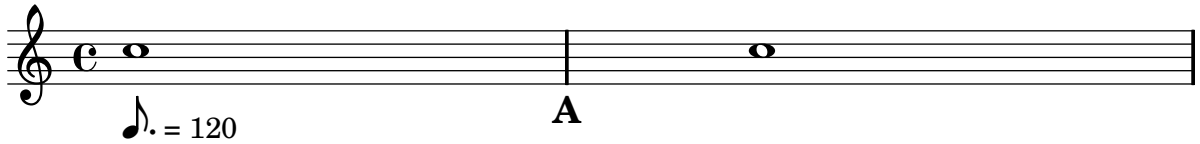
Printing metronome and rehearsal marks below the staff

By default, metronome and rehearsal marks are printed above the staff. To place them below the staff simply set the `direction` property of `MetronomeMark` or `RehearsalMark` appropriately.

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

{
  % Metronomangabe unter dem System
  \override Score.MetronomeMark #'direction = #DOWN
  \tempo 8. = 120
  c''1
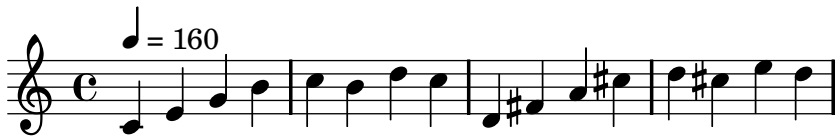
  % Übungszeichen unter dem System
  \override Score.RehearsalMark #'direction = #DOWN
  \mark \default
  c''1
}
```



*Changing the tempo without a metronome mark*To change the tempo in MIDI output without printing anything, make the metronome mark invisible:
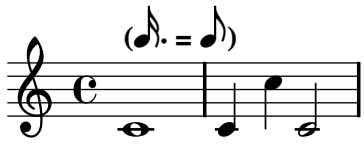
```
\score {
  \new Staff \relative c' {
    \tempo 4 = 160
    c4 e g b
    c4 b d c
```

```
    \set Score.tempoHideNote = ##t
    \tempo 4 = 96
    d,4 fis a cis
    d4 cis e d
  }
  \layout { }
  \midi { }
}
```



*Creating metronome marks in markup mode*New metronome marks can be created in markup mode, but they will not change the tempo in MIDI output.

```
\relative c' {
  \tempo \markup {
    \concat {
      (
      \smaller \general-align #Y #DOWN \note #"16." #1
      " = "
      \smaller \general-align #Y #DOWN \note #"8" #1
      )
    }
  }
  c1
  c4 c' c,2
}
```



Zu Einzelheiten siehe Abschnitt 1.8.2 [Text formatieren], Seite 170.

Siehe auch

Glossar: Abschnitt “Metronomangabe” in *Glossar*, Abschnitt “Metronombezeichnung” in *Glossar*, Abschnitt “Tempobezeichnung” in *Glossar*, Abschnitt “Metronombezeichnung” in *Glossar*.

Notationsreferenz: Abschnitt 1.8.2 [Text formatieren], Seite 170, Abschnitt 3.5 [MIDI-Ausgabe], Seite 322.
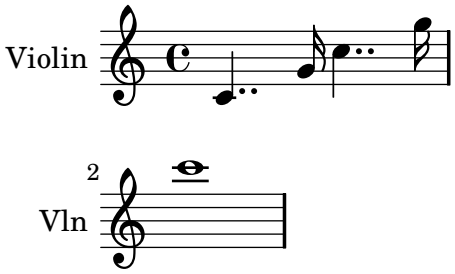
Schnipsel: Abschnitt “Staff notation” in *Schnipsel*.

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

Instrumentenbezeichnungn
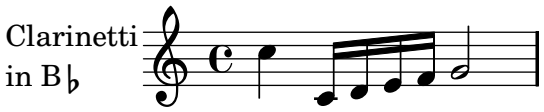
Instrumentbezeichnungen können an der linken Seite von Notensystemen im `Staff`- und `PianoStaff`-Kontext gesetzt werden. Der Wert von `instrumentName` wird für das erste System eingesetzt, der Wert von `shortInstrumentName` für alle weiteren Systeme.

```
\set Staff.instrumentName = #"Violin "
\set Staff.shortInstrumentName = #"Vln "
c4.. g'16 c4.. g'16
\break
c1
```



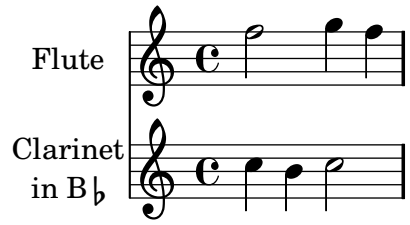
Mit dem Textbeschriftungsmodus können auch komplizierte Instrumentenbezeichnungen erstellt werden:

```
\set Staff.instrumentName = \markup {
  \column { "Clarinetti"
            \line { "in B" \smaller \flat } } }
c4 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 {
    \set Staff.instrumentName = #"Flute"
    f2 g4 f
  }
  \new Staff {
    \set Staff.instrumentName = \markup \center-column {
      Clarinet
      \line { "in B" \smaller \flat }
    }
    c4 b c2
  }
>>
```

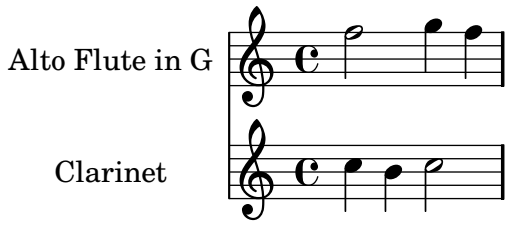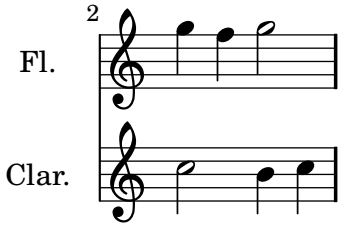


Wenn die Instrumentenbezeichnung zu lang ist, kann es vorkommen, dass die Bezeichnungen in einer Gruppe nicht zentriert werden. Um dennoch eine Zentrierung zu erhalten, müssen

die Werte des Einzugs (`indent` und `short-indent`) vergrößert werden. Zu Einzelheiten siehe [Horizontale Dimensionen], Seite 332.

```
\layout {
  indent = 3.0\cm
  short-indent = 1.5\cm
}

\relative c'' <<
  \new Staff {
    \set Staff.instrumentName = #"Alto Flute in G"
    \set Staff.shortInstrumentName = #"Fl."
    f2 g4 f \break
    g4 f g2
  }
  \new Staff {
    \set Staff.instrumentName = #"Clarinet"
    \set Staff.shortInstrumentName = #"Clar."
    c,4 b c2 \break
    c2 b4 c
  }
>>
```

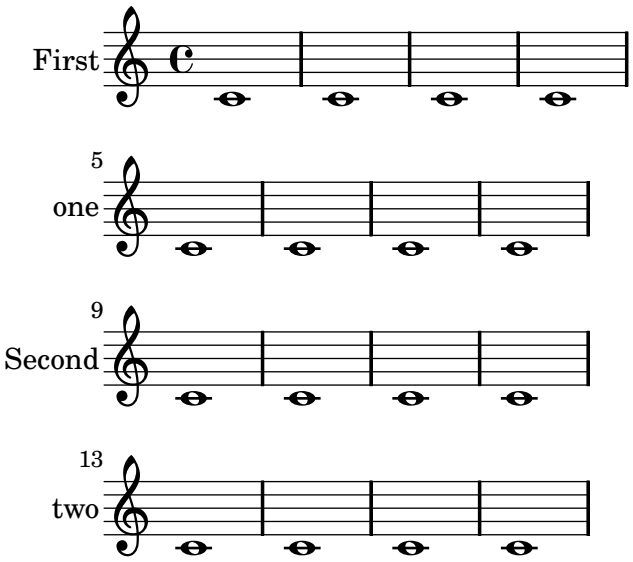




Um Instrumentenbezeichnungen zu anderen Kontexten (wie etwa `GrandStaff`, `ChoirStaff` oder `StaffGroup`) hinzuzufügen, muss der `Instrument_name_engraver` dem entsprechenden Kontext hinzugefügt werden. Zu Einzelheiten siehe Abschnitt 5.1.3 [Umgebungs-Plugins verändern], Seite 335.

Instrumentenbezeichnungen können mitten in einer Partitur geändert werden:

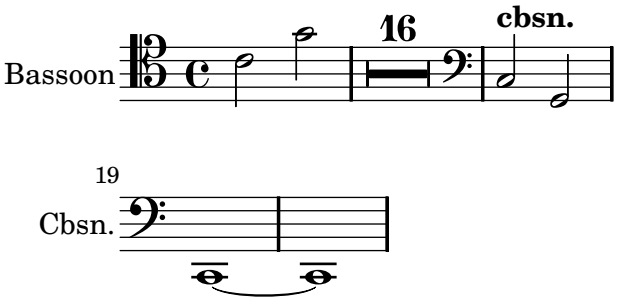
```
\set Staff.instrumentName = #"First"
\set Staff.shortInstrumentName = #"one"
c1 c c c \break
c1 c c c \break
\set Staff.instrumentName = #"Second"
\set Staff.shortInstrumentName = #"two"
c1 c c c \break
c1 c c c \break
```



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 Assosiationsliste 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
  \compressFullBarRests
  c2 g'
  R1*16
  \instrumentSwitch "contrabassoon"
  c,,2 g \break
  c,1 ~ | c1
}
```



Siehe auch

Notationsreferenz: [Horizontale Dimensionen], Seite 332, Abschnitt 5.1.3 [Umgebungs-Plugins verändern], Seite 335.

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*.

Stichnoten

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 Passage die gleichen Noten haben. In LilyPond kann man das erreichen, indem eine Stimme von der anderen *zitiert*, sodass man die Noten nicht nocheinmal eingeben muss.
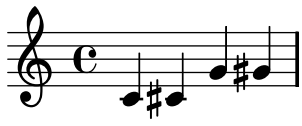
Bevor eine Stimme zitiert werden kann, muss der `\addQuote`-Befehl benutzt werden, um das zitierbare Fragment zu kennzeichnen. Dieser Befehl muss auf der höchsten Ebene der Eingabedatei benutzt werden. Das erste Argument dient zur Identifikation, das zweite ein musikalischer Ausdruck:

```
flute = \relative c'' {
  a4 gis g gis
}
\addQuote "flute" { \flute }
```

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, der Angibt, wie lange das Zitat dauern soll; normalerweise Ganztaktpausen oder unsichtbare Noten. Die entsprechenden Noten der zitierten Stimme wird an der Stelle in die aktuelle Stimme eingefügt:

```
Flöte = \relative c'' {
  a4 gis g gis
}
\addQuote "flute" { \Flöte }
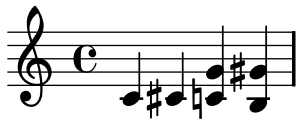
\relative c' {
  c4 cis \quoteDuring #"flute" { s2 }
}
```



Wenn der musikalische Ausdruck, der mit dem `\quoteDuring`-Befehl benutzt wird, etwas anderes als unsichtbare Noten oder Ganztaktpausen enthält, wird eine polyphone Stelle begonnen, was meistens nicht erwünscht ist:

```
Flöte = \relative c'' {
  a4 gis g gis
}
\addQuote "flute" { \Flöte }
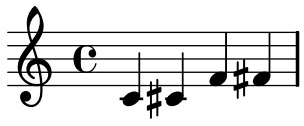
\relative c' {
  c4 cis \quoteDuring #"flute" { c4 b }
}
```



Zitate erkennen die Einstellungen von transponierten Instrumenten sowohl der Quell- als auch der Zielstimme, wenn der `\transposition`-Befehl eingesetzt wird. Zu Einzelheiten über `\transposition` siehe [Transposition von Instrumenten], Seite 17.

```
Klarinette = \relative c'' {
  \transposition bes
  a4 gis g gis
}
\addQuote "clarinet" { \Klarinette }

\relative c' {
  c4 cis \quoteDuring #"clarinet" { s2 }
}
```



Es ist möglich, Zitate mit eindeutigen Bezeichnungen zu versehen (unter Benutzung von *tags*), um sie auf unterschiedliche Weise zu verarbeiten. Einzelheiten zu diesem Vorgehen werden vorgestellt in [Marken benutzen], Seite 317.

Ausgewählte Schnipsel

*Quoting another voice with transposition*Quotations take into account the transposition of both source and target. In this example, all instruments play sounding middle C; the target is an instrument in F. The target part may be transposed using `\transpose`. In this case, all the pitches (including the quoted ones) are transposed.

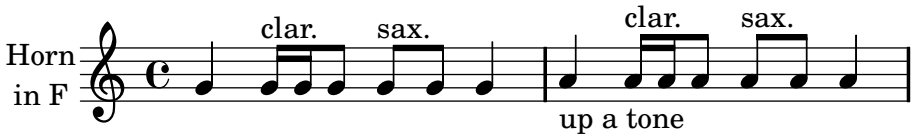
```
\addQuote clarinet {
  \transposition bes
  \repeat unfold 8 { d'16 d' d'8 }
}

\addQuote sax {
  \transposition es'
  \repeat unfold 16 { a8 }
}

zitatTest = {
  % Waldhorn
  \transposition f
  g'4
  << \quoteDuring #"clarinet" { \skip 4 } s4^"clar." >>
  << \quoteDuring #"sax" { \skip 4 } s4^"sax." >>
  g'4
}

{
  \set Staff.instrumentName =
    \markup {
      \center-column { Horn \line { in F } }
```

```
    }
  \zitatTest
  \transpose c' d' << \zitatTest s4_"up a tone" >>
}
```

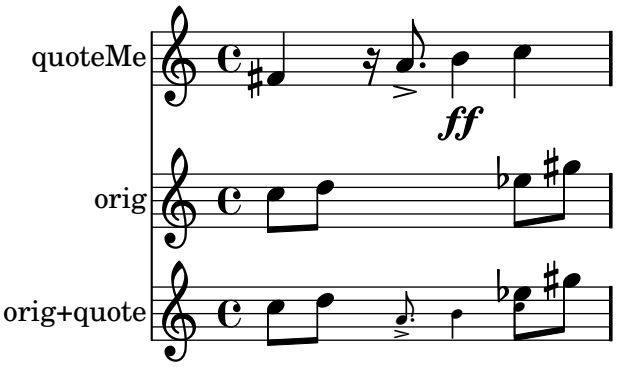


*Quoting another voice*The `quotedEventTypes` property determines the music event types that are quoted. The default value is `(note-event rest-event)`, which means that only notes and rests of the quoted voice appear in the `\quoteDuring` expression. In the following example, a 16th rest is not quoted since `rest-event` is not in `quotedEventTypes`.

```
zitiereMich = \relative c' {
  fis4 r16 a8.-> b4\ff c
}
\addQuote zitiereMich \zitiereMich

Original = \relative c'' {
  c8 d s2
  \once \override NoteColumn #'ignore-collision = ##t
  es8 gis8
}

<<
  \new Staff {
    \set Staff.instrumentName = #"quoteMe"
    \zitiereMich
  }
  \new Staff {
    \set Staff.instrumentName = #"orig"
    \Original
  }
  \new Staff \relative c'' <<
    \set Staff.instrumentName = #"orig+quote"
    \set Staff.quotedEventTypes =
      #'(note-event articulation-event)
    \Original
    \new Voice {
      s4
      \set fontSize = #-4
      \override Stem #'length-fraction = #(magstep -4)
      \quoteDuring #"quoteMe" { \skip 2. }
    }
  >>
>>
```



Siehe auch

Notationsreferenz: [Transposition von Instrumenten], Seite 17, [Marken benutzen], Seite 317.

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

Referenz der Interna: Abschnitt “QuoteMusic” 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, die Variable *Noten* kann also keine `\new` oder `\context Voice`-Einheiten enthalten, die zu einer anderen Stimme wechseln würden.

Ziernoten und Vorschläge können nicht zitiert werden und können sogar dazu führen, dass LilyPond abstürzt.

Wenn geschachtelte Triolen zitiert werden, ist das Notenbild unter Umständen sehr schlecht.

In früheren LilyPond-Versionen (vor 2.11) wurde der Befehl `addQuote` vollständig in Kleinbuchstaben geschrieben: `\addquote`.

Stichnoten formatieren
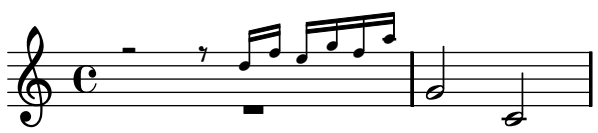
Der vorige Abschnitt zeigt, wie man Zitate erstellt. Der `\cueDuring`-Befehl (engl. cue note = Stichnote) ist eine spezialisierte Form des `\quoteDuring`-Befehls, der insbesondere dazu dient, Stichnoten zu einer Stimme hinzuzufügen. Seine Syntax lautet:

`\cueDuring #`*`Stimmenbezeichnung`* `#`*`Stimme Noten`*

Dieser Befehl kopiert die entsprechenden Takte von *Stimmenbezeichnung* in einen `CueVoice`-Kontext. Eine `CueVoice` (Stichnoten-Stimme) wird implizit erstellt und erscheint simultan mit *Noten*, wobei folglich eine polyphone Situation entsteht. Das *Stimme*-Argument entscheidet, ob die Stichnoten als eine erste oder zweite Stimme eingefügt werden sollen; `UP` entspricht der ersten Stimme, `DOWN` der zweiten.

```
Oboe = \relative c'' {
  r2 r8 d16 f e g f a
  g8 g16 g g2.
}
\addQuote "oboe" { \Oboe }

\new Voice \relative c'' {
  \cueDuring #"oboe" #UP { R1 }
  g2 c,
}
```
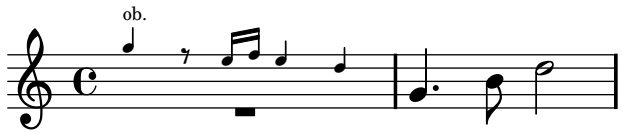


In diesem Beispiel muss der `Voice`-Kontext explizit begonnen werden, damit nicht der gesamte musikalische Ausdruck als Stichnoten-Stimme formatiert wird.

Die Bezeichnung des Instruments, von dem die Stichnoten genommen werden, kann auch ausgegeben werden, wenn die Eigenschaft `instrumentCueName` im `CueVoice`-Kontext definiert wird.

```
Oboe = \relative c''' {
  g4 r8 e16 f e4 d
}
\addQuote "oboe" { \Oboe }

\new Staff \relative c'' <<
  \new CueVoice \with {
    instrumentCueName = "ob."
  }
  \new Voice {
    \cueDuring #"oboe" #UP { R1 }
    g4. b8 d2
  }
>>
```
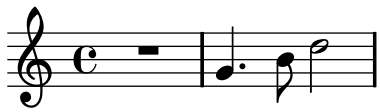


Zusätzlich zu der Instrumentenbezeichnung kann auch die Bezeichnung des Originalinstruments ausgegeben werden, und alle Änderungen, die für die Stichnoten gemacht wurden, müssen wieder rückgängig gemacht werden. Das kann mit den Befehlen `\addInstrumentDefinition` und `\instrumentSwitch` vorgenommen werden. Ein Beispiel und mehr Information findet sich in [Instrumentenbezeichnungn], Seite 142.

Der `\killCues`-Befehl entfernt Stichnoten aus einem musikalischen Ausdruck. Das kann nützlich sein, wenn die Stichnoten von einer Stimme entfernt werden sollen, aber in einer anderen Edition benötigt werden.

```
Flöte = \relative c''' {
  r2 cis2 r2 dis2
}
\addQuote "flute" { \Flöte }

\new Voice \relative c'' {
  \killCues {
    \cueDuring #"flute" #UP { R1 }
    g4. b8 d2
  }
}
```



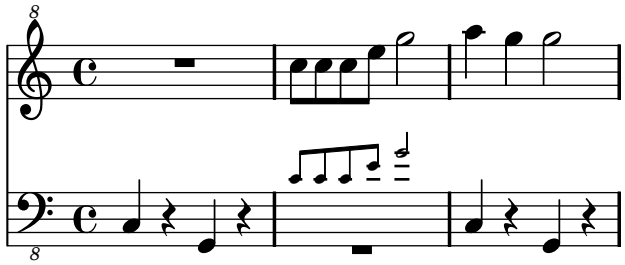
Der `\transposedCueDuring`-Befehl bietet sich an, wenn man Stichnoten eines Instrumentes mit einem vollständig anderen Register hinzufügen will. Die Syntax ähnelt der des `\cueDuring`-

Befehls, aber ein zusätzliches Argument wird benötigt, das die Transposition der Stichnoten-Stimme bezeichnet. Mehr Information zu Transposition siehe [Transposition von Instrumenten], Seite 17.

```
Piccolo = \relative c''' {
  \clef "treble^8"
  R1
  c8 c c e g2
  a4 g g2
}
\addQuote "Piccolo" { \Piccolo }

Ktrfgt = \relative c, {
  \clef "bass_8"
  c4 r g r
  \transposedCueDuring #"Piccolo" #UP c,, { R1 }
  c4 r g r
}

<<
  \new Staff = "Piccolo" \Piccolo
  \new Staff = "Ktrfgt" \Ktrfgt
>>
```



Es ist möglich, Zitate mit eindeutigen Bezeichnungen zu versehen (unter Benutzung von *tags*), um sie auf unterschiedliche Weise zu verarbeiten. Einzelheiten zu diesem Vorgehen werden vorgestellt in [Marken benutzen], Seite 317.

Siehe auch

Notationsreferenz: [Transposition von Instrumenten], Seite 17, [Instrumentenbezeichnungn], Seite 142, [Marken benutzen], Seite 317.
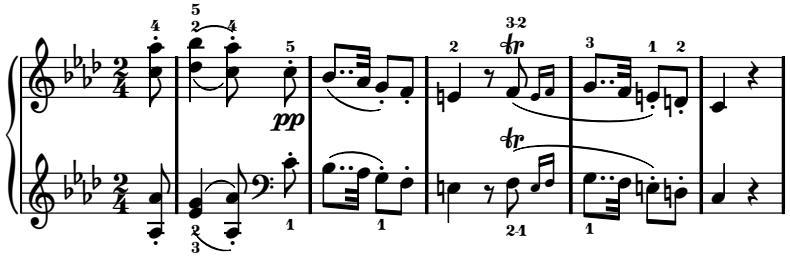
Schnipsel: Abschnitt “Staff notation” in *Schnipsel*.

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

Bekannte Probleme und Warnungen

Zusammenstöße können zwischen Pausen der Hauptstimme und den Stichnoten des `CueVoice`-Kontexts auftreten.

1.7 Anmerkungen



Dieser Abschnitt zeigt die verschiedenen Möglichkeiten, die Erscheinung der Noten zu ändern und analytische bzw. pädagogische Anmerkungen anzubringen.

1.7.1 Innerhalb des Systems

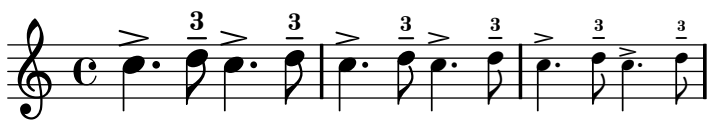
Dieser Abschnitt zeigt, wie man Elemente hervorhebt, die sich innerhalb des Notensystems befinden.

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 [Überblick über die wichtigsten Textbeschriftungsbefehle], Seite 172.
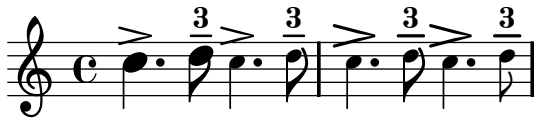
```
\huge
c4.-> d8---3
\large
c4.-> d8---3
\normalsize
c4.-> d8---3
\small
c4.-> d8---3
\tiny
c4.-> d8---3
\teeny
c4.-> d8---3
```



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 10pt 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
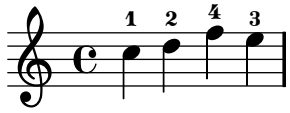
Schnipsel: Abschnitt “Editorial annotations” in *Schnipsel*.

Referenz der Interna: Abschnitt “font-interface” in *Referenz der Interna*.

Fingersatzanweisungen
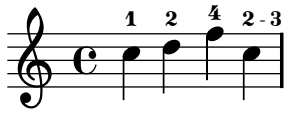
Fingersatzanweisungen können folgenderweise notiert werden: *Note-Zahl*

```
c4-1 d-2 f-4 e-3
```
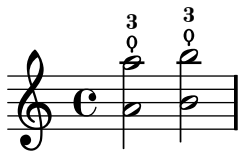


Für Fingerwechsel muss eine Textbeschriftung (markup) benutzt werden:

```
c4-1 d-2 f-4 c^\markup { \finger "2 - 3" }
```
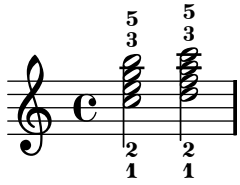


Mit dem Daumen-Befehl (`\thumb`) können die Noten bezeichnet werden, die mit dem Daumen (etwa auf dem Cello) gespielt werden sollen.

```
<a_\thumb a'-3>2 <b_\thumb b'-3>
```



Fingersätze für Akkorde können auch zu einzelnen Noten des Akkordes hinzugefügt werden, indem sie innerhalb der Akkord-Klammer direkt an die Noten angefügt werden.

```
<c-1 e-2 g-3 b-5>2 <d-1 f-2 a-3 c-5>
```



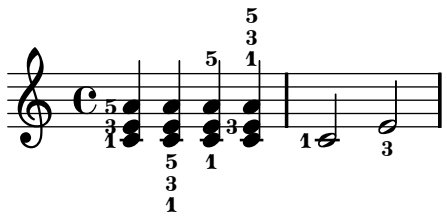
Fingersatzanweisungen können manuell oberhalb des Systems gesetzt werden, siehe Abschnitt 5.4.2 [Richtung und Platzierung], Seite 335.

Ausgewählte Schnipsel

Controlling the placement of chord fingerings

The placement of fingering numbers can be controlled precisely.

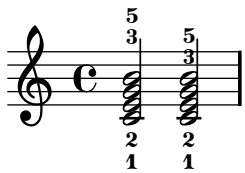
```
\relative c' {
  \set fingeringOrientations = #'(left)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(down)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(down right up)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(up)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(left)
  <c-1>2
  \set fingeringOrientations = #'(down)
  <e-3>2
}
```



Allowing fingerings to be printed inside the staff

By default, vertically oriented fingerings are positioned outside the staff. However, this behavior can be canceled.

```
\relative c' {
  <c-1 e-2 g-3 b-5>2
  \once \override Fingering #'staff-padding = #'()
  <c-1 e-2 g-3 b-5>2
}
```



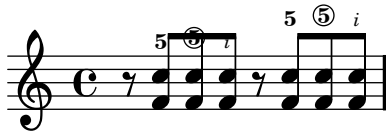
Avoiding collisions of chord fingering with beams

Fingerings and string numbers applied to individual notes will automatically avoid beams, but this is not true by default for fingerings and string numbers applied to the individual notes of chords. The following example shows how this default behavior can be overridden:

```
\relative c' {
  \set fingeringOrientations = #'(up)
  \set stringNumberOrientations = #'(up)
  \set strokeFingerOrientations = #'(up)

  % Standart
  r8
  <f c'-5>8
  <f c'\5>8
  <f c'-\rightHandFinger #2 >8

  % Korrigiert, um Zusammenstöße zu vermeiden
  r8
  \override Fingering #'add-stem-support = ##t
  <f c'-5>8
  \override StringNumber #'add-stem-support = ##t
  <f c'\5>8
  \override StrokeFinger #'add-stem-support = ##t
  <f c'-\rightHandFinger #2 >8
}
```



Siehe auch

Notationsreferenz: Abschnitt 5.4.2 [Richtung und Platzierung], Seite 335.

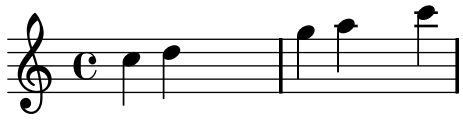
Schnipsel: Abschnitt “Editorial annotations” in *Schnipsel*.

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*.

Unsichtbare Noten
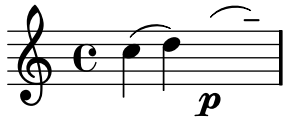
Versteckte (oder unsichtbare oder transparente) Noten können sinnvoll sein, wenn man Notation für den Theorieunterricht oder Kompositionsübungen erstellen will.

```
c4 d
\hideNotes
e4 f
\unHideNotes
g a
\hideNotes
b
\unHideNotes
c
```



Notationsobjekte, die an die unsichtbaren Noten angefügt sind, sind weiterhin sichtbar.

```
c4( d)
\hideNotes
e4(\p f)--
```



Vordefinierte Befehle

`\hideNotes`, `\unHideNotes`.

Siehe auch

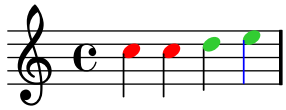
Schnipsel: Abschnitt “Editorial annotations” in *Schnipsel*.

Referenz der Interna: Abschnitt “Note_spacing_engraver” in *Referenz der Interna*, Abschnitt “NoteSpacing” in *Referenz der Interna*.

Farbige Objekte

Einzelnen Objekten können einfach eigene Farben zugewiesen werden. Gültige Farben-Bezeichnungen sind aufgelistet in Abschnitt B.5 [Liste der Farben], Seite 351.

```
\override NoteHead #'color = #red
c4 c
\override NoteHead #'color = #(x11-color 'LimeGreen)
d
\override Stem #'color = #blue
e
```
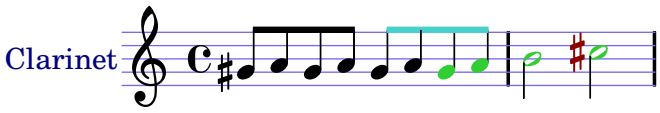


Die ganze Farbpalette, die für X11 definiert ist, kann mit der Scheme-Funktion `x11-color` benutzt werden. Diese Funktion hat ein Argument: entweder ein Symbol in der Form *'FooBar* oder eine Zeichenkette in der Form `"FooBar"`. Die erste Form ist schneller zu schreiben und effizienter. Mit der zweiten 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.

```
\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
```
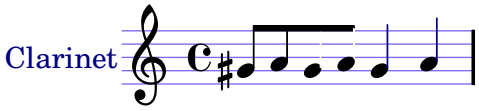
```
\override NoteHead #'color = #(x11-color "LimeGreen")
gis a
% Das ist Unsinn, die Hälsen bleiben schwarz
\override Stem #'color = #(x11-color 'Boggle)
b2 cis
```



Exakte RGB-Farben können mit Hilfe der Scheme-Funktion `rgb-color` definiert werden.

```
\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 B.5 [Liste der Farben], Seite 351, Abschnitt 5.3.4 [Der tweak-Befehl], Seite 335.

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.
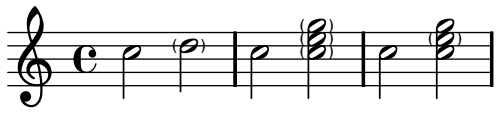
Nicht alle X11-Farben lassen sich am Webbrowser erkennen, d. h. der Unterschied etwa zwischen `'LimeGreen` und `'ForestGreen` wird eventuell nicht dargestellt. Für die Benutzung im Internet wird die Benutzung von einfachen Farben nahegelegt (z. B. `#blue`, `#green`, `#red`).

Noten in Akkorden können nicht mit `\override` eingefärbt werden, dazu muss `\tweak` benutzt werden. Siehe auch Abschnitt 5.3.4 [Der tweak-Befehl], Seite 335.

Klammern
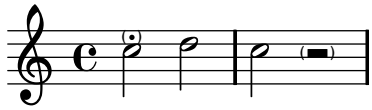
Objekte können in Klammern gesetzt werden, indem vor ihnen der Befehl `\parenthesize` geschrieben wird. Wenn ein Akkord in Klammern gesetzt wird, wirkt sich das auf jede Noten im Akkord aus. Innerhalb von einem Akkord gesetzte Befehle wirken sich auf einzelne Noten aus.

```
c2 \parenthesize d
c2 \parenthesize <c e g>
c2 <c \parenthesize e g>
```



Auch andere Objekte als Noten können in Klammern gesetzt werden.

```
c2-\parenthesize -. d
c2 \parenthesize r
```



Siehe auch

Schnipsel: Abschnitt “Editorial annotations” in *Schnipsel*.

Referenz der Interna: Abschnitt “Parenthesis_engraver” in *Referenz der Interna*, Abschnitt “ParenthesesItem” 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.

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.

Vordefinierte Befehle
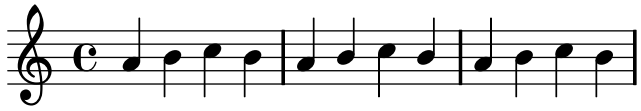
`\stemUp` (Hälse nach oben), `\stemDown` (Hälse nach unten), `\stemNeutral` (Hälse je nach Notenposition).

Ausgewählte Schnipsel

Default direction of stems on the center line of the staff

The default direction of stems on the center line of the staff is set by the `Stem` property `neutral-direction`.

```
\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 5.4.2 [Richtung und Platzierung], Seite 335.

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*.

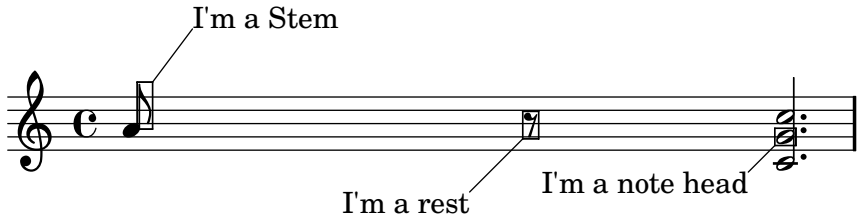
1.7.2 Außerhalb des Notensystems

Dieser Abschnitt zeigt, wie man Elemente im System von außerhalb des Systems hervorhebt.

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.

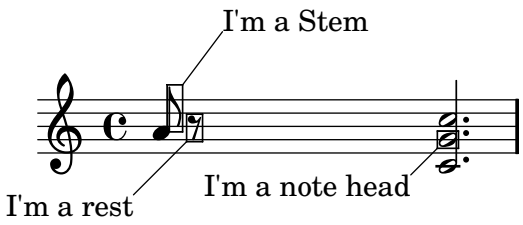
```
\new Voice \with { \consists "Balloon_engraver" }
{
  \balloonGrobText #'Stem #'(3 . 4) \markup { "I'm a Stem" }
  a8
  \balloonGrobText #'Rest #'(-4 . -4) \markup { "I'm a rest" }
  r
  <c, g'-\balloonText #'(-2 . -2) \markup { "I'm a note head" } c>2.
}
```



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 “BalloonTextItem” in *Referenz der Interna*, Abschnitt “balloon-interface” in *Referenz der Interna*.

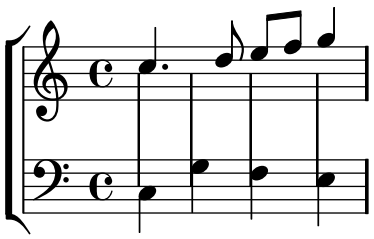
Gitternetzlinien

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 = #(ly:make-moment 1 4)
  }
  \context {
    \Score
    \consists "Grid_line_span_engraver"
  }
}

\score {
  \new ChoirStaff <<
    \new Staff \relative c'' {
      \stemUp
      c4. d8 e8 f g4
    }
    \new Staff \relative c {
      \clef bass
      \stemDown
      c4 g' f e
    }
  >>
}
```
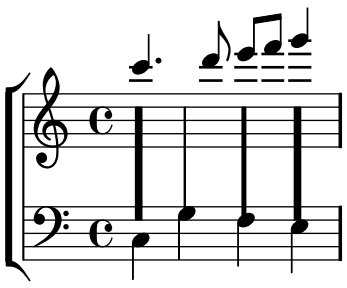


Ausgewählte Schnipsel

Grid lines: changing their appearance

The appearance of grid lines can be changed by overriding some of their properties.

```
\score {
  \new ChoirStaff <<
    \new Staff {
      \relative c'' {
        \stemUp
        c'4. d8 e8 f g4
      }
    }
    \new Staff {
      \relative c {
        % damit werden sie einen Notenlinienzwischenraum hochgeschoben
        \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
      % Gitter einstellen
      \consists "Grid_point_engraver"
      % den Gitter-Abstand auf eine Viertelnote setzen
      gridInterval = #(ly:make-moment 1 4)
    }
    \context {
      \Score
      \consists "Grid_line_span_engraver"
      % damit werden sie nach rechts um einen halben Notenlinienabstand verschoben
      \override NoteColumn #'X-offset = #-0.5
    }
  }
}
```



Siehe auch
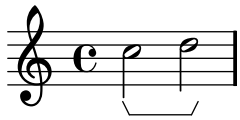
Schnipsel: Abschnitt “Editorial annotations” in *Schnipsel*.

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*.

Analyseklammern

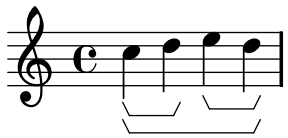
Klammern über dem System werden in der Musikanalyse benutzt, um strukturelle Einheiten der Musik zu markieren. Einfache horizontale Klammern werden von LilyPond unterstützt.

```
\layout {
  \context {
    \Voice
    \consists "Horizontal_bracket_engraver"
  }
}
\relative c'' {
  c2\startGroup
  d\stopGroup
}
```



Analysis brackets may be nested.

```
\layout {
  \context {
    \Voice
    \consists "Horizontal_bracket_engraver"
  }
}
\relative c'' {
  c4\startGroup\startGroup
  d4\stopGroup
  e4\startGroup
  d4\stopGroup\stopGroup
}
```
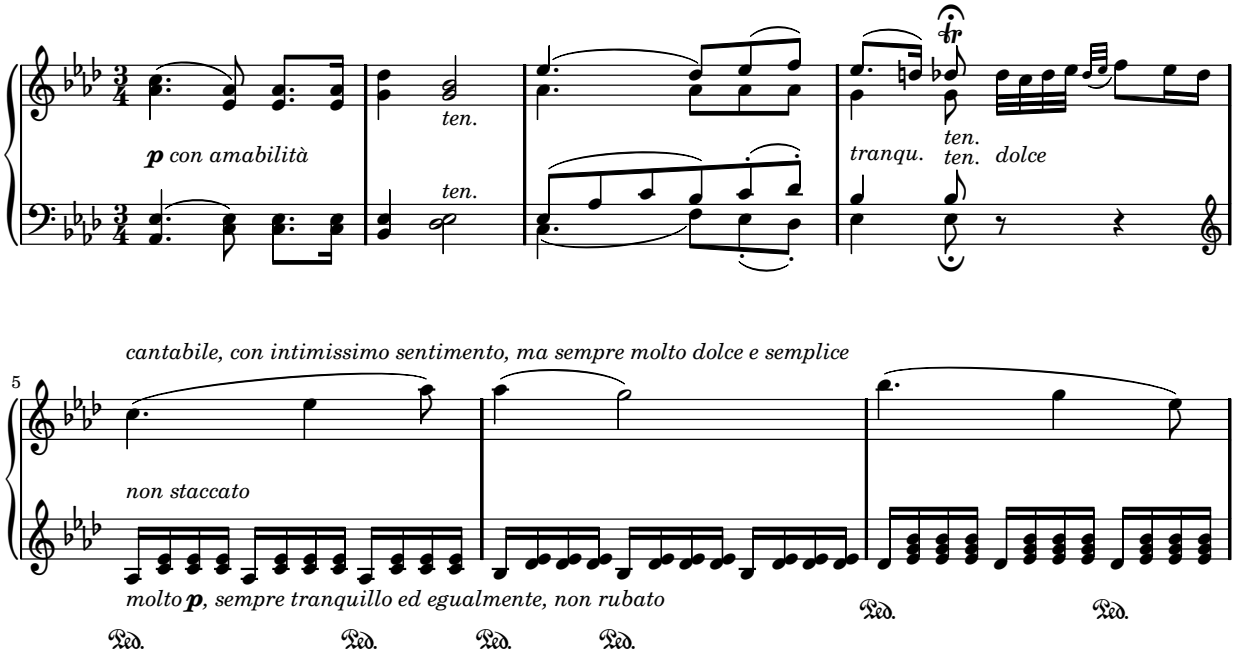


Siehe auch

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*.

1.8 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: Abschnitt 2.1 [Notation von Gesang], Seite 187, Abschnitt 3.2 [Titel], Seite 307.

1.8.1 Text eingeben
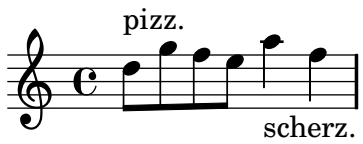
Dieser Abschnitt zeigt verschiedene Arten, wie Text in die Partitur eingefügt werden kann.

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 3.3.3 [Zeichenkodierung], Seite 320.

Textarten
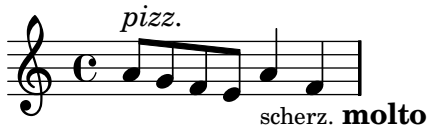
Am einfachsten kann Text mit geraden Anführungsstrichen in eine Partitur eingefügt werden, wie das folgende Beispiel zeigt. Derartiger Text kann manuell über oder unter dem Notensystem platziert werden, die Syntax hierzu ist beschrieben in Abschnitt 5.4.2 [Richtung und Platzierung], Seite 335.

```
d8^"pizz." g f e a4-"scherz." f
```
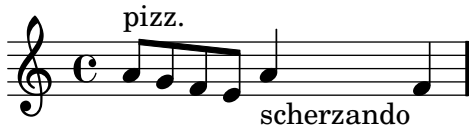


Diese Syntax ist eine Kurzform, komplexere Formatierungen können einem Text hinzugefügt werden, wenn man explizit den `\markup`-Befehl mit darauf folgenden geschweiften Klammern einsetzt, wie beschrieben in Abschnitt 1.8.2 [Text formatieren], Seite 170.

```
a8^\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.

```
a8^"pizz." g f e
\textLengthOn
a4_"scherzando" f
```



Vordefinierte Befehle

`\textLengthOn`, `\textLengthOff`.

Siehe auch

Notationsreferenz: Abschnitt 1.8.2 [Text formatieren], Seite 170, Abschnitt 5.4.2 [Richtung und Platzierung], Seite 335.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

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

Bekannte Probleme und Warnungen

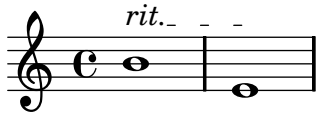
Eine Überprüfung, ob sich auch alle Textbeschriftungen und Gesangstext innerhalb der Ränder der Noten befinden, braucht verhältnismäßig viel Rechenaufwand. Diese Überprüfung ist standardmäßig ausgestellt, damit LilyPond die Dateien schneller bearbeiten kann. Man kann die Überprüfung aber mit folgendem Code einschalten:

```
\override Score.PaperColumn #'keep-inside-line = ##t
```

Text mit Verbindungslinien

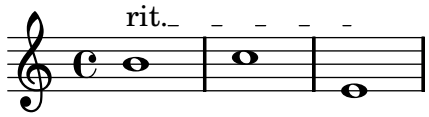
Einige Aufführungsanweisungen, etwa *rallentando* oder *accelerando*, werden als Text geschrieben, gefolgt von einer gestrichelten Linie, die anzeigt, wie weit sich die Anweisung auswirkt. Solche Objekte, „Strecker“ (engl. spanners) genannt, können von einer Note bis zu einer anderen mit folgender Anweisung erstellt werden:

```
\override TextSpanner #'(bound-details left text) = "rit."
b1\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 1.8.2 [Text formatieren], Seite 170.

```
\override TextSpanner #'(bound-details left text) =
  \markup { \upright "rit." }
b1\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 5.4.7 [Zeilenstile], Seite 336.

Vordefinierte Befehle

`\textSpannerUp`, `\textSpannerDown`, `\textSpannerNeutral`.

Siehe auch

Notationsreferenz: Abschnitt 5.4.7 [Zeilenstile], Seite 336, [Dynamik], Seite 85.
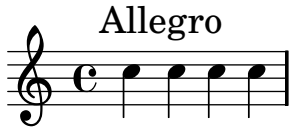
Schnipsel: Abschnitt “Text” in *Schnipsel*.

Referenz der Interna: Abschnitt “TextSpanner” in *Referenz der Interna*.
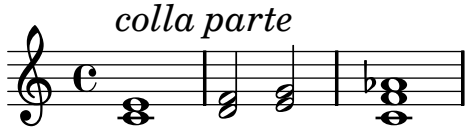
Textartige Zeichen

Verschiedene Textelemente können der Partitur hinzugefügt werden, indem man die Syntax für Zeichen einsetzen, wie beschrieben in [Übungszeichen], Seite 75:
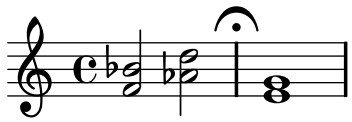
```
c4
\mark "Allegro"
c c c
```



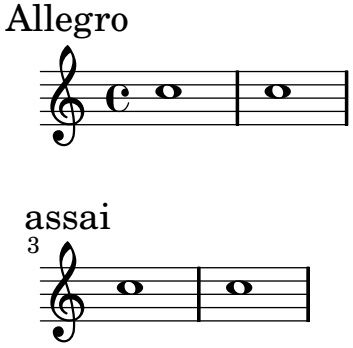
Diese Syntax ermöglicht es, beliebigen Text über eine Taktlinie zu platzieren, weitere Formatierungsmöglichkeiten sind mit dem `\markup`-Befehl gegeben, wie beschrieben in Abschnitt 1.8.2 [Text formatieren], Seite 170:



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 [Musikalische Notation innerhalb einer Textbeschriftung], Seite 180:



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.
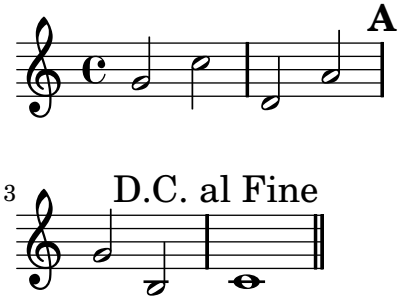


Ausgewählte Schnipsel

Printing marks at the end of a line or a score

Marks can be printed at the end of the current line, instead of the beginning of the following line. This is particularly useful when a mark has to be added at the end of a score – when there is no next line.

In such cases, the right end of the mark has to be aligned with the final bar line, as demonstrated on the second line of this example.

```
\relative c'' {
  \override Score.RehearsalMark #'break-visibility = #begin-of-line-invisible
  g2 c
  d,2 a'
  \mark \default
  \break
  g2 b,
  c1 \bar "||"
  \override Score.RehearsalMark #'self-alignment-X = #RIGHT
  \mark "D.C. al Fine"
}
```



*Aligning marks with various notation objects*If specified, text marks may be aligned with notation objects other than bar lines. These objects include `ambitus`, `breathing-sign`, `clef`, `custos`, `staff-bar`, `left-edge`, `key-cancellation`, `key-signature`, and `time-signature`.

In such cases, text marks will be horizontally centered above the object. However this can be changed, as demonstrated on the second line of this example (in a score with multiple staves, this setting should be done for all the staves).

```
\relative c' {
  e1

  % RehearsalMark zentrieren über dem Schlüssel
  \override Score.RehearsalMark #'break-align-symbols = #'(clef)
  \key a \major
  \clef treble
  \mark ""
  e1

  % RehearsalMark zentrieren über der Taktart (TimeSignature)
  \override Score.RehearsalMark #'break-align-symbols = #'(time-signature)
  \key a \major
  \clef treble
  \time 3/4
  \mark ""
  e2.

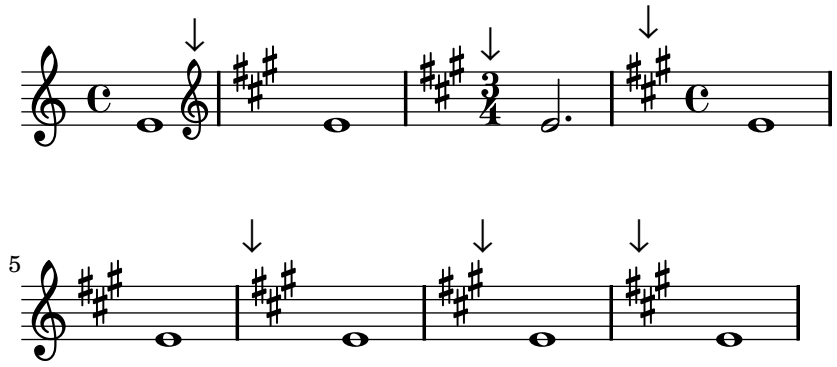
  % RehearsalMark über der Tonart (KeySignature) zentrieren
  \override Score.RehearsalMark #'break-align-symbols = #'(key-signature)
  \key a \major
  \clef treble
  \time 4/4
  \mark ""
  e1

  \break
  e1

  % RehearsalMark mit der linken Ecke von KeySignature ausrichten
  \once \override Score.KeySignature #'break-align-anchor-alignment = #LEFT
  \mark ""
  \key a \major
  e1

  % RehearsalMark mit der rechten Ecke von KeySignature ausrichten
  \once \override Score.KeySignature #'break-align-anchor-alignment = #RIGHT
  \key a \major
  \mark ""
  e1

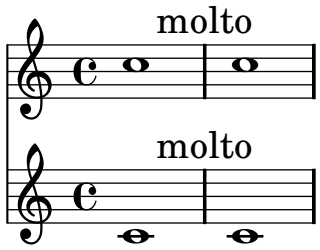
  % RehearsalMark mit der linken Ecke von KeySignature ausrichten
  % nach rechts um eine Einheit verschieben
  \once \override Score.KeySignature #'break-align-anchor = #1
  \key a \major
  \mark ""
  e1
}
```



Printing marks on every staff

Although text marks are normally only printed above the topmost staff, they may also be printed on every staff.

```
\score {
  <<
    \new Staff { c''1 \mark "molto" c'' }
    \new Staff { c'1 \mark "molto" c' }
  >>
  \layout {
    \context {
      \Score
      \remove "Mark_engraver"
      \remove "Staff_collecting_engraver"
    }
    \context {
      \Staff
      \consists "Mark_engraver"
      \consists "Staff_collecting_engraver"
    }
  }
}
```



Siehe auch

Notationsreferenz: [Übungszeichen], Seite 75, Abschnitt 1.8.2 [Text formatieren], Seite 170, [Musikalische Notation innerhalb einer Textbeschriftung], Seite 180, Abschnitt B.6 [Die Feta-Schriftart], Seite 352.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

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

Bekannte Probleme und Warnungen

Wenn ein Zeichen am Ende des letzten Taktes eines Stückes notiert wird (wo also keine nächste Zeile mehr existiert), wird das Zeichen überhaupt nicht gesetzt.

Separater Text

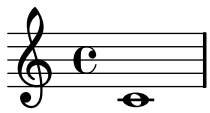
Eine \markup-Umgebung kann auch für sich alleine existieren, außerhalb einer \score-Umgebung, als ein Ausdruck auf der höchsten Ebene.

```
\markup {
  Morgen, morgen, und morgen...
}
```

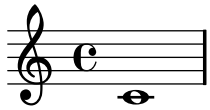
Morgen, morgen, und morgen...

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 3.1.2 [Mehrere Partituren in einem Buch], Seite 304.

```
\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 [Textbeschriftung über mehrere Seiten], Seite 182.

Vordefinierte Befehle

\markup, \markuplines.

Ausgewählte Schnipsel

Stand-alone two-column markup

Stand-alone text may be arranged in several columns using \markup commands:

```
\markup {
 \fill-line {
  \hspace #1.0
  \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 {
   \line { \italic {"O sacred feast"} }
   \line { \italic {"in which Christ is received,"} }
   \line { \italic {"the memory of His Passion is renewed,"} }
   \line { \italic {"the mind is filled with grace," } }
   \line { \italic {"and a pledge of future glory is given to us." }}
   \line { \italic {"Amen."}}
  }
  \hspace #1.0
 }
}
```

O sacrum convivium	*O sacred feast*
in quo Christus sumitur,	*in which Christ is received,*
recolitur memoria passionis ejus,	*the memory of His Passion is renewed,*
mens impletur gratia,	*the mind is filled with grace,*
futurae gloriae nobis pignus datur.	*and a pledge of future glory is given to us.*
Amen.	*Amen.*

Siehe auch

Notationsreferenz: Abschnitt 1.8.2 [Text formatieren], Seite 170, Abschnitt 3.1.3 [Die Dateistruktur], Seite 305, Abschnitt 3.1.2 [Mehrere Partituren in einem Buch], Seite 304, [Textbeschriftung über mehrere Seiten], Seite 182.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

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

1.8.2 Text formatieren

Dieser Abschnitt zeigt grundlegende und fortgeschrittene Formatierung von Text, wobei der Textbeschriftungsmodus (`\markup` benutzt wird.

Textbeschriftung (Einleitung)
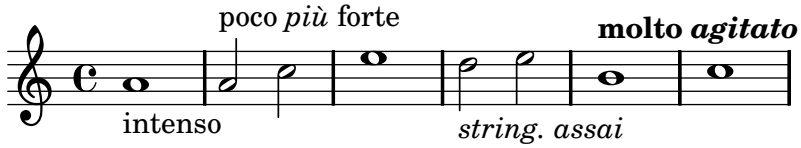
Eine `\markup`-Umgebung wird benutzt, um Text mit einer großen Anzahl von Formatierungsmöglichkeiten (im „markup-Modus“) zu setzen.

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.
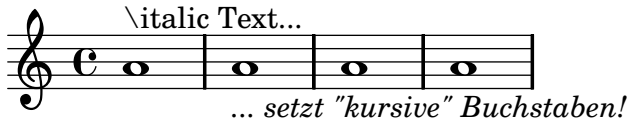
```
a1-\markup intenso
a2^\markup { poco \italic più forte  }
c e1
d2_\markup { \italic "string. assai" }
e
```

```
b1^\markup { \bold { molto \italic  agitato } }
c
```



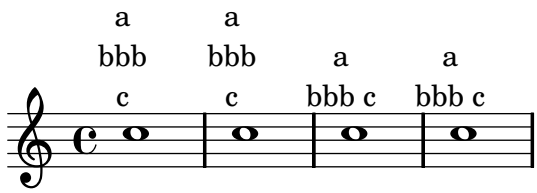
Eine \markup-Umgebung kann auch Text in Anführungszeichen beinhalten. Derartige Zeichenketten werden als ein Textausdruck angesehen, und darum werden innerhalb von ihnen Befehle oder Sonderzeichen (wie \ oder #) so ausgegeben, wie sie eingeben werden. Doppelte Anführungsstriche können gesetzt werden, indem man ihnen einen Backslash voranstellt.

```
a1^"\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:
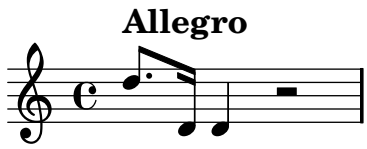
```
c1^\markup { \center-column { a bbb c } }
c1^\markup { \center-column { a { bbb c } } }
c1^\markup { \center-column { a \line { bbb c } } }
c1^\markup { \center-column { a "bbb c" } }
```



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 B.8 [Text markup commands], Seite 367.

Siehe auch

Notationsreferenz: Abschnitt B.8 [Text markup commands], Seite 367.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

Installierte Dateien: ‘`scm/markup.scm`’.
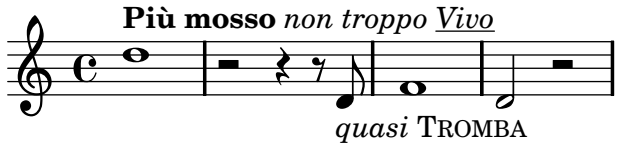
Bekannte Probleme und Warnungen

Syntaxfehler im Textbeschriftungsmodus können sehr verwirrend sein.

Überblick über die wichtigsten Textbeschriftungsbefehle

Einfache Änderungen des Schriftartschnitts können im Textbeschriftungsmodus vorgenommen werden:

```
d1^\markup {
  \bold { Più mosso }
  \italic { non troppo \underline Vivo }
}
r2 r4 r8
d,_\markup { \italic quasi \smallCaps Tromba }
f1 d2 r
```
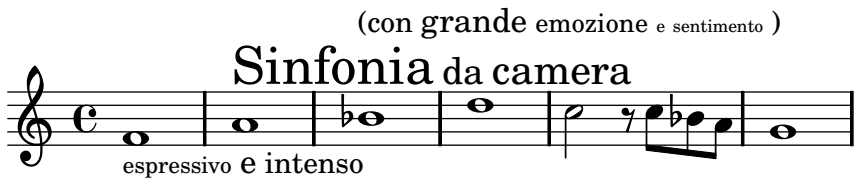


Die Größe von Buchstaben kann auf verschiedene Arten verändert werden:

- die Schriftgröße kann auf bestimmte deinierte Standardgrößen gesetzt werden,
- die Schriftgröße kann mit einem absoluten Wert gesetzt werden,
- die Schriftgröße kann relativ zur vorhergehenden Größe geändert werden.

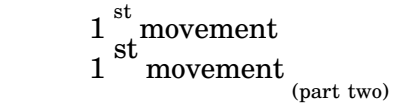
Das Beispiel unten zeigt alle drei Möglichkeiten:

```
f1_\markup {
  \tiny espressivo
  \large e
  \normalsize intenso
}
a^\markup {
  \fontsize #5 Sinfonia
  \fontsize #2 da
  \fontsize #3 camera
}
bes^\markup { (con
  \larger grande
  \smaller emozione
  \magnify #0.6 { e sentimento } )
}
d c2 r8 c bes a g1
```



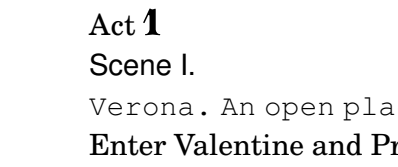
Text kann auch hoch- bzw. tiefgestellt gesetzt werden. Die so markierten Buchstaben werden automatisch in einer kleineren Schriftgröße gesetzt, aber die normale Schriftgröße kann auch eingesetzt werden:

```
\markup {
  \column {
    \line { 1 \super st movement }
    \line { 1 \normal-size-super st movement
      \sub { (part two) }  }
  }
}
```



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 `\roman` wieder herstellen lässt.

```
\markup {
  \column {
    \line { Act \number 1 }
    \line { \sans { Scene I. } }
    \line { \typewriter { Verona. An open place. } }
    \line { Enter \roman Valentine and Proteus. }
  }
}
```



Einige dieser Schriftarten, etwa die Zahlenschriftart oder die Schriftart für Dynamikzeichen, stellen nicht alle Zeichen zur Verfügung, wie beschrieben in [Neue Lautstärkezeichen], Seite 89 und [Manuelle Wiederholungszeichen], Seite 103.

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 B.8.1 [Font], Seite 367.

Es ist auch möglich, eigene Schriftfamlien zu definieren, wie erklärt in Abschnitt 1.8.3 [Schriftarten], Seite 183.

Vordefinierte Befehle

`\teeny`, `\tiny`, `\small`, `\normalsize`, `\large`, `\huge`, `\smaller`, `\larger`.

Siehe auch

Notationsreferenz: Abschnitt B.8.1 [Font], Seite 367, [Neue Lautstärkezeichen], Seite 89, [Manuelle Wiederholungszeichen], Seite 103, Abschnitt 1.8.3 [Schriftarten], Seite 183.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

Referenz der Interna: Abschnitt “TextScript” in *Referenz der Interna*.
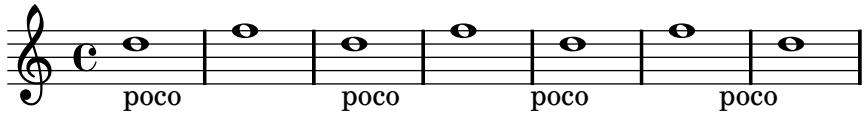
Installierte Dateien: ‘`scm/define-markup-commands.scm`’.

Textausrichtung

Dieser Abschnitt zeigt, wie man Text im Textbeschriftungsmodus eingibt. Textobjekte können auch als eine Einheit verschoben werden, wie beschrieben in Abschnitt “Moving objects” in *Handbuch zum Lernen*.

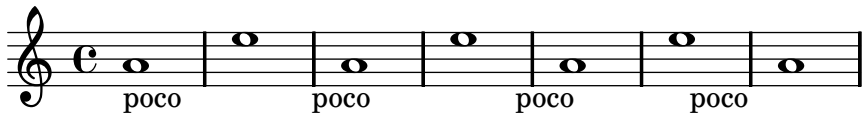
Textbeschriftungsobjekte können auf verschiedene Weise ausgerichtet werden. Standardmäßig wird ein Textobjekt an seiner linken Ecke ausgerichtet, darum wird das erste und zweite Objekt gleichermaßen an der linken Ecke ausgerichtet.

```
d1-\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:

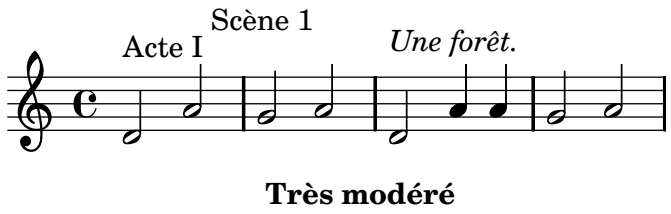
```
a1-\markup { \halign #-1 poco }
e'
a,-\markup { \halign #0 poco }
e'
a,-\markup { \halign #0.5 poco }
e'
a,-\markup { \halign #2 poco }
```



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 [Textartige Zeichen], Seite 165.

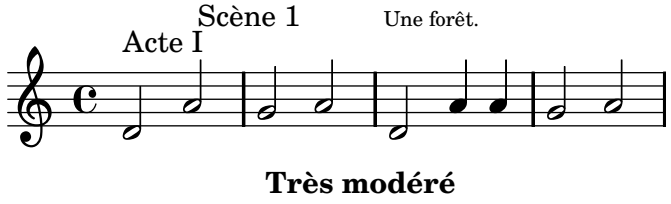
Die vertikale Ausrichtung ist etwas schwieriger. Textelemente können komplett verschoben werden, es ist aber auch möglich, nur einen Teil innerhalb der Textbeschriftung zu bewegen. In diesem Fall muss dem zu verschiebenden Objekt ein Ankerpunkt zugewiesen werden, welcher entweder ein anderes Textelement 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.

```
d2^\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:

```
d2^\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 }
  }
}
```

a
b c
d e f

a
b c
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 }
}
```

William S. Gilbert THE MIKADO Sir Arthur Sullivan

or

THE TOWN OF TITIPU

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 B.8.2 [Align], Seite 376.

Siehe auch

Handbuch zum Lernen: Abschnitt “Moving objects” in *Handbuch zum Lernen*.

Notationsreferenz: Abschnitt B.8.2 [Align], Seite 376, [Textartige Zeichen], Seite 165.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

Installierte Dateien: ‘`scm/define-markup-commands.scm`’.

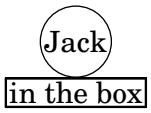
Referenz der Interna: Abschnitt “TextScript” in *Referenz der Interna*.

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 B.8.2 [Align], Seite 376.

```
\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."
}
```

Charles Ives (1874 - 1954)

THE UNANSWERED QUESTION

A Cosmic Landscape

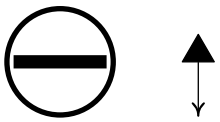
Largo to Presto

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

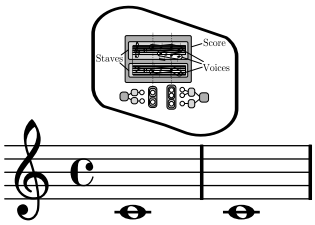
  \center-column {
    \triangle ##t
    \combine
      \draw-line #'(0 . 4)
      \arrow-head #Y #DOWN ##f
  }
}
```



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:

```
c1^\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 B.8.3 [Graphic], Seite 389.

Siehe auch

Notationsreferenz: Abschnitt B.8.3 [Graphic], Seite 389, Abschnitt 1.7 [Anmerkungen], Seite 152.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

Referenz der Interna: Abschnitt “TextScript” in *Referenz der Interna*.
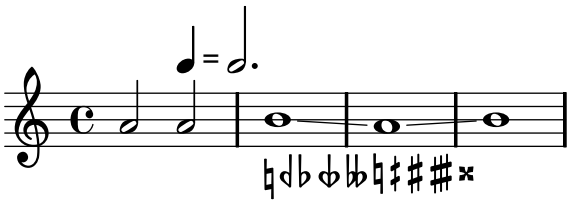
Installierte Dateien: ‘`scm/define-markup-commands.scm`’, ‘`scm/stencil.scm`’.

Musikalische Notation innerhalb einer Textbeschriftung

Auch Musikobjekte können innerhalb der Textbeschriftungsumgebung gesetzt werden.

Noten und Versetzungszeichen lassen sich mit `\markup` einfügen:

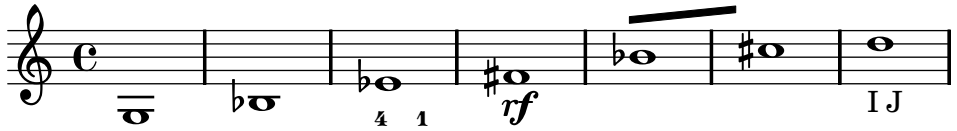
```
a2 a^\markup {
  \note #"4" #1
  =
  \note-by-number #1 #1 #1.5
}
b1_\markup {
  \natural \semiflat \flat
  \sesquiflat \doubleflat
}
\glissando
a1_\markup {
  \natural \semisharp \sharp
  \sesquisharp \doublesharp
}
\glissando b
```



Andere Notationsobjekte können auch eingefügt werden:

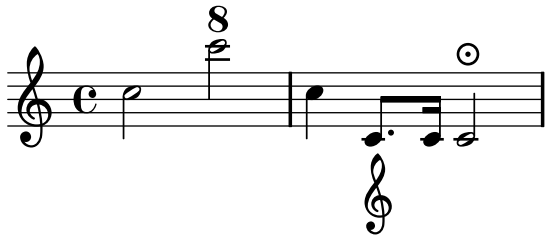
```
g1 bes
ees-\markup {
  \finger 4
```

```
  \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 B.6 [Die Feta-Schriftart], Seite 352.

```
c2
c'^\markup { \musicglyph #"eight" }
c,4
c,8._\markup { \musicglyph #"clefs.G_change" }
c16
c2^\markup { \musicglyph #"timesig.neomensural94" }
```

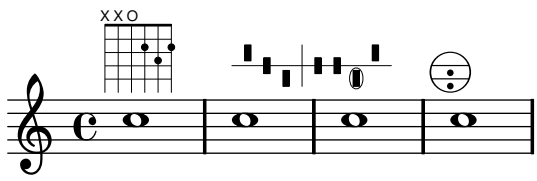


Eine andere Möglichkeit, andere als Textsymbole zu schreiben, findet sich in [Was sind Schriftarten], Seite 183.

Der Textbeschriftungsmodus unterstützt auch Diagramme für bestimmte Instrumente:

```
c1^\markup {
  \fret-diagram-terse #"x;x;o;2;3;2;"
}
c^\markup {
  \harp-pedal #"^-v|--ov^"
}
c
c^\markup {
  \combine
    \musicglyph #"accordion.accDiscant"
    \combine
      \raise #0.5 \musicglyph #"accordion.accDot"
      \raise #1.5 \musicglyph #"accordion.accDot"
```
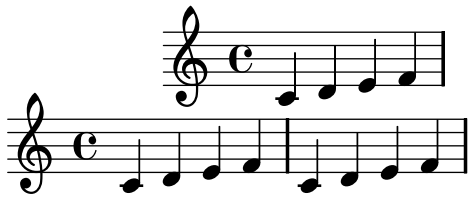
```
}
```



Derartige Digramme sind dokumentiert in Abschnitt B.8.5 [Instrument Specific Markup], Seite 396.

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:

```
c4 d^\markup {
  \score {
    \relative { c4 d e f }
    \layout { }
  }
}
e f |
c d e f
```



Eine vollständige Liste der Musiksymbol-Befehle findet sich in Abschnitt B.8.4 [Music], Seite 393.

Siehe auch

Notationsreferenz: Abschnitt B.8.4 [Music], Seite 393, Abschnitt B.6 [Die Feta-Schriftart], Seite 352, [Was sind Schriftarten], Seite 183.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

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

Installierte Dateien: ‘`scm/define-markup-commands.scm`’, ‘`scm/fret-diagrams.scm`’, ‘`scm/harp-pedals.scm`’.

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:

```
\markuplines {
  \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 B.9 [Text markup list commands], Seite 403.

Siehe auch

Notationsreferenz: Abschnitt B.9 [Text markup list commands], Seite 403, Abschnitt 6.4.4 [Neue Definitionen von Beschriftungsbefehlen für Listen], Seite 341.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

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

Installierte Dateien: ‘`scm/define-markup-commands.scm`’.

Vordefinierte Befehle

`\markuplines`.

1.8.3 Schriftarten

Dieser Abschnitt zeigt, wie Schriftarten eingesetzt werden können und wie man sie in Partituren ändern kann.
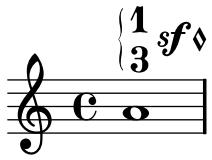
Was sind Schriftarten

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`-Schriftart zuzugreifen. Das ist die Standardschriftart für Notationselemente in LilyPond.

```
a1^\markup {
  \vcenter {
    \override #'(font-encoding . fetaBraces)
    \lookup #"brace120"
    \override #'(font-encoding . fetaNumber)
    \column { 1 3 }
    \override #'(font-encoding . fetaDynamic)
    sf
    \override #'(font-encoding . fetaMusic)
```

```
    \lookup #"noteheads.s0petrucci"
  }
}
```
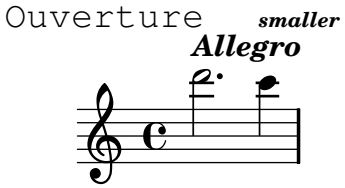


Eine einfachere, aber weniger vielfältige Syntax wird beschrieben in [Musikalische Notation innerhalb einer Textbeschriftung], Seite 180.

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 `tpyewriter` wird durch Pango entspechend 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
c4^smaller
```



Eine ähnliche Syntax kann im Textbeschriftungsmodus eingesetzt werden, hier bietet es sich aber an, die einfacheren Befehle zu verwenden, die erklärt wurden in [Überblick über die wichtigsten Textbeschriftungsbefehle], Seite 172:

```
\markup {
  \column {
    \line {
      \override #'(font-shape . italic)
      \override #'(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 vorefinierten Schriftarten umzuschalten, kann man auch eigene Schriftarten verwenden, wie erklärt in folgenden Abschnitten: [Schriftarten für einen Eintrag], Seite 185 und [Schriftart des gesamten Dokuments], Seite 186.

Siehe auch

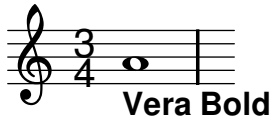
Notationsreferenz: Abschnitt B.6 [Die Feta-Schriftart], Seite 352, [Musikalische Notation innerhalb einer Textbeschriftung], Seite 180, [Überblick über die wichtigsten Textbeschriftungsbefehle], Seite 172, Abschnitt B.8.1 [Font], Seite 367.

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 = #"Charter"
\override Staff.TimeSignature #'font-size = #2
\time 3/4

a1_\markup {
  \override #'(font-name . "Vera Bold")
    { Vera Bold }
}
```



Mit folgendem Befehl erhält man eine Liste aller verfügbaren Schriftarten des Betriebssystems:

```
lilypond -dshow-available-fonts x
```

Das letzte Argument kann ein beliebiges Zeichen sein, aber es darf nicht fehlen.

Siehe auch

Notationsreferenz: [Was sind Schriftarten], Seite 183, [Schriftart des gesamten Dokuments], Seite 186.

Schnipsel: Abschnitt “Text” in *Schnipsel*.

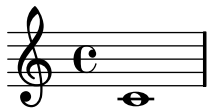
Installierte Dateien: ‘`lily/font-config-scheme.cc`’.

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 [Was sind Schriftarten], Seite 183.

```
\paper  {
  myStaffSize = #20
  #(define fonts
    (make-pango-font-tree "Times New Roman"
                          "Nimbus Sans"
                          "Luxi Mono"
                           (/ myStaffSize 20)))
}

\relative c'{
  c1-\markup {
    roman,
    \sans sans,
    \typewriter typewriter. }
}
```



roman, sans, `typewriter`.

Siehe auch

Notationsreferenz: [Was sind Schriftarten], Seite 183, [Schriftarten für einen Eintrag], Seite 185, [Überblick über die wichtigsten Textbeschriftungsbefehle], Seite 172, Abschnitt B.8.1 [Font], Seite 367.

2 Spezielle Notation

Dieser Abschnitt erklärt, wie Notation erstellt wird, die nur für ein bestimmtes Instrument oder einen Stil eingesetzt wird.

2.1 Notation von Gesang

Dieser Abschnitt erklärt, wie Vokalmusik gesetzt werden kann und die Silben von Gesangstext an den Noten ausgerichtet werden.

2.1.1 Übliche Notation für Vokalmusik

Dieser Abschnitt behandelt allgemeine Fragen der Notation von Vokalmusik und einige spezifische Vokalmusikstile.

Referenz für Vokalmusik und Gesangstext

Viele Probleme können auftreten, wenn man Vokalmusik setzt. Einige davon werden in diesem Abschnitt behandelt, während weitere sich in anderen Abschnitten befinden:

- Die meisten Vokalmusikstile benutzen Text für den Gesangstext. Eine Einleitung hierzu findet sich in Abschnitt “Setting simple songs” 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 [Textbeschriftung (Einleitung)], Seite 170.
- Liedblätter können erstellt werden, indem eine Gesangsstimme mit Akkorden kombiniert wird, Einzelheiten finden sich in Abschnitt 2.7 [Notation von Akkorden], Seite 260.
- ‚Ambitus‘ können zu Beginn der Stimmen hinzugefügt werden, dies findet sich erklärt in [Tonumfang], Seite 25.
- Gesangsstimmen können auch mit traditionellen Schlüsseln gesetzt werden, siehe [Notenschlüssel], Seite 12.
- Alte Vokalmusik ist unterstützt, Einzelheiten hierzu in Abschnitt 2.8 [Notation von alter Musik], Seite 279.

Oper

TBC

Liederhefte
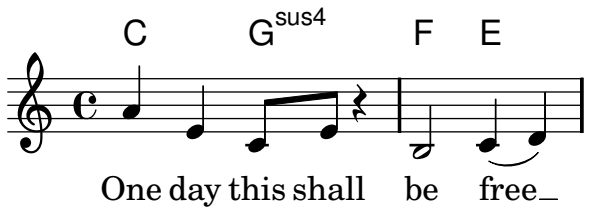
TBC

Ausgewählte Schnipsel

Simple lead sheet

When put together, chord names, a melody, and lyrics form a lead sheet:

```
<<
  \chords { c2 g:sus4 f e }
  \relative c'' {
    a4 e c8 e r4
    b2 c4( d)
  }
  \addlyrics { One day this shall be free __ }
>>
```



Siehe auch

Notationsreferenz: Abschnitt 2.7 [Notation von Akkorden], Seite 260.

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 [Besondere Notenköpfe], Seite 27.

Hymnen

TBC

Alte Vokalmusik

TBC

Siehe auch

Notationsreferenz: Abschnitt 2.8 [Notation von alter Musik], Seite 279.

2.1.2 Eingabe von Text

Was ist Gesangtext

LilyPond-Eingabedateien sind einfache Textdateien, in denen Text verwendet wird, um Notationssymbole darzustellen. Für die Notation von Gesangstext muss also sichergestellt sein, dass ein Buchstabe, etwa `d`, nicht als Note, sondern als Buchstabe „d“ interpretiert wird. Darum gibt es einen besonderen Modus, in dem Gesangstext geschreiben werden kann, den „Lyric“-Modus (engl. lyrics = Gesangstext).

Der Gesangstextmodus kann mit der Umgebung `\lyricmode` spezifiziert werden, oder indem `\addlyrics` bzw. `\lyricsto` eingesetzt wird. In diesem Modus kann Text mit Akzenten und Satzzeichen notiert werden, und das Programm geht davon aus, dass es sich auch um Text handelt. Silben werden wie Noten notiert, indem ihnen ihre Dauer angehängt wird:

```
\lyricmode { Twin-4 kle4 twin- kle litt- le star2 }
```

Es gibt zwei generelle Methoden, die horizontale Orientierung der Textsilben zu spezifizieren, 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.

Ein Wort oder eine Silbe beginnt mit einem alphabetischen Zeichen und endet mit einem Leerzeichen oder einer Zahl. Die folgenden Zeichen 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.

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*:

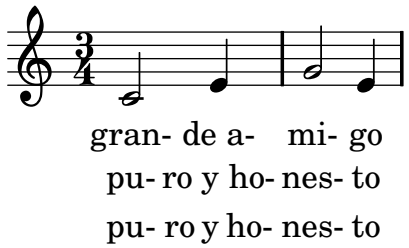
```
\override Score.LyricText #'font-shape = #'italic
```

sondern

```
\override Score . LyricText #'font-shape = #'italic
```

Um mehr als eine Silbe einer einzelnen Note zuzuweisen, kann man die Silben mit geraden Anführungszeichen umgeben (Shift+2) oder einen Unterstrich (`_`) benutzen, um Leerzeichen zwischen die Silben zu setzen, bzw. die Tilde (`~`) einsetzen, um einen Bindebogen zu erhalten.
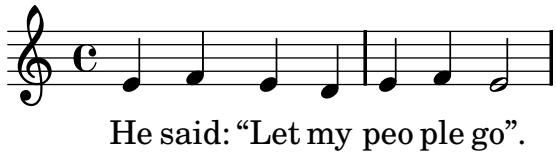
```
\time 3/4
\relative c' { c2 e4 g2 e4 }
\addlyrics { gran- de_a- mi- go }
\addlyrics { pu- "ro y ho-" nes- to }
\addlyrics { pu- ro~y~ho- nes- to }
```



Dieser Bindebogen ist definiert als das Unicode-Zeichen `U+203F`; es muss deshalb sichergestellt werden, dass eine Schriftart benutzt wird (wie etwa DejaVuLGC), die dieses Zeichen enthält. Mehr Information zur Schriftartauswahl findet sich in Abschnitt 1.8.3 [Schriftarten], Seite 183.
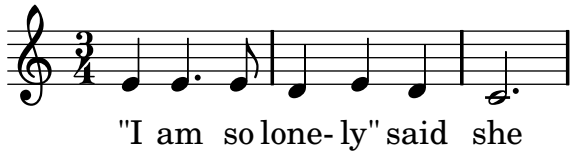
Um Gesangstext mit Akzenten, Umlauten, besonderen Zeichen oder anderen Alphabeten zu setzen, müssen diese Zeichen direkt in den Text geschrieben werden und die Datei als UTF-8 gespeichert werden. Für weitere Information siehe Abschnitt 3.3.3 [Zeichenkodierung], Seite 320.

```
\relative c' { e4 f e d e f e2 }
\addlyrics { He said: \Let my peo ple go". }
```



Um gerade Anführungszeichen im Gesangstext zu verwenden, müssen sie mit einem Backslash markiert werden, beispielsweise:

```
\relative c' { \time 3/4 e4 e4. e8 d4 e d c2. }
\addlyrics { "\"I" am so lone- "ly\"" said she }
```



Die vollständige Definition eines Wortanfangs im Gesangstextmodus ist jedoch etwas komplizierter.

Eine Silbe im Gesangstextmodus beginnt mit: einem alphabetischen Zeichen, `_`, `?`, `!`, `:`, `'`, den Kontrollzeichen `^A` bis `^F`, `^Q` bis `^W`, `^Y`, `^^`, einem beliebigen 8-Bit-Zeichen mit ASCII über 127, oder Zeichenkombinationen, in denen ein Backslas mit `` ` ``, `'`, `"` oder `^` kommbiniert wird.

Um Variablen zu definieren, in denen sich Gesangstext befindet, muss die `lyricmode`-Umgebung benutzt werden:

```
stropheEins = \lyricmode { Joy to the world the Lord is come }
\score {
  <<
    \new Voice = "eins" \relative c'' {
      \autoBeamOff
      \time 2/4
      c4 b8. a16 g4. f8 e4 d c2
    }
    \addlyrics { \stropheEins }
  >>
}
```

Siehe auch

Notationsreferenz: Abschnitt 1.8.3 [Schriftarten], Seite 183.

Referenz der Interna: Abschnitt “LyricText” in *Referenz der Interna*, Abschnitt “LyricSpace” in *Referenz der Interna*.
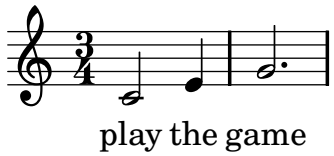
Einfache Lieder setzen

Am einfachsten kann Gesangstext zu Noten mit dem Befehl

`\addlyrics { Gesangstext }`

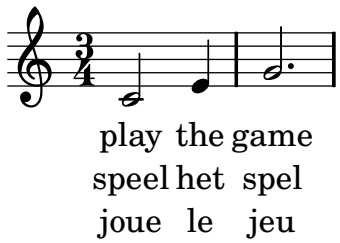
hinzugefügt werden. Hier ein Beispiel:

```
\time 3/4
\relative c' { c2 e4 g2. }
\addlyrics { play the game }
```



Weitere Strophen können hinzugefügt werden, indem weitere `\addlyrics`-Abschnitte erstellt werden:

```
\time 3/4
\relative c' { c2 e4 g2. }
\addlyrics { play the game }
\addlyrics { speel het spel }
\addlyrics { joue le jeu }
```



Der Befehl `\addlyrics` funktioniert nicht in polyphonen Situationen. In diesem Fall solte man `\lyricsto` in Kombination mit `\lyricmode` benutzen, wie erklärt in [Was ist Gesangtext], Seite 188.

Mit Gesangtexten und Bezeichnern arbeiten

Um Variablen zu definieren, die Gesangstext beinhalten, muss die `\lyricmode`-Umgebung benutzt werden. Man braucht hier keine Dauern einzugeben, wenn die Variable mit `\addlyrics` oder `\lyricsto` zu einer Melodie hinzugefügt wird.

```
stropheEins = \lyricmode { Joy to the world the Lord is come }
\score {
 <<
   \new Voice = "eins" \relative c'' {
     \autoBeamOff
     \time 2/4
     c4 b8. a16 g4. f8 e4 d c2
   }
   \addlyrics { \stropheEins }
 >>
}
```

Für eine andere Anordnung oder kompliziertere Sitationen bietet es sich an, zuerst Systeme und Gesangstextumgebungen zu definieren

```
\new ChoirStaff <<
  \new Voice = "soprano" { Noten }
  \new Lyrics = "sopranoLyrics" { s1 }
  \new Lyrics = "tenorLyrics" { s1 }
  \new Voice = "tenor" { Noten }
>>
```

und erst dann die entsprechenden Stimmen mit den dem Text zu kombinieren

```
\context Lyrics = sopranoLyrics \lyricsto "soprano"
Gesangstext
```

Siehe auch

Referenz der Interna: Abschnitt “LyricCombineMusic” in *Referenz der Interna*, Abschnitt “Lyrics” in *Referenz der Interna*.

2.1.3 Text an einer Melodie ausrichten

Gesangstext kann an einer Melodie automatisch ausgerichtet werden, aber wenn die Dauern der Silben angegeben werden, kann man sie auch manuell ausrichten. Die Ausrichtung kann angepasst werden mit leeren Noten (mit `\skip` oder `_`), Trennungsstrichen und Fülllinien.

Gesantstext wird gesetzt, wenn er sich in dem Kontext `Lyrics` befindet:

```
\new Lyrics \lyricmode ...
```

Es gibt zwei Methoden, mit denen man die horizontale Ausrichtung der Silben beeinflussen kann:

- Automatische Ausrichtung mit `\addlyrics` oder `\lyricsto`.
- Definition der Silbendauer innerhalb von `\lyricmode`.

Automatische Silbendauer

Die Silben des Gesangstextes können automatisch an einer Melodie ausgerichtet werden. Das erreicht man, indem der Gesangstext mit dem `\lyricsto`-Befehl einer Melodie zugewiesen wird:

```
\new Lyrics \lyricsto Bezeichnung ...
```

Hiermit werden die Silben an den Noten eines `Voice`-Kontexts mit der Bezeichnung *Bezeichnung* ausgerichtet. Dieser Kontext muss schon vorher definiert sein, damit er aufgerufen werden

kann. Mit dem Befehl `\lyricsto` wird in den `\lyricmode` gewechselt, so dass der Gesangstextmodus nicht mehr extra angegeben werden muss.

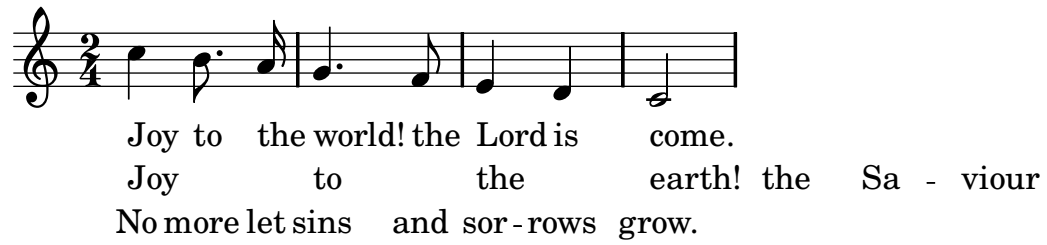
Das folgende Beispiel zeigt die Wirkung der unterschiedlichen Befehle, mit welchen Gesangtext mit einer Melodie kombiniert werden kann:

```
<<
  \new Voice = "eins" \relative c'' {
    \autoBeamOff
    \time 2/4
    c4 b8. a16 g4. f8 e4 d c2
  }

% nicht empfohlen: Silben links ausrichten
  \new Lyrics \lyricmode { Joy4 to8. the16 world!4. the8 Lord4 is come.2 }

% Falsch: brache Dauern
  \new Lyrics \lyricmode { Joy to the earth! the Sa -- viour reigns. }

%richtig
  \new Lyrics \lyricsto "eins" { No more let sins and sor -- rows grow. }
>>
```



Die zweite Strophe ist nicht richtig ausgerichtet, weil die Dauern der Silben nicht angegeben wurden. Anstelle dessen könnte besser `\lyricsto` eingesetzt werden.

Der `\addlyrics`-Befehl ist eigentlich nur eine Abkürzung für eine etwas kompliziertere LilyPond-Struktur:

```
{ Noten }
\addlyrics { Gesangstext }
```

bedeutet das Gleiche wie

```
\new Voice = "bla" { Noten }
\new Lyrics \lyricsto "bla" { Gesangstext }
```

Manuelle Silbendauer

Gesangstext kann auch ohne `\addlyrics` bezw. `\lyricsto` notiert werden. In diesem Fall werden die Silben wie Noten notiert – indem die Tonhöhen durch den Text der Silbe ersetzt werden – und die Dauer jeder Silbe muss angegeben werden. Beispielsweise so:

```
play2 the4 game2.
sink2 or4 swim2.
```
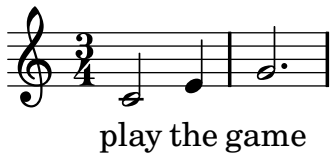
Die Ausrichtung an einer Melodie kann mit der `associatedVoice`-Eigenschaft bestimmt werden, etwa:

```
\set associatedVoice = #"lala"
```

Das Argument dieser Eigenschaft (hier `"lala"`) muss die Bezeichnung der entsprechenden Stimme sein. Ohne diese Einstellung werden Fülllinien nicht richtig formatiert.

Hier ein Beispiel, dass die manuale Ausrichtung von Gesangstext zeigt:

```
<< \new Voice = "Melodie" {
    \time 3/4
    c2 e4 g2.
 }
 \new Lyrics \lyricmode {
   \set associatedVoice = #"Melodie"
   play2 the4 game2.
 } >>
```



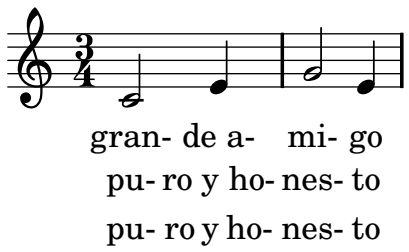
Siehe auch

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

Mehrere Silben zu einer Note

Um mehr als eine Silbe zu einer Note zuzuordnen, können die Silben mit geraden Anführungszeichen (") umgeben werden oder ein Unterstricht (_) benutzt werden, um ein Leerzeichen zwischen Silben zu setzen. Mit der Tilde (~) kann ein Bindebogen gesetzt werden. Dies erfordert, dass eine Schriftart vorhanden ist, die das entsprechende Symbol (U+203F) beinhaltet, wie etwa DejaVuLGC.

```
\time 3/4
\relative c' { c2 e4 g2 e4 }
\addlyrics { gran- de_a- mi- go }
\addlyrics { pu- "ro y ho-" nes- to }
\addlyrics { pu- ro~y~ho- nes- to }
```



Siehe auch

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

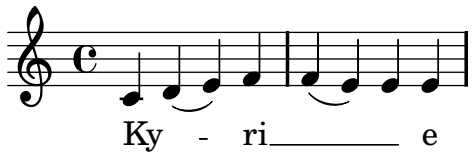
Mehrere Noten zu einer Silbe

Öfters wird eine einzige Silbe zu mehreren Noten gesungen, was als Melisma bezeichnet wird.

Melismen können direkt im Gesangstext definiert werden, indem ein Unterstrich (_) für jede Note notiert wird, die übersprungen werden soll.

Zusätzlich kann auch eine Fülllinie eingefügt werden, die das Melisma anzeigt. Sie wird notiert, indem ein doppelter Unterstrich direkt hinter die Silbe des Melismas gesetzt wird. Das Beispiel unten zeigt drei Elemente, die eingesetzt werden können: ein doppelter Bindestrich erstellt Trennungsstriche zwischen Silben, mit Unterstrichen wird eine Note übersprungen und mit einem doppelten Unterstrich wird eine Fülllinie gesetzt. Alle diese Zeichen müssen von Leerzeichen umgeben sein, damit sie erkannt werden.
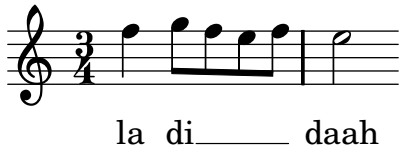
```
{ \set melismaBusyProperties = #'()
 c d( e) f f( e) e e  }
\addlyrics
 { Ky -- _ _ ri __ _ _ _  e }
```



Legatobögen können eingesetzt werden, wenn die Funktion `melismaBusyProperties` aufgerufen wird, wie in dem Beispiel oben.

Mit dem `\lyricsto`-Befehl können Melismen aber auch automatisch zugewiesen werden: unter übergebundene Noten oder Notengruppen mit einem Legatobogen wird nur eine einzige Silbe gesetzt. Wenn eine Notengruppe ohne Legatobogen als Melisma definiert werden soll, kann die Reichweite mit den Befehlen `\melisma` und `\melismaEnd` eingegrenzt werden:

```
<<
  \new Voice = "lala" {
    \time 3/4
    f4 g8
    \melisma
    f e f
    \melismaEnd
    e2
  }
  \new Lyrics \lyricsto "lala" {
    la di __ daah
  }
>>
```



Zusätzlich werden Noten als Melisma erachtet, wenn man sie manuell zu einer Balkengruppe verbindet und die automatische Bebalkung gleichzeitig ausgeschaltet ist. Siehe auch [Einstellung von automatischen Balken], Seite 57.

Ein vollständiges Beispiel für einen SATB-Chorsatz findet sich in Abschnitt “Vocal ensembles” in *Handbuch zum Lernen*.

Vordefinierte Befehle

`\melisma`, `\melismaEnd`

Siehe auch

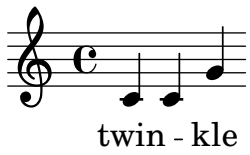
Bekannte Probleme und Warnungen

Melismen werde nicht automatisch erkannt, und Fülllinien müssen manuell gestzt werden.

Noten überspringen

Damit der Gesangstext langsamer als die Melodie fortschreitet, kann man `\skip`-Befehle einfügen. Jeder `\skip`-Befehl schiebt den Text eine Note weiter. Der Befehl muss von einer gültigen Dauer gefolgt werden, wie das Beispiel zeigt: dieser Dauerwert wird jedoch ignoriert, wenn man `\skip` im Gesangstext einsetzt.

```
\relative c' { c c g' }
\addlyrics {
  twin -- \skip 4
  kle
}
```



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.

Siehe auch

Referenz der Interna: Abschnitt “LyricExtender” in *Referenz der Interna*, Abschnitt “LyricHyphen” in *Referenz der Interna*

Gesangtext und Wiederholungen

TBC
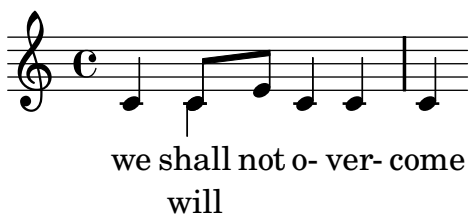
2.1.4 Besonderheiten der Gesangtextnotation

In vielen Fällen werden unterschiedliche Strophen mit einer Liedmelodie angeordnet, wobei kleine Schwankungen in der Silbenaufteilung auftreten können. Derartige Variationen können mit `\lyricsto` notiert werden.

Getrennte Texte

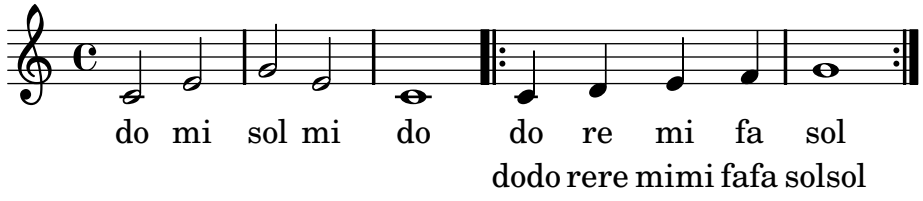
Alternative (oder *divisi* Gesangstexte können notiert werden, indem Stimmenkontexten Bezeichnungen zugewiesen werden und die Texte dann jeweils der entsprechenden Bezeichnung zugewiesen wird.
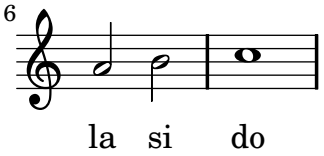
```
\score{ <<
  \new Voice = "Melodie" {
    \relative c' {
      c4
      <<
        { \voiceOne c8 e }
        \new Voice = "StimmeTeilen" { \voiceTwo c4 }
      >>
      \oneVoice c4 c | c
    }
  }
  \new Lyrics \lyricsto "Melodie" { we shall not o- ver- come }
  \new Lyrics \lyricsto "StimmeTeilen" { will }
>> }
```



Mit diesem Trick kann auch ein unterschiedlicher Text für eine wiederholte Stelle gesetzt werden:

```
\score{ <<
  \new Voice = "Melodie" \relative c' {
    c2 e | g e | c1 |
    \new Voice = "Strophe" \repeat volta 2 {c4 d e f | g1 | }
    a2 b | c1}
  \new Lyrics = "mainlyrics" \lyricsto Melodie \lyricmode {
    do mi sol mi do
    la si do }
  \context Lyrics = "mainlyrics" \lyricsto Strophe \lyricmode {
   do re mi fa sol }
  \new Lyrics = "repeatlyrics" \lyricsto Strophe \lyricmode {
   dodo rere mimi fafa solsol }
>>
}
```
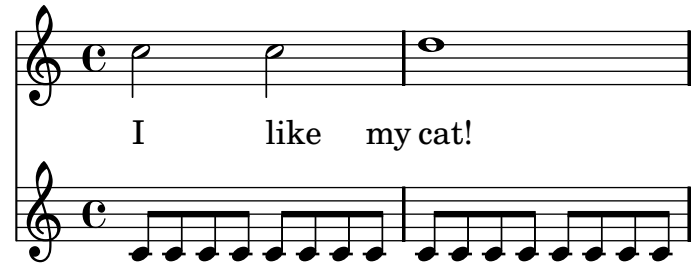




Text unabhängig von den Noten

In sehr komplexer Vokalmusik ist es manchmal erforderlich, den Gesangstext vollständig unabhängig von den Noten zu setzen. Das Beispiel unten zeigt das Vorgehen: die Noten, die für `lyricrhythm` definiert sind, verschwinden im `Devnull`-Kontext, während ihre Dauern immernoch gültig sind, um die Silben daran auszurichten.

```
Stimme = {
  c''2
  \tag #'music { c''2 }
  \tag #'lyricrhythm { c''4. c''8 }
  d''1
}

Text = \lyricmode { I like my cat! }

<<
  \new Staff \keepWithTag #'music \Stimme
  \new Devnull="nowhere" \keepWithTag #'lyricrhythm \Stimme
  \new Lyrics \lyricsto "nowhere" \Text
  \new Staff { c'8 c' c' c' c' c' c' c'
  c' c' c' c' c' c' c' c' }
>>
```

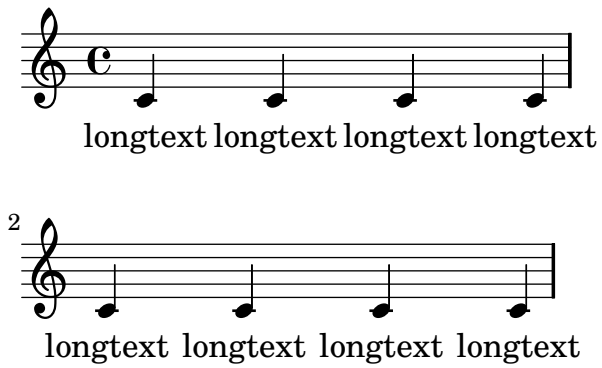


Diese Vorgehensweise ist nur empfehlenswert, wenn die Noten innerhalb des `Devnull`-Kontextes keine Melismen enthalten. Melismen werden im `Voice`-Kontext definiert. Wenn ein Gesangstext mit einem `Devnull`-Kontext verknüpft wird, wird die Verbindung von `Voice`- und `Lyrics`-Kontext aufgehoben und somit auch die Information zu Melismen. Darum werden implizite Melismen ignoriert.

Silben platzieren

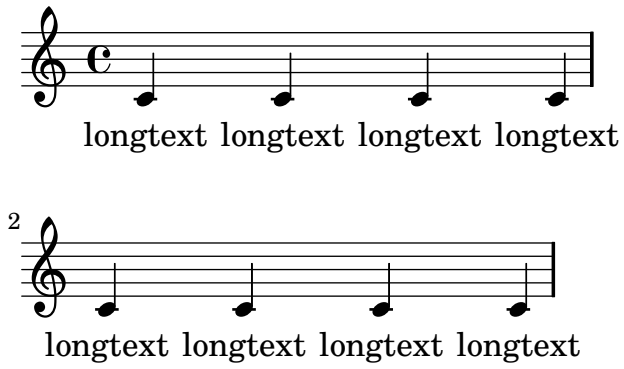
Um den Abstand zwischen Silben zu vergrößern, kann die `minimum-distance`-Eigenschaft des `LyricSpace`-Objekts gesetzt werden:

```
{
  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 c
  }
  \addlyrics {
  longtext longtext longtext longtext
  longtext longtext longtext longtext
  }
  \layout {
    \context {
      \Lyrics
      \override LyricSpace #'minimum-distance = #1.0
    }
  }
}
```



Ausgewählte Schnipsel

Eine Überprüfung, mit der sichergestellt wird, dass kein Text in die Seitenränder ragt, ist sehr rechenintensiv. Damit die Bearbeitungszeit von Dateien nicht so lange dauert, wird diese Überprüfung nicht automatisch vorgenommen. Man kann sie mit dem Befehl

```
\override Score.PaperColumn #'keep-inside-line = ##t
```

aktivieren. Damit Gesangstext auch nicht mit Taktlinien zusammenstößt, kann folgende Einstellung gesetzt werden:

```
\layout {
  \context {
    \Lyrics
      \consists "Bar_engraver"
```

```
      \consists "Separating_line_group_engraver"
      \override BarLine #'transparent = ##t
  }
}
```

Gesangtext zwischen Systemen zentrieren
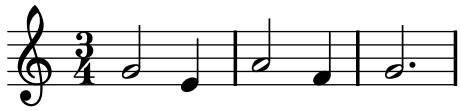
TBC

2.1.5 Strophen

Strophennummern hinzufügen

Strophennummerierung kann hinzugefügt werden:

```
\new Voice {
  \time 3/4 g2 e4 a2 f4 g2.
} \addlyrics {
  \set stanza = #"1. "
  Hi, my name is Bert.
} \addlyrics {
  \set stanza = #"2. "
  Oh, ché -- ri, je t'aime
}
```



1. Hi, my name is Bert.
2. Oh, ché - ri, je t'aime

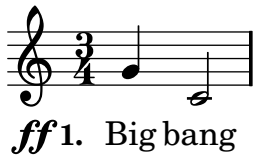
Die Zahl wird direkt vor die erste Silbe gesetzt.

Lautstärkebezeichnung zu Strophen hinzufügen

Dynamikzeichen können zur Strophennummer hinzugefügt werden. In LilyPond muss alles, was vor einer Strophe gesetzt wird, als Teil der `stanza`-Eigenschaft definiert werden, also auch Dynamikbezeichnung. Aus technischen Gründen muss die Strophe außerhalb von `lyricmode` gesetzt werden:

```
Text = {
  \set stanza = \markup { \dynamic "ff" "1. " }
  \lyricmode {
    Big bang
  }
}

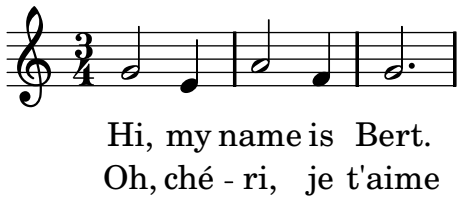
<<
  \new Voice = "Melodie" {
    \time 3/4
    g'4 c'2
  }
\new Lyrics \lyricsto "Melodie" \Text
>>
```



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 {
  \time 3/4 g2 e4 a2 f4 g2.
} \addlyrics {
  \set vocalName = #"Bert "
  Hi, my name is Bert.
} \addlyrics {
  \set vocalName = #"Ernie "
  Oh, ché -- ri, je t'aime
}
```

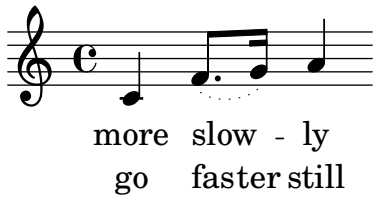


Strophen mit unterschiedlichem Rhythmus

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 c' \new Voice = "Lalala" {
    \set Staff.autoBeaming = ##f
    c4
    \slurDotted
    f8.[( g16])
    a4
  }
  \new Lyrics \lyricsto "Lalala" {
    more slow -- ly
  }
  \new Lyrics \lyricsto "Lalala" {
    go
    \set ignoreMelismata = ##t
    fas -- ter
    \unset ignoreMelismata
    still
  }
>>
```

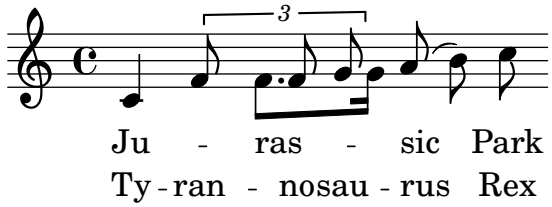


Bekannte Probleme und Warnungen

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.

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:



Der Text der ersten Strophe wird an der Stimme „lahlah“ ausgerichtet:

```
\new Lyrics \lyricsto "lahlah" {
  Ju -- ras -- sic Park
}
```

Auch die zweite Strophe wird an „lahlah“ ausgerichtet, aber für die Silbe „ran“ wird zu einer anderen Melodie gewechselt. Dazu wird der Befehl

```
\set associatedVoice = alternative
```

eingesetzt. `alternative` ist die Bezeichnung der Stimme, die die Triole enthält.

Dieser Befehl muss eine Silbe vor der Note notiert werden, auf die er sich auswirken soll, also vor „Ty“ in diesem Fall.

```
\new Lyrics \lyricsto "lahlah" {
  \set associatedVoice = alternative % applies to "ran"
  Ty --
  ran --
  no --
  \set associatedVoice = lahlah % applies to "rus"
  sau -- rus Rex
}
```

Zurück zu der alten Stimme kommt man, indem wieder „lahlah“ mit dem Text verknüpft wird.

Die Strophen am Ende ausdrucken

Manchmal soll nur eine Strophe mit der Melodie gesetzt werden und die weiteren Strophen als Text under 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:
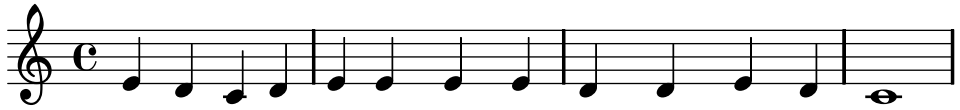
```
Melodie = \relative c' {
e d c d | e e e e |
d d e d | c1 |
}
```

```
Text = \lyricmode {
\set stanza = #"1." Ma- ry had a lit- tle lamb,
its fleece was white as snow.
}

\score{ <<
  \new Voice = "eins" { \Melodie }
  \new Lyrics \lyricsto "eins" \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."
}
```



1. Ma- ry had a lit- tle lamb, its fleece was white as snow.

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.

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.
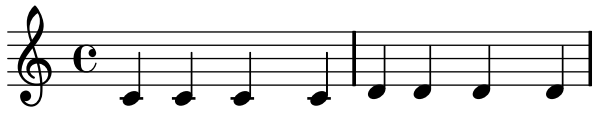
```
Melodie = \relative c' {
  c c c c | d d d d
}

Text = \lyricmode {
  \set stanza = #"1." This is verse one.
```

```
  It has two lines.
}

\score{ <<
    \new Voice = "eins" { \Melodie }
    \new Lyrics \lyricsto "eins" \Text
   >>
  \layout { }
}

\markup {
  \fill-line {
    \hspace #0.1 % Spalte vom linken Rand wegbewegen
        % Kann entfernt werden, wenn wenig Platz auf der Seite ist
     \column {
      \line { \bold "2."
        \column {
          "This is verse two."
          "It has two lines."
        }
      }
      \hspace #0.1 % Vertikaler Abstand zwischen Strophen
      \line { \bold "3."
        \column {
          "This is verse three."
          "It has two lines."
        }
      }
    }
    \hspace #0.1  % Horizontalen Abstand zwischen Spalten hinzufügen;
        % if they are still too close, add more " " pairs
        % bis das Resultat gut aussieht
     \column {
      \line { \bold "4."
        \column {
          "This is verse four."
          "It has two lines."
        }
      }
      \hspace #0.1 % Vertikaler Abstand zwischen Strophen
      \line { \bold "5."
        \column {
          "This is verse five."
          "It has two lines."
        }
      }
    }
  \hspace #0.1 % zusätzlichen Platz für den rechten Rand
      % kann entfernt werden, wenn wenig Platz auf der Seite ist
  }
}
```



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.

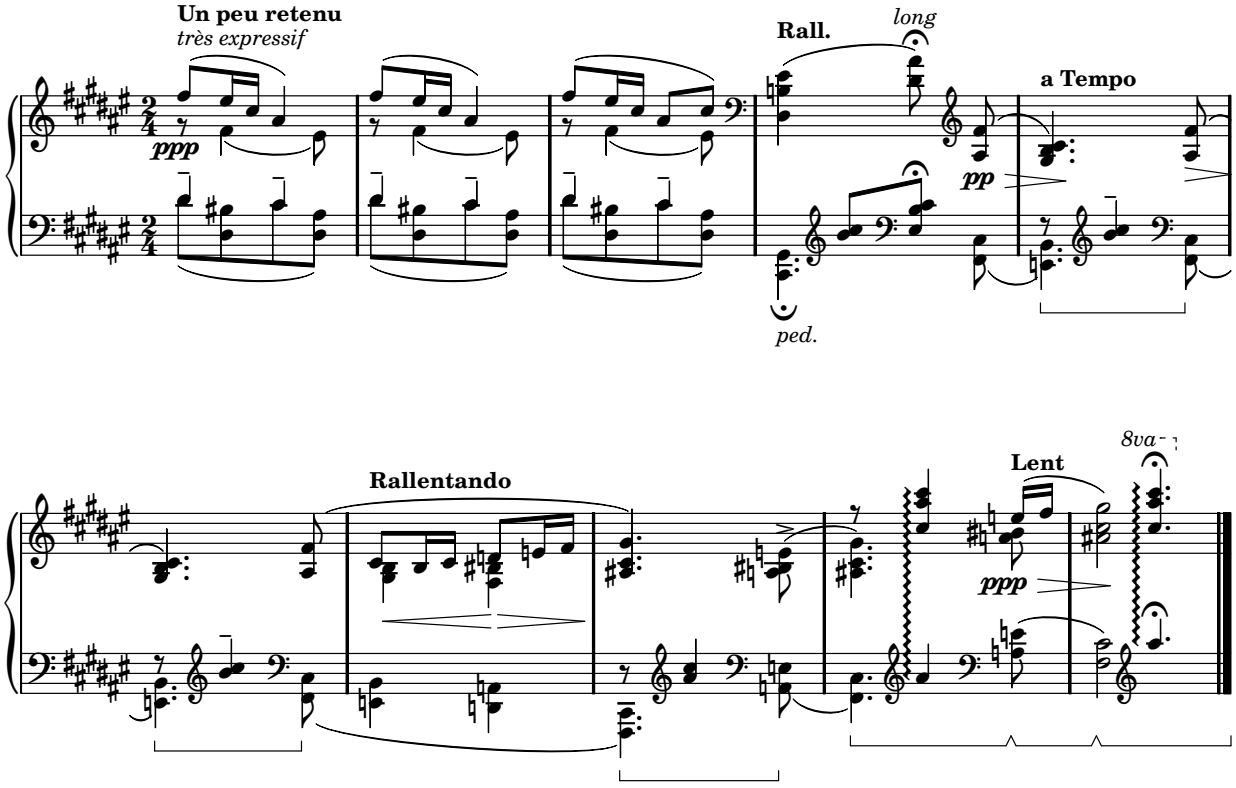
4. This is verse four.
It has two lines.

5. This is verse five.
It has two lines.

Siehe auch

Referenz der Interna: Abschnitt “LyricText” in *Referenz der Interna*, Abschnitt “StanzaNumber” in *Referenz der Interna*.

2.2 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.

2.2.1 Übliche Notation für Tasteninstrumente

Dieser Abschnitt zeigt allgemeine Eigenschaften des Notensatzes, die für die meisten Instrumente mit mehreren Systemen benötigt werden.

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 [Auflösung von Zusammenstößen], Seite 114.
- Noten für Tasteninstrumente kann auch parallel, Takt für Takt notiert werden, wie gezeigt in [Musik parallel notieren], Seite 121.
- Fingersatz wird erklärt in [Fingersatzanweisungen], Seite 153.
- Orgelpedal-Zeichen werden als Artikulationszeichen notiert, siehe Abschnitt B.10 [Liste der Arktikulationszeichen], Seite 404.
- Vertikale Rasterlinien können erstellt werden, siehe [Gitternetzlinien], Seite 160.
- Noten für Tasteninstrumente beinhalten oft *Laissez vibrer*-Bögen und Bindebögen mit Arpeggio oder Tremolo, siehe hierzu [Bindebögen], Seite 36.
- Arpeggios können auch zwischen den Systemen verbunden werden, siehe hierzu [Arpeggio], Seite 96.
- Tremolo-Zeichen finden sich in [Tremolo-Wiederholung], Seite 108.
- Viele der Optimierungen, die für Tastenmusik nötig sein können, sind demonstriert in Abschnitt “Real music example” in *Handbuch zum Lernen*.
- Unsichtbare Noten können eingesetzt werden, um Überbindungen zwischen Stimmen zu setzen, siehe Abschnitt “Other uses for tweaks” in *Handbuch zum Lernen*.

Siehe auch

Handbuch zum Lernen: Abschnitt “Real music example” in *Handbuch zum Lernen*, Abschnitt “Other uses for tweaks” in *Handbuch zum Lernen*.

Notationsreferenz: [Systeme gruppieren], Seite 125, [Instrumentenbezeichnungn], Seite 142, [Auflösung von Zusammenstößen], Seite 114, [Musik parallel notieren], Seite 121, [Fingersatzanweisungen], Seite 153, Abschnitt B.10 [Liste der Arktikulationszeichen], Seite 404, [Gitternetzlinien], Seite 160, [Bindebögen], Seite 36, [Arpeggio], Seite 96, [Tremolo-Wiederholung], Seite 108.

Schnipsel: Abschnitt “Keyboards” in *Schnipsel*.

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

Bekannte Probleme und Warnungen

Dynamikzeichen werden nicht automatische zwischen den Systemen zentriert, aber es gibt hierzu Lösungen. Eine Möglichkeit ist die Vorlage „Klavier mit zentrierten Lautstärkebezeichnungen“ im Abschnitt Abschnitt “Piano templates” in *Handbuch zum Lernen*; eine andere Möglichkeit ist es, die `staff-padding`-Eigenschaft von Lautstärkebezeichnungen zu erhöhen, wie gezeigt in Abschnitt “Moving objects” in *Handbuch zum Lernen*.

Notensysteme manuell verändern

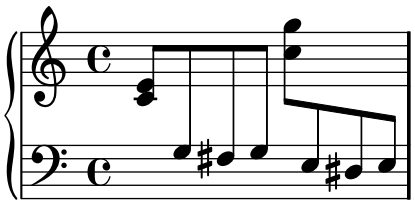
Stimmen können mit dem Befehl

`\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"`.

Balken zwischen den Systemen werden automatisch erstellt:

```
\new PianoStaff <<
  \new Staff = "oben" {
    <e' c'>8
    \change Staff = "unten"
    g8 fis g
    \change Staff = "oben"
    <g'' c''>8
    \change Staff = "unten"
    e8 dis e
    \change Staff = "oben"
  }
  \new Staff = "unten" {
    \clef bass
    % System aufrecht erhalten
    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 “Fixing overlapping notation” in *Handbuch zum Lernen*.

Siehe auch

Handbuch zum Lernen: Abschnitt “Fixing overlapping notation” in *Handbuch zum Lernen*.

Notationsreferenz: [Hälse], Seite 158, [Automatische Balken], Seite 55.

Schnipsel: Abschnitt “Keyboards” in *Schnipsel*.

Referenz der Interna: Abschnitt “Beam” in *Referenz der Interna*, Abschnitt “ContextChange” in *Referenz der Interna*.

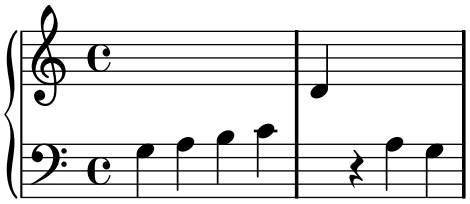
Automatischer Systemwechsel

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
  }
}
```



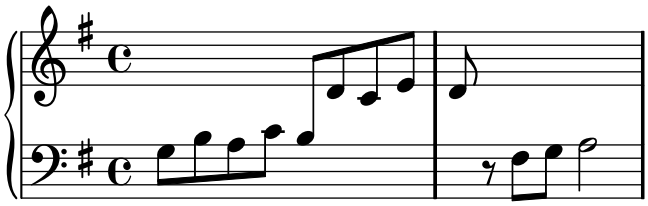
Ein `\relative`-Abschnitt, der sich außerhalb des `\autochange`-Abschnittes befindet, hat keinen Einfluss auf die Notenhöhen.

Wenn individuelle Kontrolle über die einzelen 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 c' {
        g8 b a c b d c e
        d8 r fis, g a2
      }
    }
  }
  \new Staff = "down" {
    \key g \major
    \clef bass
  }
>>
```



Siehe auch

Notationsreferenz: [Notensysteme manuell verändern], Seite 206.

Schnipsel: Abschnitt “Keyboards” in *Schnipsel*.

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

Bekannte Probleme und Warnungen

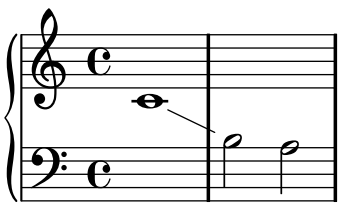
Die Auteilung auf die Systeme geschieht nicht unbedingt an optimaler Stelle. Für bessere Qualität müssen die Wechsel manuell eingestellt werden.

Akkrode werde nicht über die Systeme verteilt, sie werden dem System zugewiesen, auf dem sich ihre erste Note befinden würde.

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 = "eins" {
    \showStaffSwitch
    c1
    \change Staff = "zwei"
    b2 a
  }
  \new Staff = "zwei" {
    \clef bass
    s1*2
  }
>>
```



Vordefinierte Befehle

`\showStaffSwitch`, `\hideStaffSwitch`.

Siehe auch

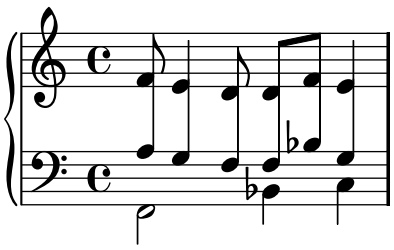
Schnipsel: Abschnitt “Keyboards” in *Schnipsel*.

Referenz der Interna: Abschnitt “Note_head_line_engraver” in *Referenz der Interna*, Abschnitt “VoiceFollower” in *Referenz der Interna*.

Hälse über beide Systeme

Akkorde, die über zwei Systeme reichen, können erstellt werden, indem die Länge der Hälse im unteren System vergrößert wird, bis sie zum oberen System hinauf reichen bzw. umgekehrt bei Hälsen, die nach unten zeigen.

```
\new PianoStaff <<
  \new Staff {
    \relative c' {
      f8 e4 d8 d f e4
    }
  }
  \new Staff {
    \relative c' {
      << {
        \clef bass
        % Hals kann das andere System überschneiden
        \override Stem #'cross-staff = ##t
        % Hälse verlängern um anderes System zu erreichen
        \override Stem #'length = #12
        % keine doppelten Fähnchen
        \override Stem #'flag-style = #'no-flag
        % Balken verhindern
        a8 g4 f8 f bes\noBeam g4
      }
      \\
      {
        f,2 bes4 c
      } >>
    }
  }
>>
```



Ausgewählte Schnipsel

Indicating cross-staff chords with arpeggio bracket

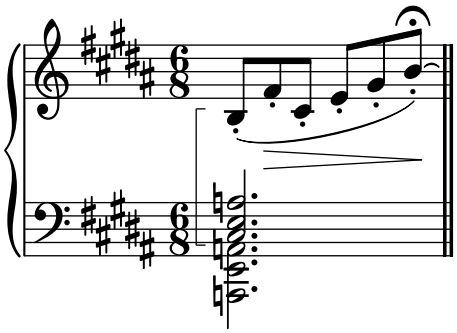
An arpeggio bracket can indicate that notes on two different staves are to be played with the same hand. In order to do this, the `PianoStaff` must be set to accept cross-staff arpeggios and the arpeggios must be set to the bracket shape in the `PianoStaff` context.

(Debussy, Les collines d'Anacapri, m. 65)

```
\paper { ragged-right = ##t }

\new PianoStaff <<
  \set PianoStaff.connectArpeggios = ##t
  \override PianoStaff.Arpeggio #'stencil = #ly:arpeggio::brew-chord-bracket
```

```
  \new Staff {
    \relative c' {
      \key b \major
      \time 6/8
      b8-.(\arpeggio fis'-.\> cis-. e-. gis-. b-.)\!\fermata^\laissezVibrer
      \bar "||"
    }
  }
  \new Staff {
    \relative c' {
      \clef bass
      \key b \major
      <<
        {
          <a e cis>2.\arpeggio
        }
        \\
        {
          <a, e a,>2.
        }
      >>
    }
  }
>>
```



Siehe auch

Schnipsel: Abschnitt “Keyboards” in *Schnipsel*.

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

2.2.2 Klavier
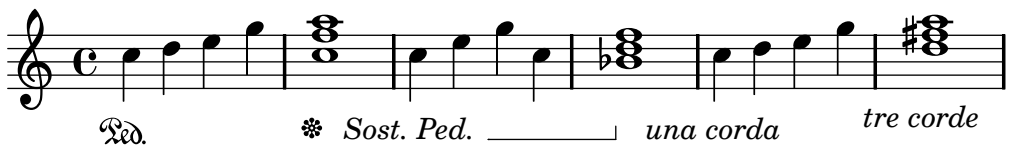
Dieser Abschnitt zeigt Eigenheiten der Notation von Klavermusik

Klavierpedal

Klaviere (teilweise auch Vibraphone und Celesta) besitzen üblicherweise drei Pedale, das linke oder Haltepedal, das rechte oder Una-corda-Pedal und das Sostenuto-Pedal. Die englischen Begriff hierzu lauten: *sustain*, *sostenuto* und *una corda*.
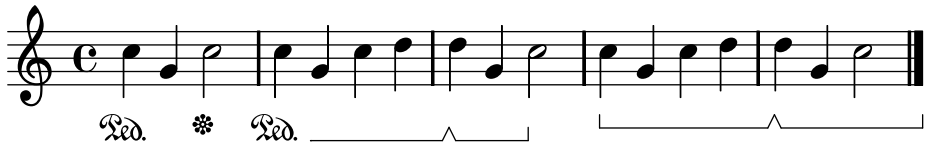
```
c4\sustainOn d e g
<c, f a>1\sustainOff
c4\sostenutoOn e g c,
<bes d f>1\sostenutoOff
```

```
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:

```
c4\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 Tatklinie zu halten, muss der letzte Pedal-hoch-Befehl weggelassen werden.

Siehe auch

Notationsreferenz: [Bindebögen], Seite 36.

Schnipsel: Abschnitt “Keyboards” 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*.

2.2.3 Akkordion

Dieser Abschnitt behandelt Notation, die nur für Akkordeonmusik benötigt wird.

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.

Ausgewählte Schnipsel

Symbole für Akkordeon-Diskantregister

Diskantregister für Akkordeon können mit `\markup` dargestellt werden. Die vertikale Position der einzelnen Elemente werden mit `\raise` angepasst.

```
Diskant = \markup {
  \musicglyph #"accordion.accDiscant"
}
punkt = \markup {
  \musicglyph #"accordion.accDot"
}

\layout { ragged-right = ##t }

% 16 voets Register
accBasson = ^\markup {
  \combine
  \Diskant
  \raise #0.5 \punkt
}

% eenkorig8en16voetsregister
accBandon = ^\markup {
  \combine
    \Diskant
    \combine
      \raise #0.5 \punkt
      \raise #1.5 \punkt
}

accVCello = ^\markup {
  \combine
    \Diskant
    \combine
      \raise #0.5 \punkt
      \combine
        \raise #1.5 \punkt
        \translate #'(1 . 0) \raise #1.5 \punkt
}

% 4-8-16voetsregister
accHarmon = ^\markup {
  \combine
    \Diskant
    \combine
      \raise #0.5 \punkt
      \combine
        \raise #1.5 \punkt
        \raise #2.5 \punkt
}

accPosaune = ^\markup {
```

```
  \combine
    \Diskant
    \combine
      \raise #0.5 \punkt
      \combine
        \raise #1.5 \punkt
        \combine
          \translate #'(1 . 0) \raise #1.5 \punkt
          \translate #'(-1 . 0) \raise #1.5 \punkt
}

% eenkorig4en16voetsregister
accOrgel = ^\markup {
  \combine
    \Diskant
    \combine
      \raise #0.5 \punkt
      \raise #2.5 \punkt
}

accHaupt = ^\markup {
  \combine
    \Diskant
    \combine
      \raise #0.5 \punkt
      \combine
        \raise #1.5 \punkt
        \combine
          \translate #'(1 . 0) \raise #1.5 \punkt
          \combine
            \translate #'(-1 . 0) \raise #1.5 \punkt
            \raise #2.5 \punkt
}

accAkkord = ^\markup {
  \combine
    \Diskant
    \combine
      \raise #1.5 \punkt
      \combine
        \translate #'(1 . 0) \raise #1.5 \punkt
        \combine
          \translate #'(-1 . 0) \raise #1.5 \punkt
          \raise #2.5 \punkt
}

accMusette = ^\markup {
  \combine
    \Diskant
    \combine
      \raise #1.5 \punkt
      \combine
```

```
        \translate #'(1 . 0) \raise #1.5 \punkt
        \translate #'(-1 . 0) \raise #1.5 \punkt
}

accCelesta = ^\markup {
  \combine
    \Diskant
    \combine
      \raise #1.5 \punkt
      \translate #'(-1 . 0) \raise #1.5 \punkt
}

accOboe = ^\markup {
  \combine
    \Diskant
    \combine
      \raise #1.5 \punkt
      \raise #2.5 \punkt
}

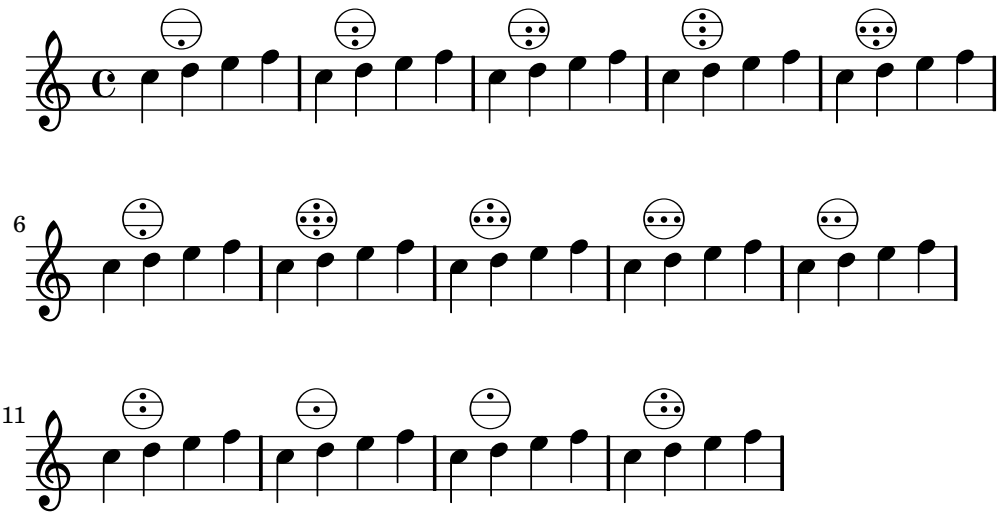
accClarin = ^\markup {
  \combine
    \Diskant
    \raise #1.5 \punkt
}

accPiccolo = ^\markup {
    \combine
       \Diskant
       \raise #2.5 \punkt
}

accGeige = ^\markup {
  \combine
    \Diskant
    \combine
      \raise #1.5 \punkt
      \combine
        \translate #'(1 . 0) \raise #1.5 \punkt
        \raise #2.5 \punkt
}

\relative c'' {
  c4 d\accBasson e f
  c4 d\accBandon e f
  c4 d\accVCello e f
  c4 d\accHarmon e f
  c4 d\accPosaune e f
  \break
  c4 d\accOrgel e f
  c4 d\accHaupt e f
  c4 d\accAkkord e f
```

```
  c4 d\accMusette e f
  c4 d\accCelesta e f
  \break
  c4 d\accOboe e f
  c4 d\accClarin e f
  c4 d\accPiccolo e f
  c4 d\accGeige e f
}
```



Siehe auch

Schnipsel: Abschnitt “Keyboards” in *Schnipsel*.

2.2.4 Harfe

Dieser Abschnitt zeigt Eigenheiten der Notation für Harfe.

Referenzen für Harfe

Einige übliche Notationseigenheiten für Harfe sind woanders behandelt:

- Glissando ist die üblichste Harfentechnik, siehe [Glissando], Seite 95.
- Ein *Bisbigliando* wird als ein Tremolo notiert, siehe [Tremolo-Wiederholung], Seite 108
- Flageolettöne werden hier beschrieben: [Flageolett], Seite 218.
- Für Arpeggio und non-arpeggio, siehe [Arpeggio], Seite 96.
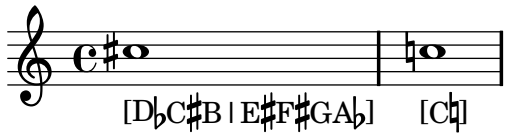
Siehe auch

Notationsreferenz: [Tremolo-Wiederholung], Seite 108 [Glissando], Seite 95 [Arpeggio], Seite 96 [Flageolett], Seite 218

Harfenpedal

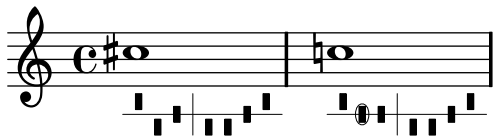
Harfe haben sieben Saiten in einer Oktave, die entweder als normaler Ton, oder aber erhöht bzw. erniedrigt klingen können. Bei einer Hakenharfe kann man jede Saite einzeln einstellen, bei Pedalharfen aber wird jede Saite mit der gleichen Notenbezeichnung von einem einzigen Pedal kontrolliert. Vom Spieler 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
cis1_\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
cis1_\markup { \harp-pedal #"^v-|vv-^" }
c!1_\markup { \harp-pedal #"^o--|vv-^" }
```

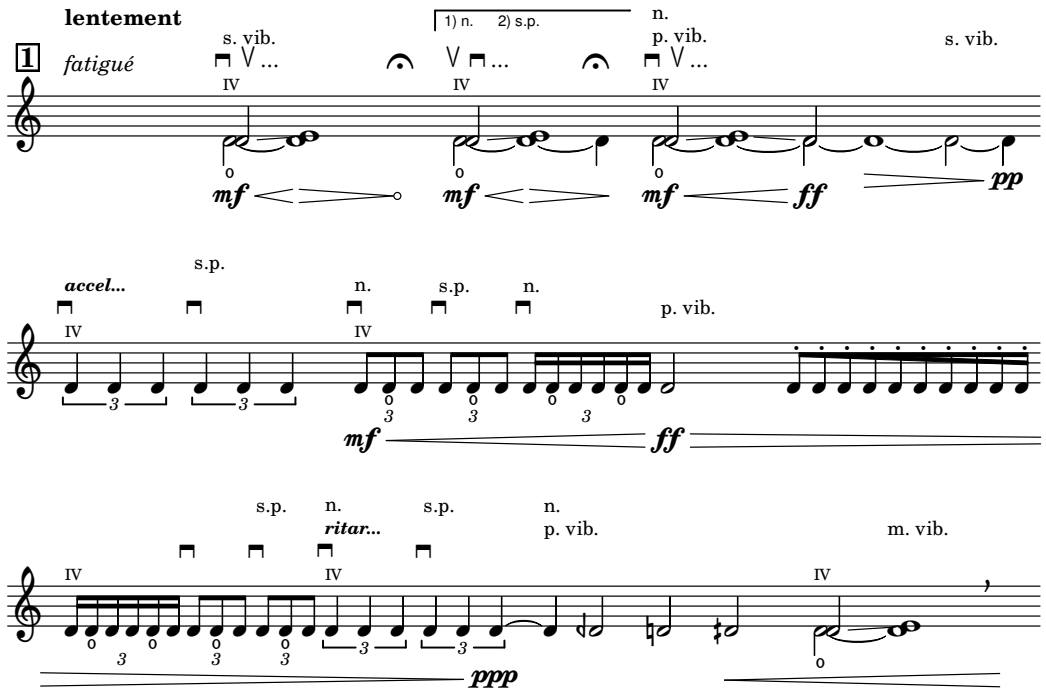


Der `\harp-pedal`-Befehl braucht eine Anzahl an Zeichen, von welchen `^` die höchste Pedalposition (erniedrigte Tonhöhe), `-` die mittlere Pedalposition (normale Tonhöhe, `v` die tiefste Pedalposition (erhöhter Ton) anzeigt. `|` ist ein Trenner. Ein `o` vor der Definition umrandet das Symbol.

Siehe auch

Notationsreferenz: [Textarten], Seite 163 Abschnitt B.8.5 [Instrument Specific Markup], Seite 396.

2.3 Bundlose Saiteninstrumente



Dieser Abschnitt stellt Information und Referenzen zur Verfügung, die beim Setzen von Noten für Saiteninstrumente ohne Bund herangezogen werden können.

2.3.1 Übliche Notation für nichtgebundene Saiteninstrumente

Es gibt wenige Spezifikat für die Notation von Saiteninstrumenten ohne Bund. 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.

Hinweise für ungebundene Saiteninstrumente

Die meisten Notationseigenschaften, die für Orchestersterstreicher eingesetzt werden, sind an anderer Stelle beschrieben:

- Textanweisungen wie „pizz.“ oder „arco“ werden als einfacher Text eingefügt, siehe [Textarten], Seite 163.
- Fingersatz, auch das Zeichen für den Daumen, ist erklärt in [Fingersatzanweisungen], Seite 153.
- Doppelgriffe werden normalerweise als Akkord notiert, siehe hierzu [Noten mit Akkorden], Seite 109. Anweisungen, wie Akkorde gespielt werden sollen, können auch hinzugefügt werden, siehe [Arpeggio], Seite 96.
- Eine Vorlage für Streichquartett findet sich in Abschnitt “String quartet” in *Handbuch zum Lernen*. Andere sind als Schnipsel zur Verfügung gestellt.

Siehe auch

Handbuch zum Lernen: Abschnitt “String quartet” in *Handbuch zum Lernen*.

Notationsreferenz: [Textarten], Seite 163, [Fingersatzanweisungen], Seite 153, [Noten mit Akkorden], Seite 109, [Arpeggio], Seite 96.
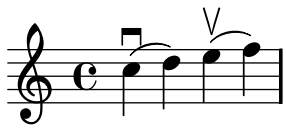
Schnipsel: Abschnitt “Unfretted strings” in *Schnipsel*.

Bezeichnung des Bogens

Hinweise zur Bogenfügung können als Artikulationen erstellt werden, wie beschrieben in [Artikulationszeichen und Verzierungen], Seite 83.
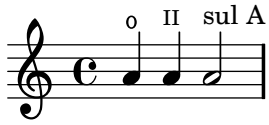
Die Befehle `\upbow` und `\downbow` werden mit Legatobögen in folgender Weise eingesetzt:

```
c4(\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: [Artikulationszeichen und Verzierungen], Seite 83, [Legatobögen], Seite 90.
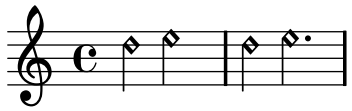
Flageolett

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.

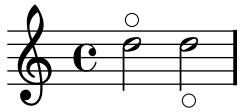
> **Achtung:** Flageolett-Töne **müssem** innerhalb von Akkorden definiert werden, auch wenn nur eine einzelne Note vorhanden ist.

```
<d\harmonic>4 <e\harmonic>2.
\set harmonicDots = ##t
<d\harmonic>4 <e\harmonic>2.
```



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:
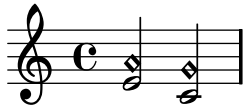
```
d2^\flageolet d_\flageolet
```



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.

```
<e a\harmonic>2  <c g'\harmonic>
```



Siehe auch

Glossar: Abschnitt “Flageolett” in *Glossar*.

Notationsreferenz: [Besondere Notenköpfe], Seite 27, [Hinweise für ungebundene Saiteninstrumente], Seite 217.

Bartók-Pizzicato

Ausgewählte Schnipsel

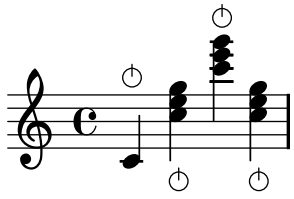
Bartók-Pizzicato

Das Bartók-Pizzicato ‚ist eine besondere Form des Pizzicato, bei dem der Spieler die Saite auf das Griffbrett aufschlagen lässt, sodass zusätzlich zum angeschlagenen Ton ein scharfes, knallendes Geräusch ertönt' (Wikipedia). Es wird dargestellt als kleiner Kreis mit einer vertikalen Linie, die vom Kreiszentrum aus nach oben weist und ein Stück außerhalb des Kreises endet. Lilypond hat keinen eigenen Glyphen für dieses Symbol; es ist aber einfach, direkt eine Definition in die Eingabedatei einzufügen.

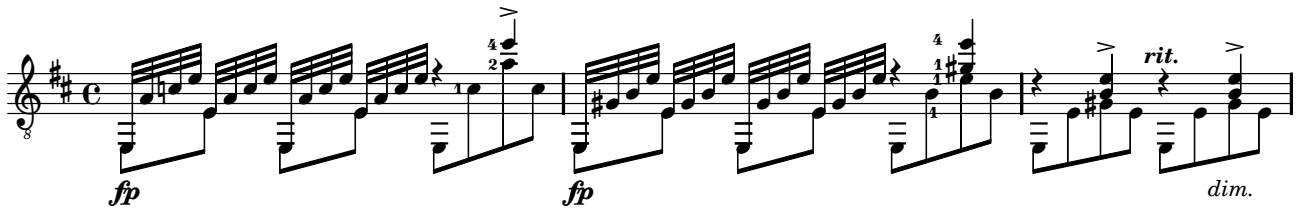
```
#(define-markup-command (snappizz layout props) ()
  (interpret-markup layout props
    (markup #:stencil
      (ly:stencil-translate-axis
        (ly:stencil-add
          (make-circle-stencil 0.7 0.1 #f)
          (ly:make-stencil
            (list 'draw-line 0.1 0 0.1 0 1)
            '(-0.1 . 0.1) '(0.1 . 1)))
        0.7 X))))
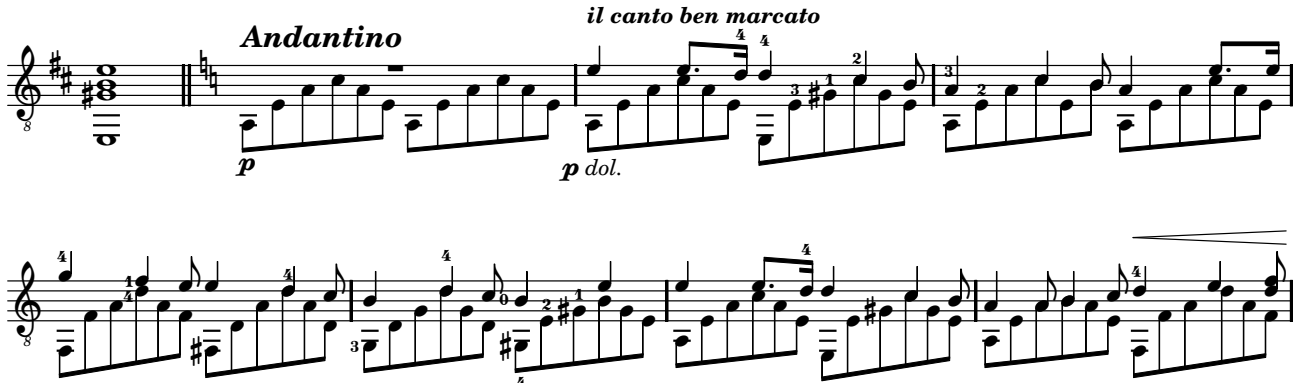
schlagPizzicato = \markup \snappizz

% jetzt kann es als \schlagPizzicato nach einer Note/Akkord benutzt werden
% Eine Richtung (-, ^ oder _) wird benötigt
\relative c' {
  c4^\schlagPizzicato
  % Das funktioniert NICHT:
  %<ceg>\schlagPizzicato
  <c' e g>-\schlagPizzicato
  <c' e g>^\schlagPizzicato
  <c, e g>_\schlagPizzicato
}
```



2.4 Saiteninstrumente mit Bünden





Dieser Abschnitt erklärt bestimmte Eigenheiten der Notation für gebundene Saiteninstrumente.

2.4.1 Übliche Notation für gebundene Saiteninstrumente

Dieser Abschnitt zeigt Besonderheiten der Notation, die allen gebundenen Saiteninstrumenten eigen ist.

Referenz für gebundene Saiteninstrumente

Noten für gebundene Saiteninstrumente 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"`) benutzen. Einige Spezifika für gebundene Instrumente sind an anderer Stelle erklärt:

- Fingersatz kann notiert werden, siehe [Fingersatzanweisungen], Seite 153.
- Anweisungen für *Laissez vibrer*-Bögen und Bögen zwischen Arpeggios und Tremolos sind beschrieben in [Bindebögen], Seite 36.
- Hinweise, wie mehrere Stimmen gesetzt werden können, finden sich in [Auflösung von Zusammenstößen], Seite 114.

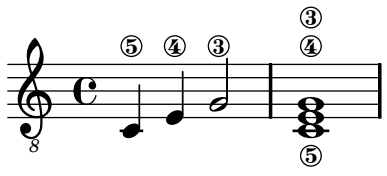
Siehe auch

Notationsreferenz: [Fingersatzanweisungen], Seite 153, [Bindebögen], Seite 36, [Auflösung von Zusammenstößen], Seite 114, [Instrumentenbezeichnungn], Seite 142, [Musik parallel notieren], Seite 121, [Arpeggio], Seite 96, Abschnitt B.10 [Liste der Arktikulationszeichen], Seite 404, [Notenschlüssel], Seite 12.

Seitennummerbezeichnung

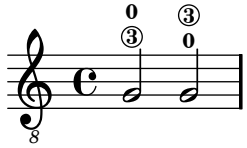
Die Nummer der Saite, auf der gespielt werden soll, kann angezeigt werden, indem `\Zahl` an eine Note innerhalb eines Akkord-Konstrukts gesetzt wird:

```
\clef "treble_8"
<c\5>4 <e\4> <g\3>2
<c,\5 e\4 g\3>1
```



Wenn Fingersatz und Saitennummer zusammen benutzt werden, wird ihre Position anhand der Reihenfolge entschieden, mit der sie im Code auftauchen:

```
\clef "treble_8"
<g\3-0>2
<g-0\3>
```

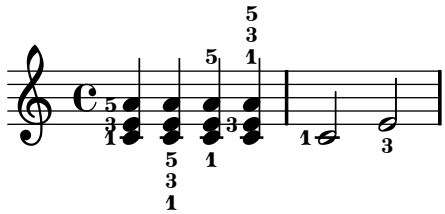


Ausgewählte Schnipsel

Controlling the placement of chord fingerings

The placement of fingering numbers can be controlled precisely.

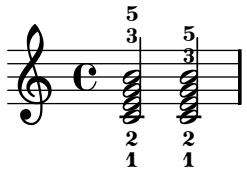
```
\relative c' {
  \set fingeringOrientations = #'(left)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(down)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(down right up)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(up)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(left)
  <c-1>2
  \set fingeringOrientations = #'(down)
  <e-3>2
}
```



Allowing fingerings to be printed inside the staff

By default, vertically oriented fingerings are positioned outside the staff. However, this behavior can be canceled.

```
\relative c' {
  <c-1 e-2 g-3 b-5>2
  \once \override Fingering #'staff-padding = #'()
  <c-1 e-2 g-3 b-5>2
}
```



Siehe auch

Notationsreferenz: [Fingersatzanweisungen], Seite 153.

Schnipsel: Abschnitt “Fretted strings” in *Schnipsel*.

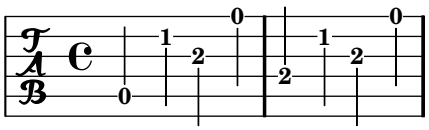
Referenz der Interna: Abschnitt “StringNumber” in *Referenz der Interna*, Abschnitt “Fingering” in *Referenz der Interna*.

Standardtabulaturen

Tabulatur-Notation wird für die Notation von Zupfinstrumenten benutzt. Tonhöhen werden hier nicht durch Notenköpfe, sondern durch Zahlen notiert. Diese Zahlen zeigen an, auf welcher Saite und welchem Bund der Ton gespielt werden soll. LilyPond bringt beschränkte Unterstützung für Tabulaturen mit.

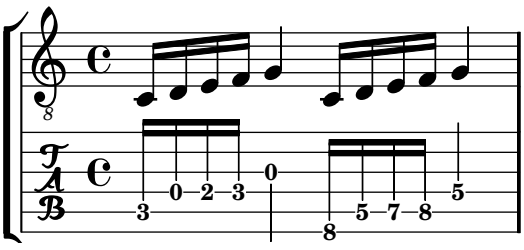
Die Saitennummer, die mit einer Note assoziiert ist, wird durch einen Backslash, gefolgt von einer Zahl, notiert. In der Standardeinstellung ist die erste Saite die höchste Saite und als Stimmung wird die übliche Gitarrenstimmung auf sechs Saiten angenommen. Die Noten werden in einer Tabulatur gesetzt, indem Abschnitt “TabStaff” in *Referenz der Interna* und Abschnitt “TabVoice” in *Referenz der Interna*-Kontexte verwendet werden.

```
\new TabStaff {
  a,4\5 c'\2 a\3 e'\1
  e\4 c'\2 a\3 e'\1
}
```



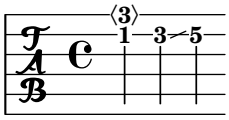
Wenn keine Saite für eine Note angegeben wird, wird die Note der Saite zugeordnet, welche die Note auf einem Bund erzeugen kann, der größer oder gleich als der Wert von `minimumFret` ist. Der Standardwert für `minimumFret` beträgt 0.

```
\new StaffGroup <<
   \new Staff \relative c {
     \clef "treble_8"
     c16 d e f g4
     c,16 d e f g4
   }
   \new TabStaff \relative c {
     c16 d e f g4
     \set TabStaff.minimumFret = #5
     c,16 d e f g4
   }
>>
```



Flageolett und Gleiten (Slide) kann zur Tabulatur hinzugefügt werden:

```
\new TabStaff {
  \new TabVoice {
    <c g'\harmonic> d\2\glissando e\2
  }
}
```

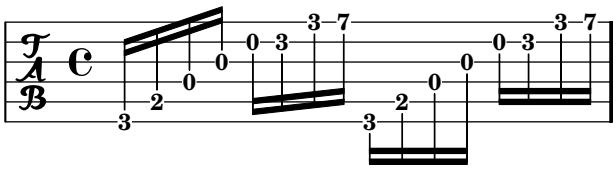


Ausgewählte Schnipsel

Stem and beam behavior in tablature

The direction of stems is controlled the same way in tablature as in traditional notation. Beams can be made horizontal, as shown in this example.

```
\new TabStaff {
  \relative c {
    g16 b d g b d g b
    \stemDown
    \override Beam #'damping = #+inf.0
    g,,16 b d g b d g b
  }
}
```



Polyphony in tablature

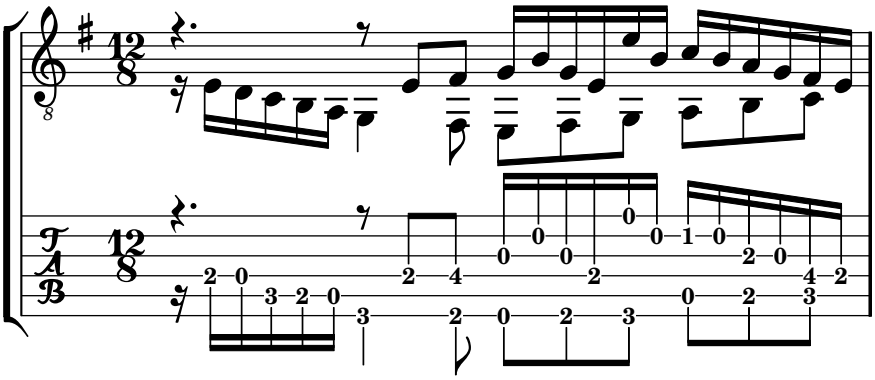
Polyphony is created the same way in a `TabStaff` as in a regular staff.

```
oben = \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
}

unten = \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 = "Gitarre traditional" <<
        \clef "treble_8"
        \context Voice = "oben" \oben
```

```
        \context Voice = "unten" \unten
      >>
      \new TabStaff = "Gitarre tab" <<
        \context TabVoice = "oben" \oben
        \context TabVoice = "unten" \unten
      >>
    >>
  >>
}
```



Siehe auch

Notationsreferenz: [Hälse], Seite 158.

Schnipsel: Abschnitt “Fretted strings” in *Schnipsel*.

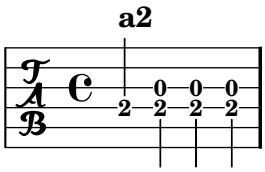
Referenz der Interna: Abschnitt “TabNoteHead” in *Referenz der Interna*, Abschnitt “TabStaff” in *Referenz der Interna*, Abschnitt “TabVoice” in *Referenz der Interna*, Abschnitt “Beam” in *Referenz der Interna*.

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:

```
Melodie = \partcombine { e4 g g g }{ e4 e e e }
<<
  \new TabStaff <<
    \new TabVoice = "eins" s1
    \new TabVoice = "zwei" s1
    \new TabVoice = "gemeinsam" s1
    \new TabVoice = "solo" s1
    { \Melodie }
  >>
>>
```



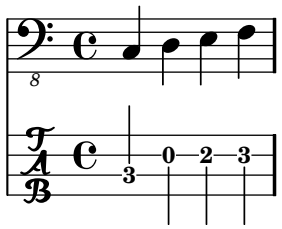
Spezialeffekte für Gitarre beschränken sich auf Flageolett und Slide.

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 und Bassgitarre. 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 Staff {
    \clef "bass_8"
    \relative c, {
      c4 d e f
    }
  }
  \new TabStaff {
    \set TabStaff.stringTunings = #bass-tuning
    \relative c, {
      c4 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 `scm/output-lib.scm`.

Die Stimmung ist 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.

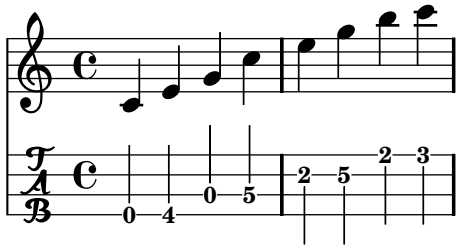
Eine Tonhöhe in der Liste der Saitenstimmungen ist der Unterschied der entsprechenden Tonhöhe zum eingestrichenen C gemessen in Halbtönen. Die Tonöhe muss eine Ganzzahl sein. LilyPond errechnet die Tonhöhe einer Saite, indem die Tonhöhe der Saitenstimmung zu der Tonhöhe von c' hinzugerechnet wird.

LilyPond erschließt die Anzahl der Saiten einer Tabulatur anhand der Anzahl der Saitenstimmungszahlen in `stringTunings`.

Jede beliebige Saitenstimmung kann erzeugt werden. Als Beispiel etwa kann die Saitenstimmung für ein viersaitiges Instrument mit den Tonhöhen `a''`, `d''`, `g'`und `c'` so definiert werden:

```
meineNoten = {
    c'4 e' g' c'' |
    e'' g'' b'' c'''
}
```

```
<<
  \new Staff {
    \clef treble
    \meineNoten
  }
  \new TabStaff {
    \set TabStaff.stringTunings = #'(21 14 7 0)
    \meineNoten
  }
>>
```



Siehe auch

Installierte Dateien: ‘`scm/output-lib.scm`’.

Schnipsel: Abschnitt “Fretted strings” in *Schnipsel*.

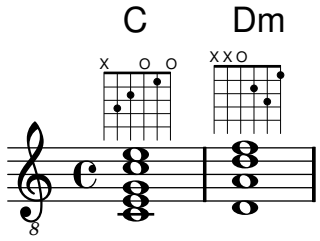
Referenz der Interna: Abschnitt “Tab_note_heads_engraver” in *Referenz der Interna*.

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 zu Textbeschriftunsbefehlen findet sich in Abschnitt B.8 [Text markup commands], Seite 367.

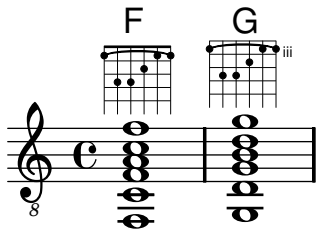
Die Standard-Bunddiagrammbeschriftung beinhaltet die Saitennummer und die Bundnummer für jeden Punkt, der notiert werden soll. Zusätzlich können offenen und nicht gespielte (schwingende) Saiten angezeigt werden.

```
<<
  \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'> ^\markup
      \fret-diagram #"6-x;5-x;4-o;3-2;2-3;1-1;"
  }
>>
```



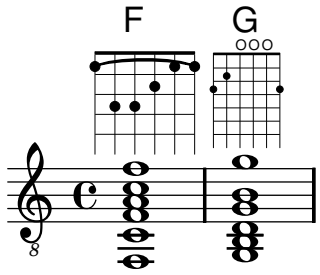
Barre kann hinzugefügt werden:

```
<<
  \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'> ^\markup
      \fret-diagram #"c:6-1-3;6-3;5-5;4-5;3-4;2-3;1-3;"
  }
>>
```



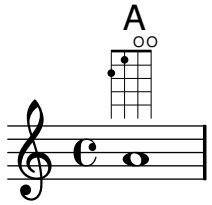
Die Größe des Bunddiagrammes und die Anzahl der Bünde im Diagramm kann geändert werden:

```
<<
  \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'> ^\markup
      \fret-diagram #"h:6;6-3;5-2;4-o;3-o;2-o;1-3;"
  }
>>
```



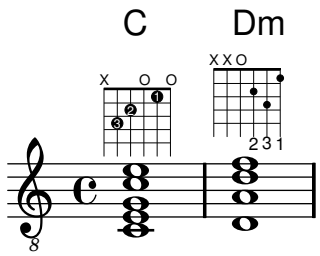
Die Anzahl der Saiten kann geändert werden, um sie für andere Instrumente anzupassen, wie etwas Banjo oder Ukulele.

```
<<
  \context ChordNames {
     \chordmode {
       a1
     }
  }
  \context Staff {
        %% Akkord für Ukulele
    a'1 ^\markup \fret-diagram #"w:4;4-2-2;3-1-1;2-o;1-o;"
  }
>>
```



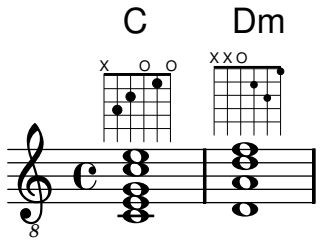
Fingersatz kann auch angezeigt werden, und die Position der Fingersatzzahlen kann kontrolliert werden.

```
<<
  \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'> ^\markup
      \fret-diagram #"f:2;6-x;5-x;4-o;3-2-2;2-3-3;1-1-1;"
  }
>>
```



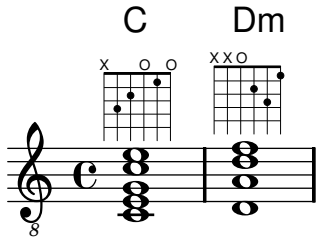
Die Größe und Position der Punkte kann geändert werden:

```
<<
  \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'> ^\markup
      \fret-diagram #"p:0.2;6-x;5-x;4-o;3-2;2-3;1-1;"
  }
>>
```



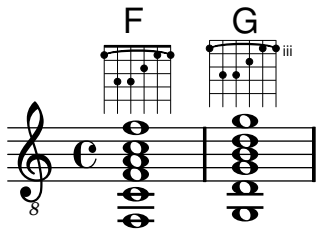
Die Beschriftungsfunktion `fret-diagram-terse` (knappe Version) lässt die Saitennummern aus: das Vorhandensein einer Saite wird durch ein Semikolon ausgedrückt. Für jede Saite des Diagramms muss ein Semikolon gesetzt werden. Das erste Semikolon entspricht der höchsten Saite, das letze der ersten Saite. 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'> ^\markup
      \fret-diagram-terse #"x;x;o;2;3;1;"
  }
>>
```



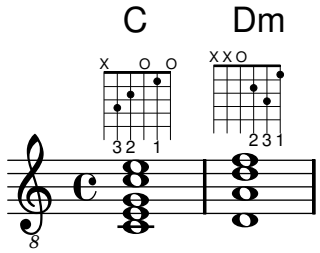
Barre kann im knappen Modus auch angezeigt werden:

```
<<
  \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'> ^\markup
      \fret-diagram-terse #"3-(;5;5;4;3;3-);"
  }
>>
```



Fingersatz kann im knappen Modus hinzugefügt werden:

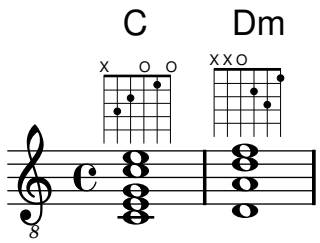
```
<<
  \context ChordNames {
     \chordmode {
       c1 d:m
     }
  }
  \context Staff {
    \override Voice.TextScript
      #'(fret-diagram-details finger-code) = #'below-string
    \clef "treble_8"
    < c e g c' e' > 1 ^\markup
      \fret-diagram-terse #"x;3-3;2-2;o;1-1;o;"
    < d a d' f'> ^\markup
      \fret-diagram-terse #"x;x;o;2-2;3-3;1-1;"
  }
>>
```



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'> ^\markup
      \fret-diagram-verbose #'(
        (mute 6)
        (mute 5)
        (open 4)
        (place-fret 3 2)
        (place-fret 2 3)
        (place-fret 1 1)
      )
  }
>>
```

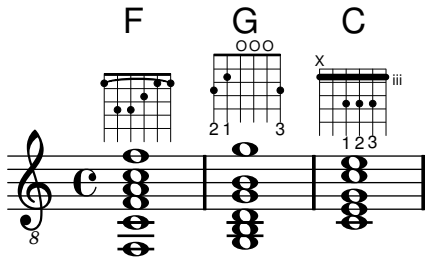


Fingersatz und Barre kann im ausführlichen Modus notiert werden. Nur im ausführlichen Modus kann ein Capo angezeigt werden, das auf dem Bunddiagramm plaziert wird. Die Capo-Anzeige ist ein dicker Strich, der alle Saiten bedeckt. Der Bund mit dem Capo ist der unterste Bund im Diagramm.

```
<<
  \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'> ^\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'> ^\markup
      \fret-diagram-verbose #'(
        (capo 3)
        (mute 6)
        (place-fret 4 5 1)
        (place-fret 3 5 2)
        (place-fret 2 5 3)
      )
  }
>>
```



Alle anderen Bunddiagramm-Eigenschaften müssen im ausführlichen Modus mit mit `\override`-Befehlen angegeben werden.

Die graphische Erscheinung eines Bunddiagramms kann den Wünschen des Notensetzers angepassen 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

*Customizing markup fret diagrams*Fret diagram properties can be set through `'fret-diagram-details`. For markup fret diagrams, overrides can be applied to the `Voice.TextScript` object or directly to the markup.

```
<<
  \chords { c1 c c d }
```

```
\new Voice = "Melodie" {
  \textLengthOn
  % Allgemeine Eigenschaften von Bund-Diagramme bestimmen
  \override TextScript #'size = #'1.2
  \override TextScript
    #'(fret-diagram-details finger-code) = #'in-dot
  \override TextScript
    #'(fret-diagram-details dot-color) = #'white

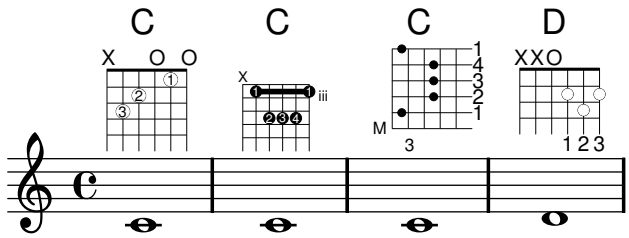
  %% C-Dur für Gitarre, kein Barre, Standardeinstellungen
     % knapper Stil
  c'1^\markup { \fret-diagram-terse #"x;3-3;2-2;o;1-1;o;" }

  %% C-Dur für Gitarre, Barre auf dem dritten Bund
     % ausführlicher Stil
     % Größe 1.0
     % römische Bundnummer, Fingersatz unter der Saite, grades Barre
  c'1^\markup {
    % Standardgröße
    \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-Dur für Gitarre, Barre auf dem dritten Bund
     % ausführlicher Stil
     % quer liegend, arabische Nummern, M (mute) für stumme Saite
     % kein Barre, Bundbezeichnung unten oder links, kleine Schrift für stumme Saite
  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))
      }
    }

    %% einfacher D-Akkord
       % knapper Stil
       % größere Punkte, zentrierte Punkte, weniger Bünde
       % Bezeichnung darunter
    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 B.8 [Text markup commands], Seite 367.

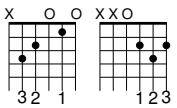
Schnipsel: Abschnitt “Fretted strings” in *Schnipsel*.

Referenz der Interna: Abschnitt “fret-diagram-interface” in *Referenz der Interna*.

Vordefinierte Bund-Diagramme

Bunddiagramme können mit dem `FretBoards`-Kontext angezeigt werden. Standardmäßig zeigt der `FretBoards`-Kontext Bunddiagramme an, die in einer Tabelle definiert sind:

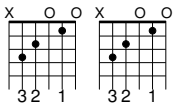
```
\include "predefined-guitar-fretboards.ly"
\context FretBoards {
  \chordmode {
    c1 d
  }
}
```



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 Gitarrenstimmtung (`guitar-tuning`). Anhand der Beispiele in dieser Datei können auch für andere Instrumente oder Stimmungen Diagramme definiert werden.

Tonhöhen von Akkorden können entweder als Akkordkonstrukte oder im Akkordmodus notiert werden (siehe auch [Überblick über den Akkord-Modus], Seite 260).

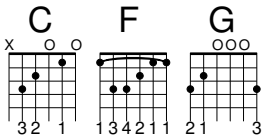
```
\include "predefined-guitar-fretboards.ly"
\context FretBoards {
  \chordmode {c1}
  <c' e' g'>1
}
```



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"
meineAkkorde = \chordmode{
  c1 f g
}

<<
  \context ChordNames {
    \meineAkkorde
  }
  \context FretBoards {
    \meineAkkorde
  }
>>
```

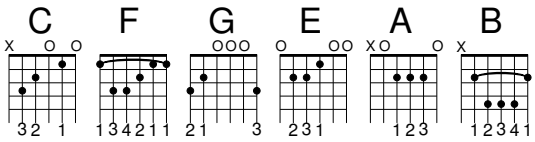


Vordefinierte Bunddiagramme können transponiert werden, solange ein Diagramm für den transponierten Akkord in der Bunddiagramm-Tabelle vorhanden ist.

```
\include "predefined-guitar-fretboards.ly"
meineAkkorde = \chordmode{
  c1 f g
}

meineAkkordliste = {
  \meineAkkorde
  \transpose c e { \meineAkkorde}
}
<<
```

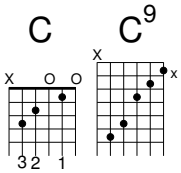
```
  \context ChordNames {
    \meineAkkordliste
  }
  \context FretBoards {
    \meineAkkordliste
  }
>>
```



Die Tabelle der vordefinierten Bunddiagramme enthält sieben Akkorde (Dur, Moll, übermäßig, vermindert, Dominantseptakkord, große Septime und kleine Septime) für alle 17 Tonarten. Eine vollständige Liste der vordefinierten Bunddiagramme findet sich in Abschnitt B.3 [Vordefinierte Bund-Diagramme], Seite 347. 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 [Automatische Bund-Diagramme], Seite 242.

```
\include "predefined-guitar-fretboards.ly"
meineAkkorde = \chordmode{
  c1 c:9
}

<<
  \context ChordNames {
    \meineAkkorde
  }
  \context FretBoards {
    \meineAkkorde
  }
>>
```



Bunddiagramme können zu der Tabelle hinzugefügt werden. Um ein Diagramm hinzuzufügen, muss der Akkord des Diagramms, die Stimmung und die Diagramm-Definition angegeben werden. Die Diagramm-Definition kann entweder eine `fret-diagram-terse`-Definition oder eine `fret-diagram-verbose`-Liste sein.

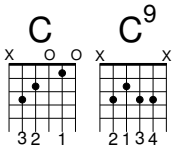
```
\include "predefined-guitar-fretboards.ly"

\storePredefinedDiagram \chordmode {c:9}
                        #guitar-tuning
                        #"x;3-2;2-1;3-3;3-4;x;"

meineAkkorde = \chordmode{
  c1 c:9
```

```
}

<<
  \context ChordNames {
    \meineAkkorde
  }
  \context FretBoards {
    \meineAkkorde
  }
>>
```



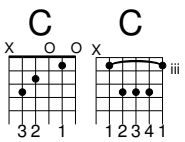
Unterschiedliche Bunddiagramme für den selben Akkord können gespeichert werden, indem unterschiedliche Oktaven für die Tonhöhe benutzt werden.

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

\storePredefinedDiagram \chordmode {c'}
                        #guitar-tuning
                        #(offset-fret 2 (chord-shape 'bes guitar-tuning))

meineAkkorde = \chordmode{
  c1 c'
}

<<
  \context ChordNames {
    \meineAkkorde
  }
  \context FretBoards {
    \meineAkkorde
  }
>>
```



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"

% Neues Akkordmuster hinzufügen
```

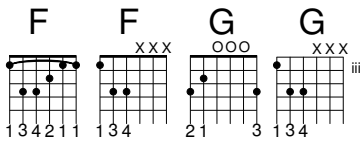
```
\addChordShape #'powerf #guitar-tuning #"1-1;3-3;3-4;x;x;x;"

% neue Akkorde basierend auf dem power-chord-Muster hinzufügen

\storePredefinedDiagram \chordmode {f'}
                        #guitar-tuning
                        #(chord-shape 'powerf guitar-tuning)
\storePredefinedDiagram \chordmode {g'}
                        #guitar-tuning
                        #(offset-fret 2 (chord-shape 'powerf guitar-tuning))

meineAkkorde = \chordmode{
  f1 f' g g'
}

<<
  \context ChordNames {
    \meineAkkorde
  }
  \context FretBoards {
    \meineAkkorde
  }
>>
```



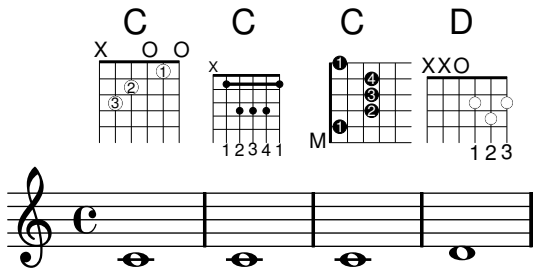
Die graphische Form eines Bunddiagramms kann entsprechend den eigenen Wünschen verändert werden, indem man die Eigenschaften der `fret-diagram-interface`-Schnittstellt verändert. Einzelheiten hierzu in Abschnitt "fret-diagram-interface" in *Referenz der Interna*. Die Schnittstelleneigenschaften eines vordefinierten Bunddiagrammes gehören dem `FretBoards.FretBoard`-Kontext an.

Ausgewählte Schnipsel

*Customizing fretboard fret diagrams*Fret diagram properties can be set through `'fret-diagram-details`. For FretBoard fret diagrams, overrides are applied to the `FretBoards.FretBoard` object. Like `Voice`, `FretBoards` is a bottom level context, therefore can be omitted in property overrides.

```
\include "predefined-guitar-fretboards.ly"
\storePredefinedDiagram \chordmode { c' }
                        #guitar-tuning
                        #"x;1-1-(;3-2;3-3;3-4;1-1-);"
<<
  \new ChordNames {
    \chordmode { c1 c c d }
  }
  \new FretBoards {
    % Allgemeine Eigenschaften von Bund-Diagramme bestimmen
    \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
      \once \override FretBoard #'size = #'1.0
      \once \override FretBoard
        #'(fret-diagram-details barre-type) = #'straight
      \once \override FretBoard
        #'(fret-diagram-details dot-color) = #'black
      \once \override FretBoard
        #'(fret-diagram-details finger-code) = #'below-string
      c'
      \once \override FretBoard
        #'(fret-diagram-details barre-type) = #'none
      \once \override FretBoard
        #'(fret-diagram-details number-type) = #'arabic
      \once \override FretBoard
        #'(fret-diagram-details orientation) = #'landscape
      \once \override FretBoard
        #'(fret-diagram-details mute-string) = #"M"
      \once \override FretBoard
        #'(fret-diagram-details label-dir) = #LEFT
      \once \override FretBoard
        #'(fret-diagram-details dot-color) = #'black
      c'
      \once \override FretBoard
        #'(fret-diagram-details finger-code) = #'below-string
      \once \override FretBoard
        #'(fret-diagram-details dot-radius) = #0.35
      \once \override FretBoard
        #'(fret-diagram-details dot-position) = #0.5
      \once \override FretBoard
        #'(fret-diagram-details fret-count) = #3
      d
    }
  }
  \new Voice {
    c'1 c' c' d'
  }
>>
```



*Defining predefined fretboards for other instruments*Predefined fret diagrams can be added for new instruments in addition to the standards 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.

```
% Bunddiagramme für Cuatro hinzufügen
%   Achtung: dieser Abschnitt gehört in eine eigene Datei
%      predefined-cuatro-fretboards.ly
%      und wird mit \include in jede Quelldatei eingefügt

cuatroStimmung = #'(11 18 14 9)

dSexte = { <a\4 b\1 d\3 fis\2> }
dDur = { <a\4 d\1 d\3 fis \2> }
aGrSept = { <a\4 cis\1 e\3 g\2> }
dGrSept = { <a\4 c\1 d\3 fis\2> }
gDur = { <b\4 b\1 d\3 g\2> }

\storePredefinedDiagram \dSexte
                        #cuatroStimmung
                        #"o;o;o;o;"
\storePredefinedDiagram \dDur
                        #cuatroStimmung
                        #"o;o;o;3-3;"
\storePredefinedDiagram \aGrSept
                        #cuatroStimmung
                        #"o;2-2;1-1;2-3;"
\storePredefinedDiagram \dGrSept
                        #cuatroStimmung
                        #"o;o;o;1-1;"
\storePredefinedDiagram \gDur
                        #cuatroStimmung
                        #"2-2;o;1-1;o;"

% Ende der eigenständigen Datei /predefined-cuatro-fretboards.ly


#(set-global-staff-size 16)

primerosBezeichnungen = \chordmode {
  d:6 d a:maj7 d:maj7
  g
}
primeros = {
  \dSexte \dDur \aGrSept \dGrSept
  \gDur
}

\score {
```

```
  <<
    \new ChordNames {
      \set chordChanges = ##t
      \primerosBezeichnungen
    }

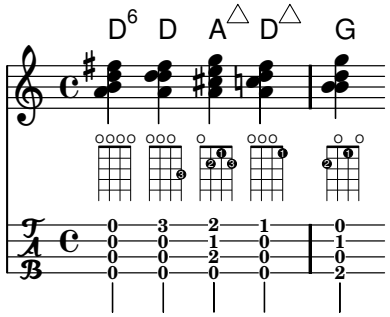
    \new Staff {
      \new Voice \with {
        \remove "New_fingering_engraver"
      }
      \relative c'' {
        \primeros
      }
    }

    \new FretBoards {
      \set stringTunings = #cuatroStimmung
      \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 = #cuatroStimmung
      \primeros
    }

  >>

  \layout {
    \context {
      \Score
      \override SpacingSpanner
        #'base-shortest-duration = #(ly:make-moment 1 16)
    }
  }
  \midi { }
}
```



Siehe auch

Notationsreferenz: [Angepasste Tabulaturen], Seite 225, [Automatische Bund-Diagramme], Seite 242, [Überblick über den Akkord-Modus], Seite 260, Abschnitt B.3 [Vordefinierte Bund-Diagramme], Seite 347.

Installierte Dateien: ‘`ly/predefined-guitar-fretboards.ly`’, ‘`ly/predefined-guitar-ninth-fretboards.ly`’.

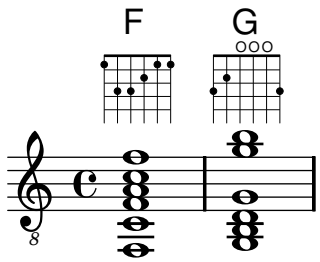
Schnipsel: Abschnitt “Fretted strings” in *Schnipsel*.

Referenz der Interna: Abschnitt “fret-diagram-interface” in *Referenz der Interna*.

Automatische Bund-Diagramme

Bunddiagramme können automatisch aus notierten Noten erstellt werden. Hierzu wird der `FretBoards`-Kontext eingesetzt. Wenn keine vordefinierten Diagramme für die ensprechenden Noten mit der aktiven Saitenstimmung (`stringTunings`) vorhanden sind, errechnet der Kontext Saiten und Bünde die benutzt werden können, um die Noten zu spielen.

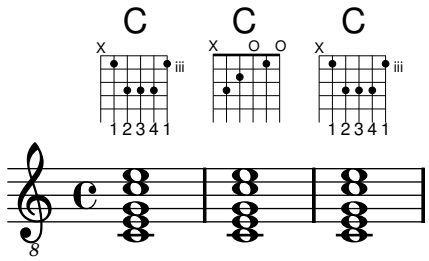
```
<<
  \context ChordNames {
     \chordmode {
       f1 g
     }
  }
  \context FretBoards {
    < f, c f a c' f'>1
    < g,\6 b, d g b g'>
  }
  \context Staff {
    \clef "treble_8"
    < f, c f a c' f'>1
    < g, b, d g b' g'>
  }
>>
```



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 <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
```

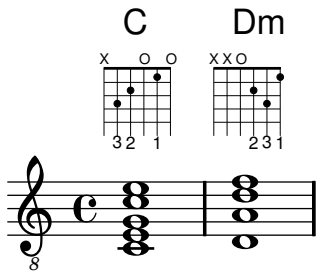
```
    }
  }
  \context FretBoards {
    <c e g c' e'>1
    \predefinedFretboardsOff
    <c e g c' e'>
    \predefinedFretboardsOn
    <c e g c' e'>
  }
  \context Staff {
    \clef "treble_8"
    <c e g c' e'>1
    <c e g c' e'>
    <c e g c' e'>
  }
>>
```



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 ensprechend durch den `FretBoards`-Kontext behandelt.

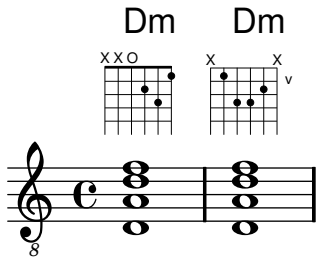
Fingersatz kann zu FretBoard-Bunddiagrammen hinzugefügt werden.

```
<<
  \context ChordNames {
     \chordmode {
       c1 d:m
     }
  }
  \context FretBoards {
    < c-3 e-2 g c'-1 e' > 1
    < d a-2 d'-3 f'-1>
  }
  \context Staff {
    \clef "treble_8"
    < c e g c' e' > 1
    < d a d' f'>
  }
>>
```



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'>
    \set FretBoards.minimumFret = #5
    < d a d' f'>
  }
  \context Staff {
    \clef "treble_8"
    < d a d' f'>
    < d a d' f'>
  }
>>
```



Die Saiten und Bünde des `FretBoards`-Kontextes hängen von der `stringTunings`-Eigesnchaft ab, die die gleiche Bedeitung wie im TabStaff-Kontext hat. Siehe auch [Angepasste Tabulaturen], Seite 225 zu Information über die `stringTunings`Eigenschaft.

Die graphische Erscheindung 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: [Angepasste Tabulaturen], Seite 225.

Schnipsel: Abschnitt “Fretted strings” in *Schnipsel*.

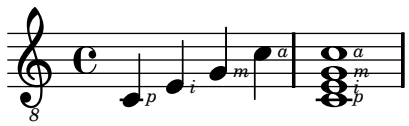
Referenz der Interna: Abschnitt “fret-diagram-interface” in *Referenz der Interna*.

Fingersatz der rechten Hand

Fingersatz für die rechte Hand in Akkorden kann mit den Bezeichnungen *p-i-m-a* notiert werden. Er muss innerhalb eines Akkord-Konstruktes notiert werden.

Achtung: Nach der Note **muss** ein Minuszeichen gesetzt werden und ein Leerzeichen nach dem schließenden `>`.

```
\clef "treble_8"
<c-\rightHandFinger #1 >4
<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`.

```
#(define RH rightHandFinger)
```

Ausgewählte Schnipsel

Placement of right-hand fingerings

It is possible to exercise greater control over the placement of right-hand fingerings by setting a specific property, as demonstrated in the following example.

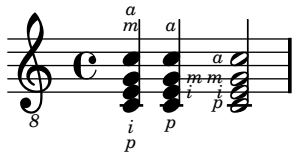
```
#(define RH rightHandFinger)

\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
}
```

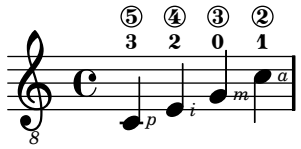


Fingerings, string indications, and right-hand fingerings

This example combines left-hand fingering, string indications, and right-hand fingering.

```
#(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 strings” in *Schnipsel*.

Referenz der Interna: Abschnitt “StrokeFinger” in *Referenz der Interna*.
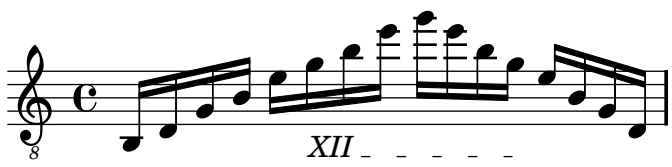
2.4.2 Gitarre

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 beinahlten. Anstattdessen sollte ein Textbearbeitungsprogramm, oder ein Satzprogramm wie GuitarTeX (für erfahrende Benutzer) eingesetzt werden.

Position und Barret anzeigen

Das Beispiel zeigt, wie man Griff- und Barreposition notieren kann.

```
\clef "treble_8"
b16 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

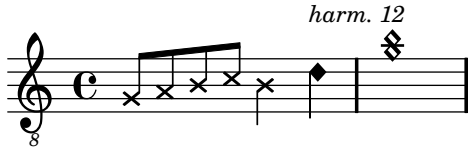
Notationsreferenz: [Text mit Verbindungslinien], Seite 164.

Schnipsel: Abschnitt “Fretted strings” in *Schnipsel*, Abschnitt “Expressive marks” in *Schnipsel*.

Flageolett und gedämpfte Noten

Besondere Notenköpfe können eingesetzt werden, um gedämpfte Noten oder Flageoletttöne anzuzeigen. Flageoletttöne werden normalerweise mit einem Text erklärt.

```
\relative c' {
  \clef "treble_8"
  \override Staff.NoteHead #'style = #'cross
  g8 a b c b4
  \override Staff.NoteHead #'style = #'harmonic-mixed
  d^\markup { \italic { \fontsize #-2 { "harm. 12" }}} <g b>1
}
```



Siehe auch

Notationsreferenz: [Besondere Notenköpfe], Seite 27, Abschnitt B.7 [Notenkopfstile], Seite 366.

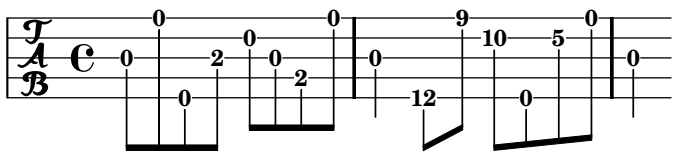
Schnipsel: Abschnitt “Fretted strings” in *Schnipsel*.

2.4.3 Banjo

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 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).

Diese Stimmungen 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

Schnipsel: Abschnitt “Fretted strings” in *Schnipsel*.

Die Datei ‘`scm/output-lib.scm`’ beinhaltet vordefinierte Stimmungen für Banjo.

2.5 Schlagzeug

2.5.1 Übliche Notation für Schlagzeug

Rhythmusnotation wird vor allem für Schlaginstrumente eingesetzt, aber hiermit kann auch der Rhythmus einer Melodie dargestellt werden.

Referenz für Schlagzeug

- Viele Schlagzeugmusik kann auf einem rhythmischen System notiert werden. Das wird gezeigt in [Melodierhythmus anzeigen], Seite 53 und [Neue Notensysteme erstellen], Seite 124.
- MIDI-Ausgabe wird behandelt in Abschnitt 3.5.6 [Schlagzeug in MIDI], Seite 331.

Siehe auch

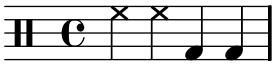
Notationsreferenz: [Melodierhythmus anzeigen], Seite 53, [Neue Notensysteme erstellen], Seite 124. Abschnitt 3.5.6 [Schlagzeug in MIDI], Seite 331.

Schnipsel: Abschnitt “Percussion” in *Schnipsel*.

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 B.11 [Schlagzeugnoten], Seite 405.

Beachten Sie, dass normale Tonhöhen (wie `cis4`) in einem `DrumStaff`-Kontext eine Fehlernachricht erzielen. Schlüssel für Schlagzeug werden automatisch hinzugefügt, aber andere Schlüssel können auch benutzt werden.

Es gibt einige Probleme mit der MIDI-Unterstützung für Schlagzeuginstrumente. Details finden sich in Abschnitt 3.5.6 [Schlagzeug in MIDI], Seite 331.

Siehe auch

Notationsreferenz: Abschnitt 3.5.6 [Schlagzeug in MIDI], Seite 331, Abschnitt B.11 [Schlagzeugnoten], Seite 405.
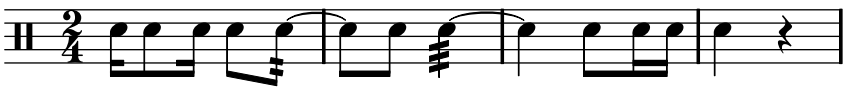
Installierte Dateien: '`ly/drumpitch-init.ly`'.

Schnipsel: Abschnitt "Percussion" in *Schnipsel*.

Trommelwirbel
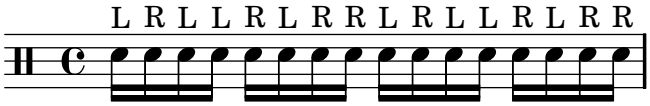
Trommelwirbel werden mit drei Balken durch den Notenhals notiert. Für Viertelnoten oder längere Noten werden die drei Balken explizit notiert, Achtel werden mit zwei Balken gezeigt (und der dritte ist der eigentliche Balken), und Trommelwirbel mit kürzeren Werten als Achtelnoten haben einen Balken zusätzlich zu den eigentlichen Balken der Noten. Dieses Verhalten wird mit der Tremolonotation erreicht, wie in [Tremolo-Wiederholung], Seite 108 gezeigt. Hier ein Beispiel kleinerer Wirbel:

```
\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 durch `^"R"` oder `^"L"` nach jeder Note. 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"
  }
}
```



Siehe auch

Schnipsel: Abschnitt "Percussion" in *Schnipsel*.

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

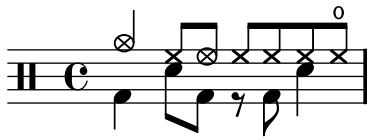
Notationsreferenz: Abschnitt 3.5.6 [Schlagzeug in MIDI], Seite 331.

Schnipsel: Abschnitt “Percussion” in *Schnipsel*.

Schlagzeugsysteme

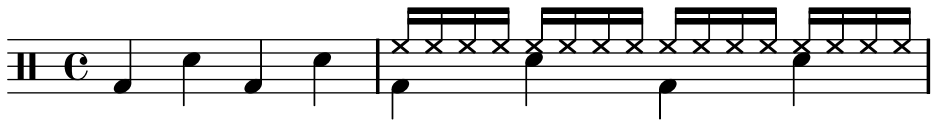
Ein Schlagzeug-System besteht üblicherweise aus einem Notensystem mit mehreren Linien, wobei jede Linie ein bestimmtes Schlagzeug-Instrument darstellt. Um die Noten darstellen zu können, müssen sie sich innerhalb von einem `DrumStaff`- und einem `DrumVoice`-Kontext befinden.

```
oben = \drummode {
  crashcymbal4 hihat8 halfopenhihat hh hh hh openhihat
}
unten = \drummode {
  bassdrum4 snare8 bd r bd sn4
}
\new DrumStaff <<
  \new DrumVoice { \voiceOne \oben }
  \new DrumVoice { \voiceTwo \unten }
>>
```



Das Beispiel zeigt ausdrücklich definierte mehrstimmige Notation. Die Kurznotation für mehrstimmige Musik, wie sie im Abschnitt Abschnitt “I’m hearing Voices” in *Handbuch zum Lernen* beschrieben wird, kann auch verwendet werden, wenn die Stimmen am Anfang explizit initialisiert werden.

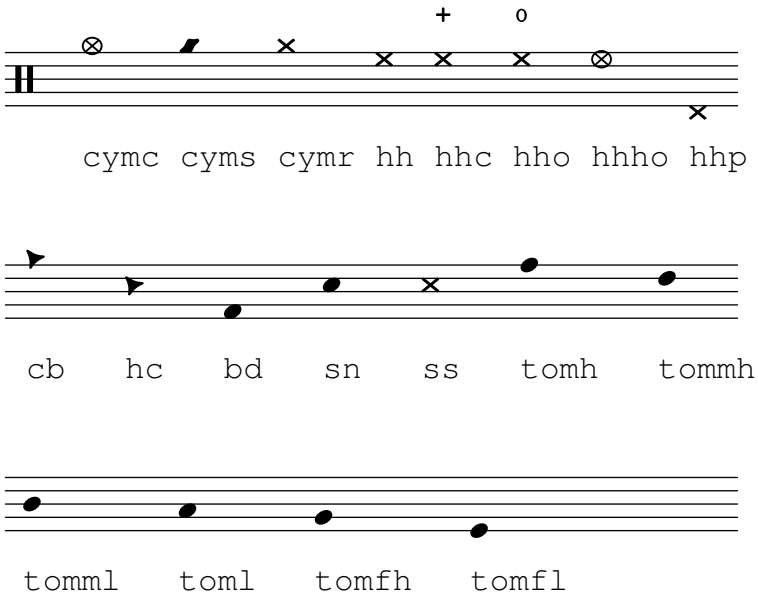
```
\new DrumStaff <<
  \new DrumVoice = "1" { s1*2 }
  \new DrumVoice = "2" { s1*2 }
  \drummode {
    bd4 sn4 bd4 sn4
    << {
      \repeat unfold 16 hh16
    } \\ {
      bd4 sn4 bd4 sn4
    } >>
  }
>>
```



Es gibt auch weitere Layout-Einstellungen. Um diese zu verwenden, muss die Eigenschaft `drumStyleTable` im `DrumVoice`-Kontext entsprechend eingestellt werden. Folgende Variablen sind vordefiniert:

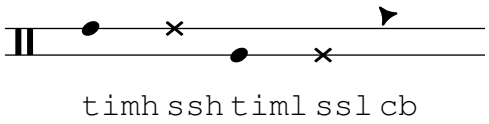
`drums-style`

Das ist die Standardeinstellung. Hiermit wird ein typisches Schlagzeug-System auf fünf Notenlinien erstellt.

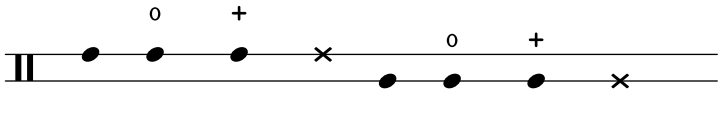


Die Schlagzeugdefinitionen unterstützen sechs unterschiedliche Tom Toms. Falls eine geringere Anzahl verwendet wird, kann man einfach die Tom Toms auswählen, deren Notation man haben will. Tom Toms auf den drei mittleren Linien werden mit den Bezeichnungen `tommh`, `tomml` und `tomfh` notiert.

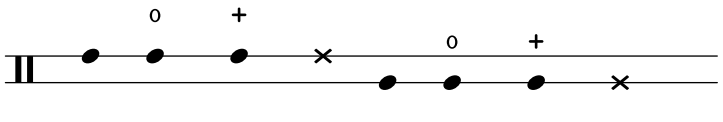
`timbales-style`
: Hiermit werden Timbale auf zwei Notenlinien gesetzt.



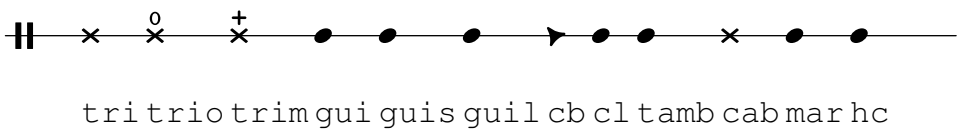
`congas-style`
: Hiermit werden Congas auf zwei Linien gesetzt.



`bongos-style`
: Hiermit werden Bongos auf zwei Linien gesetzt.



`percussion-style`
: Dieser Stil ist für alle einfachen Perkussionsinstrumente auf einer Notenlinie.

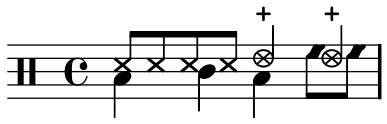


Eigene Schlagzeugsysteme

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)
         (pedalhihat   xcircle   "stopped"  2)
         (lowtom       diamond   #f         3)))
oben = \drummode { hh8 hh hh hh hhp4 hhp }
unten = \drummode { bd4 sn bd toml8 toml }

\new DrumStaff <<
  \set DrumStaff.drumStyleTable = #(alist->hash-table mydrums)
  \new DrumVoice { \voiceOne \oben }
  \new DrumVoice { \voiceTwo \unten }
>>
```



Ausgewählte Schnipsel

Hier einige Beispiele:

Zwei Holzblöcke, notiert mit wbh (hoch) und wbl (tief)

```
% Diese Zeilen definieren die Position der Holzblöcke auf dem System
% Wenn Sie wollen, können Sie das ändern oder besondere Notenköpfe benutzen
% für die Holzblöcke
#(define mydrums '((hiwoodblock default #t  3)
                   (lowoodblock default #t -2)))

System für Holz = {
  % Definiert ein System mit zwei Linien.
  % Auch die Position der zwei Linien wird festgelegt
  \override Staff.StaffSymbol #'line-positions = #'(-2 3)

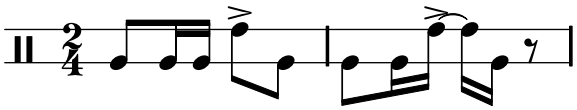
  % Das ist notwendig. Wenn nicht angegeben, wird die Taktlinie zu kurz!
  \override Staff.BarLine #'bar-size = #3
}

\new DrumStaff {
  \set DrumStaff.drumStyleTable = #(alist->hash-table mydrums)

  % jetzt neue Trommelstil-Tabelle laden
  \System für Holz

  \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-size #number` angepasst werden. Andernfalls wäre sie zu kurz. Die Position der beiden Linien muss auch definiert werden.

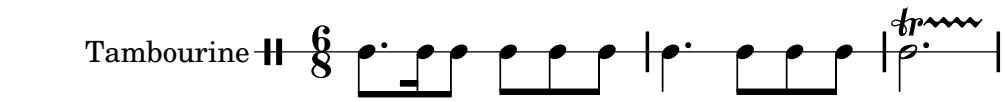
Tamburin, notiert mit `tamb`:

```
#(define mydrums '((tambourine default #t 0)))

Tambu-System = {
  \override Staff.StaffSymbol #'line-positions = #'( 0 )
  \override Staff.BarLine #'bar-size = #3
  \set DrumStaff.instrumentName = #"Tambourine"
}

\new DrumStaff {
  \Tambu-System
  \set DrumStaff.drumStyleTable = #(alist->hash-table mydrums)

  \drummode {
    \time 6/8
    tamb8. tamb16 tamb8 tamb tamb tamb |
    tamb4. tamb8 tamb tamb |
    % Trick mit der skalierten Dauer und der kürzeren Pause
    % istnötig,damitdieTrillerspannweiterichtigendet
    tamb2.*5/6 \startTrillSpan s8 \stopTrillSpan |
  }
}
```



Noten für Tam-Tam (notiert mit `tt`):

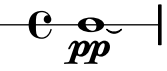
```
#(define mydrums '((tamtam default #t 0)))

tamtam-System = {
  \override Staff.StaffSymbol #'line-positions = #'( 0 )
  \override Staff.BarLine #'bar-size = #3
  \set DrumStaff.instrumentName = #"Tamtam"
}

\new DrumStaff {
  \tamtam-System
  \set DrumStaff.drumStyleTable = #(alist->hash-table mydrums)

  \drummode {
```

```
    tt 1 \pp \laissezVibrer
  }
}
```

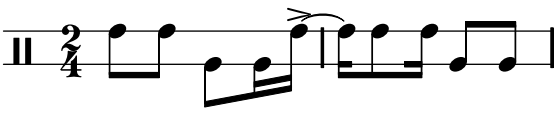
Tamtam 

Zwei Glocken, notiert mit `cb` (Kuhglocke) und `rb` (Reisterglocke)

```
#(define mydrums '((ridebell default #t  3)
                   (cowbell  default #t -2)))

Glocken-System = {
  \override DrumStaff.StaffSymbol #'line-positions = #'(-2 3)
  \set DrumStaff.drumStyleTable = #(alist->hash-table mydrums)
  \override Staff.BarLine #'bar-size = #3
  \set DrumStaff.instrumentName = #"Different Bells"
}

\new DrumStaff {
  \Glocken-System
  \drummode {
    \time 2/4
    rb8 rb cb cb16 rb-> ~ |
    rb16 rb8 rb16 cb8 cb |
  }
}
```

Different Bells 

Here an short example by maestro Stravinsky (from ‚L'histoire du Soldat')

```
#(define mydrums '((bassdrum   default #t  4)
                   (snare      default #t -4)
                   (tambourine default #t  0)))

global = {
  \time 3/8 s4.
  \time 2/4 s2*2
  \time 3/8 s4.
  \time 2/4 s2
}

TrommelA = {
  \context DrumVoice <<
    { \global }
    { \drummode {
        \autoBeamOff
        \stemDown sn8 \stemUp tamb s8 |
        sn4 \stemDown sn4 |
```

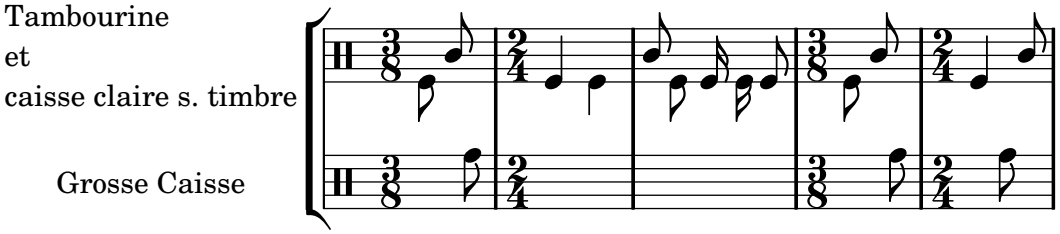
```
        \stemUp tamb8 \stemDown sn8 \stemUp sn16 \stemDown sn \stemUp sn8 |
        \stemDown sn8 \stemUp tamb s8 |
        \stemUp sn4 s8 \stemUp tamb
      }
    }
  >>
}

TrommelB = {
  \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)
      \TrommelA
    }

   \new DrumStaff {
     \set DrumStaff.instrumentName = #"Grosse Caisse"
     \set DrumStaff.drumStyleTable = #(alist->hash-table mydrums)
     \TrommelB }
  >>
}
```



Siehe auch

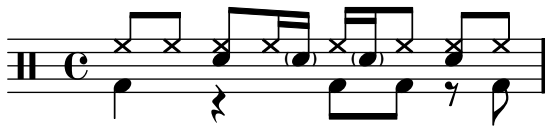
Schnipsel: Abschnitt “Percussion” in *Schnipsel*.

Referenz der Interna: Abschnitt “DrumStaff” in *Referenz der Interna*, Abschnitt “DrumVoice” in *Referenz der Interna*.

Geisternoten

Geisternoten für Schlagzeug und Perkussion können mit dem Klammer- (`\parenthesize`)-Befehl, beschrieben in [Klammern], Seite 157, 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
      }
    >>
  }
>>
```

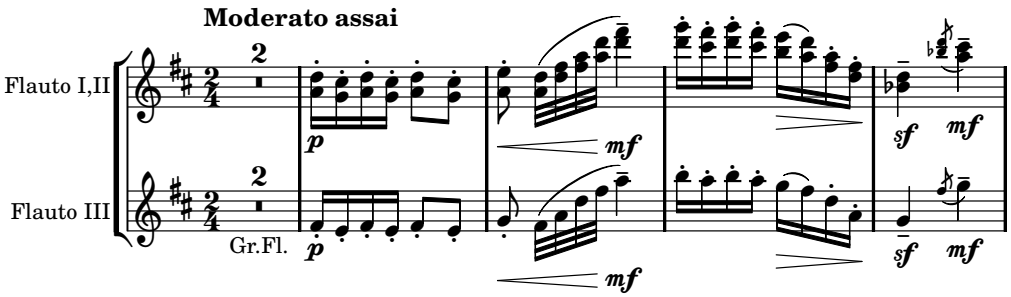


Um jede Klammer-Definition (`\parenthesize`) müssen zusätzlich die spitzen Klammern für Akkorde (`< >`) gesetzt werden.

Siehe auch

Schnipsel: Abschnitt “Percussion” in *Schnipsel*.

2.6 Blasinstrumente



Dieser Abschnitt beinhaltet einige Notationselemente, die bei der Notation von Blasinstrumenten auftreten.

2.6.1 Übliche Notation für Bläser

Dieser Abschnitt erklärt Eigenheiten, die für die meisten Blasinstrumente gültig sind.

Referenz für Blasinstrumente

Viele Besonderheiten der Blasinstrumentenotation haben mit Atmung und Spielart zu tun:

- Atmung kann durch Pausen oder mit Atemzeichen angezeigt werden,, siehe [Atemzeichen], Seite 93.
- Legato kann durch Legatobögen angezeigt werden, siehe [Legatobögen], Seite 90.
- Unterschiedliche Artikulationen, Legato, Portato, Staccato, werden normalerweise mit Artikulationszeichen angemerkt, teilweise auch in Verbindung mit Legatobögen, siehe [Artikulationszeichen und Verzierungen], Seite 83 und Abschnitt B.10 [Liste der Arktikulationszeichen], Seite 404.
- Flatterzunge wird angezeigt, indem ein Tremolozeichen und eine Anmerkung für die entsprechende Note gesetzt wird. Siehe [Tremolo-Wiederholung], Seite 108.

Es gibt auch noch weitere Aspekte der Notation, die für Blasinstrumente relevant sein können:

- Viele Instrumente sind transponierend, siehe [Transposition von Instrumenten], Seite 17.
- Das Zug-Glissando ist charakteristisch für die Posaune, aber auch andere Instrumente können Glissandos ausführen. Siehe [Glissando], Seite 95.
- Obertonreihenglissandi, die auf allen Blechblasinstrumenten möglich, aber besonders üblich für das Waldhorn sind, werden üblicherweise mit Verzierungsnoten geschrieben. Siehe [Verzierungen], Seite 77.
- Tonhöhenschwankungen am Ende eines Tons werden gezeigt in [Glissando zu unbestimmter Tonhöhe], Seite 94.
- Ventil- oder Klappenschläge werden oft als Kreuznoten dargestellt, siehe [Besondere Notenköpfe], Seite 27.
- Holzbläser können tiefe Noten überblasen. Derartige Noten werden als `flageolet`-Artikulation notiert. Siehe Abschnitt B.10 [Liste der Arktikulationszeichen], Seite 404.
- 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 [Artikulationszeichen und Verzierungen], Seite 83 und Abschnitt B.10 [Liste der Arktikulationszeichen], Seite 404.
- Gestopfte Hörner werden mit dem `stopped`-Artikulationszeichen notiert. Siehe [Artikulationszeichen und Verzierungen], Seite 83.

Ausgewählte Schnipsel

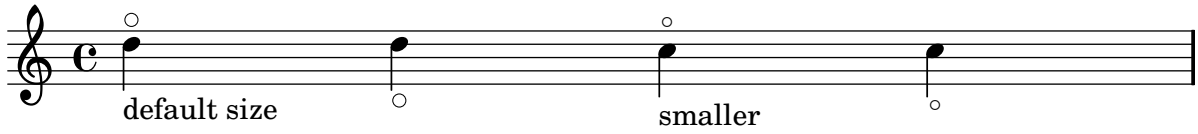
Changing \flageolet mark size

To make the `\flageolet` circle smaller use the following Scheme function.

```
kleinFlageolett = #(let ((m (make-music 'ArticulationEvent
                          'articulation-type "flageolet")))
       (set! (ly:music-property m 'tweaks)
             (acons 'font-size -3
                    (ly:music-property m 'tweaks)))
       m)

\layout { ragged-right = ##f }

\relative c'' {
  d4^\flageolet_\markup { default size } d_\flageolet
  c4^\kleinFlageolett_\markup { smaller } c_\kleinFlageolett
}
```



Siehe auch

Notationsreferenz: [Atemzeichen], Seite 93, [Legatobögen], Seite 90, [Artikulationszeichen und Verzierungen], Seite 83, Abschnitt B.10 [Liste der Arktikulationszeichen], Seite 404, [Tremolo-Wiederholung], Seite 108, [Transposition von Instrumenten], Seite 17, [Glissando], Seite 95, [Verzierungen], Seite 77, [Glissando zu unbestimmter Tonhöhe], Seite 94, [Besondere Notenköpfe], Seite 27,

Schnipsel: Abschnitt “Winds” in *Schnipsel*

Fingersatz

Alle Blasinstrumente außer der Posaune benötigen mehrere Finger, um verschiedene Tonhöhen zu produzieren.

TBC

2.6.2 Dudelsack

Dieser Abschnitt beinhaltet Information zur Notation von Dudelsackmusik.

Dudelsack-Defintionen

LilyPond besitzt spezielle Definitionen, mit der die Musik des schottischen Hochland-Dudelsacks notiert wird. Um sie zu benutzen, muss

```
\include "bagpipe.ly"
```
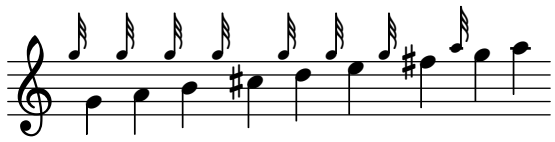
am Anfang der LilyPond-Quelldatei eingefügt werden. Hierdurch können dann bestimmte Verzierungsnoten, die für die Dudelsackmusik üblich sind, mit kurzen Befehlen eingefügt werden. So reicht etwa der Befehl `\taor`, anstatt

```
\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 `cflat` bzw. `fflat`. Gleicherweise kann das piobaireachd hohe g als `gflat` geschrieben werden, wenn es in leichter Musik vorkommt.

Siehe auch

Schnipsel: Abschnitt “Winds” in *Schnipsel*
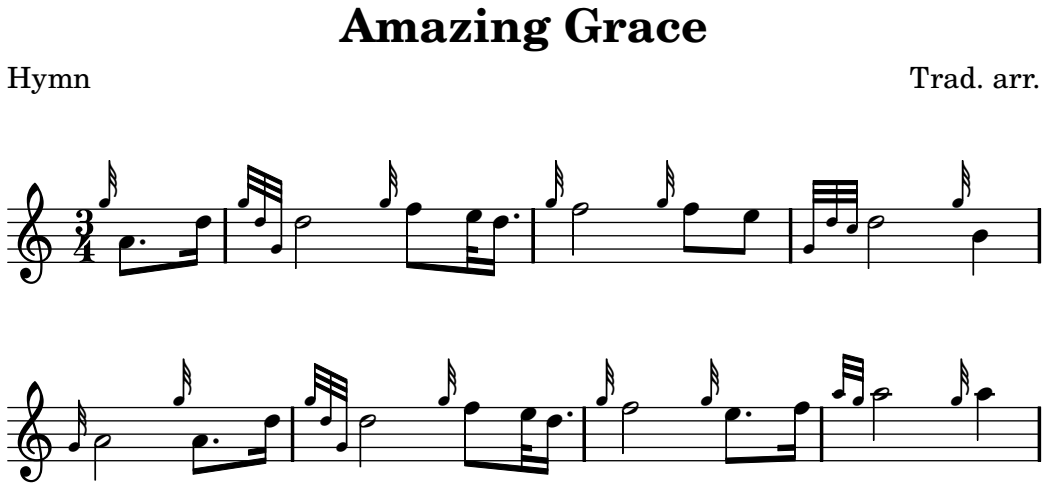
Dudelsack-Beispiele
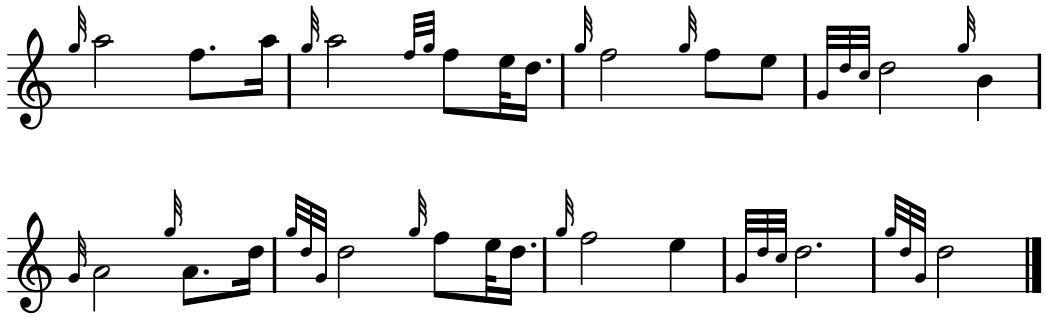
So sieht die bekannte Melodie Amazing Grace aus, wenn man sie für Dudelsack notiert.

```
\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 "|."
}
```

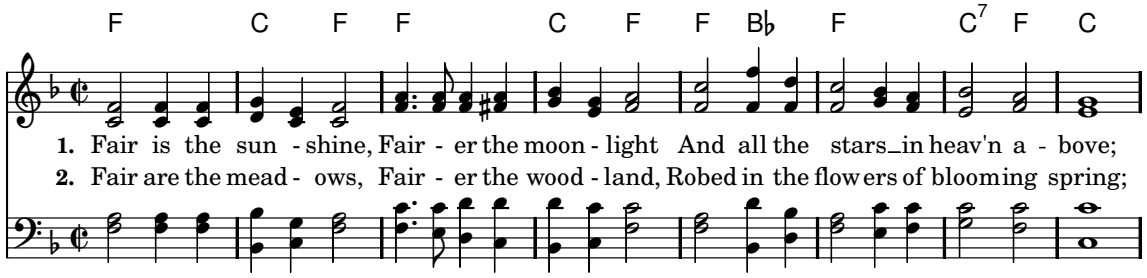




Siehe auch

Schnipsel: Abschnitt “Winds” in *Schnipsel*

2.7 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.

2.7.1 Akkord-Modus

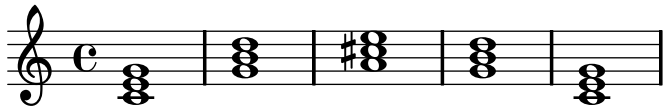
Im Akkordmodus (engl. „chord“) werden Akkorde anhand von einem Symbol der erwünschten Akkordstruktur notiert, anstatt dass die einzelnen Tonhöhen ausgeschrieben werden.

Überblick über den Akkord-Modus

Akkorde können als simultane Noten eingegeben werden, wie gezeigt in [Noten mit Akkorden], Seite 109.

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 5.4.1 [Eingabe-Modi], Seite 335.

```
\chordmode { c1 g a g c }
```

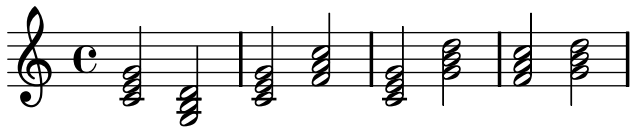


Akkorde, die im Akkordmodus eingegeben werden, sind musikalische Elemente und können genauso wie Akkorde im Notenmodus transponiert werden.

Akkordmodus und Notenmodus können gemischt verwendet werden:

```
<c e g>2 <g b d>
\chordmode { c2 f }
<c e g>2 <g' b d>
```

```
\chordmode { f2 g }
```



Siehe auch

Glossar: Abschnitt “Akkord” in *Glossar*.
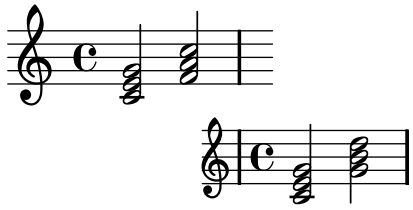
Notationsreferenz: [Noten mit Akkorden], Seite 109, Abschnitt 5.4.1 [Eingabe-Modi], Seite 335.

Schnipsel: Abschnitt “Chords” in *Schnipsel*

Bekannte Probleme und Warnungen
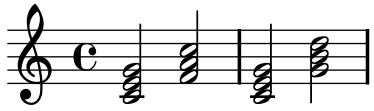
Wenn Akkord- und Notenmodus in linearer Musik abwechseln eingesetzt werden und der Akkordmodus am Anfang steht, erstellt der Notenmodus ein neues Notensyste:

```
\chordmode { c2 f }
<c e g>2 <g' b d>
```



Um dieses Verhalten zu verhindert, muss der `Staff`-Kontext explizit aufgerufen werden:

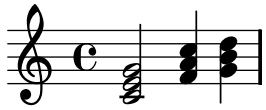
```
\new Staff {
  \chordmode { c2 f }
  <c e g>2  <g' b d>
}
```



Übliche Akkorde
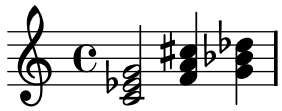
Ein Dreiklang wird mit seinem Grundton mit einer möglichen Dauer dahinter notiert:

```
\chordmode { c2 f4 g }
```



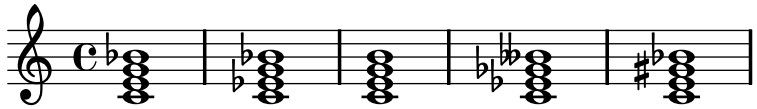
Moll- übermäßige und verminderte Dreiklänge werden notiert, indem `:` und ein Modifikator hinter der Dauer angegeben wird:

```
\chordmode { c2:m f4:aug g:dim }
```

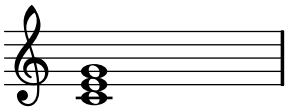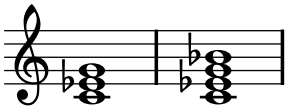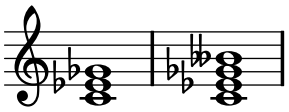
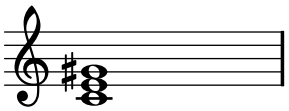
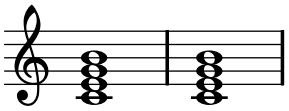


Septakkorde können erstellt werden:

```
\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 B.2 [Übliche Akkord-Variablen], Seite 344.

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 B.2 [Übliche Akkord-Variablen], Seite 344, [Erweiterte und modifizierte Akkorde], Seite 263.

Schnipsel: Abschnitt “Chords” in *Schnipsel*.

Bekannte Probleme und Warnungen

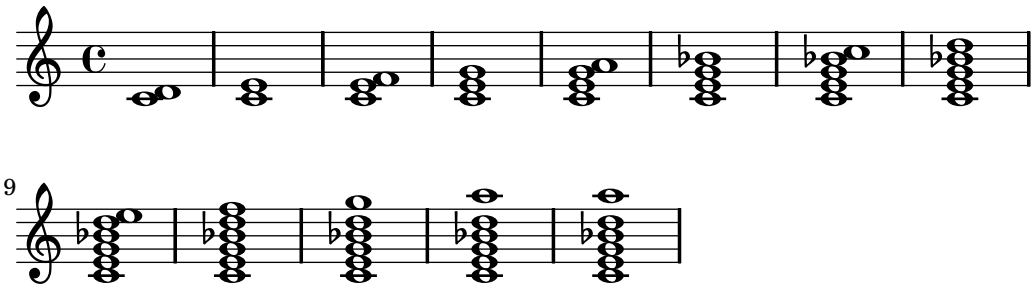
Nur ein Qualitätsmodifikator sollte pro Akkord benutzt werden, meistens für die höchste Stufe des Akkordes. Akkorde mit weiteren Qualitätsmodifikatoren werden ohne Warnung oder Fehlermeldung gelesen, aber das Ergebnis ist nicht vorhersagbar. Akkorde, die nicht mit einem einzigen Qualitätsmodifikator erreicht werden können, sollten mit einzelnen Tonhöhen alteriert werden, wie beschrieben in [Erweiterte und modifizierte Akkorde], Seite 263.

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. Umgkehrungen 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.

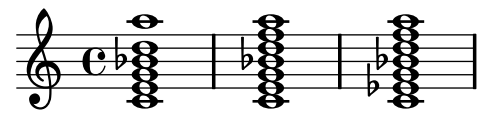
```
\chordmode {
  c1:2 c:3 c:4 c:5
  c1:6 c:7 c:8 c:9
  c1:10 c:11 c:12 c:13
  c1:14
}
```



Sowohl `c:5` als auch `c` erzeugen einen D-Dur-Dreiklang.
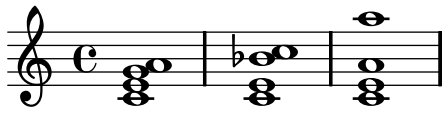
Da eine unveränderte 11 nicht gut klingt, wenn sie mit einer unveränderten 13 zusammenklingt, wird die 11 von einem `:13`-Akkord entfernt (es sei denn sie wird explizit verlangt).

```
\chordmode {
  c1:13 c:13.11 c:m13
}
```



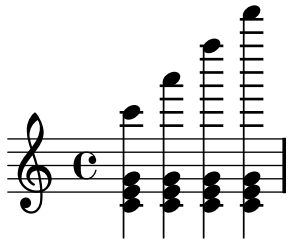
Kompliziertere Akkorde können auch konstruiert werden, indem einzelne Intervalle zu dem Grundton addiert werden. Diese Additionen werden nach dem Bereich notiert und mit Punkten voneinander getrennt. Die normale Septime, die zu einem Akkord hinzugefügt wird, ist die kleine Septime, nicht die große.

```
\chordmode {
  c1:5.6 c:3.7.8 c:3.6.13
}
```
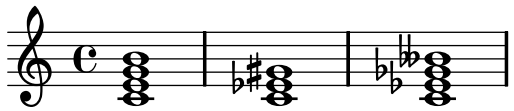


Hinzugefügte Stufen können beliebig groß sein:

```
\chordmode {
  c4:5.15 c:5.20 c:5.25 c:5.30
}
```



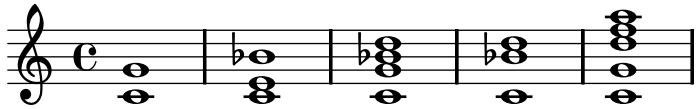
Einzelne Stufen können mit `-` oder `+` vergrößert oder verkleinert werden. Um eine Stufe zu verändert, die automatisch in den Akkord aufgenommen wurde, kann sie in veränderter Form nach dem Bereich hinzugefügt werden.

```
\chordmode {
  c1:7+ c:5+.3- c:3-.5-.7-
}
```



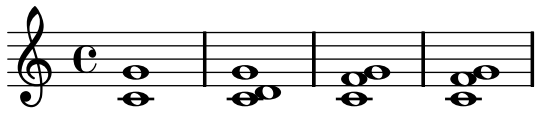
Zu entfernende Töne werden mit der gleichen Methode notiert, allerdings mit einem Dach (`^`) vor der Sequenz, die nicht erscheinen soll. Sie müssen nach den zu addierenden Tönen notiert werden. Die einzelnen zu entfernenden Töne werden mit Punkten getrennt.

```
\chordmode {
  c1^3 c:7^5 c:9^3 c:9^3.5 c:13.11^3.7
}
```



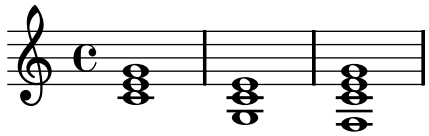
Sekund- und Quartakkorde können mit dem Modifikator `sus` notiert werden. Hiermit wird die dritte Stufe aus dem Akkord entfernt. Mit einer anschließenden `2` wird die zweite, mit einer `4` die vierte Stufe hinzugefügt. `sus` entspricht `^3` und `sus4` ist gleich `.4^3`.

```
\chordmode {
  c1:sus c:sus2 c:sus4 c:5.4^3
}
```



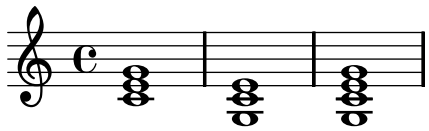
Eine Umkehrung (ein Ton des Akkordes wird unter den Grundton gesetzt) sowie auch zusätzliche Bassnoten können mit dem Schrägstrich (`/`) markiert werden:

```
\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 B.2 [Übliche Akkord-Variablen], Seite 344.

Siehe auch
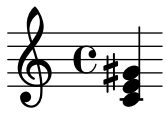
Notationsreferenz: Abschnitt B.2 [Übliche Akkord-Variablen], Seite 344.

Schnipsel: Abschnitt “Chords” in *Schnipsel*

Bekannte Probleme und Warnungen

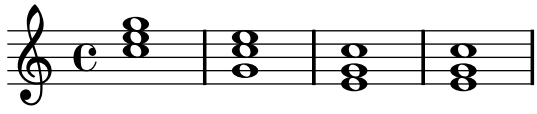
Jede Stufe kann nur einmal in einem Akkord vorkommen. Im folgenden Beispiel wird ein erweiterter Akkord erstellt, weil `5+` zuletzt gelesen wird.

```
\chordmode {
  c: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
}
```



2.7.2 Akkorde anzeigen

Akkorde können zusätzlich zur üblichen Notation als Töne auf einem Notensystem auch mit einem Akkordsymbol gesetzt werden.

Akkordbezeichnungen drucken

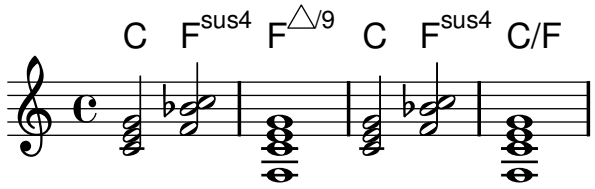
Akkordsymbole anstelle der Noten werde im `ChordNames`-Kontext notiert.

```
\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, Umgkehrungen 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
    }
  }
>>
```



`\chords { ... }` ist eine Kurznotation für die Bezeichnung `\new ChordNames { \chordmode { ... } }`.

```
\chords {
  c2 f4.:m g8:maj7
}
```




```
\new ChordNames {
  \chordmode {
```

```
    c2 f4.:m g8:maj7
  }
}
```

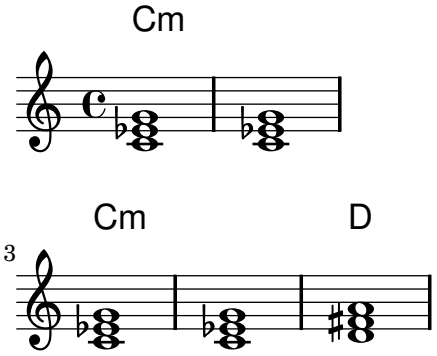


Ausgewählte Schnipsel

Showing chords at changes

Chord names can be displayed only at the start of lines and when the chord changes.
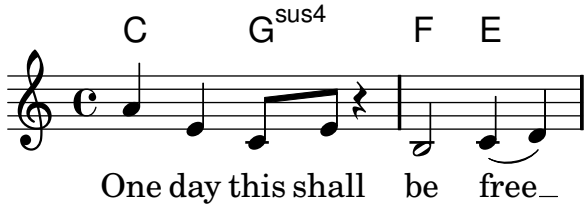
```
Harmonien = \chordmode {
  c1:m c:m \break c:m c:m d
}
<<
  \new ChordNames {
    \set chordChanges = ##t
    \Harmonien
  }
  \new Staff {
    \relative c' { \Harmonien }
  }
>>
```



Simple lead sheet

When put together, chord names, a melody, and lyrics form a lead sheet:

```
<<
  \chords { c2 g:sus4 f e }
  \relative c'' {
    a4 e c8 e r4
    b2 c4( d)
  }
  \addlyrics { One day this shall be free __ }
>>
```



Siehe auch

Glossar: Abschnitt “Akkord” in *Glossar*.

Notationsreferenz: [Musik parallel notieren], Seite 121.

Schnipsel: Abschnitt “Chords” 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

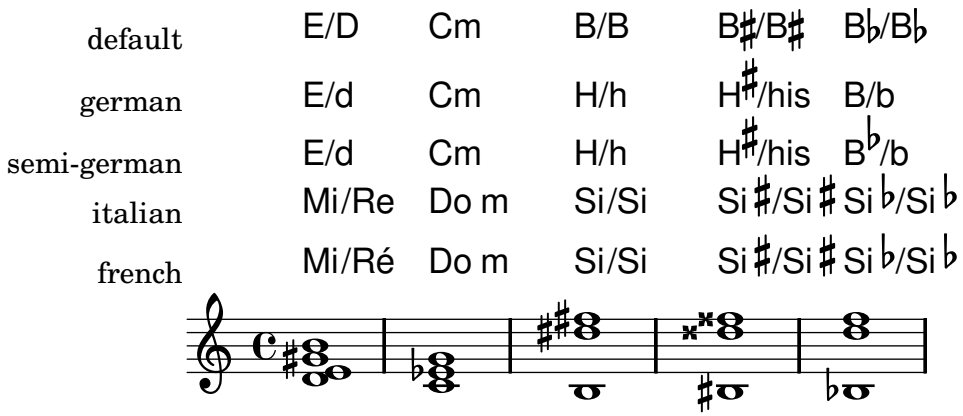
Akkorde, die Umkehrungen oder zusätzliche Basstöne beinhalten, werden nicht richtig bezeichnet, wenn sie im Notenmodus notiert werden.

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 Anhang A [Literatur], Seite 342). 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 B.1 [Liste der Akkordbezeichnungen], Seite 343.

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.



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`.

`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 einen Schrägstrich markiert. Indem `chordNameSeparator` ein anderer Wert zugewiesen wird, kann ein beliebiges Zeichen für den Trenner benutzt werden.

`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.

`chordPrefixSpacer`
: Das „m“ für Moll-Akkorde wird normalerweise direkt hinter dem Akkordbuchstaben gesetzt. Mit der Eigenschaft `chordPrefixSpacer` kann ein Abstand(halter) zwischen den Buchstaben und das „m“ gesetzt werden. Der Abstandhalter wird nicht verwendet, wenn der Grundton erhöht oder erniedrigt ist.

Vordefinierte Befehle

`\whiteTriangleMarkup`, `\blackTriangleMarkup`, `\germanChords`, `\semiGermanChords`, `\italianChords`, `\frenchChords`.

Ausgewählte Schnipsel

Chord name exceptions

The property `chordNameExceptions` can be used to store a list of special notations for specific chords.

```
% maj9 und 6(add9) verändern
% Ausnahmemusik sind Akkorde mit Textbeschriftung
chExceptionMusic = {
  <c e g b d'>1-\markup { \super "maj9" }
  <c e g a d'>1-\markup { \super "6(add9)" }
}

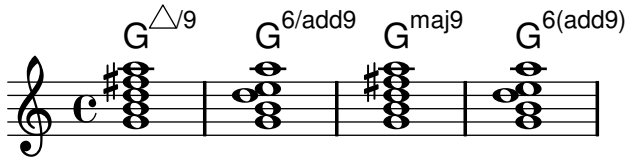
% Noten zu Liste konvertieren und an existierende Ausnahmen anhängen
chAusnahmen = #( append
  ( sequential-music-to-chord-exceptions chExceptionMusic #t)
  ignatzekExceptions)

Noten = \chordmode {
  g1:maj9 g1:6.9
  \set chordNameExceptions = #chAusnahmen
```

```
  g1:maj9 g1:6.9
}

\layout {
  ragged-right = ##t
}

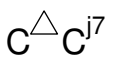
<< \context ChordNames \Noten
   \context Voice \Noten
>>
```



chord name major7

The layout of the major 7 can be tuned with `majorSevenSymbol`.

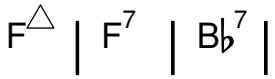
```
\chords {
  c:7+
  \set majorSevenSymbol = \markup { j7 }
  c:7+
}
```



Adding bar lines to ChordNames context

To add bar line indications in the `ChordNames` context, add the `Bar_engraver`.

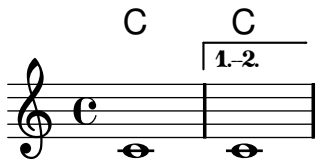
```
\new ChordNames \with {
  \override BarLine #'bar-size = #4
  \consists "Bar_engraver"
}
\chordmode {
  f1:maj7 f:7 bes:7
}
```



*Volta under chords*By adding the `Volta_engraver` to the relevant staff, volte can be put under chords.

```
\score {
  <<
    \chords {
      c1
      c1
    }
    \new Staff \with {
      \consists "Volta_engraver"
```

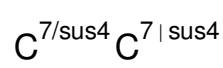
```
    }
    {
      \repeat volta 2 { c'1 }
      \alternative { c' }
    }
  >>
  \layout {
    \context {
      \Score
      \remove "Volta_engraver"
    }
  }
}
```



Changing chord separator

The separator between different parts of a chord name can be set to any markup.

```
\chords {
  c:7sus4
  \set chordNameSeparator
    = \markup { \typewriter | }
  c:7sus4
}
```



Siehe auch

Notationsreferenz: Abschnitt B.1 [Liste der Akkordbezeichnungen], Seite 343, Abschnitt B.2 [Übliche Akkord-Variablen], Seite 344.

Installierte Dateien: ‘`scm/chords-ignatzek.scm`’, ‘`scm/chord-entry.scm`’, ‘`ly/chord-modifier-init.ly`’.

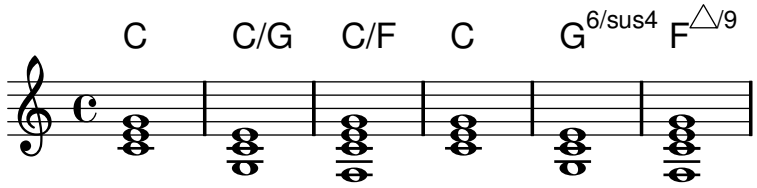
Schnipsel: Abschnitt “Chords” in *Schnipsel*.

Bekannte Probleme und Warnungen
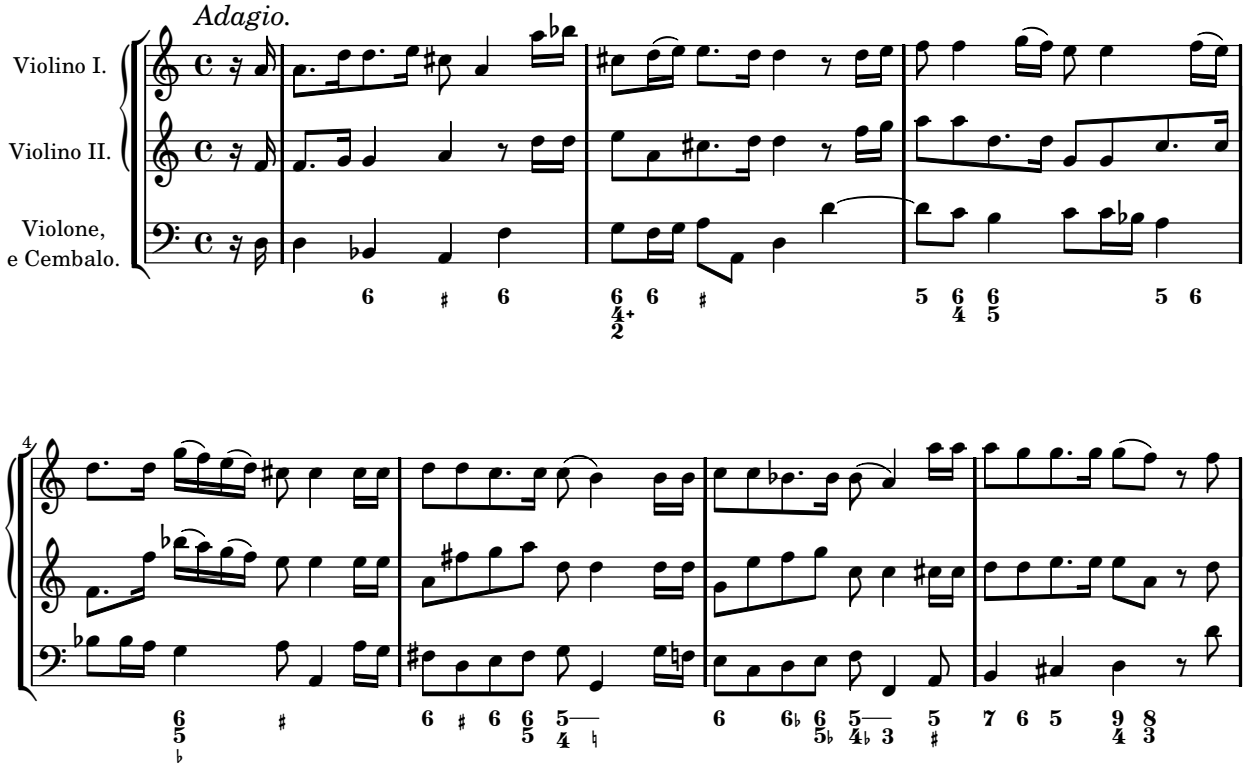
Akkordsymbole werden von den Tonhöhenbezeichnungen innerhalb des Akkordes und der Information über die Akkordstruktur, die innerhalb von `\chordmode` notiert wurde, bestimmt. Wenn der direkte Notenmodus benutzt wird, stammen unerwünschte Bezeichnungen daher, dass Umgkehrungen und zustätzliche Bassnoten nicht richtig interpretiert werden.

```
meineAkkorde = \relative c' {
  \chordmode { c1 c/g c/f }
  <c e g>1 <g c e> <f c' e g>
}
<<
  \new ChordNames { \meineAkkorde }
  \new Staff { \meineAkkorde }
```

>>



2.7.3 Generalbass

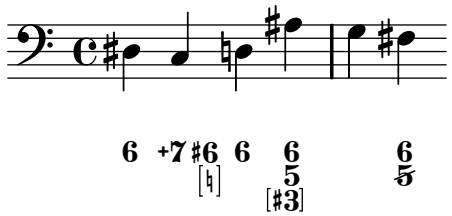


Generalbassnotation kann dargestellt werden.

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 “Generalbass” in *Glossar*.

Schnipsel: Abschnitt “Chords” in *Schnipsel*

Eingabe des Generalbass’

`\figuremode` (Zahlenmodus) wird benutzt, um den Eingabemodus auf den Zahlenmodus umzustellen. Mehr Information zu unterschiedlichen Eingabemodi findet sich in Abschnitt 5.4.1 [Eingabe-Modi], Seite 335.

Im Zahlenmodus wird eine Gruppe von Bassziffern mit den Zeichen `<` and `>` 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/>
}
```



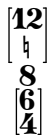
Ein Schrägstrich von links nach rechts (üblicherweise für erhöhte Sexten benutzt) kann erstellt werden:

```
\figures {
  <6> <6\\>
}
```



Vertikaler Platz und Klammern können zu den Zahlen hinzugefügt werden:

```
\figures {
  <[12 _!] 8 [6  4]>
}
```



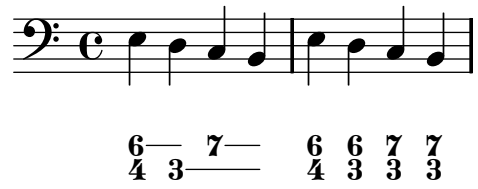
Beliebiger Text kann als Zahl notiert werden:

```
\figures {
  <\markup { \tiny \number 6 \super (1) } 5>
}
```



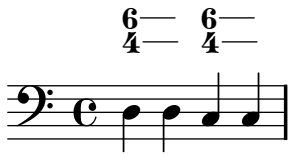
Es ist auch möglich, Fortsetzungslinien für wiederholte Ziffern zu benutzen.

```
<<
  {
    \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>
  }
>>
```

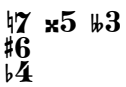
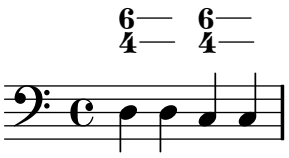


In diesem Fall werden wiederholte Ziffern immer durch eine Linie ersetzt, es sei denn, die Linie wird explizit beendet.

```
<<
  \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

Changing the positions of figured bass alterations

Accidentals and plus signs can appear before or after the numbers, depending on the `figuredBassAlterationDirection` and `figuredBassPlusDirection` properties.

```
\figures {
  <6\+> <5+> <6 4-> r
  \set figuredBassAlterationDirection = #RIGHT
  <6\+> <5+> <6 4-> r
  \set figuredBassPlusDirection = #RIGHT
  <6\+> <5+> <6 4-> r
  \set figuredBassAlterationDirection = #LEFT
  <6\+> <5+> <6 4-> r
}
```



Siehe auch

Schnipsel: Abschnitt “Chords” in *Schnipsel*.

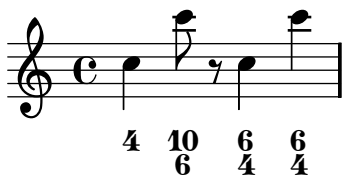
Referenz der Interna: Abschnitt “BassFigure” in *Referenz der Interna*, Abschnitt “BassFigureAlignment” in *Referenz der Interna*, Abschnitt “BassFigureLine” in *Referenz der Interna*, Abschnitt “BassFigureBracket” in *Referenz der Interna*, Abschnitt “BassFigureContinuation” in *Referenz der Interna*, Abschnitt “FiguredBass” in *Referenz der Interna*.

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 paralellen Systems.

```
<<
  \relative c'' {
    c4 c'8 r8 c,4 c'
  }
  \new FiguredBass {
    \figuremode {
      <4>4 <10 6>8 s8
      <6 4>4 <6 4>
    }
  }
>>
```

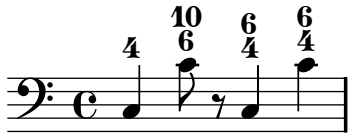


In diesem Beispiel muss der `FiguredBass`-Kontext explizit erstellt werden, damit kein zusätzliches (leeres) Notensystem erstellt wird.

Bassziffern können auch direkt einem Notensystemkontext (`Staff`) hinzugefügt werden. In diesem Fall wird ihre vertikale Position automatisch bestimmt.

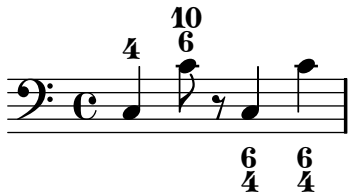
```
<<
  \new Staff = myStaff
  \figuremode {
```

```
    <4>4 <10 6>8 s8
    <6 4>4 <6 4>
  }
  %% Noten auf dem gleichen System wie die Symbole
  \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>
  }
  %% Noten auf dem gleichen System wie die Symbole
  \context Staff = myStaff
  {
    \clef bass
    c4 c'8 r8 c4 c'
  }
>>
```



Schnipsel: Abschnitt “Chords” in *Schnipsel*.
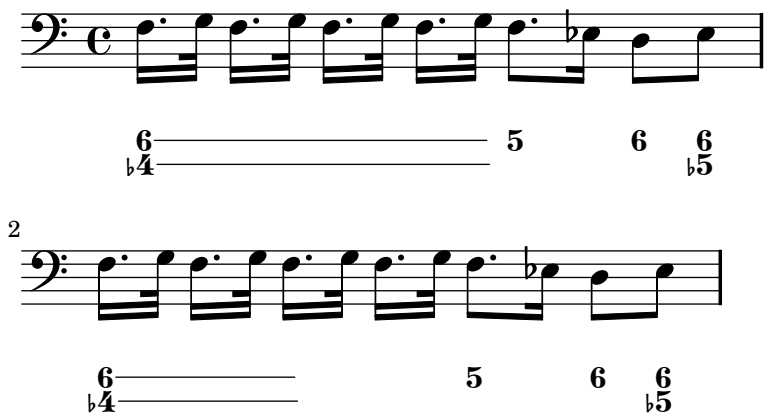
Referenz der Interna: Abschnitt “BassFigure” in *Referenz der Interna*, Abschnitt “BassFigureAlignment” in *Referenz der Interna*, Abschnitt “BassFigureLine” in *Referenz der Interna*, Abschnitt “BassFigureBracket” in *Referenz der Interna*, Abschnitt “BassFigureContinuation” in *Referenz der Interna*, 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.

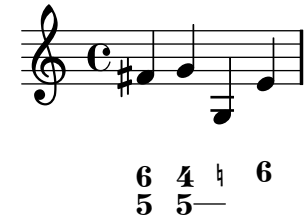
```
<<
  {
    \clef bass
```

```
    \repeat unfold 4 { f16. g32 } f8. es16 d8 es
  }
  \figures {
    \bassFigureExtendersOn
    % Linien stimmen hier, mit dem gleichen Rhythmus wie im 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
    % Linien stimmen nicht, obwohl der Rhythmus der gleiche ist
    <6 4->4 <6 4->4
    <5>8. r16 <6>8 <6\! 5->
  }
>>
```



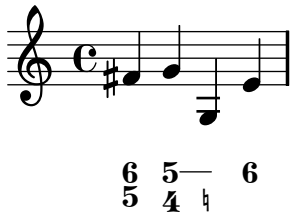
Wenn Fortsetzungslinien eingesetzt werden, können aufeinander folgende Bezifferungen mit der selben Zahl in einer anderen Position dazu führen, dass sich die Reihenfolge der Zahlen umkehrt.

```
<<
 { fis4 g g, e' }
  \figures {
    \bassFigureExtendersOn
    <6 5>4 <5\! 4> < 5 _!> <6>
  }
>>
```
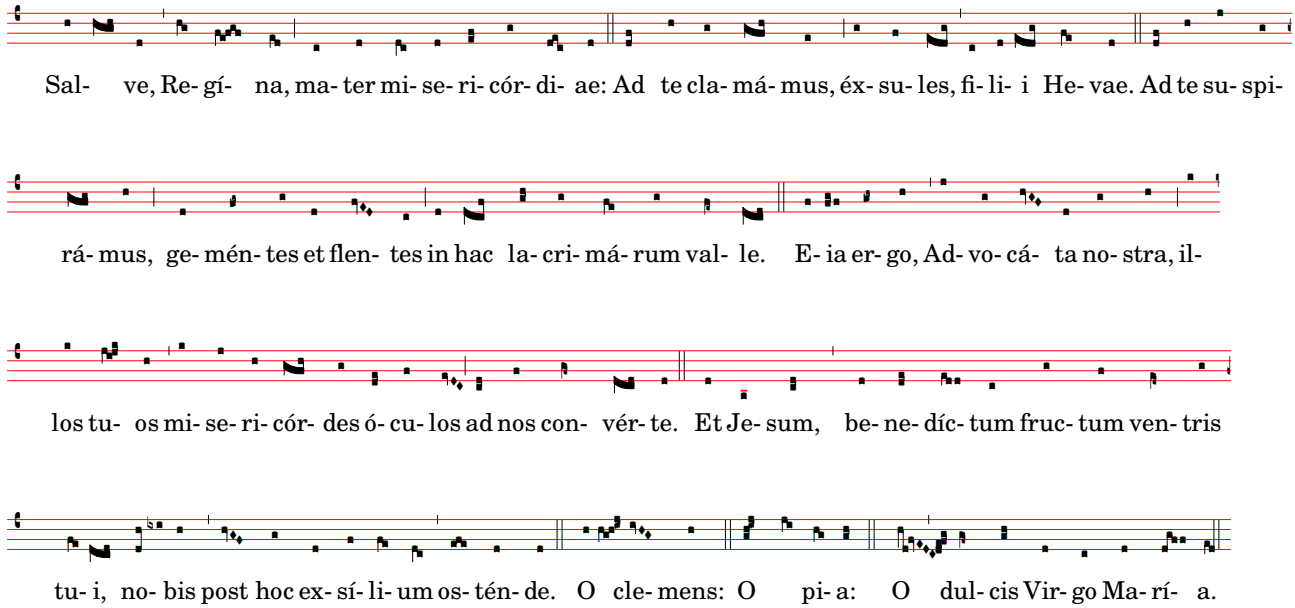


Um dieses Problem zu umgehen, kann die Fortsetzungslinie nach der Bezifferung, mit der die Linie beginnen soll, angeschaltet und am Ende der Linie wieder ausgeschaltet werden.

```
<<
 { fis4 g g, e' }
  \figures {
    <6 5>4 <5 4>
    \bassFigureExtendersOn
    < 5 _!>4 <6>
    \bassFigureExtendersOff
  }
>>
```



2.8 Notation von alter Musik



2.8.1 Einführung in die Notation Alter Musik

Notation von Alter Musik, die unterstützt wird

2.8.2 Alternative Notenzeichen

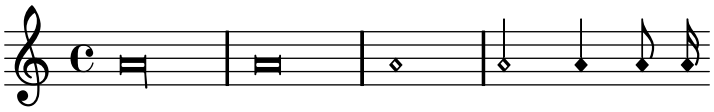
Notenköpfe Alter Musik

Für die Notation Alter Musik kann ein Notenkopfstil ausgewählt werden, der sich vom Standard (`default`) unterscheidet. Dies wird erreicht, indem die `style`-Eigenschaft der Notenkopf-(Abschnitt “NoteHead” in *Referenz der Interna*)-Objekte auf einen der Werte `baroque`, `neomensural`, `mensural` oder `petrucci` gesetzt wird. Der barocke (`baroque`) Stil

unterscheidet sich vom Standard (`default`) nur in sofern, als eine viereckige Form für die Brevis (`\breve`) benutzt wird. Der `neomensural`-Stil unterscheidet sich vom barocken Stil darin, dass hier rhomboide Notenköpfe für ganze Noten und kleinere Notenwerte eingesetzt werden. Hälse werden über oder unter den Notenköpfen zentriert. Dieser Stil ist vor allem dann sinnvoll, wenn mensurale Musik transkribiert werden soll, etwa für ein Incipit. Der mensurale (`mensural`) Stil erstellt Notenköpfe, die das Aussehen historischer Drucke des 16. Jahrhunderts imitieren. Der `petrucci`-Stil schließlich imitiert auch historische Drucke, verwendet allerdings größere Notenköpfe.

Das folgende Beispiel soll den neomensuralen Stil demonstrieren:

```
\set Score.skipBars = ##t
\override NoteHead #'style = #'neomensural
a'\longa a'\breve a'1 a'2 a'4 a'8 a'16
```



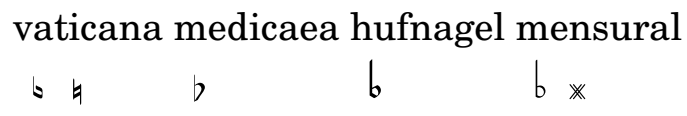
Für die Notation des Gregorianischen Chorals werden vom Abschnitt "Vaticana_ligature_engraver" in *Referenz der Interna* automatisch die richtigen Notenköpfe ausgewählt, so dass man den Stil nicht manuell setzen muss. Trotzdem kann der Stil manuell verändert werden, z. B. auf den Wert `vaticana_punctum`, um Punktum-Neumen zu produzieren. Gleichfalls erstellt der Abschnitt "Mensural_ligature_engraver" in *Referenz der Interna* automatisch Ligaturen der Mensuralnotation. Siehe [Ligaturen], Seite 287 zu einer Übersicht über die Funktion und Notation von Ligaturen.

Siehe auch

Abschnitt B.7 [Notenkopfstile], Seite 366 stellt einen Überblick über alle verfügbaren Notenkopfstile zur Verfügung.

Versetzungszeichen Alter Musik

Mit der `glyph-name-alist`-Eigenschaft der Versetzungszeichen-(Abschnitt "Accidental" in *Referenz der Interna*)- und Vorzeichen-(Abschnitt "KeySignature" in *Referenz der Interna*)-Eigenschaften können Vorzeichen und Versetzungszeichen für die Alte Musik ausgewählt werden.



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, wie in dem Beispiel `ancient-accidentals.ly` demonstriert wird.

Ähnlich wie Versetzungszeichen können auch die Vorzeichen für die Angabe der Tonart verändert werden, indem die `glyph-name-alist`-Eigenschaft des Abschnitt "KeySignature" in *Referenz der Interna*-Objektes gesetzt wird.

Siehe auch

In diesem Handbuch: Abschnitt 1.1 [Tonhöhen], Seite 1, [Versetzungszeichen], Seite 4 und [Automatische Versetzungszeichen], Seite 19 geben eine allgemeine Einführung in die Benutzung

von Versetzungszeichen. Der Abschnitt [Tonartbezeichnung], Seite 15 zeigt die allgemeine Benutzung von Vorzeichen.

Programmreferenz: Abschnitt “KeySignature” in *Referenz der Interna*.
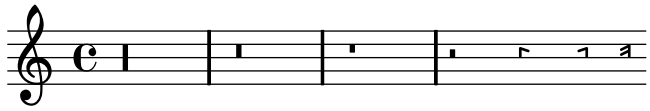
Beispiele: Abschnitt “Ancient notation” in *Schnipsel*.

Pausen Alter Musik

Besondere Pausensymbole für die Notation der Alten Musik können mit der `style`-Eigenschaft des graphischen Objektes (grob) „Pause“ (Abschnitt “Rest” in *Referenz der Interna*) 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 neomensurale Stil eignet sich gut, um z. B. das Incipit von transkribierter Musik zu notieren. Der mensurale Stil ahmt die Form von Pausen nach, wie man sie in Drucken des 16. Jahrhunderts finden kann.

Das nächste Beispiel demonstriert den neomensuralen (`neomensural`) Stil:

```
\set Score.skipBars = ##t
\override Rest #'style = #'neomensural
r\longa 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. Vgl. eine Liste aller vorhandenen Pausen in `pitches,rests`.

Für die Notation des Gregorianischen Chorals gibt es keine Pausen; anstelle dessen werden [Divisiones], Seite 286 verwendet.
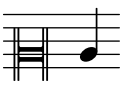
Siehe auch

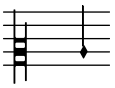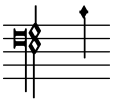
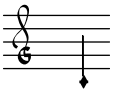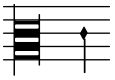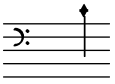
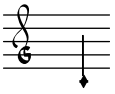
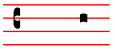
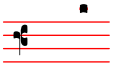
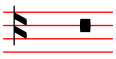
In diesem Handbuch: Der Abschnitt [Pausen], Seite 39 enthält eine allgemeine Einführung zur Benutzung von Pausen.

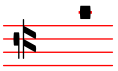
Schlüssel Alter Musik

LilyPond unterstützt eine große Anzahl von Notenschlüsseln, von denen eine ganze Anzahl für die Alte Musik geeignet ist.

In der Tabelle unten werden alle Schlüssel für die Alte Musik gezeigt, die mit dem `\clef`-Befehl erreicht werden. Manche Schlüssel benutzen den selben Schlüssel, unterscheiden sich aber in der Notenlinie, auf der der Schlüssel notiert wird. In diesem Fällen ist eine Nummer im Schlüsselnamen eingefügt. Man kann aber trotzdem eine beliebige Nummer erzwingen, wie es im Abschnitt [Notenschlüssel], Seite 12 beschrieben wird. Die Note, die rechts von jedem Schlüssel gesetzt ist, zeigt das `c'` in Bezug zu dem jeweiligen Schlüssel.

Beschreibung	**Unterstützte Schlüssel**	**Beispiel**
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	`petrucci-f`	
Mensuraler G-Schlüssel im Petrucci-Stil	`petrucci-g`	
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`	
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`	

Moderner Stil bedeutet: „Wie in modernen Editionen von transkribierter Mensuralmusik benutzt.“

Petrucci-Stil bedeutet: „Inspiriert von Drucken, die der berühmte Notensetzer Petrucci (1466–1539) produziert hat.“

Historischer Stil bedeutet: „Wie in anderen als Petruccis Editionen gedruckt oder geschrieben wurde.“

Editio XXX-Stil bedeutet: „Wie in der Editio XXX gedruckt wird.“

Petrucci verwendete C-Schlüssel mit unterschiedlich balanciertem vertikalen Balken auf der linken Seite in Abhängigkeit davon, auf welcher Notenlinie der Schlüssel gesetzt wird.

Siehe auch

In diesem Handbuch: siehe [Notenschlüssel], Seite 12.

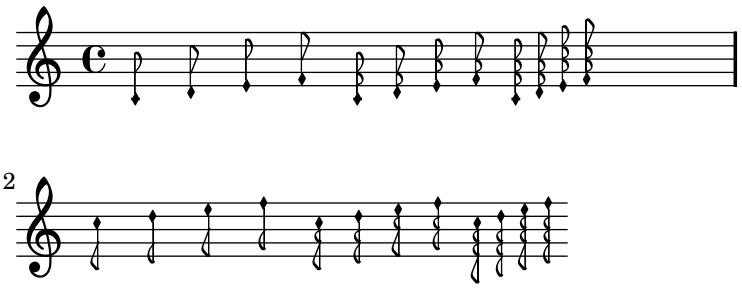
Bekannte Probleme und Warnungen

Der mensurale G-Schlüssel ist als Petrucci-G-Schlüssel deklariert.

Fähnchen Alter Musik

Mit der Fähnchen-(`flag-style`)-Eigenschaft der graphischen Objekte „Hals“ (Abschnitt “Stem” in *Referenz der Interna*) können auch Fähnchen passend zu den Notenköpfen der Alten Musik gesetzt werden. Neben dem Standardstil (`default`) ist auch ein mensuraler Stil (`mensural`) unterstützt.

```
\override Stem #'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 die „neomensurale“ Notation. Insofern sollte für das Incipit bei der Transkription mensuraler Musik der Standardstil benutzt werden. Für die Notation des Gregorianischen Chorals gibt es keine Fähnchen.

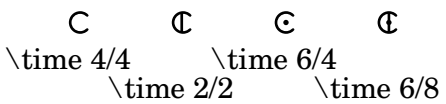
Bekannte Probleme und Warnungen

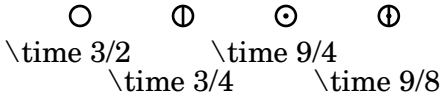
Die Positionierung der Fähnchen an den Hälsen ist leicht verschoben seit einer Änderung in einer frühen 2.3.x-Version.

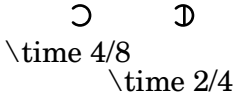
Vertikale Ausrichtung der Fähnchen an einer Notenlinie geht von der Annahme aus, dass der Hals entweder genau auf einer Notenlinie oder genau zwischen zwei Notenlinien endet. Das ist aber nicht unbedingt immer der Fall, weil LilyPond komplizierte Methoden zur Ermittlung des besten Layouts verwendet. Diese Methoden sollten aber eigentlich nicht zur Notation von mensuraler Musik eingesetzt werden.

Taktangaben Alter Musik

LilyPond besitzt grundlegende Unterstützung für mensurale Taktangaben. 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.

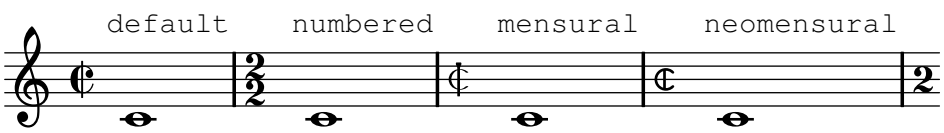


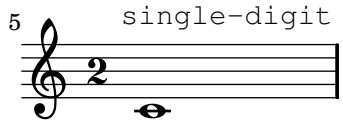




Mit der `style`-Eigenschaft des Objektes Abschnitt “TimeSignature” in *Referenz der Interna* können die Taktarten angewählt werden. Unterstützte Stile sind: `neomensural` und `mensural`. In der Tabelle oben wurde der neomensurale Stil verwendet. Dieser Stil ist geeignet, um im Incipt von transkribierter Mensuralmusik eingesetzt zu werden. Der mensurale Stil dagegen ahmt die Form historischer Druck des 16. Jahrhunderts nach.

Im folgenden Beispiel sind die unterschiedlichen Stile dargestellt.





Siehe auch

In diesem Handbuch: [Taktangabe], Seite 45 bietet eine allgemeine Übersicht über den Einsatz von Taktangaben.

Bekannte Probleme und Warnungen

Die Verhältnisse der Notenwerte ändern sich nicht, wenn die Taktart (Mensur) gewechselt wird. Zum Beispiel muss 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.

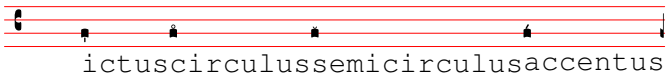
Das `old6/8alt`-Symbol (ein alternatives Symbol für 6/8) kann nicht mit dem Befehl `\time` angesprochen werden. Verwenden Sie anstatt dessen eine Textbeschriftung (`\markup`).

2.8.3 Zusätzliche Notenzeichen

Artikulationszeichen Alter Musik

Zusätzlich zu den Standardartikulationszeichen, wie sie im Abschnitt [Artikulationszeichen und Verzierungen], Seite 83 beschrieben werden, werden auch Artikulationszeichen für die Alte Musik zur Verfügung gestellt. Diese sind darauf hin geformt, dass sie mit der Notation des Editio Vaticana-Stils verwendet werden können.

```
\include "gregorian.ly"
\score {
  \new VaticanaVoice {
    \override TextScript #'font-family = #'typewriter
    \override TextScript #'font-shape = #'upright
    \override Script #'padding = #-0.1
    a\ictus_"ictus" \break
    a\circulus_"circulus" \break
    a\semicirculus_"semicirculus" \break
    a\accentus_"accentus" \break
    \[ a_"episem" \episemInitium \pes b \flexa a b \episemFinis \flexa a \]
  }
}
```





Bekannte Probleme und Warnungen

Einige Artikulationszeichen sind vertikal zu dich an den entsprechenden Notenköpfen gesetzt.

Die Episem-Linie wird in vielen Fällen nicht angezeigt. Wenn sie angezeigt wird, ist das rechte Ende der Episem-Linie oft zu weit rechts.

Custodes

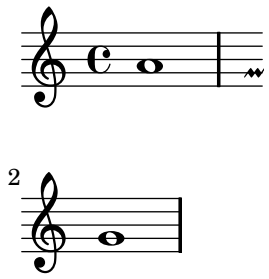
Ein *custos* (Plural: *custodes*; Lateinisch: „Beschützer“) ist ein Symbol, das am Ende jedes Notensystems erscheint. Es nimmt die Tonhöhe der ersten Note der nächsten Zeile vorweg und hilft damit dem Vortragenden, die Zeilenwechsel während der Vorführung zu bewältigen.

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 die *editio vaticana*. LilyPond stellt unterschiedliche Custos-Symbole für die unterschiedlichen Notationsstile zur Verfügung.

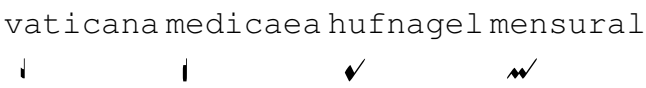
Damit Custodes angezeigt werden, muss ein Abschnitt “Custos_engraver” in *Referenz der Interna* im Abschnitt “Staff” in *Referenz der Interna*-Kontext gefordert werden. Der Aufruf folgt im Rahmen des Layout-Kontextes, wie das folgende Beispiel zeigt.

```
\layout {
  \context {
    \Staff
    \consists Custos_engraver
    Custos \override #'style = #'mensural
  }
}
```

Das Ergebnis sieht ungefähr folgendermaßen aus:



Das Custos-Zeichen wird von der `style`-Eigenschaft ausgewählt. Die unterstützten Stile sind: `vaticana`, `medicaea`, `hufnagel` und `mensural`. Sie werden im folgenden Fragment demonstriert.



Siehe auch

Programmreferenz: Abschnitt “Custos” in *Referenz der Interna*.

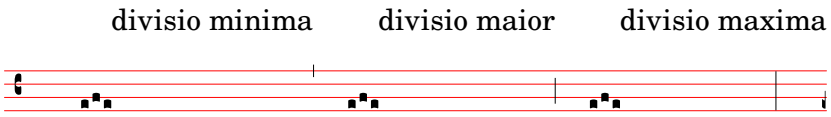
Beispiele: Abschnitt “Ancient notation” in *Schnipsel*.

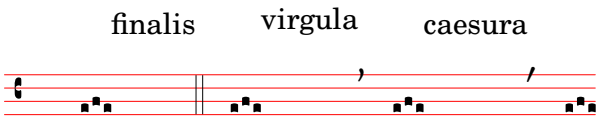
Divisiones

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 [Atemzeichen], Seite 93. 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.

Divisiones können benutzt werden, indem die Datei ‘`gregorian-init.ly`’ in die Quelldatei eingefügt wird. Hier sind die entsprechenden Definitionen schon abgelegt, so dass es genügt, die

Befehle `\divisioMinima`, `\divisioMaior`, `\divisioMaxima` und `\finalis` an den entsprechenden Stellen zu schreiben. Einige Editionen verwenden eine *virgula* oder *caesura* anstelle der divisio minima. Darum findet sich in der Datei '`gregorian-init.ly`' auch eine Definition für `\virgula` und `\caesura`.





Vordefinierte Befehle

`\virgula`, `\caesura`, `\divisioMinima`, `\divisioMaior`, `\divisioMaxima`, `\finalis`.

Siehe auch

In diesem Handbuch: [Atemzeichen], Seite 93.

Programmreferenz: Abschnitt "BreathingSign" in *Referenz der Interna*.
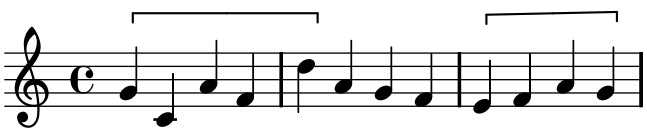
Beispiele: Abschnitt "Winds" in *Schnipsel*.

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 eingegeben, indem die dazugehörigen Noten zwischen `\[` und `\]` eingeschlossen werden. Einige Ligaturstile benötigen zusätzliche Syntax für eine bestimmte Ligatur. In der Standardeinstellung setzt der Abschnitt "LigatureBracket" in *Referenz der Interna* ganz einfach eckige Klammern über die Noten der Ligatur.

```
\transpose c c' {
  \[ g c a f d' \]
  a g f
  \[ e f a g \]
}
```



Um einen gestimmten Ligaturstil auszuwählen, muss ein entsprechender Ligatur-Engraver zum Stimmkontext hinzugefügt werden, wie in den folgenden Abschnitten erklärt wird. Nur weiße Mensuralligaturen sind unterstützt – mit Einschränkungen.

Bekannte Probleme und Warnungen

Ligaturen benötigen von klassischer Notation unterschiedliche Platzaufteilung, was sie aber noch nicht können. Darum ist fast immer zu viel Platz zwischen Ligaturen und Zeilenumbrüche sind ungenügend. Text lässt sich auch nicht richtig an Ligaturen ausrichten.

Akzidentien 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 (`\[ musik. Ausdr. \]`). Für die Konsistenz soll dies geändert werden in den Postfix-Stil (`Note\[ ... Note\]`). Alternativ kann die Datei '`gregorian-init.ly`' eingefügt werden, die eine Scheme-Funktion

`\ligature` *musik. Ausdr.*

mit der selben Wirkung zur Verfügung stellt und stabil zu sein scheint.

Weiße Mensuralligaturen

Begrenzte Unterstützung für Ligaturen der weißen Mensuralnotation.

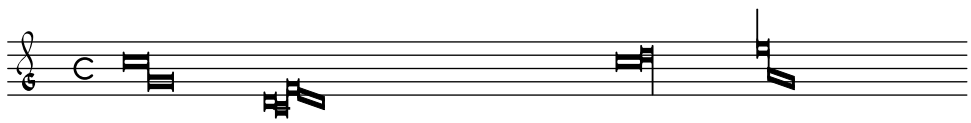
Um weiße Mensuralligaturen zu benutzen, muss innerhalb des Layout-Blocks im Abschnitt "Voice" in *Referenz der Interna*-Kontext der Abschnitt "Mensural_ligature_engraver" in *Referenz der Interna* aktiviert werden und gleichzeitig der Abschnitt "Ligature_bracket_engraver" in *Referenz der Interna* (der die Klammern über den Noten setzt) entfernt werden, wie im Beispiel.

```
\layout {
  \context {
    \Voice
    \remove Ligature_bracket_engraver
    \consists Mensural_ligature_engraver
  }
}
```
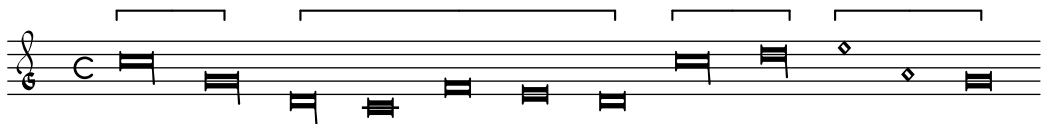
Zusätzlich zu diesen Einstellungen gibt es keine eigenen Befehle, die die Form einer Ligatur bestimmen. Die Form wird vielmehr aus Tonhöhen und Tondauern der in Klammern gesetzten Noten geschlossen. Diese Herangehensweise erfordert einige Eingewöhnung, hat aber den großen Vorteil, dass der musikalische 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.

Eine Datei kann zum Beispiel so aussehen:

```
\set Score.timing = ##f
\set Score.defaultBarType = "empty"
\override NoteHead #'style = #'neomensural
\override Staff.TimeSignature #'style = #'neomensural
\clef "petrucci-g"
\[ c'\maxima g \]
\[ d\longa c\breve f e d \]
\[ c'\maxima d'\longa \]
\[ e'1 a g\breve \]
```



Wenn der Abschnitt "Ligature_bracket_engraver" in *Referenz der Interna* nicht durch den Abschnitt "Mensural_ligature_engraver" in *Referenz der Interna* ersetzt wird, werden die Noten wie folgt ausgegeben:



Bekannte Probleme und Warnungen

Die horizontale Positionierung ist sehr schlecht.

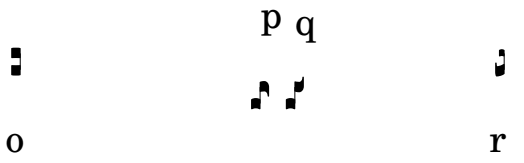
Ligaturen der gregorianischen Quadratnotation

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.

Die folgende Tabelle enthält die erweiterte Neumenliste des zweiten Bands des Antiphonale Romanum (*Liber Hymnarius*), 1983 von den Mönchen von Solesmes herausgegeben.

Neuma aut Neumarum Elementa	**Figurae Rectae**	**Figurae Liquescentes Auctae**	**Figurae Liquescentes Deminutae**
`1. Punctum`	a b	c d e	f
`2. Virga`	g		
`3. Apostropha vel Stropha`	h	i	
`4. Oriscus`	j		
`5. Clivis vel Flexa`	k	l m	n

6. Podatus vel Pes



7. Pes Quassus



8. Quilisma Pes



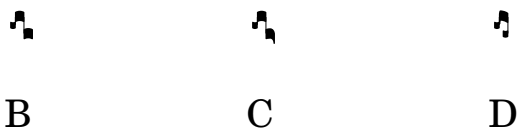
9. Podatus Initio Debilis



10. Torculus



11. Torculus Initio Debilis



B C D

12. Porrectus



E F G

13. Climacus



H I J

14. Scandicus



K L M

15. Salicus



N O

16. Trigonus



P

Anders als in den meisten Neumennotationssystemen zeigt der Quellcode von LilyPond nicht das typographische Aussehen der Ligatur an, sondern deren musikalischen Inhalt. Der Code `\[ a \pes b \flexa g \]` etwa ergibt einen Torculus, der aus drei Punctum-Köpfen besteht, während `\[ a \flexa g \pes b \]` einen Porrectus mit einer gekrümmten Flexa und einem einzelnen Punctum ausgibt. Es gibt also keinen Befehl, der explizit eine gekrümmte Flexa setzen würde; die Entscheidung, wann diese gesetzt werden soll hängt vielmehr davon ab, welcher musikalische Inhalt dargestellt werden soll. Der Sinn dieser Herangehensweise ist es, den Inhalt von der graphischen Repräsentation zu trennen. Auf diese Art kann mit der gleichen Eingabe ein anderer gregorianischer Stil gesetzt werden, ohne die Notation zu verändern.

Die folgende Tabelle zeigt Code-Fragmente, mit denen die Ligaturen der vorigen Tabelle erstellt werden können. Der Buchstabe in der ersten Spalte jeder Zeile der unteren Tabelle zeigt an, auf welche Ligatur in der vorigen Tabelle sie sich bezieht. In der zweiten Spalte erscheint die Bezeichnung der Ligatur. Die dritte Spalte enthält das Fragment, mit dem die Ligatur erzeugt wurde, wobei `g`, `a` und `b` als Beispieltonhöhen eingesetzt werden.

#	**Name**	**Input Language**
a	Punctum	`\[ b \]`
b	Punctum Inclinatum	`\[ \inclinatum b \]`
c	Punctum Auctum Ascendens	`\[ \auctum \ascendens b \]`
d	Punctum Auctum Descendens	`\[ \auctum \descendens b \]`
e	Punctum Inclinatum Auctum	`\[ \inclinatum \auctum b \]`
f	Punctum Inclinatum Parvum	`\[ \inclinatum \deminutum b \]`
g	Virga	`\[ \virga b \]`
h	Stropha	`\[ \stropha b \]`
i	Stropha Aucta	`\[ \stropha \auctum b \]`
j	Oriscus	`\[ \oriscus b \]`
k	Clivis vel Flexa	`\[ b \flexa g \]`
l	Clivis Aucta Descendens	`\[ b \flexa \auctum \descendens g \]`
m	Clivis Aucta Ascendens	`\[ b \flexa \auctum \ascendens g \]`
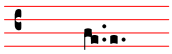
n	Cephalicus	`\[ b \flexa \deminutum g \]`
o	Podatus vel Pes	`\[ g \pes b \]`
p	Pes Auctus Descendens	`\[ g \pes \auctum \descendens b \]`
q	Pes Auctus Ascendens	`\[ g \pes \auctum \ascendens b \]`
r	Epiphonus	`\[ g \pes \deminutum b \]`
s	Pes Quassus	`\[ \oriscus g \pes \virga b \]`
t	Pes Quassus Auctus Descendens	`\[ \oriscus g \pes \auctum \descendens b \]`
u	Quilisma Pes	`\[ \quilisma g \pes b \]`
v	Quilisma Pes Auctus Descendens	`\[ \quilisma g \pes \auctum \descendens b \]`
w	Pes Initio Debilis	`\[ \deminutum g \pes b \]`
x	Pes Auctus Descendens Initio Debilis	`\[ \deminutum g \pes \auctum \descendens b \]`
y	Torculus	`\[ a \pes b \flexa g \]`
z	Torculus Auctus Descendens	`\[ a \pes b \flexa \auctum \descendens g \]`
A	Torculus Deminutus	`\[ a \pes b \flexa \deminutum g \]`
B	Torculus Initio Debilis	`\[ \deminutum a \pes b \flexa g \]`
C	Torculus Auctus Descendens Initio Debilis	`\[ \deminutum a \pes b \flexa \auctum \descendens g \]`
D	Torculus Deminutus Initio Debilis	`\[ \deminutum a \pes b \flexa \deminutum g \]`
E	Porrectus	`\[ a \flexa g \pes b \]`
F	Porrectus Auctus Descendens	`\[ a \flexa g \pes \auctum \descendens b \]`
G	Porrectus Deminutus	`\[ a \flexa g \pes \deminutum b \]`
H	Climacus	`\[ \virga b \inclinatum a \inclinatum g \]`
I	Climacus Auctus	`\[ \virga b \inclinatum a \inclinatum \auctum g \]`
J	Climacus Deminutus	`\[ \virga b \inclinatum a \inclinatum \deminutum g \]`
K	Scandicus	`\[ g \pes a \virga b \]`
L	Scandicus Auctus Descendens	`\[ g \pes a \pes \auctum \descendens b \]`

M	Scandicus Deminutus	`\[ g \pes a \pes \deminutum b \]`
N	Salicus	`\[ g \oriscus a \pes \virga b \]`
O	Salicus Auctus Descendens	`\[ g \oriscus a \pes \auctum \descendens b \]`
P	Trigonus	`\[ \stropha b \stropha b \stropha a \]`

Die Ligaturen dieser Liste dienen als begrenzter, aber doch repräsentativer Vorrat an Ligaturbeispielen des Gregorianischen Chorals. Innerhalb der Ligaturbegrenzungen `\[` und `\]` kann jedoch problemlos jede nur mögliche Anzahl an Noten gesetzt werden, und Präfixe wie `\pes`, `\flexa`, `\virga`, `\inclinatum`, usw können nach Belieben gemischt werden. Die Regeln, die der Konstruktion der Ligaturen in den Tabellen zugrunde liegen, werden entsprechend angepasst. Auf diese Weise können unendlich viele Ligaturen gesetzt werden.

Augmentum-Punkte, auch *morae* genannt, werden mit dem Befehl `\augmentum` hinzugefügt. `\augmentum` ist allerdings als eigene musikalische Funktion gebaut und nicht als ein Notenpräfix. Insofern hat der Befehl in diesem Kontext: `\augmentum \virga c` keine sichtbaren Auswirkungen. Erst mit `\virga \augmentum c` oder `\augmentum {\virga c}` funktionieren beide Befehle. Es ist auch möglich, mit `\augmentum {a g}` die Schreibweise `\augmentum a \augmentum g` abzukürzen.

```
\include "gregorian.ly"
\score {
  \new VaticanaVoice {
    \[ \augmentum a \flexa \augmentum g \]
    \augmentum g
  }
}
```



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.

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.

2.8.4 Vordefinierte Umgebungen

Gregorianische Gesangs-Kontexte

Die vordefinierten Kontexte `VaticanaVoiceContext` (für eine gregorianische Stimme) und `VaticanaStaffContext` (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:

```
\include "gregorian.ly"
\score {
  <<
    \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
    }
  >>
}
```



San- ctus, San- ctus,



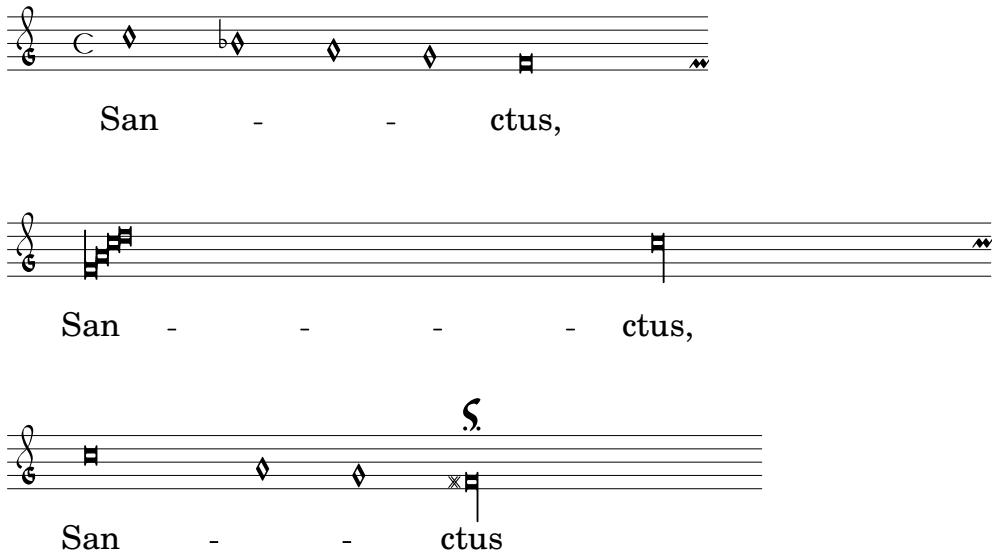
San- ctus

Mensural-Kontexte

Die vordefinierten Kontexte `MensuralVoiceContext` und `MensuralStaffContext` können eingesetzt werden, um ein Stück in Mensuralnotations zu schreiben. Die Kontexte initialisieren alle relevanten Eigenschaften der Kontexte und graphischen Objekte, so dass unmittelbar mit der Notation begonnen werden kann. Siehe das folgende Beispiel:

```
\score {
  <<
    \new MensuralVoice = "Discantus" \transpose c c' {
      \override Score.BarNumber #'transparent = ##t {
        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
    }
  >>
}
```



2.8.5 Transkription Alter Musik

Alte und moderne Edition aus einer Quelldatei

Incipite

Mensurstriche

Gregorianischen Choral transkribieren
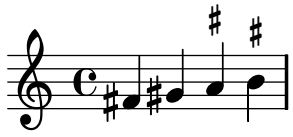
2.8.6 Herausgeberische Anmerkungen

Vorgeschlagene Versetzungszeichen

In der europäischen Musik vor 1600 wurden vom Sänger oftmals chromatische Alterationen erwartet, die nicht notiert wurden. Diese Praxis wird „Musica Ficta“ genannt. In modernen Transkripition werden die Versetzungszeichen üblicherweise oberhalb der Noten gesetzt.

Unterstützung für solche empfohlenen Versetzungszeichen ist implementiert und kann aktiviert werden, indem die Eigenschaft `suggestAccidentals` auf wahr gesetzt wird. Siehe auch das Beispiel.

```
fis gis
\set suggestAccidentals = ##t
ais bis
```



Siehe auch

Programmreferenz: Abschnitt “Accidental_engraver” in *Referenz der Interna*-Setzer und das Abschnitt “AccidentalSuggestion” in *Referenz der Interna*-Objekt.

Notation barocker Rhythmen

2.9 Weltmusik

Dieser Abschnitt soll Besonderheiten der Notation aufzeigen, die insbesondere relevant sind, um Musik nicht-westlicher Tradition zu notieren.

2.9.1 Arabische Musik

Dieser Abschnitt zeigt Möglichkeiten, wie arabische Musik notiert werden kann.

References for Arabic music

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 [Notenbezeichnungen in anderen Sprachen], Seite 7.
- Zusätzliche Taktarten können erstellt werden, siehe [Tonartbezeichnung], Seite 15.
- Komplexe Taktarten erfordern evtl., dass Noten manual gruppiert werden, wie gezeigt in [Manuelle Balken], Seite 66.
- *Takasim*, rhythmisch freie Improvisationen, können ohne Taktlinien notiert werden, siehe hierzu [Musik ohne Metrum], Seite 48.

Siehe auch

Notationsreferenz: [Notenbezeichnungen in anderen Sprachen], Seite 7, [Tonartbezeichnung], Seite 15, [Manuelle Balken], Seite 66.
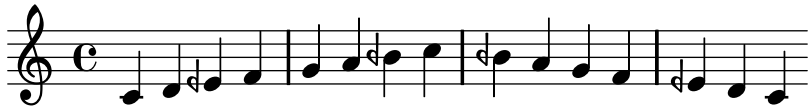
Schnipsel: Abschnitt “World music” in *Schnipsel*.

Arabic note names

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, wie gezeigt in [Notenbezeichnungen in anderen Sprachen], Seite 7.
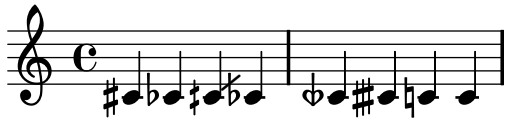
Hier ein Beispiel der arabischen *rast*-Tonleiter:

```
\include "arabic.ly"
\relative do' {
  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 do' {
  \set Staff.extraNatural = ##f
  dod dob dosd \dwn dob dobsb dodsd do do
}
```



Siehe auch

Notationsreferenz: [Notenbezeichnungen in anderen Sprachen], Seite 7.

Schnipsel: Abschnitt “World music” in *Schnipsel*.

Arabic key signatures

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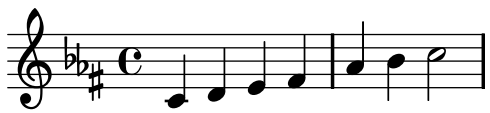
Der üblicherweise benutzte `\key`-Befehl setzt die `keySignature`-Eigenschaft im `Staff`-Kontext.

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:

```
\relative c' {
  \set Staff.keySignature = #`(((0 . 3) . ,SHARP)
                              ((0 . 5) . ,FLAT)
                              ((0 . 6) . ,FLAT))
  c4 d e fis
  aes4 bes c2
}
```



Siehe auch

Notationsreferenz: [Tonartbezeichnung], Seite 15.

Handbuch zum Lernen: Abschnitt “Accidentals and key signatures” in *Handbuch zum Lernen*.

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

Schnipsel: Abschnitt “World music” in *Schnipsel*, Abschnitt “Pitches” in *Schnipsel*.

Arabic time signatures

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.
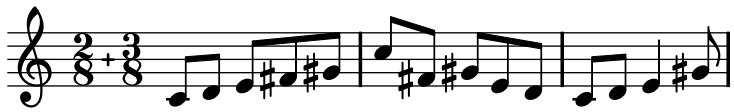
Ausgewählte Schnipsel

Zusammengesetzte Taktarten

Ungerade Taktarten werden (wie etwa "5/8") werden oft als zusammengesetzte Taktarten interpretiert (bspw. "3/8 + 2/8"), in welchen zwei oder mehr Teiltakte unterschieden werden. LilyPond kann derartige Noten produzieren, indem entsprechende Taktarten gesetzt werden und die automatische Bebalkung angepasst wird.

```
#(define ((compound-time one two num) grob)
  (grob-interpret-markup grob
    (markup #:override '(baseline-skip . 0) #:number
      (#:line (
          (#:column (one num))
          #:vcenter "+"
          (#:column (two num))))
      )))

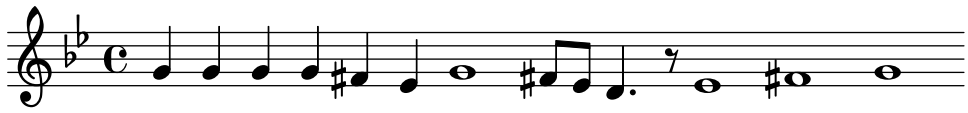
\relative c' {
  \override Staff.TimeSignature #'stencil = #(compound-time "2" "3" "8")
  \time 5/8
  #(override-auto-beam-setting '(end 1 8 5 8) 1 4)
  c8 d e fis gis
  c8 fis, gis e d
  c8 d e4 gis8
}
```



*Arabic improvisation*For improvisations or *taqasim* which are temporarily free, the time signature can be omitted and `\cadenzaOn` can be used. Adjusting the accidental style might be required, since the absence of bar lines will cause the accidental to be marked only once. Here is an example of what could be the start of a *hijaz* improvisation:

```
\include "arabic.ly"

\relative sol' {
  \key re \kurd
  #(set-accidental-style 'forget)
  \cadenzaOn
  sol4 sol sol sol fad mib sol1 fad8 mib re4. r8 mib1 fad sol
}
```



Siehe auch

Notationsreferenz: [Manuelle Balken], Seite 66, [Automatische Balken], Seite 55, [Musik ohne Metrum], Seite 48, [Automatische Versetzungszeichen], Seite 19, [Einstellung von automatischen Balken], Seite 57, [Taktangabe], Seite 45.
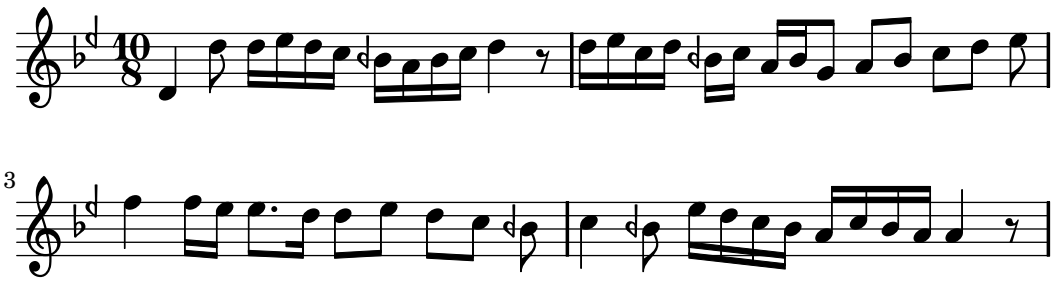
Schnipsel: Abschnitt “World music” in *Schnipsel*.

Arabic music example

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 {
  \relative re' {
    \set Staff.extraNatural = ##f
    \set Staff.autoBeaming = ##f
    \key re \bayati
    \time 10/8

    re4 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
  }
  \header {
    title = "Semai Muhayer"
    composer = "Jamil Bek"
  }
}
```



Siehe auch

Schnipsel: Abschnitt “World music” in *Schnipsel*

Weitere Literatur

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:

   - http://www.maqamworld.com/

- http://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.

   Oud-Lehrbücher folgender Autoren enthalten Beispiele vor allem türkischer und arabischer Kompositionen:

   - Charbel Rouhana
   - George Farah
   - Ibrahim Ali Darwish Al-masri

3 Allgemeine Eingabe und Ausgabe

Dieses Kapitel erklärt allgemeine Fragen zur Eingabe und Ausgabe von Notation mit LilyPond und weniger direkte Fragen der Notation.

3.1 Eingabestruktur

Das hauptsächliche Eingabeformat von LilyPond sind Textdateien. Üblicherweise werden diese Dateien mit der Endung `.ly` versehen.

3.1.1 Struktur einer Partitur

Eine `\score`-Umgebung muss einen einzelnen musikalischen Ausdruck beinhalten, der durch geschweifte Klammern begrenzt wird:
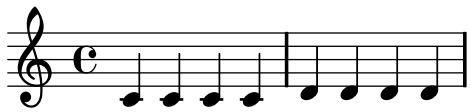
```
\score {
...
}
```

Achtung: Es darf **nur ein** äußerer musikalischer Ausdruck in der `\score`-Umgebung geschrieben werden, und er **muss** von geschweiften Klammern umgeben sein.

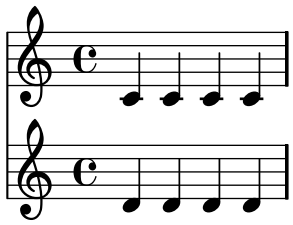
Dieser einzelne musikalische Ausdruck kann beliebige Größe annehmen und andere musikalische Ausdrücke von beliebiger Komplexität beinhalten. Alle diese Beispielse sind musikalische Ausdrücke:

```
{ 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 3.1.3 [Die Dateistruktur], Seite 305.) 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.

Siehe auch

Handbuch zum Lernen: Abschnitt “Working on input files” in *Handbuch zum Lernen*, Abschnitt “Music expressions explained” in *Handbuch zum Lernen*, Abschnitt “Score is a (single) compound musical expression” in *Handbuch zum Lernen*.

3.1.2 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 {
  ..
}
```

Wenn Sie aber mehrere Ausgabedateien aus einer einzigen `.ly`-Datei erstellen wollen, können Sie mehrere `\book`-Umgebungen notieren. Wenn Sie keine `\book`-Umgebung in Ihrer Datei angeben, interpretiert LilyPond die gesamte Datei als eine große `\book`-Umgebung (siehe auch Abschnitt 3.1.3 [Die Dateistruktur], Seite 305. 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 { ... }
  ...
}
```

3.1.3 Die Dateistruktur

Eine `.ly`-Datei kann eine belibige Anzahl an Ausdrücken auf der obersten Ebene beinhalten, wobei ein Ausdrück 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 derarte Definition desselben Typs angegeben wird, hat die spätere Vorrang.
- 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 { }
  \header { }
  }
  ```

  Dieses Verhalten kann verändert werden, indem die Variable `toplevel-music-handler` auf der oberseten 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 Beizeichnung 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 `%{ .. %}`.

Siehe auch

Hanbuch zum Lernen: Abschnitt “How LilyPond input files work” in *Handbuch zum Lernen*.

3.2 Titel

Fast alle gedruckten Noten beinhalten einen Titel und den Namen des Komponisten, teilweise wird auch noch sehr viel mehr Information zur Verfügung gestellt.

3.2.1 Titel erstellen

Überschriften können für jede `\score`-Umgebung erstellt werden, sowohl für die gesamte Datei (oder eine `\book`-Umgebung) als auch für einzelne Buchabschnitte (innerhalb einer `\bookpart`-Umgebung).

Der Inhalt der Titelei wird aus der `\header`-Umgebung übernommen. Die `\header`-Umgebung eines Buches unterstützt folgende Felder:

`dedication`
: Die Widmung der Noten, wird auf oben auf der ersten Seite gesetzt.

`title`
: Die Überschrift der Noten, wird unter der Widmung zentriert gesetzt.

`subtitle`
: Untertitel, zentriert unter der Überschrift.

`subsubtitle`
: Unteruntertitel, zentriert unter dem Untertitel.

`poet`
: Name des Dichters, linksbündig unter dem Unteruntertitel.

`instrument`
: Bezeichnung des Instruments, zentriert unter dem Unteruntertitel. Auch oben auf der Seite zentriert (andere als erste Seite).

`composer`
: Name des Komponisten, rechtsbüngig unter dem Unteruntertitel.

`meter`
: Metrum, linksbündig unter dem Dichter.

`arranger`
: Name des Bearbeiters/Arrangeurs, rechtsbündig unter dem Komponisten.

`piece`
: Bezeichnung des Stückes, linksbündig unter dem Metrum.

`opus` Bezeichnung des Opus, rechtsbündig unter dem Bearbeiter.

`breakbefore`
Hiermit beginnt der Titel auf einer neuen Steite. (Kann die Werte ##t (wahr) oder ##f (falsch) haben.)

`copyright`
Anzeige eines Copyright, zentriert unten auf der ersten Seite. Um das Copyright-Symbol zu notieren, siehe Abschnitt 3.3.3 [Zeichenkodierung], Seite 320.

`tagline` Zentriert unten auf der letzten Seite. Enthält standardmäßig: „Music engraving by LilyPond (*version*)—www.lilypond.org“

Hier eine Demonstration der möglichen Felder. Beliebige Formatierungsbefehle für Textbeschriftung können in der Titelei eingesetzt werden. Siehe hierzu auch Abschnitt 1.8.2 [Text formatieren], Seite 170.

```
\paper {
  line-width = 9.0\cm
  paper-height = 10.0\cm
}

\book {
  \header {
    dedication = "mir gewidmet"
    title = \markup \center-column { "Titel erste Zeile" "Titel zweite Zeile, länger" }
    subtitle = "Untertitel"
    subsubtitle = #(string-append "Unteruntertitel LilyPond-Version "
(lilypond-version))
    poet = "Dichter"
    composer =  \markup \center-column { "Komponist" \small "(1847-1973)" }
    texttranslator = "Übersetzer"
    meter = \markup { \teeny "m" \tiny "e" \normalsize "t" \large "r" \huge
"um" }
    arranger = \markup { \fontsize #8.5 "Be" \fontsize #2.5 "ar" \fontsize
#-2.5 "be" \fontsize #-5.3 "i" \fontsize #7.5 "ter" }
    instrument = \markup \bold \italic "Instrument"
    piece = "Stück"
  }

  \score {
    { c'1 }
    \header {
      piece = "Stück zwei"
      opus = "Opus1"
    }
  }
  \markup {
      und jetzt...
  }
  \score {
    { c'1 }
    \header {
      piece = "Stück2"
      opus = "Opus2"
```

```
    }
  }
}
```

mir gewidmet

Titel erste Zeile
Titel zweite Zeile, länger
Untertitel

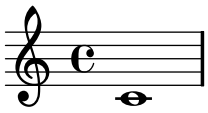
Unteruntertitel LilyPond-Version 2.12.1

Dichter ***Instrument*** Komponist

(1847-1973)

m e t r um Be ar be i ter

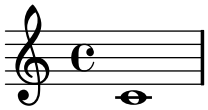
Stück zwei Opus1



und jetzt...

2 ***Instrument***

Stück2 Opus2



Music engraving by LilyPond 2.12.1—www.lilypond.org

Wie schon oben gezeigt, können mehrfache `\header`-Umgebungen eingesetzt werden. Wenn das gleiche Feld in mehreren Umgebungen, wird die letzte vorkommende Version benutzt. Hier ein kurzes Beispiel:
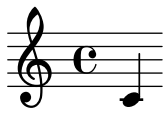
```
\header {
  composer = "Komponist"
}
\header {
```

```
  piece = "Stück"
}
\score {
  \new Staff { c'4 }
  \header {
    piece = "Neues Stück"  % überschreibt die die vorige Definition
  }
}
```

Wenn `\header` innerhalb der `\score`-Umgebung definiert wird, wird normalerweise nur die Information von `piece` und `opus` ausgegeben. Musikalische Ausdrücke innerhalb von `\score` müssen vor `\header` gesetzt werden.

```
\score {
  { c'4 }
  \header {
    title = "title"  % nicht gedruckt
    piece = "piece"
    opus = "opus"
  }
}
```

piece opus



Dieses Verhalten kann verändert werden (sodass alle Angaben aus der überschrift gesetzt werden, wenn sich `\header` innerhalb von `\score` befindet), indem man schreibt:

```
\paper{
  print-all-headers = ##t
}
```

Die Standardfußzeile ist leer mit Ausnahme der ersten Seite, auf der das `copyright`-Feld aus der `\header`-Umgebung eingefügt wird, und die letzte Seite, auf der das `tagline`-Feld eingefügt wird. Der Standardinhalt von `tagline` ist „Music engraving by LilyPond (*version*)—www.lilypond.org“. Gut gesetzte Noten werben sehr effektiv für LilyPond, darum bitten wir darum, diese Zeile stehen zu lassen, wenn es möglich ist.

Ein Titelfeld kann vollständig entfernt werden, indem es auf falsch gesetzt wird:

```
\header {
  tagline = ##f
  composer = ##f
}
```

3.2.2 Eigene Titel

Kompliziertere Anpassungen können vorgenommen werden, indem die folgenden Variablen innerhalb der `\paper`-Umgebung geändert werden. Die Init-Datei ‘`../ly/titling-init.ly`’ enthält das Standardverhalten.

`bookTitleMarkup`
: Das ist die Überschrift, die für das gesamte Dokument gilt. Üblicherweise wird hier der Komponist und die Überschrift des Werkes genannt.

`scoreTitleMarkup`
: Das ist die Überschrift, die vor jede `\score`-Umgebung gesetzt wird. Überlicherweise wird hier etwa die Bezeichnung eines Satzes notiert (im `piece`-Feld).

`oddHeaderMarkup`
: Das ist der Seitenkopf für ungerade Seiten.

`evenHeaderMarkup`
: Das ist der Seitenkopf für gerade Seiten. Wenn undefiniert, wird der ungerade Seitenkopf eingesetzt.

  Standardmäßig werden die Kopfzeilen so definiert, dass die Seitennummer sich außen befindet und das Instrument zentriert gesetzt wird.

`oddFooterMarkup`
: Das ist die Fußzeile für ungerade Seiten.

`evenFooterMarkup`
: Das ist die Fußzeile für gerade Seiten. Wenn undefiniert, wird die ungerade Fußzeile eingesetzt.

  Standardmäßig wird in der Fußzeile auf der ersten Seite das Copyright und auf der letzten Seite die Tag-Zeile gesetzt.

Die folgende Definition setzt die Überschrift linksbündig und den Komponisten rechtsbündig auf einer einzelnen Zeile:

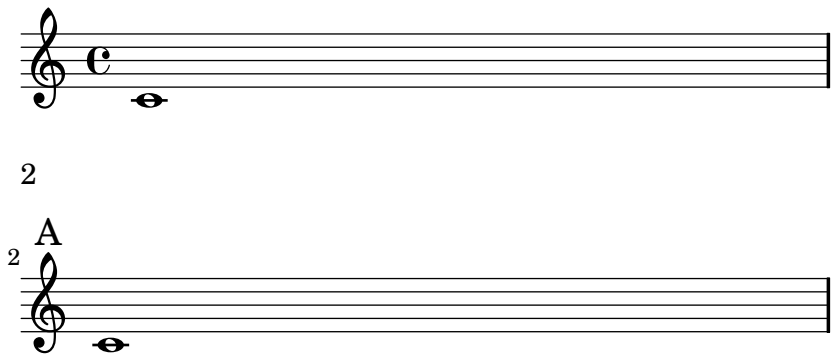
```
\paper {
  bookTitleMarkup = \markup {
   \fill-line {
     \fromproperty #'header:title
     \fromproperty #'header:composer
   }
  }
}
```

3.2.3 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'
    }
  }

  \markup { Die erste Partitur fäng auf Seite \page-ref #'ErstePartitur "0" "?" an.}
  \markup { Zeichen A befindet sich auf Seite \page-ref #'ZeichenA "0" "?". }
}
```



Die erste Partitur fäng 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 unbekann 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.

`\label \page-ref`

Vordefinierte Befehle

3.2.4 Inhaltsverzeichnis

Ein Inhaltsverzeichnis kann eingefügt werden mit dem Befehl `\markuplines \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.

```
\markuplines \table-of-contents
\pageBreak

\tocItem \markup "Erste Partitur"
\score {
  {
    c'  % ...
    \tocItem \markup "Ein bestimmter Punkt innerhalb der ersten Partitur"
    d'  % ...
  }
}

\tocItem \markup "zweite Partitur"
\score {
  {
    e' % ...
  }
}
```

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 (parser location 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

Siehe auch

Installierte Dateien: '`../ly/toc-init.ly`'.

Vordefinierte Befehle

`\table-of-contents`, `\tocItem`.

3.3 Arbeiten an Eingabe-Dateien

3.3.1 LilyPond-Dateien einfügen

Ein größeres Projekt kann in einzelne Dateien aufteteilt 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 3.3.2 [Verschiedene Editionen aus einer Quelldatei], Seite 315.

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.

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 "Other sources of 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 Komandozeile mit Option aufgerufen wird: `lilypond --verbose`. Hiermit wird neben anderer Information auch eine Liste and Pfaden und Dateien aufgeführt, die LilyPond benutzt. Die wichtigeren Dateien werden im Abschnitt Abschnitt "Other sources of information" in *Handbuch zum Lernen* besprochen. Diese Dateien können verändert werden, aber Änderungen gehen verloren, wenn eine neue LilyPond-Version installiert wird.

Eine einfache Beispiele, die die Benutzung von `\include` demonstrieren, sind dargestellt in Abschnitt "Scores and parts" in *Handbuch zum Lernen*.

Siehe auch

Handbuch zum Lernen: Abschnitt "Other sources of information" in *Handbuch zum Lernen*, Abschnitt "Scores and parts" 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.

3.3.2 Verschiedene Editionen aus einer Quelldatei
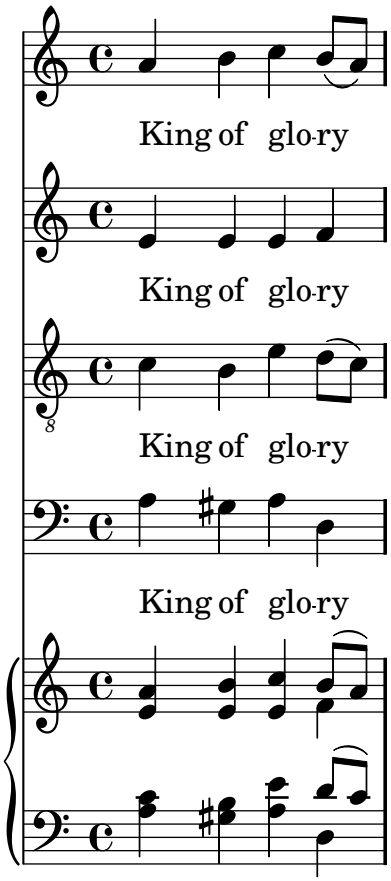
Es gibt verschiedene Funktionen, die es möglich machen, unterschiedliche Versionen einer Partitur aus der gleichen Quelldatei zu produzieren. Variablen werden am besten eingesetzt, wenn es darum geht, längere Notenpassagen und/oder Anmerkungen/Textmarken miteinander auf verschiedene Weise zu kombinieren. Tag-Marken dagegen werden am besten eingesetzt, wenn eine von mehreren kurzen alternativen Notenabschnitten ausgewählt werden soll. Egal welche Methode am Ende eingesetzt wird: Es erleichtert die Arbeit in jedem Fall, wenn die eigentlichen Noten und die Struktur der Paritur voneinander getrennt notiert werden – so kann die Struktur geändert werden, ohne dass man Änderungen an den Noten vornehmen muss.

Variablen benutzen

Wenn Notenabschnitt in Variablen definiert werden, können sie an unterschiedlichen Stellen in der Partitur eingesetzt werden, siehe auch Abschnitt "Organizing pieces with variables" 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 [Automatische Kombination von Stimmen], Seite 118. Hier ein Beispiel:

```
SopranNoten = \relative c'' { a4 b c b8( a)}
AltNoten = \relative g' { e4 e e f }
TenorNoten = \relative c' { c4 b e d8( c) }
BassNoten = \relative c' { a4 gis a d, }
alleText = \lyricmode {King of glo -- ry }
<<
  \new Staff = "Sopran" \SopranNoten
  \new Lyrics \alleText
  \new Staff = "Alt" \AltNoten
  \new Lyrics \alleText
  \new Staff = "Tenor" {
    \clef "treble_8"
    \TenorNoten
  }
  \new Lyrics \alleText
  \new Staff = "Bass" {
    \clef "bass"
    \BassNoten
  }
  \new Lyrics \alleText
  \new PianoStaff <<
    \new Staff = "RH" {
      \set Staff.printPartCombineTexts = ##f
      \partcombine
      \SopranNoten
      \AltNoten
    }
    \new Staff = "LH" {
      \set Staff.printPartCombineTexts = ##f
      \clef "bass"
      \partcombine
      \TenorNoten
      \BassNoten
    }
  >>
>>
```



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 3.3.1 [LilyPond-Dateien einfügen], Seite 314.

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:

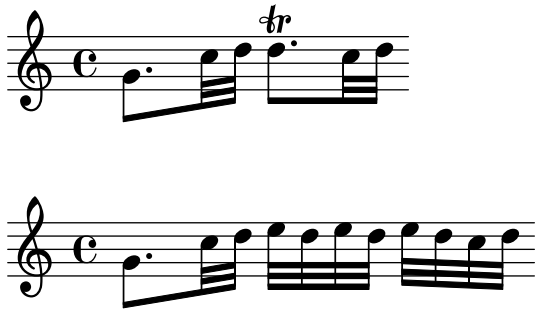
Filter	**Resultat**
Markierte Noten mit vorgesetztem `\keepWithTag #'`*`Bezeichnung`*	Unmarkierte Noten und Noten mit der Marke *Bezeichnung* werden gesetzt, Noten mit einer anderen Marke werden nicht angezeigt.
Markierte Noten mit vorgesetztem `\removeWithTag #'`*`Bezeichnung`*	Unmarkierte Noten und Noten mit einer anderen Marke als *Bezeichnung* wird angezeigt, Noten markiert mit *Bezeichnung* 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 sein (wie etwa `#'score` oder `#'part`), gefolgt von einem musikalischen Ausdruck.

Im folgenden Beispiel erscheinen zwei Versionen der Noten, eine zeigt Triller in normaler Notation, die andere zeigt sie ausgeschrieben:

```
Noten = \relative g' {
  g8. c32 d
  \tag #'trills {d8.\trill }
  \tag #'expand {\repeat unfold 3 {e32 d} }
  c32 d
 }
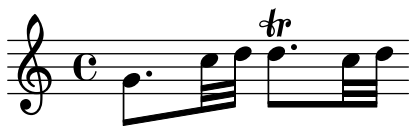
\score {
  \keepWithTag #'trills \Noten
}
\score {
  \keepWithTag #'expand \Noten
}
```
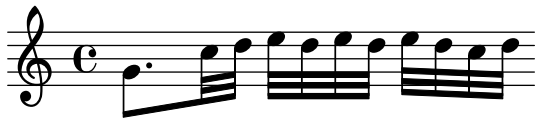


Entsprechend können auch Abschnitte ausgeschlossen werden; das erfordert manchmal weniger Schreibarbeit:

```
Noten = \relative g' {
  g8. c32 d
  \tag #'trills {d8.\trill }
  \tag #'expand {\repeat unfold 3 {e32 d} }
  c32 d
 }

\score {
  \removeWithTag #'expand
  \Noten
}
\score {
  \removeWithTag #'trills
  \Noten
}
```





Marken können auch auf Artikulationen, Text usw angewendet werden, indem man ihnen

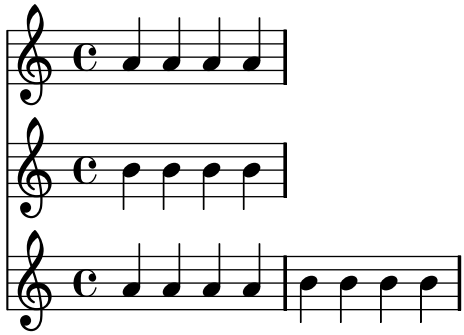
`-\tag #'`*`your-tag`*

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:

```
Noten = \relative c'' {
  \tag #'a \tag #'both { a a a a }
  \tag #'b \tag #'both { b b b b }
}
<<
\keepWithTag #'a \Noten
\keepWithTag #'b \Noten
\keepWithTag #'both \Noten
>>
```



Mehrfache `\removeWithTag`-Filter können auf einen musikalischen Ausdruck angewendet werden, um mehrere unterschiedliche markierte Abschnitte aus dem Druckbild zu entfernen.

```
Noten = \relative c'' {
\tag #'A { a a a a }
\tag #'B { b b b b }
\tag #'C { c c c c }
\tag #'D { d d d d }
}
{
\removeWithTag #'B
\removeWithTag #'C
\Noten
}
```



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.

Siehe auch

Handbuch zum Lernen: Abschnitt “Organizing pieces with variables” in *Handbuch zum Lernen*.

Notationsreferenz: [Automatische Kombination von Stimmen], Seite 118, Abschnitt 3.3.1 [LilyPond-Dateien einfügen], Seite 314.

3.3.3 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.

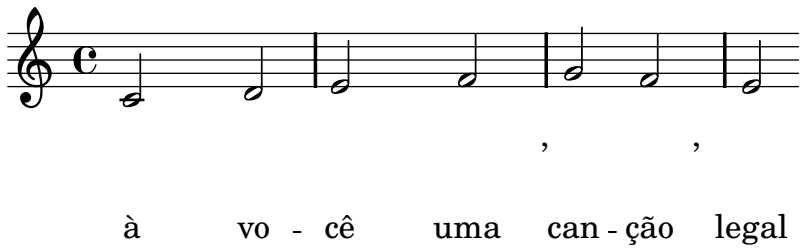
Das Aussehen des Zeichens wird bestimmt durch die gerade benutzte Schriftart (engl. font). In einer Schriftartdatei werden die Nummern der Unicode-Zeichen einem bestimmten Glyphen zugeordnet. LilyPond verwendet die Pango-Bibliothek um mehrsprachige Texte und komplexe Skripte korrekt zu setzen.

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
```

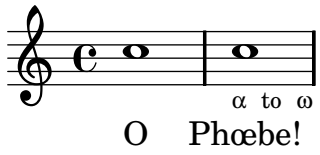
Heir ein Beispiel mit Kyrilliza, hebräischem und portugiesischem Text:



Um einen einzelnen Buchstaben zu notieren, für den die Unicode-Buchstabenfolge bekannt ist, der aber nicht auf der Tastatur zu finden ist, kann der Befehl `\char ##xhhhh` innerhalb einer `\markup`-Umgebung benutzt werden. Hierbei bedeutet `hhhh` die hexadezimale Zahl für das erforderliche Zeichen. Mit `\char ##x03BE` 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 Unicode-Zeichen an vier Stellen mit dem Zahlencode notiert: in einem Übungszeichen, als Artikulationszeichen, im Gesangstext und als normaler Text außerhalb der Partitur.

```
\score {
  \relative c'' {
    c1 \mark \markup { \char ##x03EE }
    c1_\markup { \tiny { \char ##x03B1 " to " \char ##x03C9 } }
  }
  \addlyrics { O \markup { \concat{ Ph \char ##x0153 be! } } }
}
\markup { "Copyright 2008" \char ##x00A9 }
```



Copyright 2008 ©

Um das Copyright-Zeichen zu notieren, kann folgender Code eingesetzt werden:

```
\header {
  copyright = \markup { \char ##x00A9 "2008" }
}
```

3.3.4 LilyPond-Notation anzeigen

Ein musikalischer Ausdruck in LilyPond-Notation kann mit der Funktion `\displayMusic` angezeigt werden. Der Code

```
{
  \displayLilyMusic \transpose c a, { c e g a bes }
}
```

etwa wird ausgeben:

```
{ a, cis e fis g }
```

Normalerweise werden diese Zeilen zusammen mit allen anderen Nachrichten auf der Kommandozeile ausgegeben. Um sie separat zu speichern und das Ergebnis von `\displayMusic` weiterzubenutzen, kann die Ausgabe mit folgendem Befehl in eine Datei umgeleitet werden:

```
lilypond file.ly >display.txt
```

3.4 Ausgabe kontrollieren

3.4.1 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 Defintion 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*.

3.4.2 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

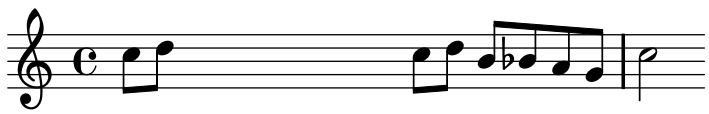
```
showLastLength = R1*5
\score { ... }
```

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'' {
  c8 d
  \set Score.skipTypesetting = ##t
  e e e e e e e
  \set Score.skipTypesetting = ##f
  c 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.

3.5 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.

Die MIDI-Ausgabe benötigt einen Kanal für jedes System und einen für globale Einstellungen. Darum sollte die Quelldatei für eine MIDI-Datei nicht mehr als 15 Systeme (oder 14 wenn kein Schlagzeug benützt wird) besitzen. Jedes weitere System bleibt stumm.

3.5.1 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 3.5.2 [Der MIDI-Block], Seite 325.

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 [Metronomangabe], Seite 140. Eine andere möglichkeit, ein eigenes MIDI-Tempo anzugeben, wird weiter unten gezeigt, siehe Abschnitt 3.5.2 [Der MIDI-Block], Seite 325.

Instrumentenbezeichnungn

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 B.4 [MIDI-Instrumente], Seite 350 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 entpsricht, wird ein Piano-Klang benutzt (`"acoustic grand"`).
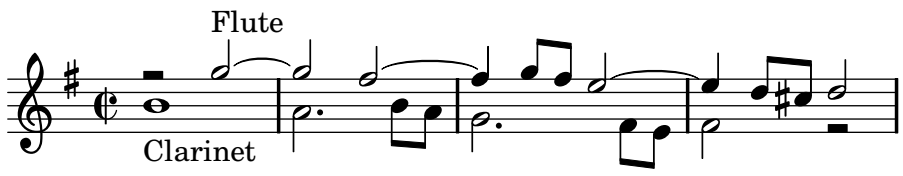
Ausgewählte Schnipsel

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"
    }
    \context {
      \Score
      tempoWholesPerMinute = #(ly:make-moment 72 2)
    }
  }
}
```



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.

3.5.2 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 {
    \context {
      \Score
      tempoWholesPerMinute = #(ly:make-moment 72 4)
    }
  }
}
```

Hier wird das Tempo auf 72 Viertelnoten pro Minute definiert. Wenn das Tempo auf diese Weise definiert wird, kann keine punktierte Note als Einheit angegeben werden. Wenn sie benötigt wird, muss man sie in kleinere Einheiten auflösen. Ein Tempo von 90 punktierten Viertelnoten pro Minute kann beispielsweise dargestellt werden als 270 Achtelnoten pro Minute:

```
tempoWholesPerMinute = #(ly:make-moment 270 8)
```

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 "Other sources of 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. Wenn eine Partitur explizit etwa mit `\new Score` begonnen wird, und sich die MIDI-Umgebung hierin befindet, wird keine Ausgabe produziert. Eine Lösung ist es, sowohl die `\new Score`- als auch die `\midi`-Umgebungen in eine `\score`-Umgebung einzuschließen.

```
\score {
  \new Score { ...Noten... }
  \midi { }
}
```

3.5.3 Was geht in die MIDI-Ausgabe

In MIDI unterstützt

Die folgenden Notationselemente werden in die MIDI-Ausgabe aufgenommen:

- Tonhöhen
- Vierteltöne (siehe [Versetzungszeichen], Seite 4. 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

In MIDI nicht unterstützt
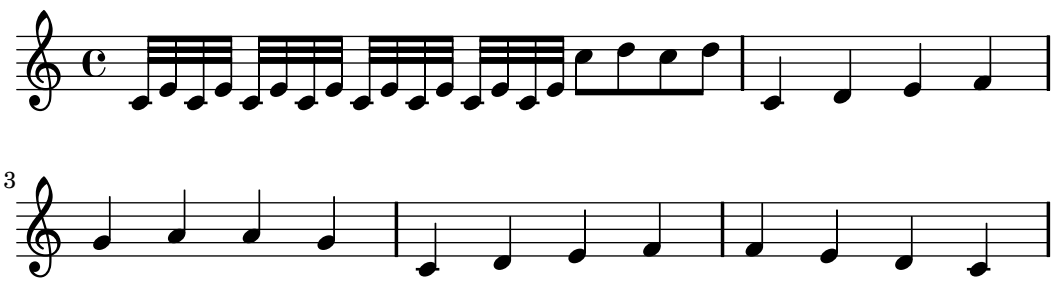
Folgende Notationselemente werden nicht in die MIDI-Ausgabe einbeozogen:

- Rhythmus, der als Anmerkung notiert wird, bspw. Swing
- Tempoveränderungen, die als Anmerkung ohne Tempobezeichnung notiert werden
- Staccato und andere Artikultationen und Ornamente
- Legato- und Phrasierungsbögen
- Crescendi, decrescendi zu einer einzelnen Note
- Tremolo, notiert mit ‚:[*number*]‘
- Bezifferter Bass

3.5.4 Wiederholungen im MIDI

Mit einigen Veränderungungen 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 "|."
```



Wenn eine Partitur mit diesem 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 { .. }
}
```

3.5.5 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.

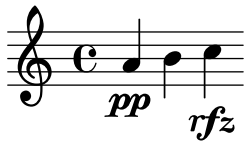
Dynamik-Zeichen

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 "Other sources of 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 c'' {
         a\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.
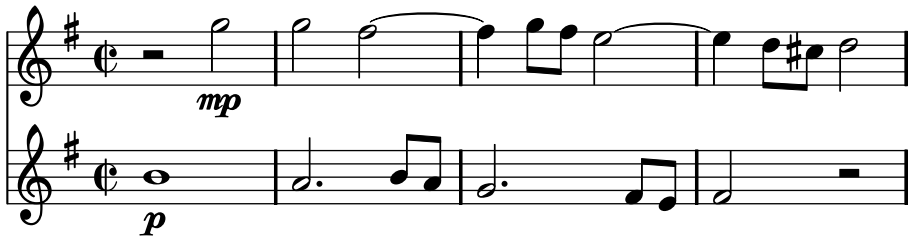
MIDI-Lautstärke

Die generellen Mindest- und Höchstwerte für die Lautstärke der MIDI-Datei wird kontrolliert, indem die Eigenschaften `midiMinimumVolume` unde `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 c''' {
        r2 g\mp g fis ~
        fis4 g8 fis e2 ~
        e4 d8 cis d2
      }
    }
    \new Staff {
      \key g \major
      \set Staff.midiInstrument = #"clarinet"
      \new Voice \relative c'' {
        b1\p a2. b8 a
        g2. fis8 e
        fis2 r
      }
    }
  >>
  \layout { }
  \midi {
    \context {
      \Score
      tempoWholesPerMinute = #(ly:make-moment 72 2)
      midiMinimumVolume = #0.2
      midiMaximumVolume = #0.5
    }
  }
}
```
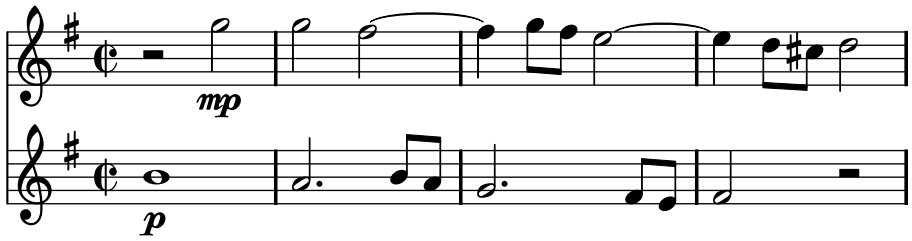


Verschiedene Instrumente angleichen (i)

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 c''' {
        r2 g\mp g fis ~
        fis4 g8 fis e2 ~
        e4 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 c'' {
        b1\p a2. b8 a
        g2. fis8 e
        fis2 r
      }
    }
  >>
  \layout { }
  \midi {
    \context {
      \Score
      tempoWholesPerMinute = #(ly:make-moment 72 2)
    }
  }
}
```



Verschidene 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 entpsrechenden 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.
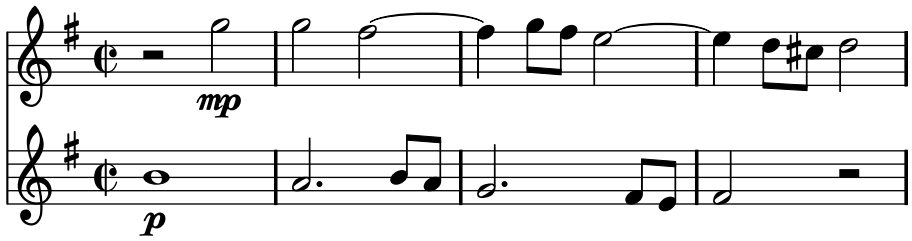
```
#(define my-instrument-equalizer-alist '())

#(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 c''' {
        r2 g\mp g fis ~
        fis4 g8 fis e2 ~
        e4 d8 cis d2
      }
    }
    \new Staff {
      \key g \major
      \set Staff.midiInstrument = #"clarinet"
      \new Voice \relative c'' {
```

```
        b1\p a2. b8 a
        g2. fis8 e
        fis2 r
      }
    }
  >>
  \layout { }
  \midi {
    \context {
      \Score
      tempoWholesPerMinute = #(ly:make-moment 72 2)
    }
  }
}
```



3.5.6 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 notiert werden. Es handelt sich um `melodic tom, taiko drum, synth drum` usw.

Viele Schlagzeuginstrumnete 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.

4 Abstände

4.1 Papier und Seiten

4.1.1 Papierformat

4.1.2 Seitenformatierung

Vertikale Dimensionen

Horizontale Dimensionen

Weitere Layout-Variablen

4.2 Notenlayout

4.2.1 Die Notensystemgröße einstellen

4.2.2 Partiturlayout

4.3 Umbrüche

4.3.1 Zeilenumbrüche

4.3.2 Seitenumbrüche

4.3.3 Optimale Seitenumbrüche

4.3.4 Optimale Umbrüche zum Blättern

4.3.5 Minimale Seitenumbrüche

4.3.6 Ausdrückliche Umbrüche

4.3.7 Eine zusätzliche Stimme für Umbrüche benutzen

4.4 Vertikale Abstände

4.4.1 Vertikale Abstände innerhalb eines Systems

4.4.2 Vertikale Abstände zwischen Systemen

4.4.3 Explizite Positionierung von Systemen

4.4.4 Vertikale Abstände mit zwei Durchgängen

4.4.5 Vermeidung von vertikalen Zusammenstößen

4.5 Horizontale Abstände

4.5.1 Überblick über horizontale Abstände

4.5.2 Eine neuer Bereich mit anderen Abständen

4.5.3 Horizontale Abstände verändern

4.5.4 Zeilenlänge

4.5.5 Proportionale Notation

4.6 Die Musik auf weniger Seiten zwingen

4.6.1 Abstände anzeigen lassen

4.6.2 Abstände verändern

Manchmal bleiben nur noch ein oder zwei Systeme auf der letzten Seite übrig. Das ist immer ärgerlich, besonders wenn es scheint, dass auf den vorigen Seiten genug Platz ist, um die Systeme noch unterzubringen.

Wenn man versucht, das Layout zu verändern, kommt einem der Befehl `annotate-spacing` zu Hilfe. Mit diesem Befehl erhält man die Werte von verschiedenen Abstandsbefehlen ausgedruckt, mehr Information im Kapitel Abschnitt 4.6.1 [Abstände anzeigen lassen], Seite 333. Anhand dieser Angaben kann dann entschieden werden, welche Werte verändert werden müssen.

Neben Rändern gibt es nämlich 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 {
    between-system-padding = #0.1
    between-system-space = #0.1
    ragged-last-bottom = ##f
    ragged-bottom = ##f
  }
  ```
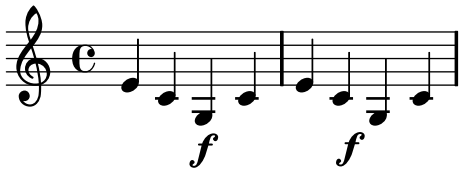
- Die Anzahl der Systeme kann reduziert werden (wenn LilyPond die Musik auf 11 Systeme verteilt, kann man die Benutzung von nur 10 Systemen erzwingen).

  ```
  \paper {
    system-count = #10
  }
  ```

- Vermeidung von Objekten, die den vertikalen Abstand von Systemen vergrößern, hilft oft. Die Verwendung von Klammern bei 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.

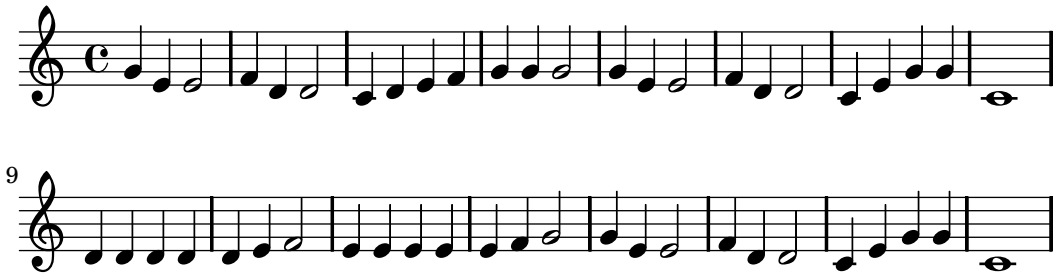
  Ein anderes Beispiel ist es, Dynamik-Zeichen, die besonders weit „hervorstehen“, zu verschieben.

  ```
  \relative c' {
    e4 c g\f c
    \override DynamicLineSpanner #'padding = #-1.8
    \override DynamicText #'extra-offset = #'( -2.1 . 0)
    e4 c g\f c
  }
  ```



- Die horizontalen Abstände können mit der `SpacingSpanner`-Eigenschaft verändert werden. Siehe Abschnitt 4.5.3 [Horizontale Abstände verändern], Seite 333 für Einzelheiten.

```
\score {
  \relative c'' {
    g4 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 = #(ly:make-moment 1 4)
    }
  }
}
```



5 Standardeinstellungen verändern

5.1 Interpretationsumgebungen

5.1.1 Was sind Umgebungen?

Score - Meister aller Kontexte

Oberste Kontexte – Container für Systeme

Mittlere Kontexte – Systeme

Unterste Kontexte – Stimmen

5.1.2 Umgebungen erstellen

5.1.3 Umgebungs-Plugins verändern

5.1.4 Die Standardeinstellungen von Umgebungen ändern

5.1.5 Neue Umgebungen definieren

5.1.6 Umgebungen aneinander ausrichten

5.2 Die Referenz der Programminterna erklärt

5.2.1 Zurechtfinden in der Programmreferenz

5.2.2 Layout-Schnittstellen

5.2.3 Die Grob-Eigenschaften

5.2.4 Benennungsübereinkommen

5.3 Eignschaften verändern

5.3.1 Überblick über verändernde Eigenschaften

5.3.2 Der `\set`-Befehl

5.3.3 Der \override-Befehl

5.3.4 Der `\tweak`-Befehl

5.3.5 `\set` versus `\override`

5.4 Nützliche Konzepte und Eigenschaften

5.4.1 Eingabe-Modi

5.4.2 Richtung und Platzierung

5.4.3 Abstände und Maße

5.4.4 Eigenschaften von Staff (Notensystem)-Symbol

5.4.5 Strecker

Das `spanner-interface` benutzen

Das `line-spanner-interface` benutzen

5.4.6 Sichtbarkeit von Objekten

Einen stencil entfernen

Objekten unsichtbar machen

Objekte weiß malen

break-visibility (unsichtbar machen) benutzen

Besonderheiten

5.4.7 Zeilenstile

5.4.8 Drehen von Objekten

Drehen von Objekten

Textbeschriftung drehen

5.5 Fortgeschrittene Optimierungen

5.5.1 Umgebungen ausrichten

`X-offset` und `Y-offset` direkt setzen

Das `side-position-interface` benutzen

Das `self-alignment-interface` benutzen

Die `aligned-on-parent`-Prozeduren benutzen

Die `centered-on-parent`-Prozeduren benutzen

Das `break-aligned-interface` benutzen

5.5.2 Vertikale Gruppierung der grafischen Objekte („grob“s)

5.5.3 stencils verändern

5.5.4 Formen verändern

Bögen verändern

6 Schnittstellen für Programmierer

6.1 Musikalische Funktionen

6.1.1 Überblick über musikalische Funktionen

6.1.2 Einfache Ersetzungsfunktionen

6.1.3 Paarige Ersetzungsfunktionen

6.1.4 Mathematik in Funktionen

6.1.5 Leere Funktionen

6.1.6 Funktionen ohne Argumente

6.1.7 Überblick über vorhandene musikalische Funktionen

`acciaccatura` - *music* (music)
(undocumented; fixme)

`addChordShape` - *key-symbol* (symbol) *tuning* (pair) *shape-definition* (unknown)
Add chord shape `shape-definition` to the `chord-shape-table` hash with the key `(cons key-symbol tuning)`.

`addInstrumentDefinition` - *name* (string) *lst* (list)
(undocumented; fixme)

`addQuote` - *name* (string) *music* (music)
Add a piece of music to be quoted

`afterGrace` - *main* (music) *grace* (music)
(undocumented; fixme)

`allowPageTurn`
Allow a page turn. May be used at toplevel (ie between scores or markups), or inside a score.

`applyContext` - *proc* (procedure)
(undocumented; fixme)

`applyMusic` - *func* (procedure) *music* (music)
(undocumented; fixme)

`applyOutput` - *ctx* (symbol) *proc* (procedure)
(undocumented; fixme)

`appoggiatura` - *music* (music)
(undocumented; fixme)

`assertBeamQuant` - *l* (pair) *r* (pair)
(undocumented; fixme)

`assertBeamSlope` - *comp* (procedure)
(undocumented; fixme)

`autochange` - *music* (music)
(undocumented; fixme)

`balloonGrobText` - *grob-name* (symbol) *offset* (pair of numbers) *text* (markup)
(undocumented; fixme)

`balloonText` - *offset* (pair of numbers) *text* (markup)
(undocumented; fixme)

`bar` - *type* (string)
(undocumented; fixme)

`barNumberCheck` - *n* (integer)
(undocumented; fixme)

`bendAfter` - *delta* (unknown)
(undocumented; fixme)

`breathe` (undocumented; fixme)

`clef` - *type* (string)
Set the current clef.

`cueDuring` - *what* (string) *dir* (direction) *main-music* (music)
(undocumented; fixme)

`displayLilyMusic` - *music* (music)
(undocumented; fixme)

`displayMusic` - *music* (music)
(undocumented; fixme)

`endSpanners` - *music* (music)
(undocumented; fixme)

`featherDurations` - *factor* (moment) *argument* (music)
Rearrange durations in ARGUMENT so there is an acceleration/deceleration.

`grace` - *music* (music)
(undocumented; fixme)

`includePageLayoutFile`
If page breaks and tweak dump is not asked, and the file <basename>-page-layout.ly exists, include it.

`instrumentSwitch` - *name* (string)
(undocumented; fixme)

`keepWithTag` - *tag* (symbol) *music* (music)
(undocumented; fixme)

`killCues` - *music* (music)
(undocumented; fixme)

`label` - *label* (symbol)
Place a bookmarking label, either at top-level or inside music.

`makeClusters` - *arg* (music)
(undocumented; fixme)

`musicMap` - *proc* (procedure) *mus* (music)
(undocumented; fixme)

`noPageBreak`
Forbid a page break. May be used at toplevel (ie between scores or markups), or inside a score.

`noPageTurn`
: Forbid a page turn. May be used at toplevel (ie between scores or markups), or inside a score.

`octaveCheck` - *pitch-note* (music)
: octave check

`oldaddlyrics` - *music* (music) *lyrics* (music)
: (undocumented; fixme)

`ottava` - *octave* (number)
: set the octavation

`overrideProperty` - *name* (string) *property* (symbol) *value* (any type)
: Set *property* to *value* in all grobs named *name*. The *name* argument is a string of the form `"Context.GrobName"` or `"GrobName"`

`pageBreak`
: Force a page break. May be used at toplevel (ie between scores or markups), or inside a score.

`pageTurn` Force a page turn between two scores or top-level markups.

`parallelMusic` - *voice-ids* (list) *music* (music)
: Define parallel music sequences, separated by '`|`' (bar check signs), and assign them to the identifiers provided in *voice-ids*.

  voice-ids: a list of music identifiers (symbols containing only letters)

  music: a music sequence, containing BarChecks as limiting expressions.

  Example:

```
\parallelMusic #'(A B C) {
  c c | d d | e e |
  d d | e e | f f |
}
<==>
  A = { c c | d d | }
  B = { d d | e e | }
  C = { e e | f f | }
```

`parenthesize` - *arg* (music)
: Tag *arg* to be parenthesized.

`partcombine` - *part1* (music) *part2* (music)
: (undocumented; fixme)

`pitchedTrill` - *main-note* (music) *secondary-note* (music)
: (undocumented; fixme)

`pointAndClickOff`
: (undocumented; fixme)

`pointAndClickOn`
: (undocumented; fixme)

`quoteDuring` - *what* (string) *main-music* (music)
: (undocumented; fixme)

`removeWithTag` - *tag* (symbol) *music* (music)
: (undocumented; fixme)

`resetRelativeOctave` - *reference-note* (music)
Set the octave inside a \relative section.

`rightHandFinger` - *finger* (number or string)
Define a StrokeFingerEvent

`scaleDurations` - *fraction* (pair of numbers) *music* (music)
(undocumented; fixme)

`scoreTweak` - *name* (string)
Include the score tweak, if exists.

`shiftDurations` - *dur* (integer) *dots* (integer) *arg* (music)
`spacingTweaks` - *parameters* (list)
Set the system stretch, by reading the 'system-stretch property of the 'parameters' assoc list.

`storePredefinedDiagram` - *chord* (music) *tuning* (pair) *diagram-definition* (unknown)
Add predefined fret diagram defined by `diagram-definition` for the chord pitches `chord` and the stringTuning `tuning`.

`tag` - *tag* (symbol) *arg* (music)
Add *tag* to the `tags` property of *arg*.

`tocItem` - *text* (markup)
Add a line to the table of content, using the `tocItemMarkup` paper variable markup

`transposedCueDuring` - *what* (string) *dir* (direction) *pitch-note* (music) *main-music* (music)
Insert notes from the part *what* into a voice called `cue`, using the transposition defined by *pitch-note*. 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-note* (music)
Set instrument transposition

`tweak` - *sym* (symbol) *val* (any type) *arg* (music)
Add `sym . val` to the `tweaks` property of *arg*.

`unfoldRepeats` - *music* (music)
(undocumented; fixme)

`withMusicProperty` - *sym* (symbol) *val* (any type) *music* (music)
Set *sym* to *val* in *music*.

6.2 Schnittstelle für Programmierer

6.2.1 Eingabevariablen und Scheme

6.2.2 Interne Repräsentation der Musik

6.3 Komplizierte Funktionen erstellen

6.3.1 Musikalische Funktionen darstellen

6.3.2 Eigenschaften von Musikobjekten

6.3.3 Verdoppelung einer Note mit Bindebögen (Beispiel)

6.3.4 Artikulationszeichen zu Noten hinzufügen (Beispiel)

6.4 Programmierungsschnittstelle für Textbeschriftungen

6.4.1 Beschriftungskonstruktionen in Scheme

6.4.2 Wie Beschriftungen intern funktionieren

6.4.3 Neue Definitionen von Beschriftungsbefehlen

6.4.4 Neue Definitionen von Beschriftungsbefehlen für Listen

6.5 Kontexte für Programmierer

6.5.1 Kontextauswertung

6.5.2 Eine Funktion auf alle Layout-Objekte anwenden

6.6 Scheme-Vorgänge als Eigenschaften

6.7 Scheme-Code anstelle von `weak` verwenden

6.8 Schwierige Korrekturen

Anhang A Literatur

Wenn Sie mehr über Notation und den Notenstich erfahren wollen, sind hier einige interessante Titel gesammelt.

Ignatzek 1995

Klaus Ignatzek, Die Jazzmethode für Klavier. Schott's Söhne 1995. Mainz, Germany ISBN 3-7957-5140-3.

Eine praktische Einführung zum Spielen von Jazz auf dem Klavier. Eins der ersten Kapitel enthält einen Überblick über die Akkorde, die im Jazz verwendet werden.

Gerou 1996

Tom Gerou and Linda Lusk, Essential Dictionary of Music Notation. Alfred Publishing, Van Nuys CA ISBN 0-88284-768-6.

Eine ausführliche, alphabetische Liste vieler Belange des Musiksatzes und der Notation; die üblichen Fälle werden behandelt.

Read 1968

Gardner Read, Music Notation: A Manual of Modern Practice. Taplinger Publishing, New York (2nd edition).

Ein Klassiker für die Musiknotation.

Ross 1987 Ted Ross, Teach yourself the art of music engraving and processing. Hansen House, Miami, Florida 1987.

Dieses Buch handelt vom Musiksatz, also vom professionellen Notenstich. Hier sind Anweisungen über Stempel, die Benutzung von Stiften und nationale Konventionen versammelt. Die Kapitel zu Reproduktionstechniken und der historische Überblick sind auch interessant.

Schirmer 2001

The G.Schirmer/AMP Manual of Style and Usage. G.Schirmer/AMP, NY, 2001.

Dieses Handbuch setzt den Fokus auf die Herstellung von Drucksachen für den Schirmer-Verlag. Hier werden viele Details behandelt, die sich in anderen Notationshandbüchern nicht finden. Es gibt auch einen guten Überblick, was nötig ist, um Drucke in publikationstauglicher Qualität zu produzieren.

Stone 1980

Kurt Stone, Music Notation in the Twentieth Century. Norton, New York 1980.
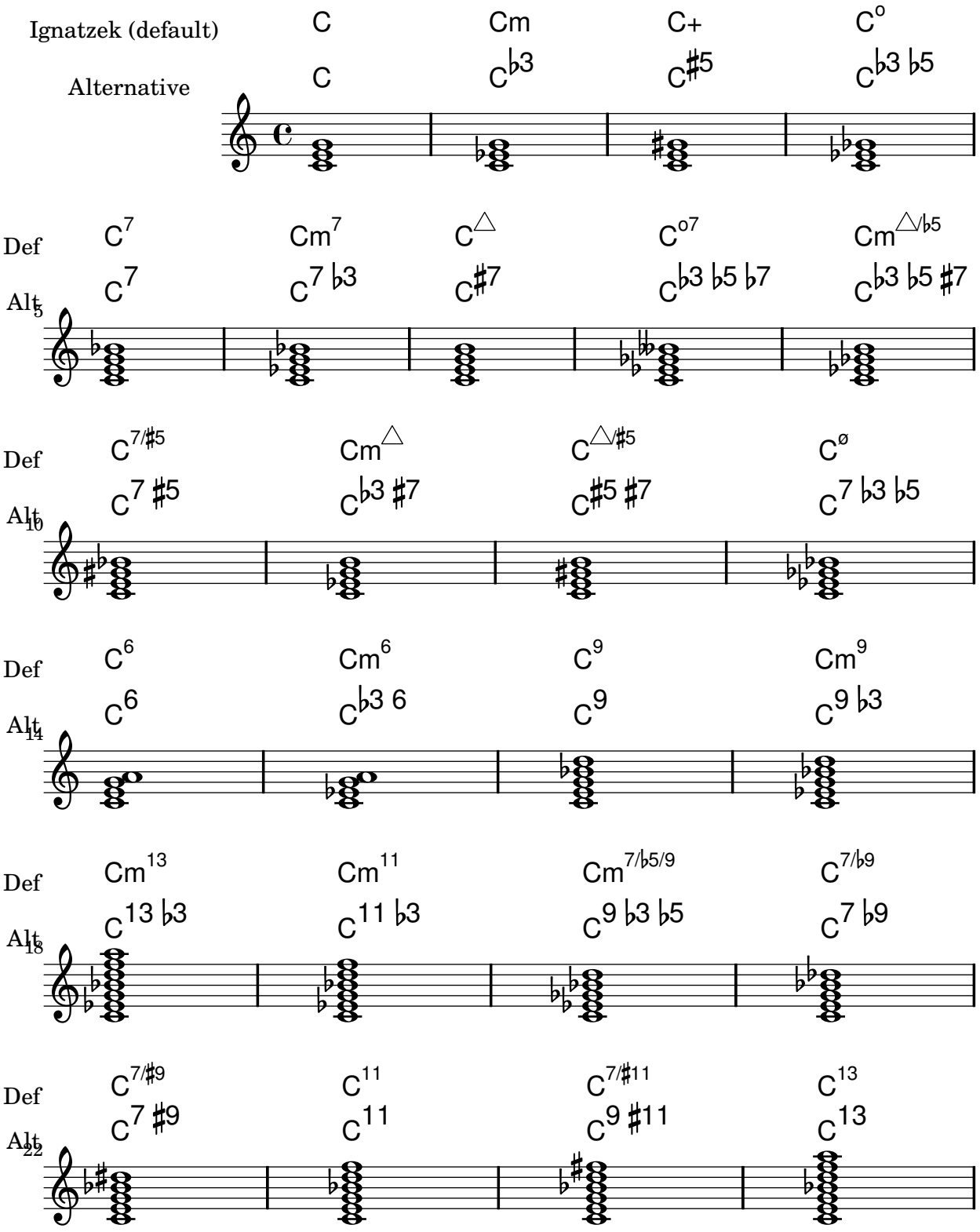
Dieses Buch enthält einen Überblick über die Notation von moderner E-Musik, beginnt aber mit einem Überblick über bereits existente Praktiken.

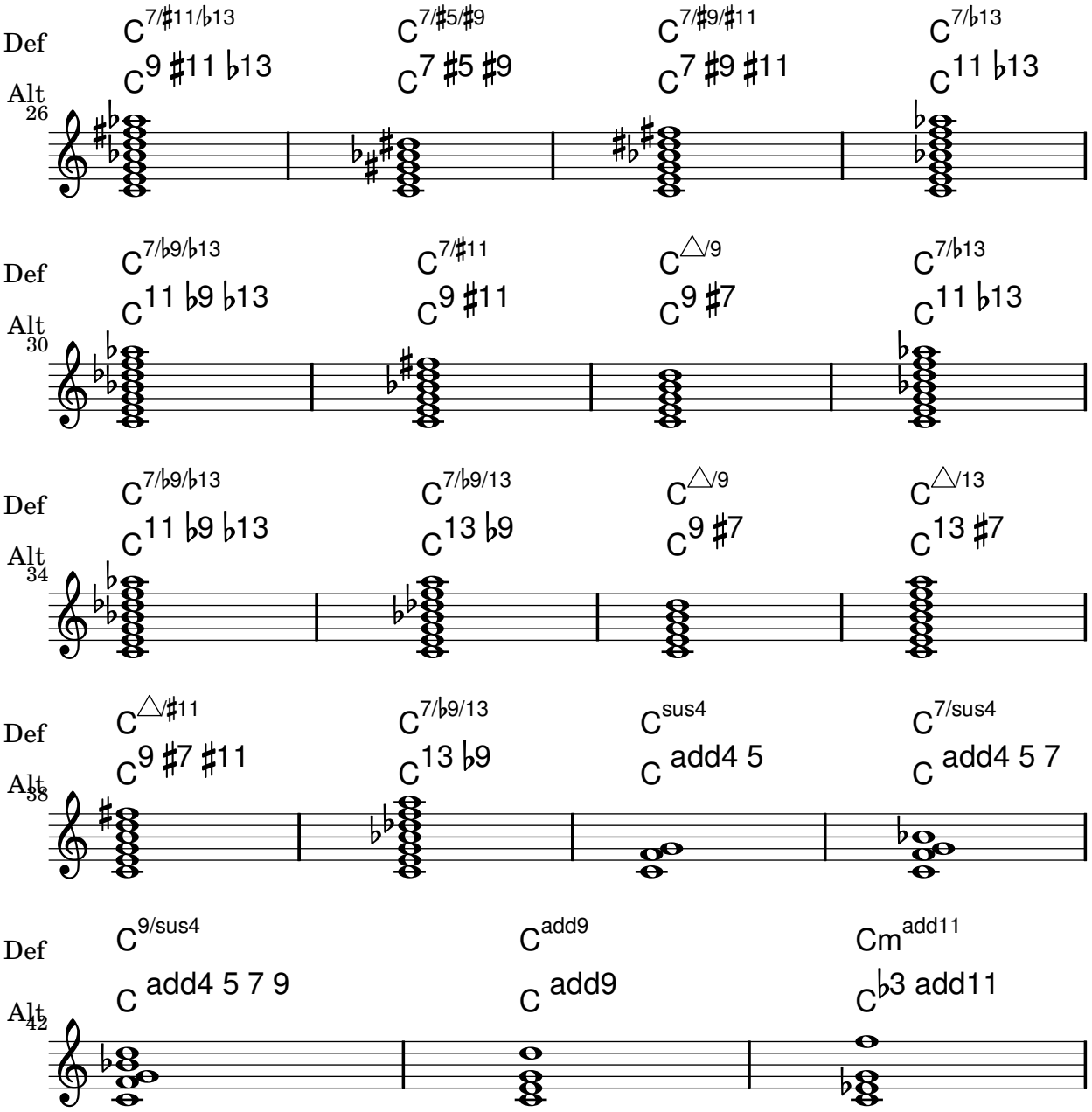
Das Quellenarchiv enthält eine ausführlichere BibTEX-Bibliographie mit über 100 Titeln in '`Documentation/bibliography/`'.

Anhang B Notationsübersicht

B.1 Liste der Akkordbezeichnungen

Die Tabelle zeigt die zwei üblichen Möglichkeiten, wie Akkordbezeichnungen ausgegeben werden. Es wird auch die entsprechende Note ausgegeben.

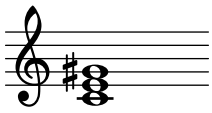
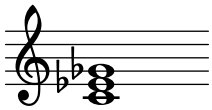
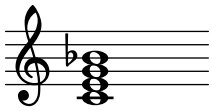
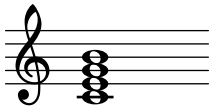
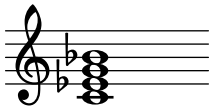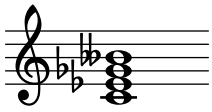
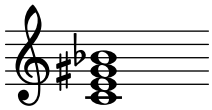
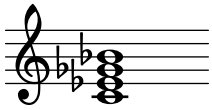
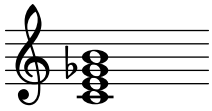


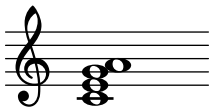
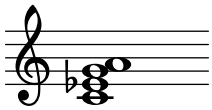
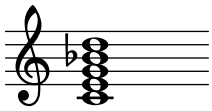
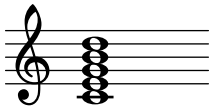
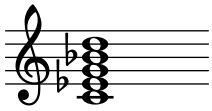
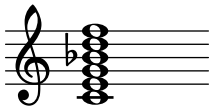
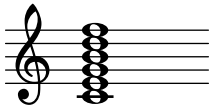
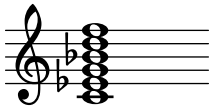
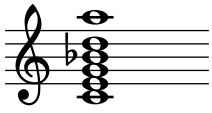


B.2 Übliche Akkord-Variablen

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`	

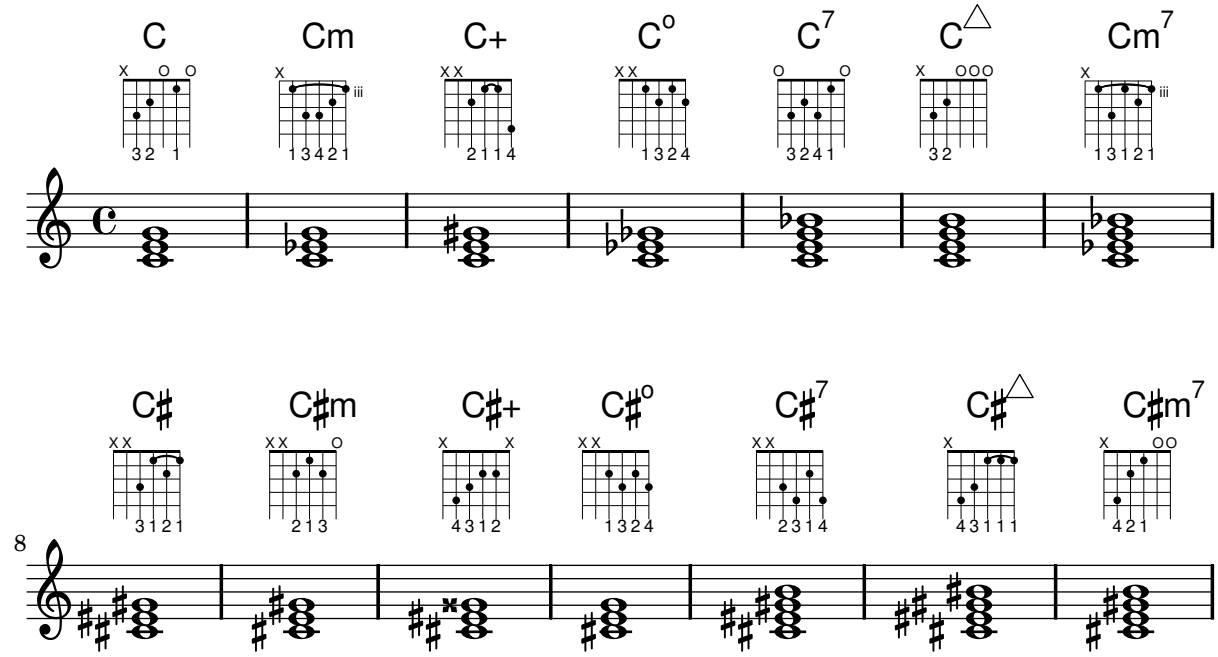
Übermäßig	Große Terz, übermäßige Quinte	`aug`	
Vermindert	Kleine Terz, verminderte Quinte	`dim`	
Dominantsieben	Durdreiklang, kleine Septime	`7`	
Große Septime	Durdreiklang, große Septime	`maj7` oder `maj`	
Kleine Septime	Molldreiklang, kleine Septime	`m7`	
Verminderte Septime	Verminderter Dreiklang, verminderte Septime	`dim7`	
Übermäßige Septime	Übermäßiger Dreiklang, kleine Septime	`aug7`	
halbverminderte Septime	Verminderter Dreklang, kleine Sept	`m7.5-`	
Kleine MollSept	Molldreiklang, Durseptime	`maj7.5-`	

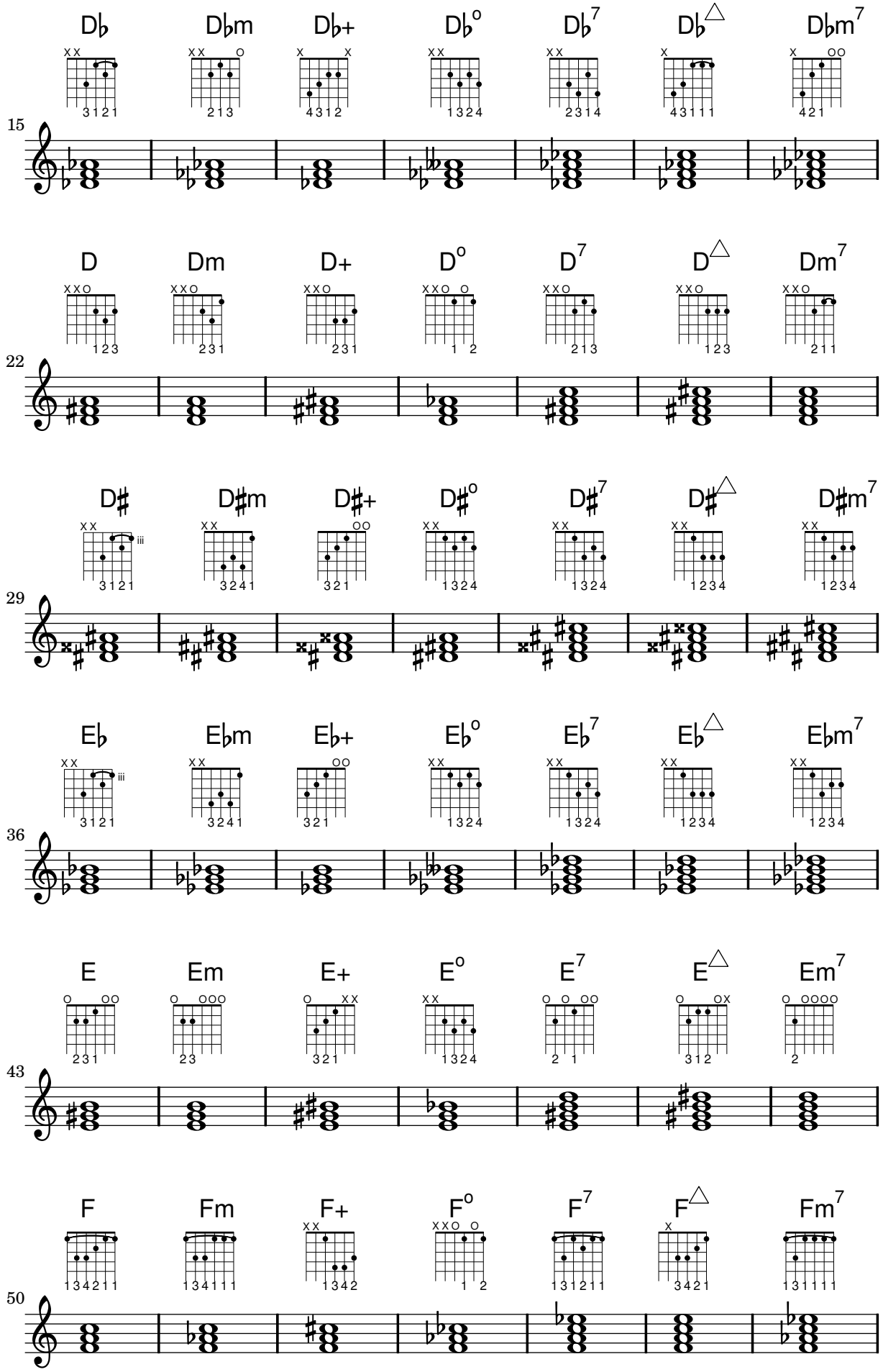
Große Sexte	Durdreiklang, Sexte	`6`	
Kleine Sexte	Molldreiklang, Sexte	`m6`	
Dominantnone	Dominantsept, große None	`9`	
Dur None	Große None, große Septime	`maj9`	
Moll None	Große None, kleine Septime	`m9`	
Dominantundezime	Dominantnone, perfekte Undezime	`11`	
Durundezime	Große None, perfekte Unidezime	`maj11`	
Mollundezime	Kleine None, perfekte Undezime	`m11`	
Dominant-13	Dominantnone, große 13	`13`	

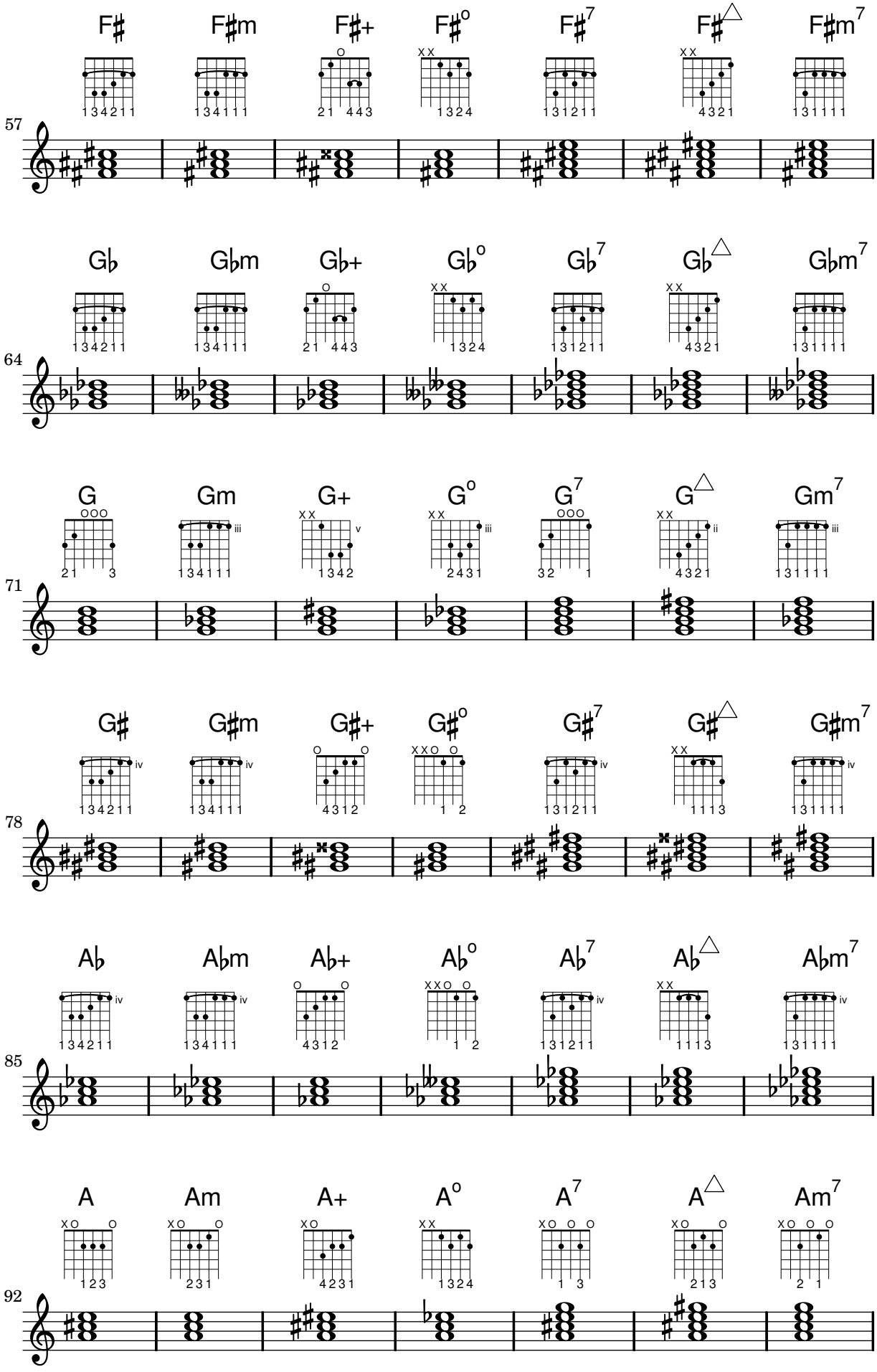
Dominant-13		`13.11`
Dur-13	Große Undezime, große 13	`maj13.11`
Moll-13	Kleine Undezime, große 13	`m13.11`
Sekundakkord	große Sekunde, perfekte Quinte	`sus2`
Quartakkord	perfekte Quarte, perfekte Quinte	`sus4`

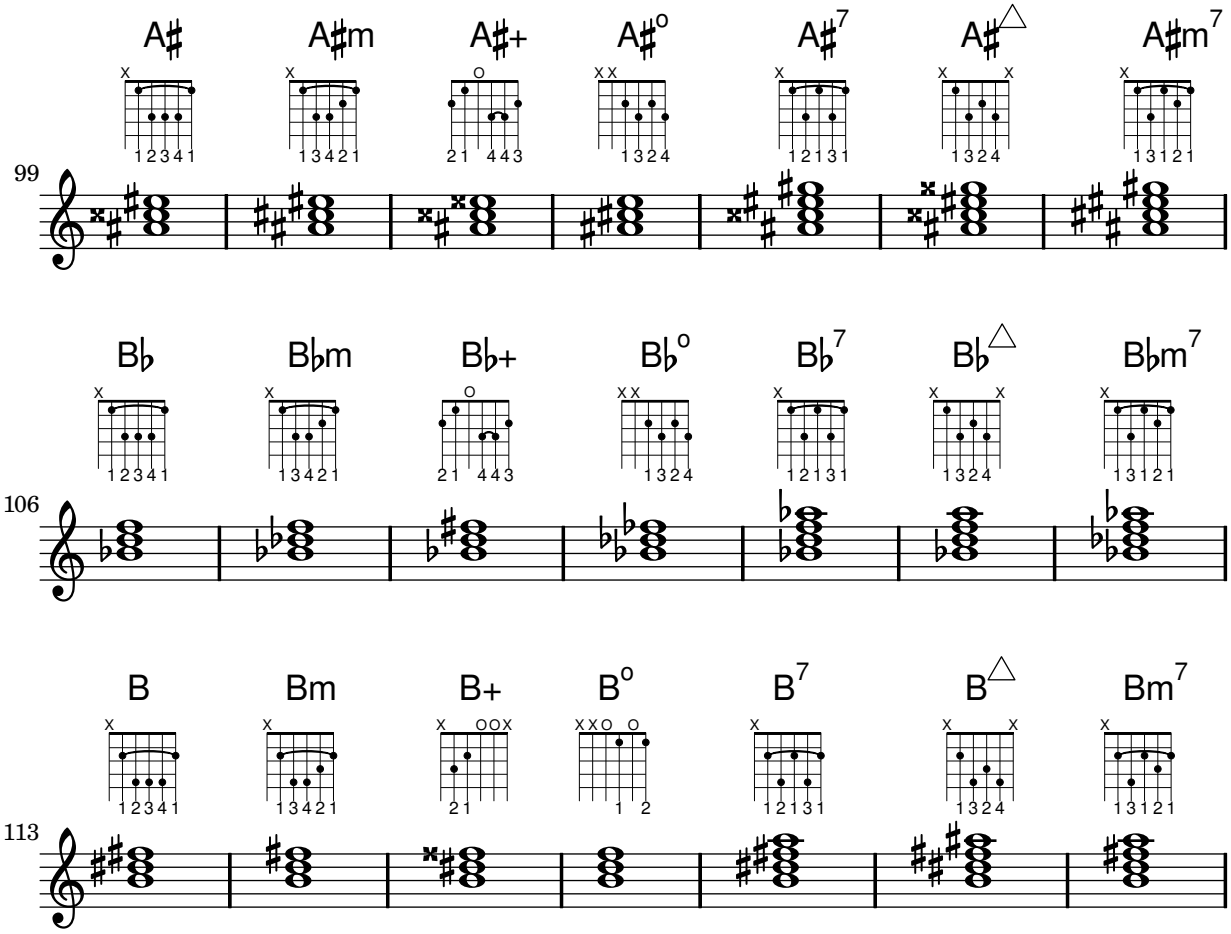
B.3 Vordefinierte Bund-Diagramme

Die Tabelle zeigt alle vordefinierten Bunddiagramme.




D♭ D♭m D♭+ D♭° D♭7 D♭△ D♭m7
15
D Dm D+ D° D7 D△ Dm7
22
D♯ D♯m D♯+ D♯° D♯7 D♯△ D♯m7
29
E♭ E♭m E♭+ E♭° E♭7 E♭△ E♭m7
36
E Em E+ E° E7 E△ Em7
43
F Fm F+ F° F7 F△ Fm7
50


F♯ F♯m F♯+ F♯° F♯7 F♯△ F♯m7
57
G♭ G♭m G♭+ G♭° G♭7 G♭△ G♭m7
64
G Gm G+ G° G7 G△ Gm7
71
G♯ G♯m G♯+ G♯° G♯7 G♯△ G♯m7
78
A♭ A♭m A♭+ A♭° A♭7 A♭△ A♭m7
85
A Am A+ A° A7 A△ Am7
92



B.4 MIDI-Instrumente

Hier eine Liste von Musikinstrumentenbezeichnungen, die als Name für `midiInstrument` benutzt werden können.

```
acoustic grand
bright acoustic
electric grand
honky-tonk
electric piano 1
electric piano 2
harpsichord
clav
celesta
glockenspiel
music box
vibraphone
marimba
xylophone
tubular bells
dulcimer
drawbar organ
percussive organ
rock organ
church organ
reed organ
accordion
harmonica
contrabass
tremolo strings
pizzicato strings
orchestral strings
timpani
string ensemble 1
string ensemble 2
synthstrings 1
synthstrings 2
choir aahs
voice oohs
synth voice
orchestra hit
trumpet
trombone
tuba
muted trumpet
french horn
brass section
synthbrass 1
synthbrass 2
soprano sax
alto sax
lead 7 (fifths)
lead 8 (bass+lead)
pad 1 (new age)
pad 2 (warm)
pad 3 (polysynth)
pad 4 (choir)
pad 5 (bowed)
pad 6 (metallic)
pad 7 (halo)
pad 8 (sweep)
fx 1 (rain)
fx 2 (soundtrack)
fx 3 (crystal)
fx 4 (atmosphere)
fx 5 (brightness)
fx 6 (goblins)
fx 7 (echoes)
fx 8 (sci-fi)
sitar
banjo
shamisen
koto
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)
```

B.5 Liste der Farben

Normale Farben

Die Syntax zur Benutzung findet sich im Abschnitt [Farbige Objekte], Seite 156.

```
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.

B.6 Die Feta-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 (bspw. `g^\markup { \musicglyph #"scripts.segno" }`). Siehe auch Abschnitt 1.8.2 [Text formatieren], Seite 170.

```
space                           plus                            +
```

comma	,	hyphen	-
period	.	zero	0
one	1	two	2
three	3	four	4
five	5	six	6
seven	7	eight	8
nine	9	f	*f*
m	*m*	p	*p*
r	*r*	s	*s*
z	*z*	rests.0	𝄻
rests.1	𝄺	rests.0o	𝄼
rests.1o	𝄽	rests.M3	𝄩
rests.M2	𝄨	rests.M1	𝄧
rests.2	𝄾	rests.2classical	𝄽
rests.3	𝄿	rests.4	𝅀

rests.5 

rests.6 

rests.7 

accidentals.sharp 

accidentals
.sharp.arrowup 

accidentals
.sharp.arrowdown 

accidentals
.sharp.arrowboth 

accidentals.sharp
.slashslash.stem 

accidentals.sharp
.slashslashslash.stemstem 

accidentals.sharp
.slashslashslash.stem 

accidentals.sharp
.slashslash.stemstemstem 

accidentals.natural 

accidentals
.natural.arrowup 

accidentals
.natural.arrowdown 

accidentals
.natural.arrowboth 

accidentals.flat 

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.slash 

accidentals.doublesharp 

accidentals.rightparen

accidentals.leftparen

arrowheads.open.01

arrowheads.open.0M1		arrowheads.open.11	
arrowheads.open.1M1		arrowheads.close.01	
arrowheads.close.0M1		arrowheads.close.11	
arrowheads.close.1M1		dots.dot	
noteheads.uM2		noteheads.dM2	
noteheads.sM1		noteheads.s0	
noteheads.s1		noteheads.s2	
noteheads.s0diamond		noteheads.s1diamond	
noteheads.s2diamond		noteheads.s0triangle	
noteheads.d1triangle		noteheads.u1triangle	
noteheads.u2triangle		noteheads.d2triangle	
noteheads.s0slash		noteheads.s1slash	
noteheads.s2slash		noteheads.s0cross	
noteheads.s1cross		noteheads.s2cross	
noteheads.s2xcircle		noteheads.s0do	

noteheads.d1do		noteheads.u1do	
noteheads.d2do		noteheads.u2do	
noteheads.s0re		noteheads.u1re	
noteheads.d1re		noteheads.u2re	
noteheads.d2re		noteheads.s0mi	
noteheads.s1mi		noteheads.s2mi	
noteheads.u0fa		noteheads.d0fa	
noteheads.u1fa		noteheads.d1fa	
noteheads.u2fa		noteheads.d2fa	
noteheads.s0la		noteheads.s1la	
noteheads.s2la		noteheads.s0ti	
noteheads.u1ti		noteheads.d1ti	
noteheads.u2ti		noteheads.d2ti	
scripts.ufermata		scripts.dfermata	
scripts.ushortfermata		scripts.dshortfermata	

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.stopped		scripts.upbow	
scripts.downbow		scripts.reverseturn	
scripts.turn		scripts.trill	
scripts.upedalheel		scripts.dpedalheel	
scripts.upedaltoe		scripts.dpedaltoe	
scripts.flageolet		scripts.segno	
scripts.coda		scripts.varcoda	

scripts.rcomma

scripts.lcomma

scripts.rvarcomma 

scripts.lvarcomma 

scripts.arpeggio 

scripts.trill_element

scripts.arpeggio.arrow.M1 

scripts.arpeggio.arrow.1 

scripts.trilelement

scripts.prall

scripts.mordent 

scripts.prallprall

scripts.prallmordent

scripts.upprall

scripts.upmordent

scripts.pralldown

scripts.downprall

scripts.downmordent

scripts.prallup

scripts.lineprall

scripts.caesura.curved

scripts.caesura.straight

flags.u3 

flags.u4 

flags.u5 

flags.u6 

flags.u7 

flags.d3 

flags.ugrace		flags.dgrace	
flags.d4		flags.d5	
flags.d6		flags.d7	
clefs.C		clefs.C_change	
clefs.F		clefs.F_change	
clefs.G		clefs.G_change	
clefs.percussion		clefs.percussion_change	
clefs.tab		clefs.tab_change	
timesig.C44		timesig.C22	
pedal.*		pedal.M	
pedal..	.	pedal.P	
pedal.d		pedal.e	
pedal.Ped		brackettips.up	
brackettips.down		accordion.accDiscant	

accordion.accDot

accordion.accFreebase

accordion.accStdbase

accordion.accBayanbase

accordion.accOldEE

rests.M3neomensural

rests.M2neomensural

rests.M1neomensural

rests.0neomensural

rests.1neomensural

rests.2neomensural

rests.3neomensural

rests.4neomensural

rests.M3mensural

rests.M2mensural

rests.M1mensural

rests.0mensural

rests.1mensural

rests.2mensural

rests.3mensural

rests.4mensural

noteheads.s1neomensural

noteheads.sM3neomensural

noteheads.sM2neomensural

noteheads.sM1neomensural

noteheads.s0harmonic

noteheads.s2harmonic

noteheads.s0neomensural

noteheads.s1neomensural

noteheads.s2neomensural

noteheads.s1mensural 

noteheads.sM3mensural 

noteheads.sM2mensural 

noteheads.sM1mensural 

noteheads.s0mensural

noteheads.s1mensural

noteheads.s2mensural

noteheads.s0petrucci 

noteheads.s1petrucci 

noteheads.s2petrucci 

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

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

noteheads.smedicaea .inclinatum



noteheads .smedicaea.punctum

noteheads .smedicaea.rvirga



noteheads .smedicaea.virga



noteheads .shufnagel.punctum

noteheads .shufnagel.virga



noteheads.shufnagel.lpes

clefs.vaticana.do

clefs.vaticana.do_change

clefs.vaticana.fa



clefs.vaticana.fa_change



clefs.medicaea.do



clefs.medicaea.do_change



clefs.medicaea.fa



clefs.medicaea.fa_change



clefs.neomensural.c



clefs.neomensural .c_change



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.mensural.c		clefs.mensural.c_change	
clefs.petrucci.f		clefs.petrucci.f_change	
clefs.mensural.f		clefs.mensural.f_change	
clefs.petrucci.g		clefs.petrucci.g_change	
clefs.mensural.g		clefs.mensural.g_change	
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	
custodes.medicaea.u0		custodes.medicaea.u1	
custodes.medicaea.u2		custodes.medicaea.d0	
custodes.medicaea.d1		custodes.medicaea.d2	

custodes.vaticana.u0		custodes.vaticana.u1	
custodes.vaticana.u2		custodes.vaticana.d0	
custodes.vaticana.d1		custodes.vaticana.d2	
custodes.mensural.u0		custodes.mensural.u1	
custodes.mensural.u2		custodes.mensural.d0	
custodes.mensural.d1		custodes.mensural.d2	
accidentals.medicaeaM1		accidentals.vaticanaM1	
accidentals.vaticana0		accidentals.mensural1	
accidentals.mensuralM1		accidentals.hufnagelM1	
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 

timesig.neomensural44 

timesig.neomensural22 

timesig.neomensural32 

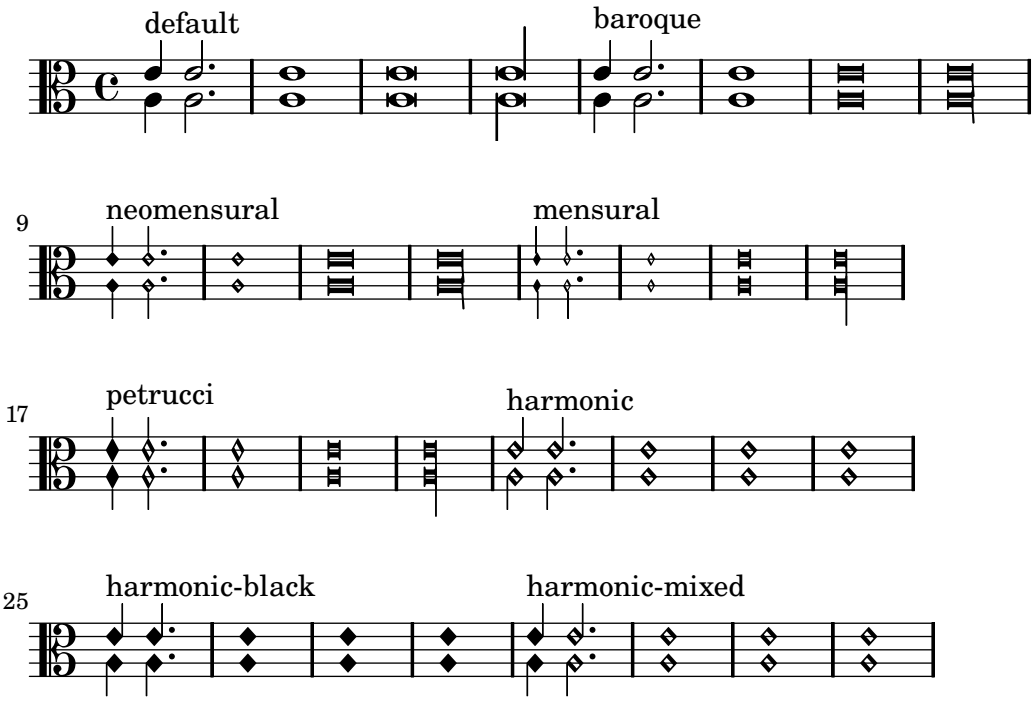
timesig.neomensural64 

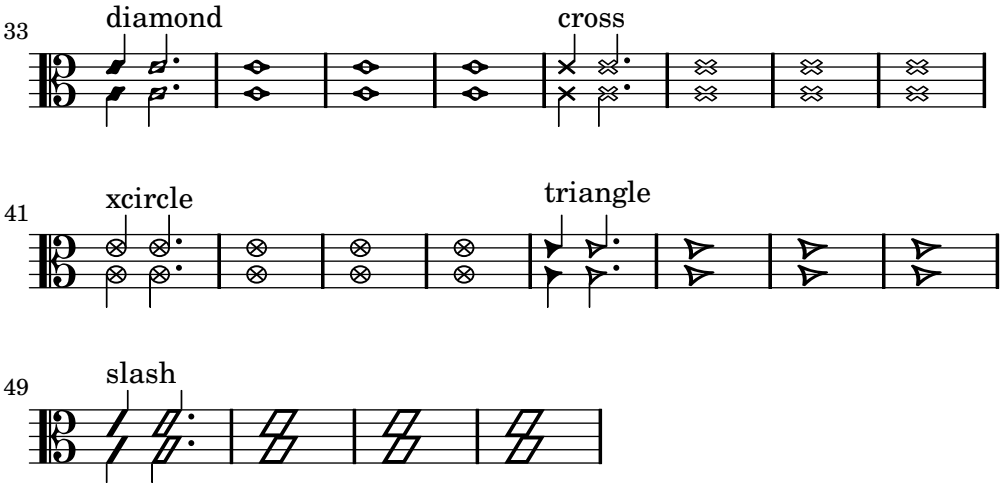
timesig.neomensural94 

timesig.neomensural34	𝅜	timesig.neomensural68	𝅝
timesig.neomensural98	𝅜	timesig.neomensural48	𝅝
timesig.neomensural68alt	𝅝	timesig.neomensural24	𝅜
scripts.ictus	ˌ	scripts.uaccentus	ˏ
scripts.daccentus	ˎ	scripts.usemicirculus	˓
scripts.dsemicirculus	˒	scripts.circulus	˚
scripts.augmentum	.	scripts .usignumcongruentiae	𝄐
scripts .dsignumcongruentiae	𝄐	dots.dotvaticana	.

B.7 Notenkopfstile

Folgende Stile können zur Darstellung der Notenköpfe verwendet werden:





B.8 Text markup commands

The following commands can all be used inside `\markup { }`.

B.8.1 Font

`\abs-fontsize` *size* (number) *arg* (markup)

Use *size* as the absolute font size to display *arg*. Adjusts `baseline-skip` and `word-space` 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

`\bold` *arg* (markup)

Switch to bold font-series.

```
\markup {
  default
  \hspace #2
  \bold
  bold
}
```

default **bold**

`\box` *arg* (markup)

Draw a box round *arg*. Looks at `thickness`, `box-padding` and `font-size` properties to determine line thickness and padding around the markup.

```
\markup {
  \override #'(box-padding . 0.5)
  \box
  \line { V. S. }
}
```

V. S.

Used properties:

- `box-padding` (0.2)
- `font-size` (0)
- `thickness` (1)

`\caps` *arg* (markup)

Copy of the `\smallCaps` command.

```
\markup {
  default
  \hspace #2
  \caps {
    Text in small caps
  }
}
```

default TEXT IN SMALL CAPS

`\dynamic` *arg* (markup)

Use the dynamic font. This font only contains **s**, **f**, **m**, **z**, **p**, and **r**. 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

`\finger` *arg* (markup)

Set *arg* as small numbers.

```
\markup {
  \finger {
    1 2 3 4 5
  }
}
```

1 2 3 4 5

`\fontCaps` *arg* (markup)

Set `font-shape` to `caps`

Note: `\fontCaps` requires the installation and selection of fonts which support the `caps` font shape.

`\fontsize` *increment* (number) *arg* (markup)

Add *increment* to the font-size. Adjusts `baseline-skip` accordingly.

```
\markup {
  default
  \hspace #2
  \fontsize #-1.5
  smaller
```

```
}
```

default smaller

Used properties:

- `baseline-skip` (2)
- `word-space` (1)
- `font-size` (0)

`\huge` *arg* (markup)
Set font size to +2.

```
\markup {
  default
  \hspace #2
  \huge
  huge
}
```

default huge

`\italic` *arg* (markup)
Use italic `font-shape` for *arg*.

```
\markup {
  default
  \hspace #2
  \italic
  italic
}
```

default *italic*

`\large` *arg* (markup)
Set font size to +1.

```
\markup {
  default
  \hspace #2
  \large
  large
}
```

default large

`\larger` *arg* (markup)
Increase the font size relative to the current setting.

```
\markup {
  default
  \hspace #2
  \larger
  larger
}
```

default larger

`\magnify` *sz* (number) *arg* (markup)
Set the font magnification for its argument. In the following example, the middle A is 10% larger:

```
A \magnify #1.1 { A } A
```

Note: Magnification only works if a font name is explicitly selected. Use `\fontsize` otherwise.

```
\markup {
  default
  \hspace #2
  \magnify #1.5 {
    50% larger
  }
}
```

default 50% larger

`\medium` *arg* (markup)
Switch to medium font-series (in contrast to bold).

```
\markup {
  \bold {
    some bold text
    \hspace #2
    \medium {
      medium font series
    }
    \hspace #2
    bold again
  }
}
```

some bold text medium font series **bold again**

`\normal-size-sub` *arg* (markup)
Set *arg* in subscript with a normal font size.

```
\markup {
  default
  \normal-size-sub {
    subscript in standard size
  }
}
```

default $_{\text{subscript in standard size}}$

Used properties:

- `baseline-skip`

`\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:

- `baseline-skip`

`\normal-text` *arg* (markup)
: Set all font related properties (except the size) to get the default normal text font, no matter what font was used earlier.

```
\markup {
  \huge \bold \sans \caps {
    Some text with font overrides
    \hspace #2
    \normal-text {
      Default text, same font-size
    }
    \hspace #2
    More text as before
  }
}
```

SOME TEXT WITH FONT OVERRIDES Default text, same font-size **MORE TEX**

`\normalsize` *arg* (markup)
: Set font size to default.

```
\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)
: Set font family to `number`, which yields the font used for time signatures and fingerings. This font contains numbers and some punctuation; it has no letters.

```
\markup {
  \number {
    0 1 2 3 4 5 6 7 8 9 . ,
```

```
  }
}
```

0123456789.,

`\roman` *arg* (markup)
Set font family to `roman`.

```
\markup {
  \sans \bold {
    sans serif, bold
    \hspace #2
    \roman {
      text in roman font family
    }
    \hspace #2
    return to sans
  }
}
```

sans serif, bold text in roman font family return to sans

`\sans` *arg* (markup)
Switch to the sans serif font family.

```
\markup {
  default
  \hspace #2
  \sans {
    sans serif
  }
}
```

default sans serif

`\simple` *str* (string)
A simple text string; `\markup { foo }` is equivalent with `\markup { \simple #"foo" }`.

Note: for creating standard text markup or defining new markup commands, the use of `\simple` is unnecessary.

```
\markup {
  \simple #"simple"
  \simple #"text"
  \simple #"strings"
}
```

simple text strings

`\small` *arg* (markup)
Set font size to -1.

```
\markup {
  default
  \hspace #2
```

```
  \small
  small
}
```

default small

`\smallCaps` *arg* (markup)
: Emit *arg* as small caps.

  Note: `\smallCaps` does not support accented characters.

```
\markup {
  default
  \hspace #2
  \smallCaps {
    Text in small caps
  }
}
```

default TEXT IN SMALL CAPS

`\smaller` *arg* (markup)
: Decrease the font size relative to the current setting.

```
\markup {
  \fontsize #3.5 {
    some large text
    \hspace #2
    \smaller {
      a bit smaller
    }
    \hspace #2
    more large text
  }
}
```

some large text a bit smaller more large text

`\sub` *arg* (markup)
: Set *arg* in subscript.

```
\markup {
  \concat {
    H
    \sub {
      2
    }
    O
  }
}
```

H_2O

Used properties:

- `baseline-skip`
- `font-size` (0)

`\super` *arg* (markup)
Set *arg* in superscript.

```
\markup {
  E =
  \concat {
    mc
    \super
    2
  }
}
```

$E = mc^{2}$

Used properties:

- `baseline-skip`
- `font-size` (0)

`\teeny` *arg* (markup)
Set font size to -3.

```
\markup {
  default
  \hspace #2
  \teeny
  teeny
}
```

default teeny

`\text` *arg* (markup)
Use a text font instead of music symbol or music alphabet font.

```
\markup {
  \number {
    1, 2,
    \text {
      three, four,
    }
    5
  }
}
```

1, 2, three, four, **5**

`\tiny` *arg* (markup)
Set font size to -2.

```
\markup {
  default
  \hspace #2
  \tiny
```

```
  tiny
}
```

default tiny

`\typewriter` *arg* (markup)
Use `font-family` typewriter for *arg*.

```
\markup {
  default
  \hspace #2
  \typewriter
  typewriter
}
```

default `typewriter`

`\underline` *arg* (markup)
Underline *arg*. Looks at `thickness` to determine line thickness and y-offset.

```
\markup {
  default
  \hspace #2
  \override #'(thickness . 2)
  \underline {
    underline
  }
}
```

default underline

Used properties:

- `thickness` (1)

`\upright` *arg* (markup)
Set `font-shape` to `upright`. This is the opposite of `italic`.

```
\markup {
  \italic {
    italic text
    \hspace #2
    \upright {
      upright text
    }
    \hspace #2
    italic again
  }
}
```

italic text upright text *italic again*

B.8.2 Align

`\center-align` *arg* (markup)
Align `arg` to its X center.

```
\markup {
  \column {
    one
    \center-align
    two
    three
  }
}
```

one
two
three

`\center-column` *args* (list of markups)
Put `args` in a centered column.

```
\markup {
  \center-column {
    one
    two
    three
  }
}
```

one
two
three

Used properties:

- `baseline-skip`

`\column` *args* (list of markups)
Stack the markups in *args* vertically. The property `baseline-skip` determines the space between markups in *args*.

```
\markup {
  \column {
    one
    two
    three
  }
}
```

one
two
three

Used properties:

- `baseline-skip`

`\combine` *arg1* (markup) *arg2* (markup)
: Print two markups on top of each other.

  Note: `\combine` cannot take a list of markups enclosed in curly braces as an argument; the follow example will not compile:

```
\combine { a list }
```

```
\markup {
  \fontsize #5
  \override #'(thickness . 2)
  \combine
    \draw-line #'(0 . 4)
    \arrow-head #Y #DOWN ##f
}
```



`\concat` *args* (list of markups)
: Concatenate *args* in a horizontal line, without spaces in between. Strings and simple markups are concatenated on the input level, allowing ligatures. For example, `\concat { "f" \simple #"i" }` is equivalent to `"fi"`.

```
\markup {
  \concat {
    one
    two
    three
  }
}
```

onetwothree

`\dir-column` *args* (list of markups)
: Make a column of *args*, going up or down, depending on the setting of the `direction` layout property.

```
\markup {
  \override #`(direction . ,UP) {
    \dir-column {
      going up
    }
  }
  \hspace #1
  \dir-column {
    going down
  }
  \hspace #1
  \override #'(direction . 1) {
    \dir-column {
      going up
    }
  }
}
```

up up
going going going
down

Used properties:

- `baseline-skip`
- `direction`

`\fill-line` *args* (list of markups)
Put *markups* in a horizontal line of width *line-width*. The markups are spaced or flushed to fill the entire line. If there are no arguments, return an empty stencil.

```
\markup {
  \column {
    \fill-line {
      Words evenly spaced across the page
    }
    \null
    \fill-line {
      \line { Text markups }
      \line {
        \italic { evenly spaced }
      }
      \line { across the page }
    }
  }
}
```

Words evenly spaced across the page

Text markups *evenly spaced* across the page

Used properties:

- `line-width` (#f)
- `word-space` (1)
- `text-direction` (1)

`\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)
: Set horizontal alignment. If *dir* is `-1`, then it is left-aligned, while `+1` is right. Values in between interpolate 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

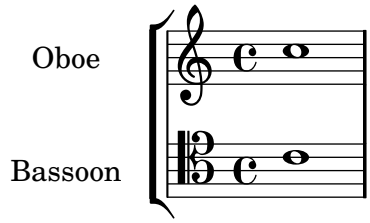
one
two
three

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 horizontal space *amount*.

```
\markup {
  one
  \hspace #2
  two
  \hspace #8
  three
}
```

one two three

`\justify-field` *symbol* (symbol)
: Justify the data which has been assigned to *symbol*.

```
\header {
  title = "My title"
  description = "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:description
    }
  }
}

\markup {
  \null
}
```

My title

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.

`\justify` *args* (list of markups)
: 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.

Used properties:

- `text-direction` (1)
- `word-space`
- `line-width` (#f)
- `baseline-skip`

`\justify-string` *arg* (string)

Justify a string. Paragraphs may be separated with double newlines

```
\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

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* (list of markups)
Put `args` in a left-aligned column.

```
\markup {
  \left-column {
    one
    two
    three
  }
}
```

one
two
three

Used properties:

- `baseline-skip`

`\line` *args* (list of markups)
Put *args* in a horizontal line. The property `word-space` determines the space between markups in *args*.

```
\markup {
  \line {
    one two three
  }
}
```

one two three

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`.

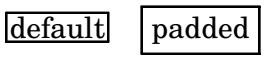
```
\markup {
  one
  \lower #3
  two
  three
}
```

one three
two

`\pad-around` *amount* (number) *arg* (markup)
Add padding *amount* all around *arg*.

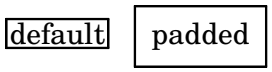
```
\markup {
  \box {
    default
```

```
  }
  \hspace #2
  \box {
    \pad-around #0.5 {
      padded
    }
  }
}
```



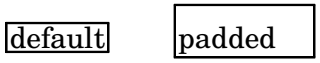
`\pad-markup` *amount* (number) *arg* (markup)
Add space around a markup object.

```
\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.

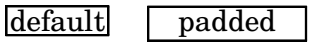
```
\markup {
  \box {
    default
  }
  \hspace #4
  \box {
    \pad-to-box #'(0 . 10) #'(0 . 3) {
      padded
    }
  }
}
```



`\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
    }
  }
}
```



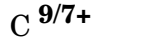
`\put-adjacent` *axis* (integer) *dir* (direction) *arg1* (markup) *arg2* (markup)
Put *arg2* next to *arg1*, without moving *arg1*.

`\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. `\raise` and `\super` raise objects in relation to their surrounding markups.

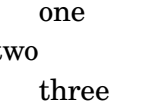
If the text object itself is positioned above or below the staff, then `\raise` cannot be used to move it, since the mechanism that positions it next to the staff cancels any shift made with `\raise`. For vertical positioning, use the `padding` and/or `extra-offset` properties.

```
\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
  }
}
```



`\right-column` *args* (list of markups)
Put `args` in a right-aligned column.

```
\markup {
  \right-column {
    one
    two
    three
  }
}
```

one
two
three

Used properties:

- `baseline-skip`

`\rotate` *ang* (number) *arg* (markup)
: Rotate object with *ang* degrees around its center.

```
\markup {
  default
  \hspace #2
  \rotate #45
  \line {
    rotated 45˚
  }
}
```



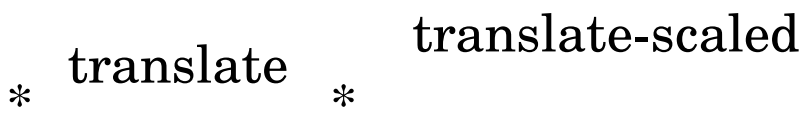
`\translate` *offset* (pair of numbers) *arg* (markup)
: Translate *arg* relative to its surroundings. *offset* is a pair of numbers representing the displacement in the X and Y axis.

```
\markup {
  *
  \translate #'(2 . 3)
  \line { translated two spaces right, three up }
}
```

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`.

```
\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
}
```

one two three

`\wordwrap-field` *symbol* (symbol)
: Wordwrap the data which has been assigned to *symbol*.

```
\header {
  title = "My title"
  description = "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:descr
    }
  }
}

\markup {
  \null
}
```

My title

`\wordwrap` *args* (list of markups)
: Simple wordwrap. Use `\override #'(line-width . X)` to set the line width, where *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.

Used properties:

- `text-direction` (1)
- `word-space`
- `line-width` (#f)
- `baseline-skip`

`\wordwrap-string` *arg* (string)

Wordwrap a string. Paragraphs may be separated with double newlines.

```
\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

Used properties:

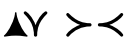
- `text-direction` (1)
- `word-space`
- `line-width`
- `baseline-skip`

B.8.3 Graphic

`\arrow-head` *axis* (integer) *dir* (direction) *filled* (boolean)

Produce an arrow head in specified direction and axis. Use the filled head if *filled* is specified.

```
\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)

Create a beam with the specified parameters.

```
\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 `thickness`, `circle-padding` and `font-size` properties to determine line thickness and padding around the markup.

```
\markup {
  \circle {
    Hi
  }
}
```



Used properties:

- `circle-padding` (0.2)
- `font-size` (0)
- `thickness` (1)

`\draw-circle` *radius* (number) *thickness* (number) *filled* (boolean)
A circle of radius *radius* and thickness *thickness*, optionally filled.

```
\markup {
  \draw-circle #2 #0.5 ##f
  \hspace #2
  \draw-circle #2 #0 ##t
}
```



`\draw-line` *dest* (pair of numbers)
A simple line.

```
\markup {
  \draw-line #'(4 . 4)
  \override #'(thickness . 5)
  \draw-line #'(-3 . 0)
}
```

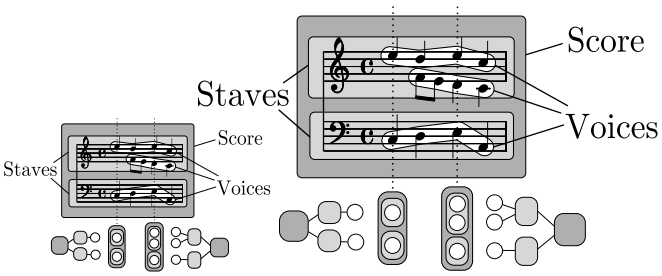


Used properties:

- `thickness` (1)

`\epsfile` *axis* (number) *size* (number) *file-name* (string)
Inline an EPS image. The image is scaled along *axis* to *size*.

```
\markup {
  \general-align #Y #DOWN {
    \epsfile #X #20 #"context-example.eps"
    \epsfile #Y #20 #"context-example.eps"
  }
}
```



`\filled-box` *xext* (pair of numbers) *yext* (pair of numbers) *blot* (number)
Draw a box with rounded corners of dimensions *xext* and *yext*. 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
  \filled-box #'(1 . 8) #'(0 . 7) #0.2
  \with-color #white
  \filled-box #'(-4.5 . -2.5) #'(3.5 . 5.5) #0.7
}
```



`\hbracket` *arg* (markup)
: Draw horizontal brackets around *arg*.

```
\markup {
  \hbracket {
    \line {
      one two three
    }
  }
}
```
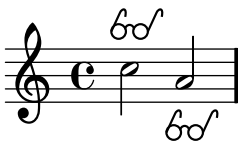
one two three

`\postscript` *str* (string)
: This inserts *str* directly into the output as a PostScript command string.

```
eyeglassesps = #"
  0.15 setlinewidth
  -0.9 0 translate
  1.1 1.1 scale
  1.2 0.7 moveto
  0.7 0.7 0.5 0 361 arc
  stroke
  2.20 0.70 0.50 0 361 arc
  stroke
  1.45 0.85 0.30 0 180 arc
  stroke
  0.20 0.70 moveto
  0.80 2.00 lineto
  0.92 2.26 1.30 2.40 1.15 1.70 curveto
  stroke
  2.70 0.70 moveto
  3.30 2.00 lineto
  3.42 2.26 3.80 2.40 3.65 1.70 curveto
  stroke"
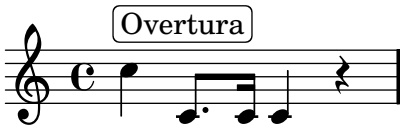
eyeglasses = \markup {
  \with-dimensions #'(0 . 4.4) #'(0 . 2.5)
  \postscript #eyeglassesps
}
```

```
\relative c'' {
  c2^\eyeglasses
  a2_\eyeglasses
}
```



`\rounded-box` *arg* (markup)
: Draw a box with rounded corners around *arg*. Looks at `thickness`, `box-padding` and `font-size` properties to determine line thickness and padding around the markup; the `corner-radius` property makes it possible to define another shape for the corners (default is 1).

```
c4^\markup {
  \rounded-box {
    Overtura
  }
}
c,8. c16 c4 r
```

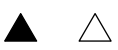


Used properties:

- `box-padding` (0.5)
- `font-size` (0)
- `corner-radius` (1)
- `thickness` (1)

`\triangle` *filled* (boolean)
: A triangle, either filled or empty.

```
\markup {
  \triangle ##t
  \hspace #2
  \triangle ##f
}
```



Used properties:

- `baseline-skip` (2)
- `font-size` (0)
- `thickness` (0.1)

`\with-url` *url* (string) *arg* (markup)
: Add a link to URL *url* around *arg*. This only works in the PDF backend.

```
\markup {
  \with-url #"http://lilypond.org/web/" {
    LilyPond ... \italic {
      music notation for everyone
    }
  }
}
```

LilyPond ... *music notation for everyone*

B.8.4 Music

`\doubleflat`

Draw a double flat symbol.

```
\markup {
  \doubleflat
}
```



`\doublesharp`

Draw a double sharp symbol.

```
\markup {
  \doublesharp
}
```


`\flat`

Draw a flat symbol.

```
\markup {
  \flat
}
```



`\musicglyph` *glyph-name* (string)

glyph-name is converted to a musical symbol; for example, `\musicglyph #"accidentals.natural"` selects the natural sign from the music font. See Abschnitt "Die Feta-Schriftart" in *Notationsreferenz* for a complete listing of the possible glyphs.

```
\markup {
  \musicglyph #"f"
  \musicglyph #"rests.2"
  \musicglyph #"clefs.G_change"
}
```



`\natural`

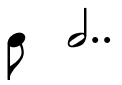
Draw a natural symbol.

```
\markup {
  \natural
}
```



`\note-by-number` *log* (number) *dot-count* (number) *dir* (number)
: Construct a note symbol, with stem. By using fractional values for *dir*, longer or shorter stems can be obtained.

```
\markup {
  \note-by-number #3 #0 #DOWN
  \hspace #2
  \note-by-number #1 #2 #0.8
}
```

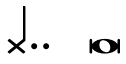


Used properties:

- `style` ('())
- `font-size` (0)

`\note` *duration* (string) *dir* (number)
: This produces a note with a stem pointing in *dir* direction, with the *duration* for the note head type and augmentation dots. For example, `\note #"4." #-0.75` creates a dotted quarter note, with a shortened down stem.

```
\markup {
  \override #'(style . cross) {
    \note #"4.." #UP
  }
  \hspace #2
  \note #"breve" #0
}
```



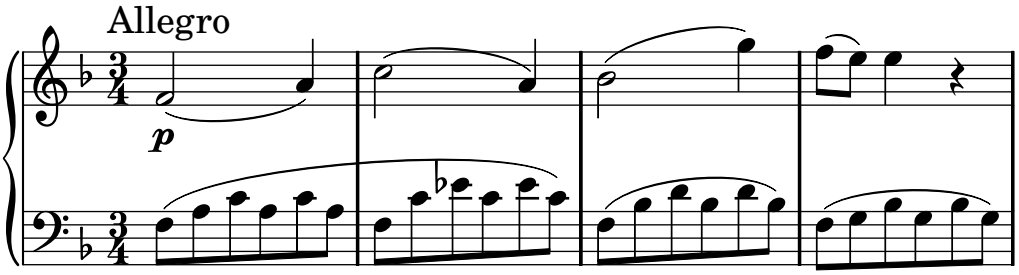
Used properties:

- `style` ('())
- `font-size` (0)

`\score` *score* (unknown)
: Inline an image of music.

```
\markup {
  \score {
    \new PianoStaff <<
      \new Staff \relative c' {
        \key f \major
        \time 3/4
        \mark \markup { Allegro }
        f2\p( a4)
        c2( a4)
```

```
            bes2( g'4)
            f8( e) e4 r
          }
          \new Staff \relative c {
            \clef bass
            \key f \major
            \time 3/4
            f8( a c a c a
            f c' es c es c)
            f,( bes d bes d bes)
            f( g bes g bes g)
          }
        >>
        \layout {
          indent = 0.0\cm
          \context {
            \Score
            \override RehearsalMark #'break-align-symbols =
              #'(time-signature key-signature)
            \override RehearsalMark #'self-alignment-X = #LEFT
          }
          \context {
            \Staff
            \override TimeSignature #'break-align-anchor-alignment = #LEFT
          }
        }
      }
    }
```



`\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
}
```



`\tied-lyric` *str* (string)
Like simple-markup, but use tie characters for ‚~‘ tilde symbols.

```
\markup {
  \tied-lyric #"Lasciate~i monti"
}
```

Lasciate i monti

B.8.5 Instrument Specific Markup

`\fret-diagram` *definition-string* (string)
Make a (guitar) fret diagram. For example, say

```
\markup \fret-diagram #"s:0.75;6-x;5-x;4-o;3-2;2-3;1-2;"
```

for fret spacing 3/4 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 (in staff spaces). Default: 0.05.
  - `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.

Here is an example

```
\markup \fret-diagram-terse #"x;x;o;2;3;2;"
```

for 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*.

For example,

```
\markup \fret-diagram-verbose
  #'((mute 6) (mute 5) (open 4)
     (place-fret 3 2) (place-fret 2 3) (place-fret 1 2))
```

produces a standard D chord diagram without fingering indications.

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`*`)`
: Place a fret playing indication on string *string-number* at fret *fret-number* with an optional fingering label *finger-value*. By default, the fret playing indicator is a solid dot. This can be changed by setting the value of the variable *dot-color*. 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.

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.

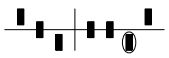
Possible elements in *definition-string*:

`^`	pedal is up
`-`	pedal is neutral
`v`	pedal is down
`|`	vertical divider line
`o`	the following pedal should be circled (indicating a change)

The function also checks if the string has the typical form of three pedals, then the divider and then the remaining four pedals. If not it prints out a warning. However, in any case, it will also print each symbol in the order as given. This means you can place the divider (even multiple dividers) anywhere you want, but you'll have to live with the warnings.

The appearance of the diagram can be tweaked inter alia using the size property of the TextScript grob (`\override Voice.TextScript #'size = #0.3`) for the overall, the thickness property (`\override Voice.TextScript #'thickness = #3`) for the line thickness of the horizontal line and the divider. The remaining configuration (box sizes, offsets and spaces) is done by the harp-pedal-details list of properties (`\override Voice.TextScript #'harp-pedal-details #'box-width = #1`). It contains the following settings: `box-offset` (vertical shift of the box center for up/down pedals), `box-width`, `box-height`, `space-before-divider` (the spacing between two boxes before the divider) and `space-after-divider` (box spacing after the divider).

```
\markup \harp-pedal #"^-v|--ov^"
```



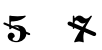
Used properties:

- `thickness` (0.5)
- `harp-pedal-details`
- `size` (1.0)

B.8.6 Other

`\backslashed-digit` *num* (integer)
: A feta number, with 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 character. Characters encoded in hexadecimal format require the prefix `#x`.

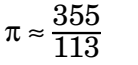
```
\markup {
  \char #65 \char ##x00a9
}
```



`\fraction` *arg1* (markup) *arg2* (markup)
: Make a fraction of two markups.

```
\markup {

  \fraction 355 113
}
```



Used properties:

- `font-size` (0)

`\fromproperty` *symbol* (symbol)
: Read the *symbol* from property settings, and produce a stencil from the markup contained within. If *symbol* is not defined, it returns an empty markup.

```
\header {
  myTitle = "myTitle"
  title = \markup {
    from
    \italic
    \fromproperty #'header:myTitle
  }
}
\markup {
  \null
}
```

from *myTitle*

`\lookup` *glyph-name* (string)
: Lookup a glyph by name.

```
\markup {
  \override #'(font-encoding . fetaBraces) {
    \lookup #"brace200"
    \hspace #2
    \rotate #180
    \lookup #"brace180"
  }
}
```



`\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
}
```

I AA

`\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
}
```

J AB

`\null`

An empty markup with extents of a single point.

```
\markup {
  \null
}
```

`\on-the-fly` *procedure* (symbol) *arg* (markup)

Apply the *procedure* markup command to *arg*. *procedure* should take a single argument.

`\override` *new-prop* (pair) *arg* (markup)

Add the argument *new-prop* to the property list. Properties may be any property supported by Abschnitt “font-interface” in *Referenz der Interna*, Abschnitt “text-interface” in *Referenz der Interna* and Abschnitt “instrument-specific-markup-interface” in *Referenz der Interna*.

```
\markup {
  \line {
    \column {
      default
      baseline-skip
    }
    \hspace #2
    \override #'(baseline-skip . 4) {
      \column {
        increased
        baseline-skip
      }
    }
  }
}
```

default baseline-skip increased baseline-skip

`\page-ref` *label* (symbol) *gauge* (markup) *default* (markup)

Reference to a page number. *label* is the label set on the referenced page (using the `\label` command), *gauge* a markup used to estimate the maximum width of the page number, and *default* the value to display when *label* is not found.

`\slashed-digit` *num* (integer)

A feta number, with slash. This is for use in the context of figured bass notation.

```
\markup {
  \slashed-digit #5
  \hspace #2
  \override #'(thickness . 3)
  \slashed-digit #7
}
```

5 7

Used properties:

- `thickness` (1.6)
- `font-size` (0)

`\stencil` *stil* (unknown)
Use a stencil 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.

`\transparent` *arg* (markup)
Make *arg* transparent.

```
\markup {
  \transparent {
    invisible text
  }
}
```

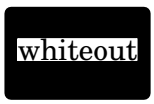
`\verbatim-file` *name* (string)
Read the contents of file *name*, and include it verbatim.

```
\markup {
  \verbatim-file #"simple.ly"
}
```

```
%% A simple piece in LilyPond, a scale.
\relative c' {
  c d e f g a b c
}
%% Optional helper for automatic updating by convert-ly.  May be omit
\version "2.12.0"
```

`\whiteout` *arg* (markup)
Provide a white background for *arg*.

```
\markup {
  \combine
    \filled-box #'(-1 . 10) #'(-3 . 4) #1
    \whiteout whiteout
}
```



`\with-color` *color* (list) *arg* (markup)
Draw *arg* in color specified by *color*.

```
\markup {
  \with-color #red
  red
  \hspace #2
  \with-color #green
  green
  \hspace #2
  \with-color #blue
  blue
}
```

red green blue

`\with-dimensions` *x* (pair of numbers) *y* (pair of numbers) *arg* (markup)
Set the dimensions of *arg* to *x* and *y*.

B.9 Text markup list commands

The following commands can all be used with `\markuplines`.

`\column-lines` *args* (list of markups)
Like `\column`, but return a list of lines instead of a single markup. `baseline-skip` determines the space between each markup in *args*.

Used properties:

- `baseline-skip`

`\justified-lines` *args* (list of markups)
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* (list of markups)
Like `\override`, for markup lists.

`\wordwrap-internal` *justify* (boolean) *args* (list of markups)
Internal markup list command used to define `\justify` and `\wordwrap`.

Used properties:

- `text-direction` (1)
- `word-space`
- `line-width` (#f)

`\wordwrap-lines` *args* (list of markups)
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`
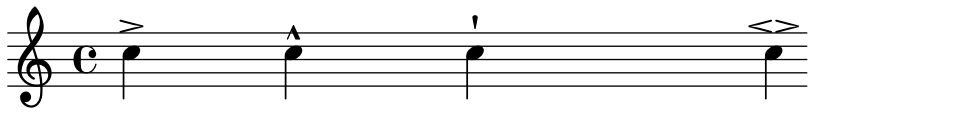
`\wordwrap-string-internal` *justify* (boolean) *arg* (string)
Internal markup list command used to define `\justify-string` and `\wordwrap-string`.
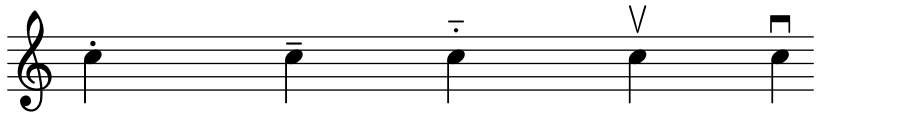
Used properties:
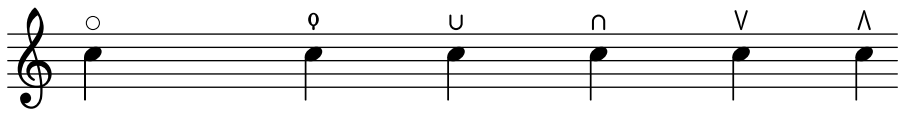
- `text-direction` (1)
- `word-space`
- `line-width`
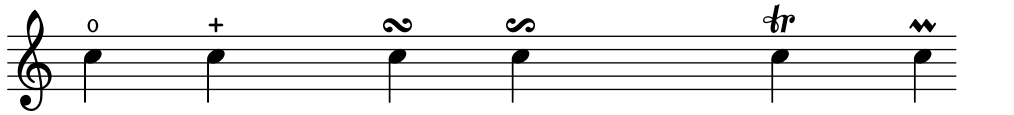
B.10 Liste der Arktikulationszeichen

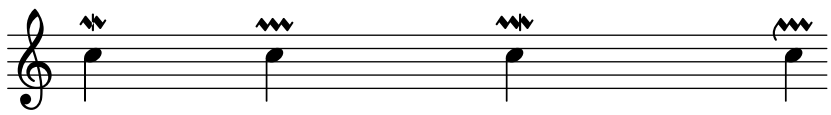
Hier ist eine Liste, die alle möglichen Zeichen darstellt:

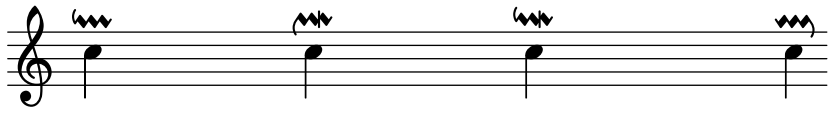


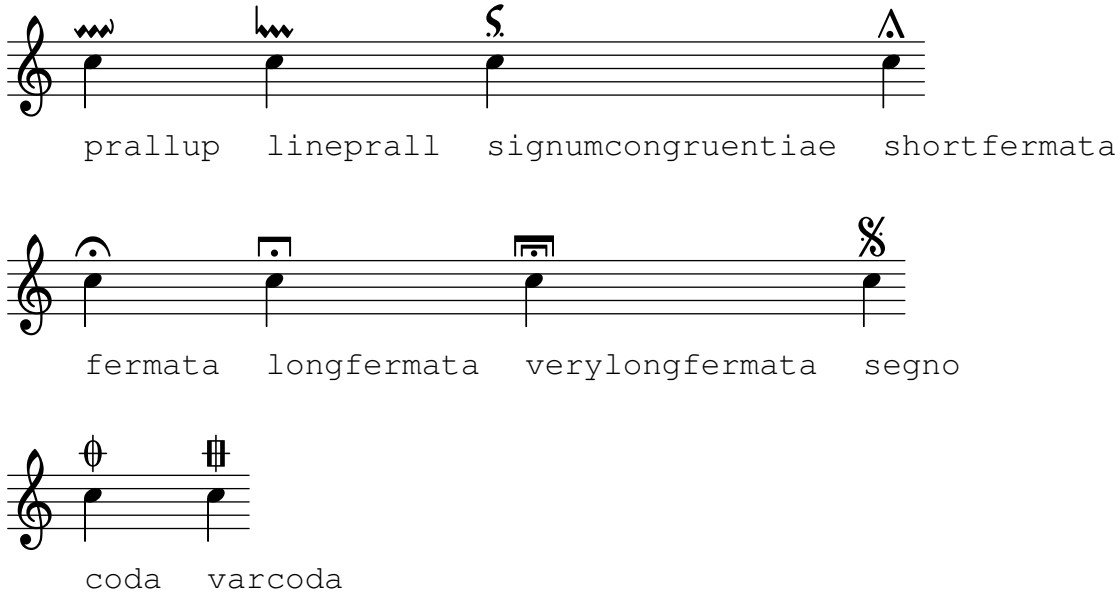




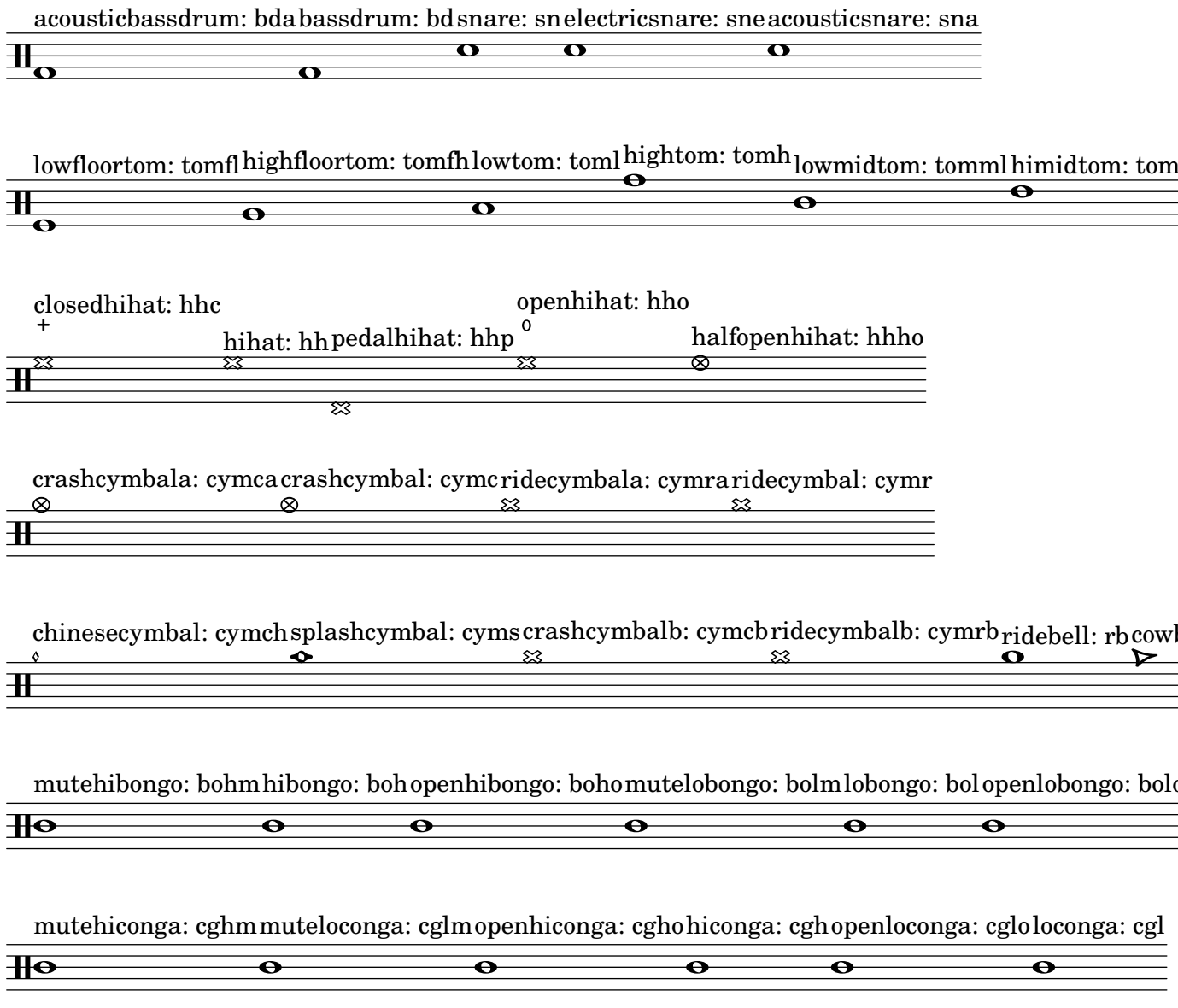


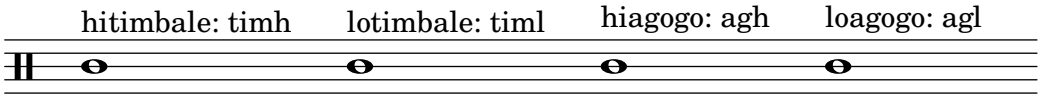


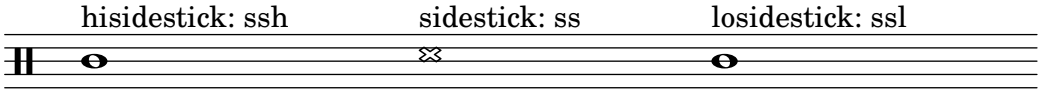


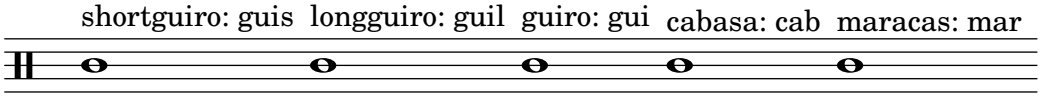


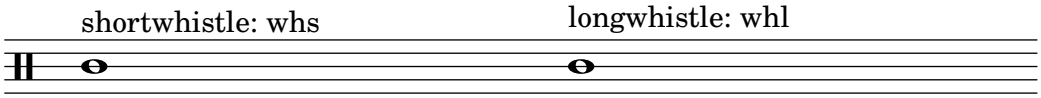
B.11 Schlagzeugnoten

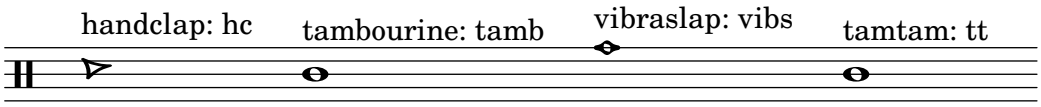


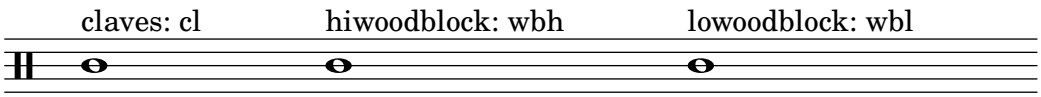


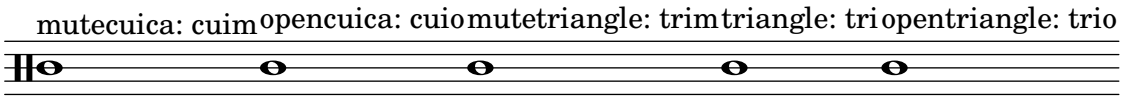


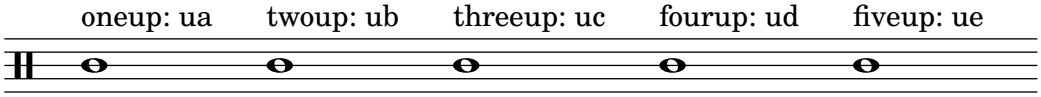


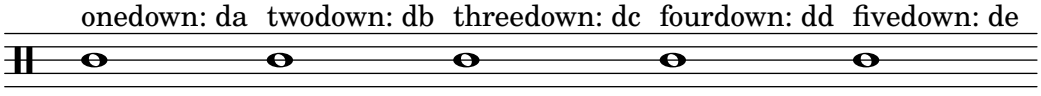












B.12 Alle Kontexteigenschaften

`aDueText` (markup)
: Text to print at a unisono passage.

`alignAboveContext` (string)
: Where to insert newly created context in vertiical alignment.

`alignBassFigureAccidentals` (boolean)
: If true, then the accidentals are aligned in bass figure context.

`alignBelowContext` (string)
: Where to insert newly created context in vertiical alignment.

`associatedVoice` (string)
: Name of the `Voice` that has the melody for this `Lyrics` line.

`autoAccidentals` (list)
: List of different ways to typeset an accidental.

  For determining when to print an accidental, several different rules are tried. The rule that gives the highest number of accidentals is used.

  Each entry in the list is either a symbol or a procedure.

  symbol
  : The symbol is the name of the context in which the following rules are to be applied. For example, if *context* is Abschnitt "Score" in *Referenz der Interna* then all staves share accidentals, and if *context* is Abschnitt "Staff" in *Referenz der Interna* then all voices in the same staff share accidentals, but staves do not.

  procedure
  : The procedure represents an accidental rule to be applied to the previously specified context.

    The procedure takes the following arguments:

    `context`
    : The current context to which the rule should be applied.

    `pitch`
    : The pitch of the note to be evaluated.

    `barnum`
    : The current bar number.

    `measurepos`
    : The current measure position.

    The procedure returns a pair of booleans. The first states whether an extra natural should be added. The second states whether an accidental should be printed. `(#t . #f)` does not make sense.

`autoBeamCheck` (procedure)
: A procedure taking three arguments, *context*, *dir* [start/stop (-1 or 1)], and *test* [shortest note in the beam]. A non-`#f` return value starts or stops the auto beam.

`autoBeamSettings` (list)
: Specifies when automatically generated beams should begin and end. See Abschnitt "Einstellung von automatischen Balken" in *Notationsreferenz* for more information.

`autoBeaming` (boolean)
: If set to true then beams are generated automatically.

`autoCautionaries` (list)
: List similar to `autoAccidentals`, but it controls cautionary accidentals rather than normal ones. Both lists are tried, and the one giving the most accidentals wins. In case of draw, a normal accidental is typeset.

`automaticBars` (boolean)
: If set to false then bar lines will not be printed automatically; they must be explicitly created with a `\bar` command. Unlike the `\cadenzaOn` keyword, measures are still counted. Bar line generation will resume according to that count if this property is unset.

`barAlways` (boolean)
: If set to true a bar line is drawn after each note.

`barCheckSynchronize` (boolean)
: If true then reset `measurePosition` when finding a bar check.

`barNumberVisibility` (procedure)
: A Procedure that takes an integer and returns whether the corresponding bar number should be printed.

`bassFigureFormatFunction` (procedure)
: A procedure that is called to produce the formatting for a `BassFigure` grob. It takes a list of `BassFigureEvent`s, a context, and the grob to format.

`bassStaffProperties` (list)
: An alist of property settings to apply for the down staff of `PianoStaff`. Used by `\autochange`.

`beatGrouping` (list)
: A list of beatgroups, e.g., in 5/8 time `'(2 3)`.

`beatLength` (moment)
: The length of one beat in this time signature.

`chordChanges` (boolean)
: Only show changes in chords scheme?

`chordNameExceptions` (list)
: An alist of chord exceptions. Contains (*`chord . markup`*) entries.

`chordNameExceptionsFull` (list)
: An alist of full chord exceptions. Contains (*`chord . markup`*) entries.

`chordNameExceptionsPartial` (list)
: An alist of partial chord exceptions. Contains (*`chord . (prefix-markup suffix-markup)`*) entries.

`chordNameFunction` (procedure)
: The function that converts lists of pitches to chord names.

`chordNameSeparator` (markup)
: The markup object used to separate parts of a chord name.

`chordNoteNamer` (procedure)
: A function that converts from a pitch object to a text markup. Used for single pitches.

`chordPrefixSpacer` (number)
: The space added between the root symbol and the prefix of a chord name.

`chordRootNamer` (procedure)
: A function that converts from a pitch object to a text markup. Used for chords.

`clefGlyph` (string)
: Name of the symbol within the music font.

`clefOctavation` (integer)
: Add this much extra octavation. Values of 7 and -7 are common.

`clefPosition` (number)
: Where should the center of the clef symbol go, measured in half staff spaces from the center of the staff.

`completionBusy` (boolean)
: Whether a completion-note head is playing.

`connectArpeggios` (boolean)
: If set, connect arpeggios across piano staff.

`countPercentRepeats` (boolean)
: If set, produce counters for percent repeats.

`createKeyOnClefChange` (boolean)
: Print a key signature whenever the clef is changed.

`createSpacing` (boolean)
: Create `StaffSpacing` objects? Should be set for staves.

`crescendoSpanner` (symbol)
: The type of spanner to be used for crescendi. Available values are '`hairpin`' and '`text`'. If unset, a hairpin crescendo is used.

`crescendoText` (markup)
: The text to print at start of non-hairpin crescendo, i.e., '`cresc.`'.

`currentBarNumber` (integer)
: Contains the current barnumber. This property is incremented at every bar line.

`decrescendoSpanner` (symbol)
: The type of spanner to be used for decrescendi. Available values are '`hairpin`' and '`text`'. If unset, a hairpin decrescendo is used.

`decrescendoText` (markup)
: The text to print at start of non-hairpin decrescendo, i.e., '`dim.`'.

`defaultBarType` (string)
: Set the default type of bar line. See `whichBar` for information on available bar types.

  This variable is read by Abschnitt "Timing_translator" in *Referenz der Interna* at Abschnitt "Score" in *Referenz der Interna* level.

`doubleRepeatType` (string)
: Set the default bar line for double repeats.

`doubleSlurs` (boolean)
: If set, two slurs are created for every slurred note, one above and one below the chord.

`drumPitchTable` (hash table)
: A table mapping percussion instruments (symbols) to pitches.

`drumStyleTable` (hash table)
: A hash table which maps drums to layout settings. Predefined values: '`drums-style`', '`timbales-style`', '`congas-style`', '`bongos-style`', and '`percussion-style`'.

  The layout style is a hash table, containing the drum-pitches (e.g., the symbol '`hihat`') as keys, and a list `(`*`notehead-style script vertical-position`*`)` as values.

`explicitClefVisibility` (vector)
: '`break-visibility`' function for clef changes.

`explicitKeySignatureVisibility` (vector)
: '`break-visibility`' function for explicit key changes. '`\override`' of the `break-visibility` property will set the visibility for normal (i.e., at the start of the line) key signatures.

`extendersOverRests` (boolean)
: Whether to continue extenders as they cross a rest.

`extraNatural` (boolean)
: Whether to typeset an extra natural sign before accidentals changing from a non-natural to another non-natural.

`figuredBassAlterationDirection` (direction)
: Where to put alterations relative to the main figure.

`figuredBassCenterContinuations` (boolean)
: Whether to vertically center pairs of extender lines. This does not work with three or more lines.

`figuredBassFormatter` (procedure)
: A routine generating a markup for a bass figure.

`figuredBassPlusDirection` (direction)
: Where to put plus signs relative to the main figure.

`fingeringOrientations` (list)
: A list of symbols, containing '`left`', '`right`', '`up`' and/or '`down`'. This list determines where fingerings are put relative to the chord being fingered.

`firstClef` (boolean)
: If true, create a new clef when starting a staff.

`followVoice` (boolean)
: If set, note heads are tracked across staff switches by a thin line.

`fontSize` (number)
: The relative size of all grobs in a context.

`forbidBreak` (boolean)
: If set to `##t`, prevent a line break at this point.

`forceClef` (boolean)
: Show clef symbol, even if it has not changed. Only active for the first clef after the property is set, not for the full staff.

`gridInterval` (moment)
: Interval for which to generate `GridPoint`s.

`harmonicAccidentals` (boolean)
: If set, harmonic notes in chords get accidentals.

`harmonicDots` (boolean)
: If set, harmonic notes in dotted chords get dots.

`highStringOne` (boolean)
: Whether the first string is the string with highest pitch on the instrument. This used by the automatic string selector for tablature notation.

`ignoreBarChecks` (boolean)
: Ignore bar checks.

`ignoreFiguredBassRest` (boolean)
: Don't swallow rest events.

`ignoreMelismata` (boolean)
: Ignore melismata for this Abschnitt "Lyrics" in *Referenz der Interna* line.

`implicitBassFigures` (list)
: A list of bass figures that are not printed as numbers, but only as extender lines.

`implicitTimeSignatureVisibility` (vector)
: break visibility for the default time signature.

`instrumentCueName` (markup)
: The name to print if another instrument is to be taken.

`instrumentEqualizer` (procedure)
: A function taking a string (instrument name), and returning a `(`*`min . max`*`)` pair of numbers for the loudness range of the instrument.

`instrumentName` (markup)
: The name to print left of a staff. The `instrument` property labels the staff in the first system, and the `instr` property labels following lines.

`instrumentTransposition` (pitch)
: Define the transposition of the instrument. Its value is the pitch that sounds like middle C. This is used to transpose the MIDI output, and `\quote`s.

`internalBarNumber` (integer)
: Contains the current barnumber. This property is used for internal timekeeping, among others by the `Accidental_engraver`.

`keepAliveInterfaces` (list)
: A list of symbols, signifying grob interfaces that are worth keeping a staff with `remove-empty` set around for.

`keyAlterationOrder` (list)
: An alist that defines in what order alterations should be printed. The format is `(`*`step . alter`*`)`, where *step* is a number from 0 to 6 and *alter* from -2 (sharp) to 2 (flat).

`keySignature` (list)
: The current key signature. This is an alist containing `(`*`step . alter`*`)` or `((`*`octave . step`*`) . alter)`, where *step* is a number in the range 0 to 6 and *alter* a fraction, denoting alteration. For alterations, use symbols, e.g. `keySignature = #`((6 . ,FLAT))`.

`lyricMelismaAlignment` (direction)
: Alignment to use for a melisma syllable.

`majorSevenSymbol` (markup)
: How should the major 7th be formatted in a chord name?

`markFormatter` (procedure)
: A procedure taking as arguments the context and the rehearsal mark. It should return the formatted mark as a markup object.

`maximumFretStretch` (number)
: Don't allocate frets further than this from specified frets.

`measureLength` (moment)
: Length of one measure in the current time signature.

`measurePosition` (moment)
: How much of the current measure have we had. This can be set manually to create incomplete measures.

`melismaBusyProperties` (list)
: A list of properties (symbols) to determine whether a melisma is playing. Setting this property will influence how lyrics are aligned to notes. For example, if set to `#'(melismaBusy beamMelismaBusy)`, only manual melismata and manual beams are considered. Possible values include `melismaBusy`, `slurMelismaBusy`, `tieMelismaBusy`, and `beamMelismaBusy`.

`metronomeMarkFormatter` (procedure)
: How to produce a metronome markup. Called with four arguments: text, duration, count and context.

`middleCClefPosition` (number)
: The position of the middle C, as determined only by the clef. This can be calculated by looking at `clefPosition` and `clefGlyph`.

`middleCOffset` (number)
: The offset of middle C from the position given by `middleCClefPosition` This is used for ottava brackets.

`middleCPosition` (number)
: The place of the middle C, measured in half staff-spaces. Usually determined by looking at `middleCClefPosition` and `middleCOffset`.

`midiInstrument` (string)
: Name of the MIDI instrument to use.

`midiMaximumVolume` (number)
: Analogous to `midiMinimumVolume`.

`midiMinimumVolume` (number)
: Set the minimum loudness for MIDI. Ranges from 0 to 1.

`minimumFret` (number)
: The tablature auto string-selecting mechanism selects the highest string with a fret at least `minimumFret`.

`minimumPageTurnLength` (moment)
: Minimum length of a rest for a page turn to be allowed.

`minimumRepeatLengthForPageTurn` (moment)
: Minimum length of a repeated section for a page turn to be allowed within that section.

`noteToFretFunction` (procedure)
: How to produce a fret diagram. Parameters: A list of note events and a list of tabstring events.

`ottavation` (markup)
: If set, the text for an ottava spanner. Changing this creates a new text spanner.

`output` (unknown)
: The output produced by a score-level translator during music interpretation.

`pedalSostenutoStrings` (list)
: See `pedalSustainStrings`.

`pedalSostenutoStyle` (symbol)
: See `pedalSustainStyle`.

`pedalSustainStrings` (list)
: A list of strings to print for sustain-pedal. Format is `(up updown down)`, where each of the three is the string to print when this is done with the pedal.

`pedalSustainStyle` (symbol)
: A symbol that indicates how to print sustain pedals: `text`, `bracket` or `mixed` (both).

`pedalUnaCordaStrings` (list)
: See `pedalSustainStrings`.

`pedalUnaCordaStyle` (symbol)
: See `pedalSustainStyle`.

`predefinedDiagramTable` (hash table)
: The hash table of predefined fret diagrams to use in FretBoards.

`printKeyCancellation` (boolean)
: Print restoration alterations before a key signature change.

`printOctaveNames` (boolean)
: Print octave marks for the `NoteNames` context.

`printPartCombineTexts` (boolean)
: Set ‚Solo‘ and ‚A due‘ texts in the part combiner?

`proportionalNotationDuration` (moment)
: Global override for shortest-playing duration. This is used for switching on proportional notation.

`recordEventSequence` (procedure)
: When `Recording_group_engraver` is in this context, then upon termination of the context, this function is called with current context and a list of music objects. The list of contains entries with start times, music objects and whether they are processed in this context.

`rehearsalMark` (integer)
: The last rehearsal mark printed.

`repeatCommands` (list)
: This property is a list of commands of the form `(list 'volta` *x*`)`, where *x* is a string or `#f`. `'end-repeat` is also accepted as a command.

`repeatCountVisibility` (procedure)
: A procedure taking as arguments an integer and context, returning whether the corresponding percent repeat number should be printed when `countPercentRepeats` is set.

`restNumberThreshold` (number)
: If a multimeasure rest has more measures than this, a number is printed.

`shapeNoteStyles` (vector)
: Vector of symbols, listing style for each note head relative to the tonic (qv.) of the scale.

`shortInstrumentName` (markup)
: See `instrument`.

`shortVocalName` (markup)
: Name of a vocal line, short version.

`skipBars` (boolean)
: If set to true, then skip the empty bars that are produced by multimeasure notes and rests. These bars will not appear on the printed output. If not set (the default), multimeasure notes and rests expand into their full length, printing the appropriate number of empty bars so that synchronization with other voices is preserved.

```
{
  r1 r1*3 R1*3
  \set Score.skipBars= ##t
  r1*3 R1*3
}
```

`skipTypesetting` (boolean)
: If true, no typesetting is done, speeding up the interpretation phase. Useful for debugging large scores.

`soloIIText` (markup)
: The text for the start of a solo for voice ‚two‘ when part-combining.

`soloText` (markup)
: The text for the start of a solo when part-combining.

`squashedPosition` (integer)
: Vertical position of squashing for Abschnitt “Pitch_squash_engraver” in *Referenz der Interna*.

`staffLineLayoutFunction` (procedure)
: Layout of staff lines, `traditional`, or `semitone`.

`stanza` (markup)
: Stanza ‚number‘ to print before the start of a verse. Use in `Lyrics` context.

`stemLeftBeamCount` (integer)
: Specify the number of beams to draw on the left side of the next note. Overrides automatic beaming. The value is only used once, and then it is erased.

`stemRightBeamCount` (integer)
: See `stemLeftBeamCount`.

`stringNumberOrientations` (list)
: See `fingeringOrientations`.

`stringOneTopmost` (boolean)
: Whether the first string is printed on the top line of the tablature.

`stringTunings` (list)
: The tablature strings tuning. It is a list of the pitch (in semitones) of each string (starting with the lower one).

`strokeFingerOrientations` (list)
: See `fingeringOrientations`.

`subdivideBeams` (boolean)
: If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

`suggestAccidentals` (boolean)
: If set, accidentals are typeset as cautionary suggestions over the note.

`systemStartDelimiter` (symbol)
: Which grob to make for the start of the system/staff? Set to `SystemStartBrace`, `SystemStartBracket` or `SystemStartBar`.

`systemStartDelimiterHierarchy` (pair)
: A nested list, indicating the nesting of a start delimiters.

`tablatureFormat` (procedure)
: A function formatting a tablature note head. Called with three arguments: string number, context and event. It returns the text as a string.

`tempoHideNote` (boolean)
: Hide the note=count in tempo marks.

`tempoText` (markup)
: Text for tempo marks.

`tempoUnitCount` (number)
: Count for specifying tempo.

`tempoUnitDuration` (duration)
: Unit for specifying tempo.

`tempoWholesPerMinute` (moment)
: The tempo in whole notes per minute.

`tieWaitForNote` (boolean)
: If true, tied notes do not have to follow each other directly. This can be used for writing out arpeggios.

`timeSignatureFraction` (pair of numbers)
: A pair of numbers, signifying the time signature. For example, `#'(4 . 4)` is a 4/4 time signature.

`timing` (boolean)
: Keep administration of measure length, position, bar number, etc.? Switch off for cadenzas.

`tonic` (pitch)
: The tonic of the current scale.

`trebleStaffProperties` (list)
: An alist of property settings to apply for the up staff of `PianoStaff`. Used by `\autochange`.

`tremoloFlags` (integer)
: The number of tremolo flags to add if no number is specified.

`tupletFullLength` (boolean)
: If set, the tuplet is printed up to the start of the next note.

`tupletFullLengthNote` (boolean)
: If set, end at the next note, otherwise end on the matter (time signatures, etc.) before the note.

`tupletSpannerDuration` (moment)
: Normally, a tuplet bracket is as wide as the `\times` expression that gave rise to it. By setting this property, you can make brackets last shorter.

```
{
  \set tupletSpannerDuration = #(ly:make-moment 1 4)
  \times 2/3 { c8 c c c c c }
}
```

`useBassFigureExtenders` (boolean)
: Whether to use extender lines for repeated bass figures.

`verticallySpacedContexts` (list)
: List of symbols, containing context names whose vertical axis groups should be taken into account for vertical spacing of systems.

`vocalName` (markup)
: Name of a vocal line.

`voltaSpannerDuration` (moment)
: This specifies the maximum duration to use for the brackets printed for `\alternative`. This can be used to shrink the length of brackets in the situation where one alternative is very large.

`whichBar` (string)
: This property is read to determine what type of bar line to create.

  Example:

```
\set Staff.whichBar = "|:"
```

This will create a start-repeat bar in this staff only. Valid values are described in Abschnitt “bar-line-interface” in *Referenz der Interna*.

B.13 Eigenschaften des Layouts

`X-extent` (pair of numbers)
: Hard coded extent in X direction.

`X-offset` (number)
: The horizontal amount that this object is moved relative to its X-parent.

`Y-extent` (pair of numbers)
: Hard coded extent in Y direction.

`Y-offset` (number)
: The vertical amount that this object is moved relative to its Y-parent.

`add-stem-support` (boolean)
: If set, the `Stem` object is included in this script's support.

`after-line-breaking` (boolean)
: Dummy property, used to trigger callback for `after-line-breaking`.

`align-dir` (direction)
: Which side to align? `-1`: left side, `0`: around center of width, `1`: right side.

`allow-loose-spacing` (boolean)
: If set, column can be detached from main spacing.

`allow-span-bar` (boolean)
: If false, no inter-staff bar line will be created below this bar line.

`alteration` (number)
: Alteration numbers for accidental.

`alteration-alist` (list)
: List of `(`*`pitch . accidental`*`)` pairs for key signature.

`annotation` (string)
: Annotate a grob for debug purposes.

`arpeggio-direction` (direction)
: If set, put an arrow on the arpeggio squiggly line.

`arrow-length` (number)
: Arrow length.

`arrow-width` (number)
: Arrow width.

`auto-knee-gap` (dimension, in staff space)
: If a gap is found between note heads where a horizontal beam fits that is larger than this number, make a kneed beam.

`average-spacing-wishes` (boolean)
: If set, the spacing wishes are averaged over staves.

`avoid-note-head` (boolean)
: If set, the stem of a chord does not pass through all note heads, but starts at the last note head.

`avoid-slur` (symbol)
: Method of handling slur collisions. Choices are `around`, `inside`, `outside`. If unset, scripts and slurs ignore each other. `around` only moves the script if there is a collision; `outside` always moves the script.

`axes` (list)
: List of axis numbers. In the case of alignment grobs, this should contain only one number.

`bar-size` (dimension, in staff space)
: The size of a bar line.

`base-shortest-duration` (moment)
: Spacing is based on the shortest notes in a piece. Normally, pieces are spaced as if notes at least as short as this are present.

`baseline-skip` (dimension, in staff space)
: Distance between base lines of multiple lines of text.

`beam-thickness` (dimension, in staff space)
: Beam thickness, measured in `staff-space` units.

`beam-width` (dimension, in staff space)
: Width of the tremolo sign.

`beamed-stem-shorten` (list)
: How much to shorten beamed stems, when their direction is forced. It is a list, since the value is different depending on the number of flags and beams.

`beaming` (pair)
: Pair of number lists. Each number list specifies which beams to make. `0` is the central beam, `1` is the next beam toward the note, etc. This information is used to determine how to connect the beaming patterns from stem to stem inside a beam.

`before-line-breaking` (boolean)
: Dummy property, used to trigger a callback function.

`between-cols` (pair)
: Where to attach a loose column to.

`bound-details` (list)
: An alist of properties for determining attachments of spanners to edges.

`bound-padding` (number)
: The amount of padding to insert around spanner bounds.

`bracket-flare` (pair of numbers)
: A pair of numbers specifying how much edges of brackets should slant outward. Value `0.0` means straight edges.

`bracket-visibility` (boolean or symbol)
: This controls the visibility of the tuplet bracket. Setting it to false prevents printing of the bracket. Setting the property to `if-no-beam` makes it print only if there is no beam associated with this tuplet bracket.

`break-align-anchor` (number)
: Grobs aligned to this break-align grob will have their X-offsets shifted by this number. In bar lines, for example, this is used to position grobs relative to the (visual) center of the bar line.

`break-align-anchor-alignment` (number)
: Read by `ly:break-aligned-interface::calc-extent-aligned-anchor` for aligning an anchor to a grob's extent

`break-align-orders` (vector)
: Defines the order in which prefatory matter (clefs, key signatures) appears. The format is a vector of length 3, where each element is one order for end-of-line, middle of line, and start-of-line, respectively. An order is a list of symbols.

  For example, clefs are put after key signatures by setting

```
\override Score.BreakAlignment #'break-align-orders =
  #(make-vector 3 '(span-bar
                    breathing-sign
                    staff-bar
                    key
                    clef
                    time-signature))
```

`break-align-symbol` (symbol)
: This key is used for aligning and spacing breakable items.

`break-align-symbols` (list)
: A list of symbols that determine which break-aligned grobs to align this to. If the grob selected by the first symbol in the list is invisible due to break-visibility, we will align to the next grob (and so on).

`break-overshoot` (pair of numbers)
: How much does a broken spanner stick out of its bounds?

`break-visibility` (vector)
: A vector of 3 booleans, `#(`*`end-of-line unbroken begin-of-line`*`)`. `#t` means visible, `#f` means killed.

`breakable` (boolean)
: Allow breaks here.

`c0-position` (integer)
: An integer indicating the position of middle C.

`circled-tip` (boolean)
: Put a circle at start/end of hairpins (al/del niente).

`clip-edges` (boolean)
: Allow outward pointing beamlets at the edges of beams?

`collapse-height` (dimension, in staff space)
: Minimum height of system start delimiter. If equal or smaller, the bracket/brace/line is removed.

`color` (list)
: The color of this grob.

`common-shortest-duration` (moment)
: The most common shortest note length. This is used in spacing. Enlarging this sets the score tighter.

`concaveness` (number)
: A beam is concave if its inner stems are closer to the beam than the two outside stems. This number is a measure of the closeness of the inner stems. It is used for damping the slope of the beam.

`connect-to-neighbor` (pair)
: Pair of booleans, indicating whether this grob looks as a continued break.

`control-points` (list)
: List of offsets (number pairs) that form control points for the tie, slur, or bracket shape. For Béziers, this should list the control points of a third-order Bézier curve.

`damping` (number)
: Amount of beam slope damping.

`dash-fraction` (number)
: Size of the dashes, relative to `dash-period`. Should be between `0.0` (no line) and `1.0` (continuous line).

`dash-period` (number)
: The length of one dash together with whitespace. If negative, no line is drawn at all.

`default-direction` (direction)
: Direction determined by note head positions.

`digit-names` (unknown)
: Names for string finger digits.

`direction` (direction)
: If `side-axis` is `0` (or `#X`), then this property determines whether the object is placed `#LEFT`, `#CENTER` or `#RIGHT` with respect to the other object. Otherwise, it determines whether the object is placed `#UP`, `#CENTER` or `#DOWN`. Numerical values may also be used: `#UP=1`, `#DOWN=-1`, `#LEFT=-1`, `#RIGHT=1`, `#CENTER=0`.

`dot-count` (integer)
: The number of dots.

`dot-negative-kern` (number)
: The space to remove between a dot and a slash in percent repeat glyphs. Larger values bring the two elements closer together.

`dot-placement-list` (list)
: List consisting of `(`*`description string-number fret-number finger-number`*`)` entries used to define fret diagrams.

`duration-log` (integer)
: The 2-log of the note head duration, i.e., `0` = whole note, `1` = half note, etc.

`eccentricity` (number)
: How asymmetrical to make a slur. Positive means move the center to the right.

`edge-height` (pair)
: A pair of numbers specifying the heights of the vertical edges: `(`*`left-height . right-height`*`)`.

`edge-text` (pair)
: A pair specifying the texts to be set at the edges: `(`*`left-text . right-text`*`)`.

`expand-limit` (integer)
: Maximum number of measures expanded in church rests.

`extra-X-extent` (pair of numbers)
: A grob is enlarged in X dimension by this much.

`extra-Y-extent` (pair of numbers)
: A grob is enlarged in Y dimension by this much.

`extra-dy` (number)
: Slope glissandi this much extra.

`extra-offset` (pair of numbers)
: A pair representing an offset. This offset is added just before outputting the symbol, so the typesetting engine is completely oblivious to it. The values are measured in `staff-space` units of the staff's `StaffSymbol`.

`extra-spacing-height` (pair of numbers)
: In the horizontal spacing problem, we increase the height of each item by this amount (by adding the ‚car‘ to the bottom of the item and adding the ‚cdr‘ to the top of the item. In order to make a grob infinitely high (to prevent the horizontal spacing problem from placing any other grobs above or below this grob), set this to `(-inf.0 . +inf.0)`.

`extra-spacing-width` (pair of numbers)
: In the horizontal spacing problem, we pad each item by this amount (by adding the ‚car‘ on the left side of the item and adding the ‚cdr‘ on the right side of the item). In order to make a grob take up no horizontal space at all, set this to `(+inf.0 . -inf.0)`.

`flag` (unknown)
: A function returning the full flag stencil for the `Stem`, which is passed to the function as the only argument. The default ly:stem::calc-stencil function uses the `flag-style` property to determine the correct glyph for the flag. By providing your own function, you can create arbitrary flags.

`flag-count` (number)
: The number of tremolo beams.

`flag-style` (symbol)
: A symbol determining what style of flag glyph is typeset on a `Stem`. Valid options include `'()` for standard flags, `'mensural` and `'no-flag`, which switches off the flag.

`font-encoding` (symbol)
: The font encoding is the broadest category for selecting a font. Currently, only lilypond's system fonts (Emmentaler and Aybabtu) are using this property. Available values are `fetaMusic` (Emmentaler), `fetaBraces` (Aybabtu), `fetaNumber` (Emmentaler), and `fetaDynamic` (Emmentaler).

`font-family` (symbol)
: The font family is the broadest category for selecting text fonts. Options include: `sans`, `roman`.

`font-name` (string)
: Specifies a file name (without extension) of the font to load. This setting overrides selection using `font-family`, `font-series` and `font-shape`.

`font-series` (symbol)
: Select the series of a font. Choices include `medium`, `bold`, `bold-narrow`, etc.

`font-shape` (symbol)
: Select the shape of a font. Choices include `upright`, `italic`, `caps`.

`font-size` (number)
: The font size, compared to the ‚normal‘ size. `0` is style-sheet's normal size, `-1` is smaller, `+1` is bigger. Each step of 1 is approximately 12% larger; 6 steps are exactly a factor 2 larger. Fractional values are allowed.

`force-hshift` (number)
: This specifies a manual shift for notes in collisions. The unit is the note head width of the first voice note. This is used by Abschnitt “note-collision-interface” in *Referenz der Interna*.

`fraction` (pair of numbers)
: Numerator and denominator of a time signature object.

`french-beaming` (boolean)
: Use French beaming style for this stem. The stem stops at the innermost beams.

`fret-diagram-details` (list)
: An alist of detailed grob properties for fret diagrams. Each alist entry consists of a (`property . value`) pair. The properties which can be included in fret-diagram-details include the following:

  - `barre-type` – Type of barre indication used. Choices include `curved`, `straight`, and `none`. Default `curved`.
  - `capo-thickness` – Thickness of capo indicator, in multiples of fret-space. Default value 0.5.
  - `dot-color` – Color of dots. Options include `black` and `white`. Default `black`.
  - `dot-label-font-mag` – Magnification for font used to label fret dots. Default value 1.
  - `dot-radius` – Radius of dots, in terms of fret spaces. Default value 0.425 for labeled dots, 0.25 for unlabeled dots.
  - `finger-code` – Code for the type of fingering indication used. Options include `none`, `in-dot`, and `below-string`. Default `none` for markup fret diagrams, `below-string` for `FretBoards` fret diagrams.
  - `fret-count` – The number of frets. Default 4.
  - `fret-label-font-mag` – The magnification of the font used to label the lowest fret number. Default 0.5
  - `fret-label-vertical-offset` – The vertical offset of the fret label from the fret. Default -0.2
  - `label-dir` – Side to which the fret label is attached. `-1`, `#LEFT`, or `#DOWN` for left or down; `1`, `#RIGHT`, or `#UP` for right or up. Default `#RIGHT`.
  - `mute-string` – Character string to be used to indicate muted string. Default `"x"`.
  - `number-type` – Type of numbers to use in fret label. Choices include `roman-lower`, `roman-upper`, and `arabic`. Default `roman-lower`.
  - `open-string` – Character string to be used to indicate open string. Default `"o"`.
  - `orientation` – Orientation of fret-diagram. Options include `normal` and `landscape`. Default `normal`.
  - `string-count` – The number of strings. Default 6.
  - `string-label-font-mag` – The magnification of the font used to label fingerings at the string, rather than in the dot. Default value 0.6.
  - `top-fret-thickness` – The thickness of the top fret line, as a multiple of the standard thickness. Default value 3.
  - `xo-font-magnification` – Magnification used for mute and open string indicators. Default value 0.5.
  - `xo-padding` – Padding for open and mute indicators from top fret. Default value 0.25.

`full-length-padding` (number)
: How much padding to use at the right side of a full-length tuplet bracket.

`full-length-to-extent` (boolean)
: Run to the extent of the column for a full-length tuplet bracket.

`full-size-change` (boolean)
: Don't make a change clef smaller.

`gap` (dimension, in staff space)
: Size of a gap in a variable symbol.

`gap-count` (integer)
: Number of gapped beams for tremolo.

`glyph` (string)
: A string determining what ‚style' of glyph is typeset. Valid choices depend on the function that is reading this property.

`glyph-name-alist` (list)
: An alist of key-string pairs.

`grow-direction` (direction)
: Crescendo or decrescendo?

`hair-thickness` (number)
: Thickness of the thin line in a bar line.

`harp-pedal-details` (list)
: An alist of detailed grob properties for harp pedal diagrams. Each alist entry consists of a (`property` . `value`) pair. The properties which can be included in harp-pedal-details include the following:

  - `box-offset` – Vertical shift of the center of flat / sharp pedal boxes above / below the horizontal line. Default value 0.8.
  - `box-width` – Width of each pedal box. Default value 0.4.
  - `box-height` – Height of each pedal box. Default value 1.0.
  - `space-before-divider` – Space between boxes before the first divider (so that the diagram can be made symmetric). Default value 0.8.
  - `space-after-divider` – Space between boxes after the first divider. Default value 0.8.
  - `circle-thickness` – Thickness (in unit of the line-thickness) of the ellipse around circled pedals. Default value 0.5.
  - `circle-x-padding` – Padding in X direction of the ellipse around circled pedals. Default value 0.15.
  - `circle-y-padding` – Padding in Y direction of the ellipse around circled pedals. Default value 0.2.

`head-direction` (direction)
: Are the note heads left or right in a semitie?

`height` (dimension, in staff space)
: Height of an object in `staff-space` units.

`height-limit` (dimension, in staff space)
: Maximum slur height: The longer the slur, the closer it is to this height.

`horizontal-shift` (integer)
: An integer that identifies ranking of `NoteColumn`s for horizontal shifting. This is used by Abschnitt "note-collision-interface" in *Referenz der Interna*.

`horizontal-skylines` (unknown)
: Two skylines, one to the left and one to the right of this grob.

`ignore-collision` (boolean)
: If set, don't do note collision resolution on this `NoteColumn`.

`implicit` (boolean)
: Is this an implicit bass figure?

`inspect-index` (integer)
: If debugging is set, set beam and slur configuration to this index, and print the respective scores.

`inspect-quants` (pair of numbers)
: If debugging is set, set beam and slur quants to this position, and print the respective scores.

`keep-fixed-while-stretching` (boolean)
: A grob with this property set to true is fixed relative to the staff above it when systems are stretched.

`keep-inside-line` (boolean)
: If set, this column cannot have objects sticking into the margin.

`kern` (dimension, in staff space)
: Amount of extra white space to add. For bar lines, this is the amount of space after a thick line.

`knee` (boolean)
: Is this beam kneed?

`knee-spacing-correction` (number)
: Factor for the optical correction amount for kneed beams. Set between `0` for no correction and `1` for full correction.

`labels` (list)
: List of labels (symbols) placed on a column

`layer` (integer)
: The output layer (a value between 0 and 2: Layers define the order of printing objects. Objects in lower layers are overprinted by objects in higher layers.

`ledger-line-thickness` (pair of numbers)
: The thickness of ledger lines. It is the sum of 2 numbers: The first is the factor for line thickness, and the second for staff space. Both contributions are added.

`left-bound-info` (list)
: An alist of properties for determining attachments of spanners to edges.

`left-padding` (dimension, in staff space)
: The amount of space that is put left to an object (e.g., a group of accidentals).

`length` (dimension, in staff space)
: User override for the stem length of unbeamed stems.

`length-fraction` (number)
: Multiplier for lengths. Used for determining ledger lines and stem lengths.

`line-break-penalty` (number)
: Penalty for a line break at this column. This affects the choices of the line breaker; it avoids a line break at a column with a positive penalty and prefers a line break at a column with a negative penalty.

`line-break-permission` (symbol)
: Instructs the line breaker on whether to put a line break at this column. Can be `force` or `allow`.

`line-break-system-details` (list)
: An alist of properties to use if this column is the start of a system.

`line-count` (integer)
: The number of staff lines.

`line-positions` (list)
: Vertical positions of staff lines.

`line-thickness` (number)
: The thickness of the tie or slur contour.

`long-text` (markup)
: Text markup. See Abschnitt "Text formatieren" in *Notationsreferenz*.

`max-beam-connect` (integer)
: Maximum number of beams to connect to beams from this stem. Further beams are typeset as beamlets.

`max-stretch` (number)
: The maximum amount that this `VerticalAxisGroup` can be vertically stretched (for example, in order to better fill a page).

`measure-count` (integer)
: The number of measures for a multi-measure rest.

`measure-length` (moment)
: Length of a measure. Used in some spacing situations.

`merge-differently-dotted` (boolean)
: Merge note heads in collisions, even if they have a different number of dots. This is normal notation for some types of polyphonic music.

  `merge-differently-dotted` only applies to opposing stem directions (i.e., voice 1 & 2).

`merge-differently-headed` (boolean)
: Merge note heads in collisions, even if they have different note heads. The smaller of the two heads is rendered invisible. This is used in polyphonic guitar notation. The value of this setting is used by Abschnitt "note-collision-interface" in *Referenz der Interna*.

  `merge-differently-headed` only applies to opposing stem directions (i.e., voice 1 & 2).

`minimum-X-extent` (pair of numbers)
: Minimum size of an object in X dimension, measured in `staff-space` units.

`minimum-Y-extent` (pair of numbers)
: Minimum size of an object in Y dimension, measured in `staff-space` units.

`minimum-distance` (dimension, in staff space)
: Minimum distance between rest and notes or beam.

`minimum-length` (dimension, in staff space)
: Try to make a spanner at least this long, normally in the horizontal direction. This requires an appropriate callback for the `springs-and-rods` property. If added to a `Tie`, this sets the minimum distance between noteheads.

`minimum-length-fraction` (number)
: Minimum length of ledger line as fraction of note head size.

`minimum-space` (dimension, in staff space)
: Minimum distance that the victim should move (after padding).

`neutral-direction` (direction)
: Which direction to take in the center of the staff.

`neutral-position` (number)
: Position (in half staff spaces) where to flip the direction of custos stem.

`next` (layout object)
: Object that is next relation (e.g., the lyric syllable following an extender.

`no-alignment` (boolean)
: If set, don't place this grob in a `VerticalAlignment`; rather, place it using its own `Y-offset` callback.

`no-ledgers` (boolean)
: If set, don't draw ledger lines on this object.

`no-stem-extend` (boolean)
: If set, notes with ledger lines do not get stems extending to the middle staff line.

`non-default` (boolean)
: Set for manually specified clefs.

`non-musical` (boolean)
: True if the grob belongs to a `NonMusicalPaperColumn`.

`note-names` (vector)
: Vector of strings containing names for easy-notation note heads.

`outside-staff-horizontal-padding` (number)
: By default, an outside-staff-object can be placed so that is it very close to another grob horizontally. If this property is set, the outside-staff-object is raised so that it is not so close to its neighbor.

`outside-staff-padding` (number)
: The padding to place between this grob and the staff when spacing according to `outside-staff-priority`.

`outside-staff-priority` (number)
: If set, the grob is positioned outside the staff in such a way as to avoid all collisions. In case of a potential collision, the grob with the smaller `outside-staff-priority` is closer to the staff.

`packed-spacing` (boolean)
: If set, the notes are spaced as tightly as possible.

`padding` (dimension, in staff space)
: Add this much extra space between objects that are next to each other.

`padding-pairs` (list)
: An alist mapping `(`*`name . name`*`)` to distances.

`page-break-penalty` (number)
: Penalty for page break at this column. This affects the choices of the page breaker; it avoids a page break at a column with a positive penalty and prefers a page break at a column with a negative penalty.

`page-break-permission` (symbol)
: Instructs the page breaker on whether to put a page break at this column. Can be `force` or `allow`.

`page-turn-penalty` (number)
: Penalty for a page turn at this column. This affects the choices of the page breaker; it avoids a page turn at a column with a positive penalty and prefers a page turn at a column with a negative penalty.

`page-turn-permission` (symbol)
: Instructs the page breaker on whether to put a page turn at this column. Can be `force` or `allow`.

`parenthesized` (boolean)
: Parenthesize this grob.

`positions` (pair of numbers)
: Pair of staff coordinates `(`*`left . right`*`)`, where both *left* and *right* are in `staff-space` units of the current staff. For slurs, this value selects which slur candidate to use; if extreme positions are requested, the closest one is taken.

`prefer-dotted-right` (boolean)
: For note collisions, prefer to shift dotted up-note to the right, rather than shifting just the dot.

`ratio` (number)
: Parameter for slur shape. The higher this number, the quicker the slur attains its `height-limit`.

`remove-empty` (boolean)
: If set, remove group if it contains no interesting items.

`remove-first` (boolean)
: Remove the first staff of an orchestral score?

`restore-first` (boolean)
: Print a natural before the accidental.

`rhythmic-location` (rhythmic location)
: Where (bar number, measure position) in the score.

`right-bound-info` (list)
: An alist of properties for determining attachments of spanners to edges.

`right-padding` (dimension, in staff space)
: Space to insert on the right side of an object (e.g., between note and its accidentals).

`rotation` (list)
: Number of degrees to rotate this object, and what point to rotate around. For example, `#'(45 0 0)` rotates by 45 degrees around the center of this object.

`same-direction-correction` (number)
: Optical correction amount for stems that are placed in tight configurations. This amount is used for stems with the same direction to compensate for note head to stem distance.

`script-priority` (number)
: A sorting key that determines in what order a script is within a stack of scripts.

`self-alignment-X` (number)
: Specify alignment of an object. The value `-1` means left aligned, `0` centered, and `1` right-aligned in X direction. Other numerical values may also be specified.

`self-alignment-Y` (number)
: Like `self-alignment-X` but for the Y axis.

`shorten-pair` (pair of numbers)
: The lengths to shorten a text-spanner on both sides, for example a pedal bracket. Positive values shorten the text-spanner, while negative values lengthen it.

`shortest-duration-space` (dimension, in staff space)
: Start with this much space for the shortest duration. This is expressed in `spacing-increment` as unit. See also Abschnitt "spacing-spanner-interface" in *Referenz der Interna*.

`shortest-playing-duration` (moment)
: The duration of the shortest note playing here.

`shortest-starter-duration` (moment)
: The duration of the shortest note that starts here.

`side-axis` (number)
: If the value is `#X` (or equivalently `0`), the object is placed horizontally next to the other object. If the value is `#Y` or `1`, it is placed vertically.

`side-relative-direction` (direction)
: Multiply direction of `direction-source` with this to get the direction of this object.

`size` (number)
: Size of object, relative to standard size.

`skyline-horizontal-padding` (number)
: For determining the vertical distance between two staves, it is possible to have a configuration which would result in a tight interleaving of grobs from the top staff and the bottom staff. The larger this parameter is, the farther apart the staves are placed in such a configuration.

`slash-negative-kern` (number)
: The space to remove between slashes in percent repeat glyphs. Larger values bring the two elements closer together.

`slope` (number)
: The slope of this object.

`slur-padding` (number)
: Extra distance between slur and script.

`space-alist` (list)
: A table that specifies distances between prefatory items, like clef and time-signature. The format is an alist of spacing tuples: `(break-align-symbol type . distance)`, where *type* can be the symbols `minimum-space` or `extra-space`.

`space-to-barline` (boolean)
: If set, the distance between a note and the following non-musical column will be measured to the bar line instead of to the beginning of the non-musical column. If there is a clef change followed by a bar line, for example, this means that we will try to space the non-musical column as though the clef is not there.

`spacing-increment` (number)
: Add this much space for a doubled duration. Typically, the width of a note head. See also Abschnitt "spacing-spanner-interface" in *Referenz der Interna*.

`springs-and-rods` (boolean)
: Dummy variable for triggering spacing routines.

`stacking-dir` (direction)
: Stack objects in which direction?

`staff-padding` (dimension, in staff space)
: Maintain this much space between reference points and the staff. Its effect is to align objects of differing sizes (like the dynamics **p** and **f**) on their baselines.

`staff-position` (number)
: Vertical position, measured in half staff spaces, counted from the middle line.

`staff-space` (dimension, in staff space)
: Amount of space between staff lines, expressed in global `staff-space`.

`stem-attachment` (pair of numbers)
: An `(x . y)` pair where the stem attaches to the notehead.

`stem-end-position` (number)
: Where does the stem end (the end is opposite to the support-head)?

`stem-spacing-correction` (number)
: Optical correction amount for stems that are placed in tight configurations. For opposite directions, this amount is the correction for two normal sized stems that overlap completely.

`stemlet-length` (number)
: How long should a stem over a rest be?

`stencil` (unknown)
: The symbol to print.

`stencils` (list)
: Multiple stencils, used as intermediate value.

`strict-grace-spacing` (boolean)
: If set, main notes are spaced normally, then grace notes are put left of the musical columns fot the main notes.

`strict-note-spacing` (boolean)
: If set, unbroken columns with non-musical material (clefs, bar lines, etc.) are not spaced separately, but put before musical columns.

`stroke-style` (string)
: Set to `"grace"` to turn stroke through flag on.

`style` (symbol)
: This setting determines in what style a grob is typeset. Valid choices depend on the `stencil` callback reading this property.

`text` (markup)
: Text markup. See Abschnitt “Text formatieren” in *Notationsreferenz*.

`text-direction` (direction)
: This controls the ordering of the words. The default `RIGHT` is for roman text. Arabic or Hebrew should use `LEFT`.

`thick-thickness` (number)
: Bar line thickness, measured in `line-thickness`.

`thickness` (number)
: Line thickness, generally measured in `line-thickness`.

`thin-kern` (number)
: The space after a hair-line in a bar line.

`threshold` (pair of numbers)
: `(min . max)`, where *min* and *max* are dimensions in staff space.

`tie-configuration` (list)
: List of `(position . dir)` pairs, indicating the desired tie configuration, where *position* is the offset from the center of the staff in staff space and *dir* indicates the direction of the tie (`1`=>up, `-1`=>down, `0`=>center). A non-pair entry in the list causes the corresponding tie to be formatted automatically.

`to-barline` (boolean)
: If true, the spanner will stop at the bar line just before it would otherwise stop.

`toward-stem-shift` (number)
: Amount by which scripts are shifted toward the stem if their direction coincides with the stem direction. `0.0` means keep the default position (centered on the note head), `1.0` means centered on the stem. Interpolated values are possible.

`transparent` (boolean)
: This makes the grob invisible.

`uniform-stretching` (boolean)
: If set, items stretch proportionally to their durations. This looks better in complex polyphonic patterns.

`used` (boolean)
: If set, this spacing column is kept in the spacing problem.

`vertical-skylines` (unknown)
: Two skylines, one above and one below this grob.

`when` (moment)
: Global time step associated with this column happen?

`width` (dimension, in staff space)
: The width of a grob measured in staff space.

`word-space` (dimension, in staff space)
: Space to insert between words in texts.

`zigzag-length` (dimension, in staff space)
: The length of the lines of a zigzag, relative to `zigzag-width`. A value of `1` gives 60-degree zigzags.

`zigzag-width` (dimension, in staff space)
: The width of one zigzag squiggle. This number is adjusted slightly so that the glissando line can be constructed from a whole number of squiggles.

B.14 Bezeichner

`acciaccatura` - *music* (music)
: (undocumented; fixme)

`addChordShape` - *key-symbol* (symbol) *tuning* (pair) *shape-definition* (unknown)
: Add chord shape `shape-definition` to the `chord-shape-table` hash with the key `(cons key-symbol tuning)`.

`addInstrumentDefinition` - *name* (string) *lst* (list)
: (undocumented; fixme)

`addQuote` - *name* (string) *music* (music)
: Add a piece of music to be quoted

`afterGrace` - *main* (music) *grace* (music)
(undocumented; fixme)

`allowPageTurn`
Allow a page turn. May be used at toplevel (ie between scores or markups), or inside a score.

`applyContext` - *proc* (procedure)
(undocumented; fixme)

`applyMusic` - *func* (procedure) *music* (music)
(undocumented; fixme)

`applyOutput` - *ctx* (symbol) *proc* (procedure)
(undocumented; fixme)

`appoggiatura` - *music* (music)
(undocumented; fixme)

`assertBeamQuant` - *l* (pair) *r* (pair)
(undocumented; fixme)

`assertBeamSlope` - *comp* (procedure)
(undocumented; fixme)

`autochange` - *music* (music)
(undocumented; fixme)

`balloonGrobText` - *grob-name* (symbol) *offset* (pair of numbers) *text* (markup)
(undocumented; fixme)

`balloonText` - *offset* (pair of numbers) *text* (markup)
(undocumented; fixme)

`bar` - *type* (string)
(undocumented; fixme)

`barNumberCheck` - *n* (integer)
(undocumented; fixme)

`bendAfter` - *delta* (unknown)
(undocumented; fixme)

`breathe` (undocumented; fixme)

`clef` - *type* (string)
Set the current clef.

`cueDuring` - *what* (string) *dir* (direction) *main-music* (music)
(undocumented; fixme)

`displayLilyMusic` - *music* (music)
(undocumented; fixme)

`displayMusic` - *music* (music)
(undocumented; fixme)

`endSpanners` - *music* (music)
(undocumented; fixme)

`featherDurations` - *factor* (moment) *argument* (music)
Rearrange durations in ARGUMENT so there is an acceleration/deceleration.

`grace` - *music* (music)
(undocumented; fixme)

`includePageLayoutFile`
If page breaks and tweak dump is not asked, and the file <basename>-page-layout.ly exists, include it.

`instrumentSwitch` - *name* (string)
(undocumented; fixme)

`keepWithTag` - *tag* (symbol) *music* (music)
(undocumented; fixme)

`killCues` - *music* (music)
(undocumented; fixme)

`label` - *label* (symbol)
Place a bookmarking label, either at top-level or inside music.

`makeClusters` - *arg* (music)
(undocumented; fixme)

`musicMap` - *proc* (procedure) *mus* (music)
(undocumented; fixme)

`noPageBreak`
Forbid a page break. May be used at toplevel (ie between scores or markups), or inside a score.

`noPageTurn`
Forbid a page turn. May be used at toplevel (ie between scores or markups), or inside a score.

`octaveCheck` - *pitch-note* (music)
octave check

`oldaddlyrics` - *music* (music) *lyrics* (music)
(undocumented; fixme)

`ottava` - *octave* (number)
set the octavation

`overrideProperty` - *name* (string) *property* (symbol) *value* (any type)
Set *property* to *value* in all grobs named *name*. The *name* argument is a string of the form `"Context.GrobName"` or `"GrobName"`

`pageBreak`
Force a page break. May be used at toplevel (ie between scores or markups), or inside a score.

`pageTurn` Force a page turn between two scores or top-level markups.

`parallelMusic` - *voice-ids* (list) *music* (music)
Define parallel music sequences, separated by '`|`' (bar check signs), and assign them to the identifiers provided in *voice-ids*.

voice-ids: a list of music identifiers (symbols containing only letters)

music: a music sequence, containing BarChecks as limiting expressions.

Example:

```
  \parallelMusic #'(A B C) {
    c c | d d | e e |
    d d | e e | f f |
  }
<==>
  A = { c c | d d | }
  B = { d d | e e | }
  C = { e e | f f | }
```

`parenthesize` - *arg* (music)
: Tag *arg* to be parenthesized.

`partcombine` - *part1* (music) *part2* (music)
: (undocumented; fixme)

`pitchedTrill` - *main-note* (music) *secondary-note* (music)
: (undocumented; fixme)

`pointAndClickOff`
: (undocumented; fixme)

`pointAndClickOn`
: (undocumented; fixme)

`quoteDuring` - *what* (string) *main-music* (music)
: (undocumented; fixme)

`removeWithTag` - *tag* (symbol) *music* (music)
: (undocumented; fixme)

`resetRelativeOctave` - *reference-note* (music)
: Set the octave inside a \relative section.

`rightHandFinger` - *finger* (number or string)
: Define a StrokeFingerEvent

`scaleDurations` - *fraction* (pair of numbers) *music* (music)
: (undocumented; fixme)

`scoreTweak` - *name* (string)
: Include the score tweak, if exists.

`shiftDurations` - *dur* (integer) *dots* (integer) *arg* (music)
`spacingTweaks` - *parameters* (list)
: Set the system stretch, by reading the 'system-stretch property of the 'parameters' assoc list.

`storePredefinedDiagram` - *chord* (music) *tuning* (pair) *diagram-definition* (unknown)
: Add predefined fret diagram defined by `diagram-definition` for the chord pitches `chord` and the stringTuning `tuning`.

`tag` - *tag* (symbol) *arg* (music)
: Add *tag* to the `tags` property of *arg*.

`tocItem` - *text* (markup)
: Add a line to the table of content, using the `tocItemMarkup` paper variable markup

`transposedCueDuring` - *what* (string) *dir* (direction) *pitch-note* (music) *main-music* (music)
: Insert notes from the part *what* into a voice called `cue`, using the transposition defined by *pitch-note*. 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-note* (music)
: Set instrument transposition

`tweak` - *sym* (symbol) *val* (any type) *arg* (music)
: Add `sym . val` to the `tweaks` property of *arg*.

`unfoldRepeats` - *music* (music)
: (undocumented; fixme)

`withMusicProperty` - *sym* (symbol) *val* (any type) *music* (music)
: Set *sym* to *val* in *music*.

B.15 Scheme-Funktionen

`dispatcher` *x* [Funktion]
: Is *x* a `Dispatcher` object?

`listener` *x* [Funktion]
: Is *x* a `Listener` object?

`ly:add-file-name-alist` *alist* [Funktion]
: Add mappings for error messages from *alist*.

`ly:add-interface` *a b c* [Funktion]
: Add an interface description.

`ly:add-listener` *list disp cl* [Funktion]
: Add the listener *list* to the dispatcher *disp*. Whenever *disp* hears an event of class *cl*, it is forwarded to *list*.

`ly:add-option` *sym val description* [Funktion]
: Add a program option *sym* with default *val*.

`ly:all-grob-interfaces` [Funktion]
: Get a hash table with all interface descriptions.

`ly:all-options` [Funktion]
: Get all option settings in an alist.

`ly:all-stencil-expressions` [Funktion]
: Return all symbols recognized as stencil expressions.

`ly:assoc-get` *key alist default-value* [Funktion]
: Return value if *key* in *alist*, else `default-value` (or `#f` if not specified).

`ly:book-add-bookpart!` *book-smob book-part* [Funktion]
: Add *book_part* to *book-smob* book part list.

`ly:book-add-score!` *book-smob score* [Funktion]
: Add *score* to *book-smob* score list.

`ly:book-process` *book-smob default-paper default-layout output* [Funktion]
: Print book. *output* is passed to the backend unchanged. For example, it may be a string (for file based outputs) or a socket (for network based output).

`ly:book-process-to-systems` *book-smob default-paper default-layout output* [Funktion]
: Print book. *output* is passed to the backend unchanged. For example, it may be a string (for file based outputs) or a socket (for network based output).

`ly:box?` *x* [Funktion]
Is *x* a `Box` object?

`ly:bp` *num* [Funktion]
num bigpoints (1/72th inch).

`ly:bracket` *a iv t p* [Funktion]
Make a bracket in direction *a*. The extent of the bracket is given by *iv*. The wings protrude by an amount of *p*, which may be negative. The thickness is given by *t*.

`ly:broadcast` *disp ev* [Funktion]
Send the stream event *ev* to the dispatcher *disp*.

`ly:camel-case->lisp-identifier` *name-sym* [Funktion]
Convert `FooBar_Bla` to `foo-bar-bla` style symbol.

`ly:chain-assoc-get` *key achain dfault* [Funktion]
Return value for *key* from a list of alists *achain*. If no entry is found, return *dfault* or `#f` if no *dfault* is specified.

`ly:clear-anonymous-modules` [Funktion]
Plug a GUILE 1.6 and 1.7 memory leak by breaking a weak reference pointer cycle explicitly.

`ly:cm` *num* [Funktion]
num cm.

`ly:command-line-code` [Funktion]
The Scheme code specified on command-line with '`-e`'.

`ly:command-line-options` [Funktion]
The Scheme options specified on command-line with '`-d`'.

`ly:command-line-verbose?` [Funktion]
Was `be_verbose_global` set?

`ly:connect-dispatchers` *to from* [Funktion]
Make the dispatcher *to* listen to events from *from*.

`ly:context-event-source` *context* [Funktion]
Return `event-source` of context *context*.

`ly:context-events-below` *context* [Funktion]
Return a `stream-distributor` that distributes all events from *context* and all its subcontexts.

`ly:context-find` *context name* [Funktion]
Find a parent of *context* that has name or alias *name*. Return `#f` if not found.

`ly:context-grob-definition` *context name* [Funktion]
Return the definition of *name* (a symbol) within *context* as an alist.

`ly:context-id` *context* [Funktion]
Return the ID string of *context*, i.e., for `\context Voice = one ...` return the string `one`.

`ly:context-name` *context* [Funktion]
Return the name of *context*, i.e., for `\context Voice = one ...` return the symbol `Voice`.

`ly:context-now` *context* [Funktion]
Return `now-moment` of context *context*.

`ly:context-parent` *context* [Funktion]
Return the parent of *context*, `#f` if none.

`ly:context-property` *c name* [Funktion]
Return the value of *name* from context *c*.

`ly:context-property-where-defined` *context name* [Funktion]
Return the context above *context* where *name* is defined.

`ly:context-pushpop-property` *context grob eltprop val* [Funktion]
Do a single `\override` or `\revert` operation in *context*. The grob definition *grob* is extended with *eltprop* (if *val* is specified) or reverted (if unspecified).

`ly:context-set-property!` *context name val* [Funktion]
Set value of property *name* in context *context* to *val*.

`ly:context-unset-property` *context name* [Funktion]
Unset value of property *name* in context *context*.

`ly:context?` *x* [Funktion]
Is *x* a `Context` object?

`ly:default-scale` [Funktion]
Get the global default scale.

`ly:dimension?` *d* [Funktion]
Return *d* as a number. Used to distinguish length variables from normal numbers.

`ly:dir?` *s* [Funktion]
A type predicate. The direction `s` is `-1`, `0` or `1`, where `-1` represents left or down and `1` represents right or up.

`ly:duration->string` *dur* [Funktion]
Convert *dur* to a string.

`ly:duration-dot-count` *dur* [Funktion]
Extract the dot count from *dur*.

`ly:duration-factor` *dur* [Funktion]
Extract the compression factor from *dur*. Return it as a pair.

`ly:duration-length` *dur* [Funktion]
The length of the duration as a `moment`.

`ly:duration-log` *dur* [Funktion]
Extract the duration log from *dur*.

`ly:duration<?` *p1 p2* [Funktion]
Is *p1* shorter than *p2*?

`ly:duration?` *x* [Funktion]
Is *x* a `Duration` object?

`ly:effective-prefix` [Funktion]
Return effective prefix.

`ly:error` *str rest* [Funktion]
A Scheme callable function to issue the error *str*. The error is formatted with `format` and *rest*.

`ly:eval-simple-closure` *delayed closure scm-start scm-end* [Funktion]
Evaluate a simple *closure* with the given *delayed* argument. If *scm-start* and *scm-end* are defined, evaluate it purely with those start and end points.

`ly:event-deep-copy` *m* [Funktion]
Copy *m* and all sub expressions of *m*.

`ly:event-property` *sev sym* [Funktion]
Get the property *sym* of stream event *mus*. If *sym* is undefined, return `'()`.

`ly:event-set-property!` *ev sym val* [Funktion]
Set property *sym* in event *ev* to *val*.

`ly:expand-environment` *str* [Funktion]
Expand `$VAR` and `${VAR}` in *str*.

`ly:export` *arg* [Funktion]
Export a Scheme object to the parser so it is treated as an identifier.

`ly:find-accidentals-simple` *keysig pitch-scm barnum laziness octaveness* [Funktion]
Checks the need for an accidental and a ‚restore‘ accidental against a key signature. The *laziness* is the number of bars for which reminder accidentals are used (ie. if *laziness* is zero, we only cancel accidentals in the same bar; if *laziness* is three, we cancel accidentals up to three bars after they first appear. *octaveness* is either `'same-octave` or `'any-octave` and it specifies whether accidentals should be canceled in different octaves.

`ly:find-file` *name* [Funktion]
Return the absolute file name of *name*, or `#f` if not found.

`ly:font-config-add-directory` *dir* [Funktion]
Add directory *dir* to FontConfig.

`ly:font-config-add-font` *font* [Funktion]
Add font *font* to FontConfig.

`ly:font-config-display-fonts` [Funktion]
Dump a list of all fonts visible to FontConfig.

`ly:font-config-get-font-file` *name* [Funktion]
Get the file for font *name*.

`ly:font-design-size` *font* [Funktion]
Given the font metric *font*, return the design size, relative to the current output-scale.

`ly:font-file-name` *font* [Funktion]
Given the font metric *font*, return the corresponding file name.

`ly:font-get-glyph` *font name* [Funktion]
Return a stencil from *font* for the glyph named *name*. If the glyph is not available, return an empty stencil.

Note that this command can only be used to access glyphs from fonts loaded with `ly:system-font-load`; currently, this means either the Emmentaler or Aybabtu fonts, corresponding to the font encodings `fetaMusic` and `fetaBraces`, respectively.

`ly:font-glyph-name-to-charcode` *font name* [Funktion]
Return the character code for glyph *name* in *font*.

Note that this command can only be used to access glyphs from fonts loaded with `ly:system-font-load`; currently, this means either the Emmentaler or Aybabtu fonts, corresponding to the font encodings `fetaMusic` and `fetaBraces`, respectively.

`ly:font-glyph-name-to-index` *font name* [Funktion]
Return the index for *name* in *font*.

Note that this command can only be used to access glyphs from fonts loaded with `ly:system-font-load`; currently, this means either the Emmentaler or Aybabtu fonts, corresponding to the font encodings `fetaMusic` and `fetaBraces`, respectively.

`ly:font-index-to-charcode` *font index* [Funktion]
Return the character code for *index* in *font*.

Note that this command can only be used to access glyphs from fonts loaded with `ly:system-font-load`; currently, this means either the Emmentaler or Aybabtu fonts, corresponding to the font encodings `fetaMusic` and `fetaBraces`, respectively.

`ly:font-magnification` *font* [Funktion]
Given the font metric *font*, return the magnification, relative to the current output-scale.

`ly:font-metric?` *x* [Funktion]
Is *x* a `Font_metric` object?

`ly:font-name` *font* [Funktion]
Given the font metric *font*, return the corresponding name.

`ly:font-sub-fonts` *font* [Funktion]
Given the font metric *font* of an OpenType font, return the names of the subfonts within *font*.

`ly:format` *str rest* [Funktion]
LilyPond specific format, supporting `~a` and `~[0-9]f`.

`ly:format-output` *context* [Funktion]
Given a global context in its final state, process it and return the `Music_output` object in its final state.

`ly:get-all-function-documentation` [Funktion]
Get a hash table with all LilyPond Scheme extension functions.

`ly:get-all-translators` [Funktion]
Return a list of all translator objects that may be instantiated.

`ly:get-glyph` *font index* [Funktion]
Retrieve a stencil for the glyph numbered *index* in *font*.

Note that this command can only be used to access glyphs from fonts loaded with `ly:system-font-load`; currently, this means either the Emmentaler or Aybabtu fonts, corresponding to the font encodings `fetaMusic` and `fetaBraces`, respectively.

`ly:get-listened-event-classes` [Funktion]
Return a list of all event classes that some translator listens to.

`ly:get-option` *var* [Funktion]
Get a global option setting.

`ly:gettext` *original* [Funktion]
A Scheme wrapper function for `gettext`.

`ly:grob-alist-chain` *grob global* [Funktion]
Get an alist chain for grob *grob*, with *global* as the global default. If unspecified, `font-defaults` from the layout block is taken.

`ly:grob-array-length` *grob-arr* [Funktion]
Return the length of *grob-arr*.

`ly:grob-array-ref` *grob-arr index* [Funktion]
Retrieve the *index*th element of *grob-arr*.

`ly:grob-array?` *x* [Funktion]
Is *x* a `Grob_array` object?

`ly:grob-basic-properties` *grob* [Funktion]
Get the immutable properties of *grob*.

`ly:grob-common-refpoint` *grob other axis* [Funktion]
Find the common refpoint of *grob* and *other* for *axis*.

`ly:grob-common-refpoint-of-array` *grob others axis* [Funktion]
Find the common refpoint of *grob* and *others* (a grob-array) for *axis*.

`ly:grob-default-font` *grob* [Funktion]
Return the default font for grob *gr*.

`ly:grob-extent` *grob refp axis* [Funktion]
Get the extent in *axis* direction of *grob* relative to the grob *refp*.

`ly:grob-interfaces` *grob* [Funktion]
Return the interfaces list of grob *grob*.

`ly:grob-layout` *grob* [Funktion]
Get `\layout` definition from grob *grob*.

`ly:grob-object` *grob sym* [Funktion]
Return the value of a pointer in grob *g* of property *sym*. It returns `'()` (end-of-list) if *sym* is undefined in *g*.

`ly:grob-original` *grob* [Funktion]
Return the unbroken original grob of *grob*.

`ly:grob-parent` *grob axis* [Funktion]
Get the parent of *grob*. *axis* is 0 for the X-axis, 1 for the Y-axis.

`ly:grob-pq<?` *a b* [Funktion]
Compare two grob priority queue entries. This is an internal function.

`ly:grob-properties` *grob* [Funktion]
Get the mutable properties of *grob*.

`ly:grob-property` *grob sym deflt* [Funktion]
Return the value of a value in grob *g* of property *sym*. It returns `'()` (end-of-list) or *deflt* (if specified) if *sym* is undefined in *g*.

`ly:grob-property-data` *grob sym* [Funktion]
Retrieve *sym* for *grob* but don't process callbacks.

`ly:grob-relative-coordinate` *grob refp axis* [Funktion]
Get the coordinate in *axis* direction of *grob* relative to the grob *refp*.

`ly:grob-robust-relative-extent` *grob refp axis* [Funktion]
Get the extent in *axis* direction of *grob* relative to the grob *refp*, or `(0,0)` if empty.

`ly:grob-script-priority-less` *a b* [Funktion]
Compare two grobs by script priority. For internal use.

`ly:grob-set-property!` *grob sym val* [Funktion]
Set *sym* in grob *grob* to value *val*.

`ly:grob-staff-position` *sg* [Funktion]
Return the Y-position of *sg* relative to the staff.

`ly:grob-suicide!` *grob* [Funktion]
Kill *grob*.

`ly:grob-system` *grob* [Funktion]
Return the system grob of *grob*.

`ly:grob-translate-axis!` *grob d a* [Funktion]
Translate *g* on axis *a* over distance *d*.

`ly:grob?` *x* [Funktion]
Is *x* a `Grob` object?

`ly:gulp-file` *name size* [Funktion]
Read the file *name*, and return its contents in a string. The file is looked up using the search path.

`ly:hash-table-keys` *tab* [Funktion]
Return a list of keys in *tab*.

`ly:inch` *num* [Funktion]
num inches.

`ly:input-both-locations` *sip* [Funktion]
Return input location in *sip* as `(file-name first-line first-column last-line last-column)`.

`ly:input-file-line-char-column` *sip* [Funktion]
Return input location in *sip* as `(file-name line char column)`.

`ly:input-location?` *x* [Funktion]
Is *x* an `input-location`?

`ly:input-message` *sip msg rest* [Funktion]
Print *msg* as a GNU compliant error message, pointing to the location in *sip*. *msg* is interpreted similar to `format`'s argument, using *rest*.

`ly:interpret-music-expression` *mus ctx* [Funktion]
Interpret the music expression *mus* in the global context *ctx*. The context is returned in its final state.

`ly:interpret-stencil-expression` *expr func arg1 offset* [Funktion]
Parse *expr*, feed bits to *func* with first arg *arg1* having offset *offset*.

`ly:intlog2` *d* [Funktion]
The 2-logarithm of $1/d$.

`ly:is-listened-event-class` *sym* [Funktion]
Is *sym* a listened event class?

`ly:item-break-dir` *it* [Funktion]
The break status direction of item *it*. `-1` means end of line, `0` unbroken, and `1` beginning of line.

`ly:item?` *g* [Funktion]
Is *g* an `Item` object?

`ly:iterator?` *x* [Funktion]
Is *x* a `Music_iterator` object?

`ly:lexer-keywords` *lexer* [Funktion]
Return a list of `(KEY . CODE)` pairs, signifying the LilyPond reserved words list.

`ly:lily-lexer?` *x* [Funktion]
Is *x* a `Lily_lexer` object?

`ly:lily-parser?` *x* [Funktion]
Is *x* a `Lily_parser` object?

`ly:make-book` *paper header scores* [Funktion]
Make a `\book` of *paper* and *header* (which may be `#f` as well) containing `\scores`.

`ly:make-book-part` *scores* [Funktion]
Make a `\bookpart` containing `\scores`.

`ly:make-dispatcher` [Funktion]
Return a newly created dispatcher.

`ly:make-duration` *length dotcount num den* [Funktion]
length is the negative logarithm (base 2) of the duration: 1 is a half note, 2 is a quarter note, 3 is an eighth note, etc. The number of dots after the note is given by the optional argument *dotcount*.

The duration factor is optionally given by *num* and *den*.

A duration is a musical duration, i.e., a length of time described by a power of two (whole, half, quarter, etc.) and a number of augmentation dots.

`ly:make-global-context` *output-def* [Funktion]
Set up a global interpretation context, using the output block *output_def*. The context is returned.

`ly:make-global-translator` *global* [Funktion]
Create a translator group and connect it to the global context *global*. The translator group is returned.

`ly:make-listener` *callback* [Funktion]
Create a listener. Any time the listener hears an object, it will call *callback* with that object.

callback should take exactly one argument.

`ly:make-moment` *n d gn gd* [Funktion]
Create the rational number with main timing n/d, and optional grace timing gn/gd.

A *moment* is a point in musical time. It consists of a pair of rationals (m, g), where m is the timing for the main notes, and g the timing for grace notes. In absence of grace notes, g is zero.

`ly:make-music` *props* [Funktion]
Make a C++ `Music` object and initialize it with *props*.

This function is for internal use and is only called by `make-music`, which is the preferred interface for creating music objects.

`ly:make-music-function` *signature func* [Funktion]
Make a function to process music, to be used for the parser. `func` is the function, and `signature` describes its arguments. `signature` is a list containing either `ly:music?` predicates or other type predicates.

`ly:make-output-def` [Funktion]
Make an output definition.

`ly:make-page-label-marker` *label* [Funktion]
Return page marker with label.

`ly:make-page-permission-marker` *symbol permission* [Funktion]
Return page marker with page breaking and turning permissions.

`ly:make-pango-description-string` *chain size* [Funktion]
Make a `PangoFontDescription` string for the property alist *chain* at size *size*.

`ly:make-paper-outputter` *port format* [Funktion]
Create an outputter that evaluates within `output-`*format*, writing to *port*.

`ly:make-pitch` *octave note alter* [Funktion]
octave is specified by an integer, zero for the octave containing middle C. *note* is a number from 0 to 6, with 0 corresponding to pitch C and 6 corresponding to pitch B. *alter* is a rational number of whole tones for alteration.

`ly:make-prob` *type init rest* [Funktion]
Create a `Prob` object.

`ly:make-scale` *steps* [Funktion]
Create a scale. The argument is a vector of rational numbers, each of which represents the number of tones of a pitch above the tonic.

`ly:make-score` *music* [Funktion]
Return score with *music* encapsulated in *score*.

`ly:make-simple-closure` *expr* [Funktion]
Make a simple closure. *expr* should be form of `(func a1 A2 ...)`, and will be invoked as `(func delayed-arg a1 a2 ...)`.

`ly:make-stencil` *expr xext yext* [Funktion]
Stencils are device independent output expressions. They carry two pieces of information:

1. A specification of how to print this object. This specification is processed by the output backends, for example '`scm/output-ps.scm`'.
2. The vertical and horizontal extents of the object, given as pairs. If an extent is unspecified (or if you use `(1000 . -1000)` as its value), it is taken to be empty.

`ly:make-stream-event` *cl proplist* [Funktion]
Create a stream event of class *cl* with the given mutable property list.

`ly:message` *str rest* [Funktion]
A Scheme callable function to issue the message *str*. The message is formatted with `format` and *rest*.

`ly:minimal-breaking` *pb* [Funktion]
Break (pages and lines) the `Paper_book` object *pb* without looking for optimal spacing: stack as many lines on a page before moving to the next one.

`ly:mm` *num* [Funktion]
num mm.

`ly:module->alist` *mod* [Funktion]
Dump the contents of module *mod* as an alist.

`ly:module-copy` *dest src* [Funktion]
Copy all bindings from module *src* into *dest*.

`ly:modules-lookup` *modules sym def* [Funktion]
Look up *sym* in the list *modules*, returning the first occurence. If not found, return *def* or `#f` if *def* isn't specified.

`ly:moment-add` *a b* [Funktion]
Add two moments.

`ly:moment-div` *a b* [Funktion]
Divide two moments.

`ly:moment-grace-denominator` *mom* [Funktion]
Extract denominator from grace timing.

`ly:moment-grace-numerator` *mom* [Funktion]
Extract numerator from grace timing.

`ly:moment-main-denominator` *mom* [Funktion]
Extract denominator from main timing.

`ly:moment-main-numerator` *mom* [Funktion]
Extract numerator from main timing.

`ly:moment-mod` *a b* [Funktion]
Modulo of two moments.

`ly:moment-mul` *a b* [Funktion]
Multiply two moments.

`ly:moment-sub` *a b* [Funktion]
Subtract two moments.

`ly:moment<?` *a b* [Funktion]
Compare two moments.

`ly:moment?` *x* [Funktion]
Is *x* a `Moment` object?

`ly:music-compress` *m factor* [Funktion]
Compress music object *m* by moment *factor*.

`ly:music-deep-copy` *m* [Funktion]
Copy *m* and all sub expressions of *m*.

`ly:music-duration-compress` *mus fact* [Funktion]
Compress *mus* by factor *fact*, which is a `Moment`.

`ly:music-duration-length` *mus* [Funktion]
Extract the duration field from *mus* and return the length.

`ly:music-function-extract` *x* [Funktion]
Return the Scheme function inside *x*.

`ly:music-function?` *x* [Funktion]
Is *x* a `music-function`?

`ly:music-length` *mus* [Funktion]
Get the length of music expression *mus* and return it as a `Moment` object.

`ly:music-list?` *lst* [Funktion]
Type predicate: Return true if *lst* is a list of music objects.

`ly:music-mutable-properties` *mus* [Funktion]
Return an alist containing the mutable properties of *mus*. The immutable properties are not available, since they are constant and initialized by the `make-music` function.

`ly:music-output?` *x* [Funktion]
Is *x* a `Music_output` object?

`ly:music-property` *mus sym dfault* [Funktion]
Get the property *sym* of music expression *mus*. If *sym* is undefined, return `'()`.

`ly:music-set-property!` *mus sym val* [Funktion]
Set property *sym* in music expression *mus* to *val*.

`ly:music-transpose` *m p* [Funktion]
Transpose *m* such that central C is mapped to *p*. Return *m*.

`ly:music?` *obj* [Funktion]
Type predicate.

`ly:note-head::stem-attachment` *font-metric glyph-name* [Funktion]
Get attachment in *font-metric* for attaching a stem to notehead *glyph-name*.

`ly:number->string` *s* [Funktion]
Convert *num* to a string without generating many decimals.

`ly:optimal-breaking` *pb* [Funktion]
Optimally break (pages and lines) the `Paper_book` object *pb* to minimize badness in bother vertical and horizontal spacing.

`ly:option-usage` [Funktion]
Print `ly:set-option` usage.

`ly:otf->cff` *otf-file-name* [Funktion]
Convert the contents of an OTF file to a CFF file, returning it as a string.

`ly:otf-font-glyph-info` *font glyph* [Funktion]
Given the font metric *font* of an OpenType font, return the information about named glyph *glyph* (a string).

`ly:otf-font-table-data` *font tag* [Funktion]
Extract a table *tag* from *font*. Return empty string for non-existent *tag*.

`ly:otf-font?` *font* [Funktion]
Is *font* an OpenType font?

`ly:otf-glyph-list` *font* [Funktion]
Return a list of glyph names for *font*.

`ly:output-def-clone` *def* [Funktion]
Clone output definition *def*.

`ly:output-def-lookup` *pap sym def* [Funktion]
Look up *sym* in the *pap* output definition (e.g., `\paper`). Return the value or *def* (which defaults to `'()`) if undefined.

`ly:output-def-parent` *def* [Funktion]
Get the parent output definition of *def*.

`ly:output-def-scope` *def* [Funktion]
Get the variable scope inside *def*.

`ly:output-def-set-variable!` *def sym val* [Funktion]
Set an output definition *def* variable *sym* to *val*.

`ly:output-def?` *def* [Funktion]
Is *def* a layout definition?

`ly:output-description` *output-def* [Funktion]
Return the description of translators in *output-def*.

`ly:output-formats` [Funktion]
Formats passed to '`--format`' as a list of strings, used for the output.

`ly:outputter-close` *outputter* [Funktion]
Close port of *outputter*.

`ly:outputter-dump-stencil` *outputter stencil* [Funktion]
Dump stencil *expr* onto *outputter*.

`ly:outputter-dump-string` *outputter str* [Funktion]
Dump *str* onto *outputter*.

`ly:outputter-output-scheme` *outputter expr* [Funktion]
Eval *expr* in module of *outputter*.

`ly:outputter-port` *outputter* [Funktion]
Return output port for *outputter*.

`ly:page-marker?` *x* [Funktion]
Is *x* a `Page_marker` object?

`ly:page-turn-breaking` *pb* [Funktion]
Optimally break (pages and lines) the `Paper_book` object *pb* such that page turns only happen in specified places, returning its pages.

`ly:pango-font-physical-fonts` *f* [Funktion]
Return alist of `(ps-name file-name font-index)` lists for Pango font *f*.

`ly:pango-font?` *f* [Funktion]
Is *f* a pango font?

`ly:paper-book-pages` *pb* [Funktion]
Return pages in book *pb*.

`ly:paper-book-paper` *pb* [Funktion]
Return pages in book *pb*.

`ly:paper-book-performances` *paper-book* [Funktion]
Return performances in book *paper-book*.

`ly:paper-book-scopes` *book* [Funktion]
Return scopes in layout book *book*.

`ly:paper-book-systems` *pb* [Funktion]
Return systems in book *pb*.

`ly:paper-book?` *x* [Funktion]
Is *x* a `Paper_book` object?

`ly:paper-fonts` *bp* [Funktion]
Return fonts from the `\paper` block *bp*.

`ly:paper-get-font` *paper-smob chain* [Funktion]
Return a font metric satisfying the font-qualifiers in the alist chain *chain*. (An alist chain is a list of alists, containing grob properties.)

`ly:paper-get-number` *layout-smob name* [Funktion]
Return the layout variable *name*.

`ly:paper-outputscale` *bp* [Funktion]
Get output-scale for *bp*.

`ly:paper-score-paper-systems` *paper-score* [Funktion]
Return vector of `paper_system` objects from *paper-score*.

`ly:paper-system-minimum-distance` *sys1 sys2* [Funktion]
Measure the minimum distance between these two paper-systems, using their stored skylines if possible and falling back to their extents otherwise.

`ly:paper-system?` *obj* [Funktion]
Type predicate.

`ly:parse-file` *name* [Funktion]
Parse a single `.ly` file. Upon failure, throw `ly-file-failed` key.

`ly:parser-clear-error` *parser* [Funktion]
Clear the error flag for the parser.

`ly:parser-clone` *parser-smob* [Funktion]
Return a clone of *parser-smob*.

`ly:parser-define!` *parser-smob symbol val* [Funktion]
Bind *symbol* to *val* in *parser-smob*'s module.

`ly:parser-error` *parser msg input* [Funktion]
Display an error message and make the parser fail.

`ly:parser-has-error?` *parser* [Funktion]
Does *parser* have an error flag?

`ly:parser-lexer` *parser-smob* [Funktion]
Return the lexer for *parser-smob*.

`ly:parser-lookup` *parser-smob symbol* [Funktion]
Look up *symbol* in *parser-smob*'s module. Return `'()` if not defined.

`ly:parser-output-name` *parser* [Funktion]
Return the base name of the output file.

`ly:parser-parse-string` *parser-smob ly-code* [Funktion]
Parse the string *ly-code* with *parser-smob*. Upon failure, throw `ly-file-failed` key.

`ly:parser-set-note-names` *parser names* [Funktion]
Replace current note names in *parser*. *names* is an alist of symbols. This only has effect if the current mode is notes.

`ly:performance-write` *performance filename* [Funktion]
Write *performance* to *filename*.

`ly:pfb->pfa` *pfb-file-name* [Funktion]
Convert the contents of a PFB file to PFA.

`ly:pitch-alteration` *pp* [Funktion]
Extract the alteration from pitch *pp*.

`ly:pitch-diff` *pitch root* [Funktion]
Return pitch *delta* such that *pitch* transposed by *delta* equals *root*.

`ly:pitch-negate` *p* [Funktion]
Negate *p*.

`ly:pitch-notename` *pp* [Funktion]
Extract the note name from pitch *pp*.

`ly:pitch-octave` *pp* [Funktion]
Extract the octave from pitch *pp*.

`ly:pitch-quartertones` *pp* [Funktion]
Calculate the number of quarter tones of *pp* from middle C.

`ly:pitch-semitones` *pp* [Funktion]
Calculate the number of semitones of *pp* from middle C.

`ly:pitch-steps` *p* [Funktion]
Number of steps counted from middle C of the pitch *p*.

`ly:pitch-transpose` *p delta* [Funktion]
Transpose *p* by the amount *delta*, where *delta* is relative to middle C.

`ly:pitch<?` *p1 p2* [Funktion]
Is *p1* lexicographically smaller than *p2*?

`ly:pitch?` *x* [Funktion]
Is *x* a `Pitch` object?

`ly:position-on-line?` *sg spos* [Funktion]
Return whether *pos* is on a line of the staff associated with the the grob *sg* (even on an extender line).

`ly:prob-immutable-properties` *prob* [Funktion]
Retrieve an alist of mutable properties.

`ly:prob-mutable-properties` *prob* [Funktion]
Retrieve an alist of mutable properties.

`ly:prob-property` *obj sym dfault* [Funktion]
Return the value for *sym*.

`ly:prob-property?` *obj sym* [Funktion]
Is boolean prop *sym* set?

`ly:prob-set-property!` *obj sym value* [Funktion]
Set property *sym* of *obj* to *value*.

`ly:prob-type?` *obj type* [Funktion]
Is *obj* the specified prob-type?

`ly:prob?` *x* [Funktion]
Is *x* a `Prob` object?

`ly:programming-error` *str rest* [Funktion]
A Scheme callable function to issue the internal warning *str*. The message is formatted with `format` and *rest*.

`ly:progress` *str rest* [Funktion]
A Scheme callable function to print progress *str*. The message is formatted with `format` and *rest*.

`ly:property-lookup-stats` *sym* [Funktion]
Return hash table with a property access corresponding to *sym*. Choices are `prob`, `grob`, and `context`.

`ly:protects` [Funktion]
Return hash of protected objects.

`ly:pt` *num* [Funktion]
num printer points.

`ly:register-stencil-expression` *symbol* [Funktion]
Add *symbol* as head of a stencil expression.

`ly:relative-group-extent` *elements common axis* [Funktion]
Determine the extent of *elements* relative to *common* in the *axis* direction.

`ly:reset-all-fonts` [Funktion]
Forget all about previously loaded fonts.

`ly:round-filled-box` *xext yext blot* [Funktion]
Make a `Stencil` object that prints a black box of dimensions *xext*, *yext* and roundness *blot*.

`ly:round-filled-polygon` *points blot* [Funktion]
Make a `Stencil` object that prints a black polygon with corners at the points defined by *points* (list of coordinate pairs) and roundness *blot*.

`ly:run-translator` *mus output-def* [Funktion]
Process *mus* according to *output-def*. An interpretation context is set up, and *mus* is interpreted with it. The context is returned in its final state.

Optionally, this routine takes an object-key to to uniquely identify the score block containing it.

`ly:score-add-output-def!` *score def* [Funktion]
Add an output definition *def* to *score*.

`ly:score-embedded-format` *score layout* [Funktion]
Run *score* through *layout* (an output definition) scaled to correct output-scale already, returning a list of layout-lines. This function takes an optional `Object_key` argument.

`ly:score-error?` *score* [Funktion]
Was there an error in the score?

`ly:score-header` *score* [Funktion]
Return score header.

`ly:score-music` *score* [Funktion]
Return score music.

`ly:score-output-defs` *score* [Funktion]
All output definitions in a score.

`ly:score-set-header!` *score module* [Funktion]
Set the score header.

`ly:score?` *x* [Funktion]
Is *x* a `Score` object?

`ly:set-default-scale` *scale* [Funktion]
Set the global default scale.

`ly:set-grob-creation-callback` *cb* [Funktion]
Specify a procedure that will be called every time a new grob is created. The callback will receive as arguments the grob that was created, the name of the C++ source file that caused the grob to be created, and the corresponding line number in the C++ source file.

`ly:set-grob-modification-callback` *cb* [Funktion]
Specify a procedure that will be called every time LilyPond modifies a grob property. The callback will receive as arguments the grob that is being modified, the name of the C++ file in which the modification was requested, the line number in the C++ file in which the modification was requested, the name of the function in which the modification was requested, the property to be changed, and the new value for the property.

`ly:set-middle-C!` *context* [Funktion]
Set the `middleCPosition` variable in *context* based on the variables `middleCClefPosition` and middleCOffset.

`ly:set-option` *var val* [Funktion]
Set a program option.

`ly:set-point-and-click` *what* [Funktion]
Deprecated.

`ly:set-property-cache-callback` *cb* [Funktion]
Specify a procedure that will be called whenever lilypond calculates a callback function and caches the result. The callback will receive as arguments the grob whose property it is, the name of the property, the name of the callback that calculated the property, and the new (cached) value of the property.

`ly:simple-closure?` *clos* [Funktion]
Type predicate.

`ly:skyline-pair?` *x* [Funktion]
Is *x* a `Skyline_pair` object?

`ly:skyline?` *x* [Funktion]
Is *x* a `Skyline` object?

`ly:smob-protects` [Funktion]
Return LilyPond's internal smob protection list.

`ly:solve-spring-rod-problem` *springs rods length ragged* [Funktion]
Solve a spring and rod problem for *count* objects, that are connected by *count*-1 *springs*, and an arbitrary number of *rods*. *count* is implicitly given by *springs* and *rods*. The *springs* argument has the format `(ideal, inverse_hook)` and *rods* is of the form `(idx1, idx2, distance)`.

length is a number, *ragged* a boolean.

The function returns a list containing the force (positive for stretching, negative for compressing and `#f` for non-satisfied constraints) followed by *spring-count*+1 positions of the objects.

`ly:source-file?` *x* [Funktion]
Is *x* a `Source_file` object?

`ly:spanner-bound` *slur dir* [Funktion]
Get one of the bounds of *slur*. *dir* is `-1` for left, and `1` for right.

`ly:spanner-broken-into` *spanner* [Funktion]
Return broken-into list for *spanner*.

`ly:spanner?` *g* [Funktion]
Is *g* a spanner object?

`ly:staff-symbol-line-thickness` *grob* [Funktion]
Returns the line-thickness of the staff associated with *grob*.

`ly:start-environment` [Funktion]
Return the environment (a list of strings) that was in effect at program start.

`ly:stderr-redirect` *file-name mode* [Funktion]
Redirect stderr to *file-name*, opened with *mode*.

`ly:stencil-add` *args* [Funktion]
Combine stencils. Takes any number of arguments.

`ly:stencil-aligned-to` *stil axis dir* [Funktion]
Align *stil* using its own extents. *dir* is a number. `-1` and `1` are left and right, respectively. Other values are interpolated (so `0` means the center).

`ly:stencil-combine-at-edge` *first axis direction second padding minimum* [Funktion]
Construct a stencil by putting *second* next to *first*. *axis* can be 0 (x-axis) or 1 (y-axis). *direction* can be -1 (left or down) or 1 (right or up). The stencils are juxtaposed with *padding* as extra space. If this puts the reference points closer than *minimum*, they are moved by the latter amount. *first* and *second* may also be `'()` or `#f`.

`ly:stencil-empty?` *stil* [Funktion]
Return whether *stil* is empty.

`ly:stencil-expr` *stil* [Funktion]
Return the expression of *stil*.

`ly:stencil-extent` *stil axis* [Funktion]
Return a pair of numbers signifying the extent of *stil* in *axis* direction (`0` or `1` for x and y axis, respectively).

`ly:stencil-fonts` *s* [Funktion]
Analyze *s*, and return a list of fonts used in *s*.

`ly:stencil-in-color` *stc r g b* [Funktion]
Put *stc* in a different color.

`ly:stencil-rotate` *stil angle x y* [Funktion]
Return a stencil *stil* rotated *angle* degrees around the relative offset (*x*, *y*). E.g. an offset of (-1, 1) will rotate the stencil around the left upper corner.

`ly:stencil-rotate-absolute` *stil angle x y* [Funktion]
Return a stencil *stil* rotated *angle* degrees around point (*x*, *y*), given in absolute coordinates.

`ly:stencil-translate` *stil offset* [Funktion]
Return a *stil*, but translated by *offset* (a pair of numbers).

`ly:stencil-translate-axis` *stil amount axis* [Funktion]
Return a copy of *stil* but translated by *amount* in *axis* direction.

`ly:stencil?` *x* [Funktion]
Is *x* a `Stencil` object?

`ly:stream-event?` *x* [Funktion]
Is *x* a `Stream_event` object?

`ly:string-substitute` *a b s* [Funktion]
Replace string *a* by string *b* in string *s*.

`ly:system-font-load` *name* [Funktion]
Load the OpenType system font '`name.otf`'. Fonts loaded with this command must contain three additional SFNT font tables called `LILC`, `LILF`, and `LILY`, needed for typesetting musical elements. Currently, only the Emmentaler and the Aybabtu fonts fulfill these requirements.

Note that only `ly:font-get-glyph` and derived code (like `\lookup`) can access glyphs from the system fonts; text strings are handled exclusively via the Pango interface.

`ly:system-print` *system* [Funktion]
Draw the system and return the prob containing its stencil.

`ly:system-stretch` *system amount-scm* [Funktion]
Stretch the system vertically by the given amount. This must be called before the system is drawn (for example with `ly:system-print`).

`ly:text-dimension` *font text* [Funktion]
Given the font metric in *font* and the string *text*, compute the extents of that text in that font. The return value is a pair of number-pairs.

`ly:text-interface::interpret-markup` [Funktion]
Convert a text markup into a stencil. Takes three arguments, *layout*, *props*, and *markup*.

layout is a `\layout` block; it may be obtained from a grob with `ly:grob-layout`. *props* is an alist chain, i.e. a list of alists. This is typically obtained with `(ly:grob-alist-chain (ly:layout-lookup layout 'text-font-defaults))`. *markup* is the markup text to be processed.

`ly:translator-description` *me* [Funktion]
Return an alist of properties of translator *me*.

`ly:translator-group?` *x* [Funktion]
Is *x* a `Translator_group` object?

`ly:translator-name` *trans* [Funktion]
Return the type name of the translator object *trans*. The name is a symbol.

`ly:translator?` *x* [Funktion]
Is *x* a `Translator` object?

`ly:transpose-key-alist` *l pit* [Funktion]
Make a new key alist of *l* transposed by pitch *pit*.

`ly:truncate-list!` *lst i* [Funktion]
Take at most the first *i* of list *lst*.

`ly:ttf->pfa` *ttf-file-name idx* [Funktion]
Convert the contents of a TrueType font file to PostScript Type 42 font, returning it as a string. The optional *idx* argument is useful for TrueType collections (TTC) only; it specifies the font index within the TTC. The default value of *idx* is 0.

`ly:ttf-ps-name` *ttf-file-name idx* [Funktion]
Extract the PostScript name from a TrueType font. The optional *idx* argument is useful for TrueType collections (TTC) only; it specifies the font index within the TTC. The default value of *idx* is 0.

`ly:unit` [Funktion]
Return the unit used for lengths as a string.
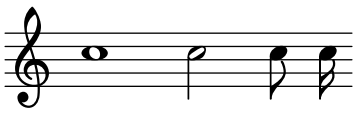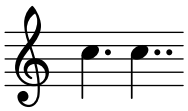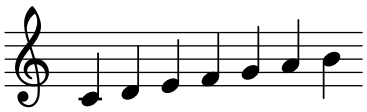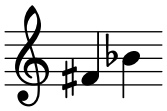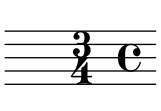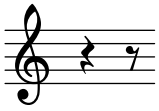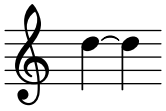
`ly:usage` [Funktion]
Print usage message.
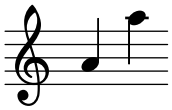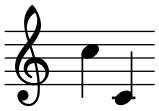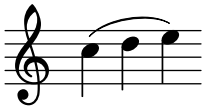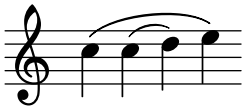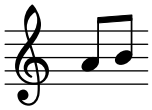
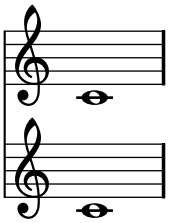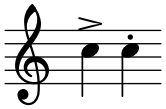
`ly:version` [Funktion]
Return the current lilypond version as a list, e.g., `(1 3 127 uu1)`.
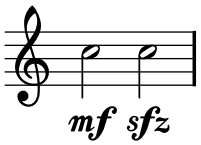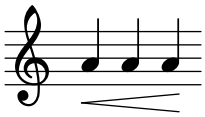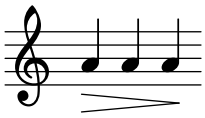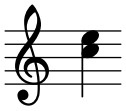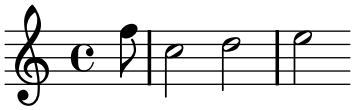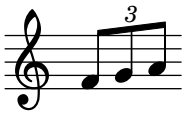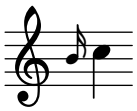
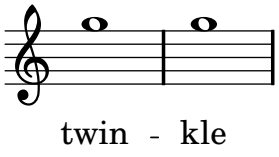
`ly:warning` *str rest* [Funktion]
A Scheme callable function to issue the warning `str`. The message is formatted with `format` and `rest`.
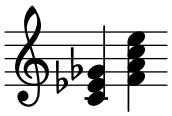
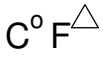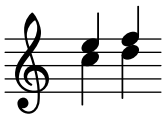
`ly:wide-char->utf-8` *wc* [Funktion]
Encode the Unicode codepoint *wc*, an integer, as UTF-8.

Anhang C 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	
`\times 2/3 {f g a}`	Triolen	
`\grace`	Verzierungen	
`\lyricmode { twinkle }`	Texteingabe	twinkle
`\new Lyrics`	Gesangtext	
`twin -- kle`	Gesangtext-Trennstrich	

`\chordmode { c:dim f:maj7 }`	Akkorde	
`\context ChordNames`	Akkordsymbole drucken	
`<<{e f} \\ {c d}>>`	Mehrstimmigkeit	
`s4 s8 s16`	unsichtbare Pausen	

Anhang D GNU Free Documentation License

Version 1.1, March 2000

Copyright © 2000 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307, USA

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 written 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 that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The ‚Document‘, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as ‚you‘.

   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. (For example, 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.

   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 ‚Transparent‘ copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and 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 has been designed to thwart or discourage subsequent modification by readers is not Transparent. 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 designed for human modification. Opaque formats include PostScript, PDF, 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 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.

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 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 publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. 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 less than five).

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', and 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. In any section entitled ‚Acknowledgments' or ‚Dedications', preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgments 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 as ‚Endorsements' or to conflict in title with any Invariant Section.

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.

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 ‚Acknowledgments', 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, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an ‚aggregate', and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they 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 quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around 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 provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

   You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

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 http://www.gnu.org/copyleft/.

    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.

Anhang: Wie kann die Lizenz für eigene Dokumente verwendet werden

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 1.1
  or any later version published by the Free Software Foundation;
  with the Invariant Sections being list their titles, with the
  Front-Cover Texts being list, and with the Back-Cover Texts being list.
  A copy of the license is included in the section entitled ,GNU
Free Documentation License‘
.
```

If you have no Invariant Sections, write ,with no Invariant Sections‘ instead of saying which ones are invariant. If you have no Front-Cover Texts, write ,no Front-Cover Texts‘ instead of ,Front-Cover Texts being *list*‘; likewise for Back-Cover Texts.

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 E Index der LilyPond-Befehle

Dieser Index listet alle LilyPond Befehle und Schlüsselwörter auf, versehen mit Verweisen zu den Abschnitten im Handbuch, die den Befehl beschreiben oder seine Verwendung diskutieren. Der erste Teil zeigt auf die genaue Stelle im Handbuch, an der der Befehl oder das Schlüsselwort erscheint, der zweite Teil zeigt auf den entsprechenden Abschnitt.

!

! 4

'

' 1

(

(begin * * * *) 56
(end * * * *) 56

,

, ... 1

-

- 81

.

. 30

/

/ 259
/+ 259

:

: 106

<

< 107
<...> 107

=

= 7

>

> 107

?

? 4

[

[..... 65

]

] 65

^

^ 259

_

_ 185, 189

\

\! 83
\(..... 90
\) 90
\< 83
\> 83
\abs-fontsize 460
\accepts 380, 381
\addChordShape 233
\addInstrumentDefinition 142
\addlyrics 187
\addQuote 143
\aeolian 14
\afterGrace 76
\aikenHeads 27
\allowPageTurn 342
\alternative 98
\AncientRemoveEmptyStaffContext 135
\applyContext 430
\applyOutput 431
\arpeggio 94
\arpeggioArrowDown 94
\arpeggioArrowUp 94
\arpeggioBracket 94
\arpeggioNormal 94
\arpeggioParenthesis 94
\arrow-head 175, 482
\ascendens 287, 294
\auctum 287, 294
\augmentum 294
\autoBeamOff 64
\autoBeamOn 64
\autochange 202
\backslashed-digit 492
\balloonGrobText 155
\balloonLengthOff 155
\balloonLengthOn 155
\balloonText 155
\bar 68
\beam 482
\bendAfter 92
\bold 168, 460
\book 305, 306

\bookpart ... 306, 341
\box ... 174, 460
\bracket ... 87, 174, 482
\break ... 339
\breathe ... 91
\breve ... 30
\breve ... 37
\cadenzaOff ... 47
\cadenzaOn ... 47
\caesura ... 286
\caps ... 460
\cavum ... 287, 294
\center-align ... 170, 468
\center-column ... 172, 469
\change ... 201
\char ... 492
\chordmode ... 3, 11, 230
\chords ... 261
\circle ... 174, 482
\clef ... 11
\cm ... 393
\column ... 172, 469
\column-lines ... 496
\combine ... 175, 469
\compressFullBarRests ... 40
\concat ... 470
\context ... 376
\cr ... 83
\crescHairpin ... 84
\crescTextCresc ... 84
\cueDuring ... 146
\decr ... 83
\defaultTimeSignature ... 44
\deminutum ... 287, 294
\denies ... 381
\descendens ... 287, 294
\dimHairpin ... 84
\dimTextDecr ... 84
\dimTextDecresc ... 84
\dimTextDim ... 84
\dir-column ... 470
\displayLilyMusic ... 321, 423
\displayMusic ... 422
\divisioMaior ... 286
\divisioMaxima ... 286
\divisioMinima ... 286
\dorian ... 14
\dotsDown ... 31
\dotsNeutral ... 31
\dotsUp ... 31
\doubleflat ... 486
\doublesharp ... 486
\downbow ... 213
\draw-circle ... 175, 483
\draw-line ... 175, 483
\drummode ... 122
\dynamic ... 87, 461
\dynamicDown ... 84
\dynamicNeutral ... 84
\dynamicUp ... 84
\easyHeadsOff ... 26
\easyHeadsOn ... 26
\epsfile ... 176, 483
\espressivo ... 84
\expandFullBarRests ... 40
\f ... 83
\featherDurations ... 67
\ff ... 83
\fff ... 83
\ffff ... 83
\fill-line ... 172, 470
\filled-box ... 175, 483
\finalis ... 286
\finger ... 150, 461
\flat ... 486
\flexa ... 294
\fontCaps ... 461
\fontsize ... 169, 461
\fp ... 83
\fraction ... 492
\frenchChords ... 263
\fret-diagram ... 222, 489
\fret-diagram-terse ... 224, 490
\fret-diagram-verbose ... 226, 490
\fromproperty ... 492
\general-align ... 172, 471
\germanChords ... 263
\glissando ... 93
\grace ... 75
\halign ... 171, 472
\harmonic ... 213
\harp-pedal ... 491
\hbracket ... 174, 484
\hcenter-in ... 473
\header ... 306
\hideKeySignature ... 253
\hideNotes ... 152
\hideStaffSwitch ... 204
\hspace ... 473
\huge ... 149
\huge ... 170, 462
\improvisationOff ... 29
\improvisationOn ... 29
\in ... 393
\inclinatum ... 287, 294
\include ... 315
\instrumentSwitch ... 142
\ionian ... 14
\italianChords ... 263
\italic ... 168, 462
\justified-lines ... 496
\justify ... 173, 474
\justify-field ... 474
\justify-string ... 475
\keepWithTag ... 317
\key ... 14, 27
\killCues ... 147
\label ... 313
\laissezVibrer ... 35
\large ... 149
\large ... 170, 462
\larger ... 169, 170, 462
\layout ... 306, 338
\left-align ... 170, 475
\left-column ... 476
\line ... 476
\linea ... 287, 294
\locrian ... 14
\longa ... 30
\longa ... 37

\lookup ... 493
\lower ... 171, 476
\lydian ... 14
\lyricmode ... 184, 187
\lyricsto ... 187
\magnify ... 169, 462
\major ... 14
\makeClusters ... 108
\mark ... 74, 162
\markalphabet ... 493
\markletter ... 493
\markup ... 165, 167
\markuplines ... 166, 179
\maxima ... 30
\maxima ... 37
\medium ... 463
\melisma ... 191
\melismaEnd ... 191
\mergeDifferentlyDottedOff ... 112
\mergeDifferentlyDottedOn ... 112
\mergeDifferentlyHeadedOff ... 112
\mergeDifferentlyHeadedOn ... 112
\mf ... 83
\midi ... 306
\minor ... 14
\mixolydian ... 14
\mm ... 393
\mp ... 83
\musicglyph ... 486
\natural ... 486
\new ... 376
\noBreak ... 339
\noPageBreak ... 341
\noPageTurn ... 342
\normal-size-sub ... 463
\normal-size-super ... 463
\normal-text ... 464
\normalsize ... 149
\normalsize ... 170, 464
\note ... 487
\note-by-number ... 487
\null ... 494
\number ... 464
\numericTimeSignature ... 44
\octaveCheck ... 7
\on-the-fly ... 494
\oneVoice ... 109
\open ... 213
\oriscus ... 287, 294
\ottava ... 16
\override ... 388, 494
\override-lines ... 496
\p ... 83
\pad-around ... 174, 476
\pad-markup ... 174, 477
\pad-to-box ... 174, 477
\pad-x ... 174, 477
\page-ref ... 313, 494
\pageBreak ... 341
\pageTurn ... 342
\paper ... 306, 333
\parallelMusic ... 119
\parenthesize ... 154
\partcombine ... 116
\partial ... 46
\partial ... 98, 99
\pes ... 294
\phrasingSlurDashed ... 90
\phrasingSlurDotted ... 90
\phrasingSlurDown ... 90
\phrasingSlurNeutral ... 90
\phrasingSlurSolid ... 90
\phrasingSlurUp ... 90
\phrygian ... 14
\pitchedTrill ... 97
\postscript ... 176, 484
\pp ... 83
\ppp ... 83
\pppp ... 83
\ppppp ... 83
\predefinedFretboardsOff ... 237
\predefinedFretboardsOn ... 237
\property in \lyricmode ... 184
\pt ... 393
\put-adjacent ... 478
\quilisma ... 287, 294
\quoteDuring ... 143
\raise ... 171, 478
\relative ... 2, 3, 11, 203
\RemoveEmptyRhythmicStaffContext ... 135
\RemoveEmptyStaffContext ... 134, 135
\removeWithTag ... 317
\repeat ... 98
\repeat percent ... 104
\repeat tremolo ... 106
\repeatTie ... 35, 99
\rest ... 37
\rfz ... 83
\right-align ... 170, 478
\right-column ... 478
\rightHandFinger ... 240
\roman ... 464
\rotate ... 479
\rounded-box ... 174, 485
\sacredHarpHeads ... 27
\sans ... 465
\scaleDurations ... 34
\score ... 304, 306, 487
\semiflat ... 488
\semiGermanChords ... 263
\semisharp ... 488
\sesquiflat ... 489
\sesquisharp ... 489
\set ... 387
\sf ... 83
\sff ... 83
\sfz ... 83
\sharp ... 489
\shiftOff ... 112
\shiftOn ... 112
\shiftOnn ... 112
\shiftOnnn ... 112
\showKeySignature ... 253
\showStaffSwitch ... 204
\simple ... 465
\skip ... 39
\slashed-digit ... 494
\slurDashed ... 89
\slurDotted ... 89
\slurDown ... 89

\slurNeutral ... 89
\slurSolid ... 89
\slurUp ... 89
\small ... 149
\small ... 170, 465
\smallCaps ... 465
\smaller ... 169, 170, 466
\sostenutoOff ... 206
\sostenutoOn ... 206
\sp ... 83
\spp ... 83
\startGroup ... 158
\startStaff ... 129, 130
\startTrillSpan ... 96
\stemDown ... 155
\stemNeutral ... 155
\stemUp ... 155
\stencil ... 495
\stopGroup ... 158
\stopStaff ... 129, 130
\stopTrillSpan ... 96
\storePredefinedDiagram ... 233
\stropha ... 287, 294
\strut ... 495
\sub ... 169, 466
\super ... 169, 466
\sustainOff ... 206
\sustainOn ... 206
\table-of-contents ... 314
\tag ... 317
\taor ... 253
\teeny ... 149
\teeny ... 170, 467
\tempo ... 137
\text ... 467
\textLengthOff ... 161
\textLengthOn ... 161
\thumb ... 150
\tied-lyric ... 489
\tieDashed ... 36
\tieDotted ... 36
\tieDown ... 36
\tieNeutral ... 36
\tieSolid ... 36
\tieUp ... 36
\time ... 43
\times ... 31
\tiny ... 149
\tiny ... 170, 467
\tocItem ... 314
\translate ... 172, 479
\translate-scaled ... 172, 479
\transparent ... 495
\transpose ... 3, 8, 11
\transposedCueDuring ... 147
\transposition ... 16, 143
\treCorde ... 206
\triangle ... 175, 485
\trill ... 96
\tupletDown ... 31
\tupletNeutral ... 31
\tupletUp ... 31
\tweak ... 389
\typewriter ... 467
\unaCorda ... 206
\underline ... 168, 468
\unfoldRepeats ... 326
\unHideNotes ... 152
\unset ... 387
\upbow ... 213
\upright ... 468
\vcenter ... 480
\verbatim-file ... 495
\virga ... 287, 294
\virgula ... 286
\voiceFourStyle ... 112
\voiceNeutralStyle ... 112
\voiceOne ... 109
\voiceOne ... \voiceFour ... 109
\voiceOneStyle ... 112
\voiceThreeStyle ... 112
\voiceTwoStyle ... 112
\whiteout ... 495
\with ... 377
\with-color ... 153, 495
\with-dimensions ... 496
\with-url ... 485
\wordwrap ... 173, 480
\wordwrap-field ... 480
\wordwrap-internal ... 496
\wordwrap-lines ... 496
\wordwrap-string ... 481
\wordwrap-string-internal ... 497

|

| ... 73

~

~ ... 34

A

add ChordShape ... 233
addInstrumentDefinition ... 142
addQuote ... 143
aeolian ... 14
after-title-space ... 333
aikenHeads ... 27
alignAboveContext ... 381
alignBelowContext ... 381
AncientRemoveEmptyStaffContext ... 135
annotate-spacing ... 370
arpeggio ... 94
arpeggioArrowDown ... 94
arpeggioArrowUp ... 94
arpeggioBracket ... 94
arpeggioNormal ... 94
arpeggioParenthesis ... 94
arranger ... 308
aug ... 256
auto-first-page-number ... 335
autoBeaming ... 56
autoBeamSettings ... 56
autochange ... 202

B

Balloon_engraver ... 155
balloonGrobText ... 155
balloonLengthOff ... 155
balloonLengthOn ... 155
balloonText ... 155
banjo-c-tuning ... 242
banjo-modal-tuning ... 242
banjo-open-d-tuning ... 242
banjo-open-dm-tuning ... 242
barCheckSynchronize ... 73
barNumberVisibility ... 70
base-shortest-duration ... 360
before-title-space ... 333
bendAfter ... 92
between-system-padding ... 333
between-system-space ... 333
between-title-space ... 333
blank-last-page-force ... 335
blank-page-force ... 335
bookTitleMarkup ... 311
bottom-margin ... 333
bracket ... 87, 206
breakable ... 55
breakbefore ... 308
breathe ... 91

C

change ... 201
chordmode ... 3, 11, 230
chordNameExceptions ... 263
ChordNames ... 230
chordNameSeparator ... 263
chordNoteNamer ... 263
chordPrefixSpacer ... 263
chordRootNamer ... 263
clef ... 11
color ... 153
common-shortest-duration ... 360
composer ... 308
controlpitch ... 7
copyright ... 308
cr ... 83
crescHairpin ... 84
crescTextCresc ... 84
cross ... 26
cross-staff ... 204
cueDuring ... 146
currentBarNumber ... 70, 80

D

decr ... 83
dedication ... 308
default ... 18, 19
defaultBarType ... 70
dim ... 256
dimHairpin ... 84
dimTextDecr ... 84
dimTextDecresc ... 84
dimTextDim ... 84
dodecaphonic ... 22
dorian ... 14
drummode ... 122
dynamic ... 87
dynamicDown ... 84
DynamicLineSpanner ... 84
dynamicNeutral ... 84
dynamicUp ... 84

E

easyHeadsOff ... 26
easyHeadsOn ... 26
espressivo ... 84
evenFooterMarkup ... 312
evenHeaderMarkup ... 311

F

f ... 83
ff ... 83
fff ... 83
ffff ... 83
finger ... 150
first-page-number ... 335
flag-style ... 204
followVoice ... 204
font-interface ... 149
font-interface ... 180
font-size ... 149
fontSize ... 149
foot-separation ... 333
forget ... 23
four-string-banjo ... 242
fp ... 83
fret-diagram ... 222
fret-diagram-interface ... 228
fret-diagram-terse ... 224
fret-diagram-verbose ... 226
FretBoards ... 230

G

glissando ... 93
Grid_line_span_engraver ... 156
Grid_point_engraver ... 156
gridInterval ... 156

H

head-separation ... 333
hideKeySignature ... 253
hideNotes ... 152
hideStaffSwitch ... 204
horizontal-shift ... 335
Horizontal_bracket_engraver ... 158
huge ... 149

I

improvisationOff ... 29
improvisationOn ... 29
indent ... 140, 335, 363
instrument ... 308
instrumentSwitch ... 142
ionian ... 14

K

key 14, 27
killCues 147

L

large 149
layout file 337
left-margin 335
length 204
line-width 335, 363
locrian 14
ly:minimal-breaking 342
ly:optimal-breaking 341
ly:page-turn-breaking 341
lydian 14

M

m 256
magstep 149
magstep 394
maj 256
major 14
major seven symbols 263
majorSevenSymbol 263
make-dynamic-script 87
make-pango-font-tree 182
makeClusters 108
measureLength 80
measurePosition 80
mergeDifferentlyDottedOff 112
mergeDifferentlyDottedOn 112
mergeDifferentlyHeadedOff 112
mergeDifferentlyHeadedOn 112
meter 308
mf 83
minimumFret 217
minimumPageTurnLength 342
minimumRepeatLengthForPageTurn 342
minor 14
mixed 206
mixolydian 14
modern 20
modern-cautionary 20
modern-voice 21
modern-voice-cautionary 21
mp 83

N

neo-modern 22
neo-modern-cautionary 22
no-reset 23
normalsize 149

O

octaveCheck 7
oddFooterMarkup 311
oddHeaderMarkup 311
oneVoice 109
opus 308
ottava 16
outside-staff-horizontal-padding 358
outside-staff-padding 358
outside-staff-priority 358

P

p 83
page-breaking-between-system-padding 336
page-count 336
page-limit-inter-system-space 336
page-limit-inter-system-space-factor 336
page-spacing-weight 336
page-top-space 333
paper-height 333
paper-width 335
parallelMusic 119
parenthesize 154
partcombine 116
pedalSustainStyle 206
percent 104
phrasingSlurDashed 90
phrasingSlurDotted 90
phrasingSlurDown 90
phrasingSlurNeutral 90
phrasingSlurSolid 90
phrasingSlurUp 90
phrygian 14
piano 21
piano-cautionary 21
PianoStaff 200, 202
piece 308
pipeSymbol 73
pitchedTrill 97
poet 308
pp 83
ppp 83
pppp 83
ppppp 83
predefinedFretboardsOff 237
predefinedFretboardsOn 237
print-all-headers 311, 336
print-first-page-number 336
print-page-number 336

Q

quotedEventTypes 145
quoteDuring 143

R

r 37
R 40
ragged-bottom 336
ragged-last 336, 363
ragged-last-bottom 336
ragged-right 336, 363
relative 2, 3, 11, 203
RemoveEmptyRhythmicStaffContext 135
RemoveEmptyStaffContext 134, 135
repeatCommands 101
rfz 83
rgb-color 153
rightHandFinger 240

S

s ... 39
sacredHarpHeads ... 27
scoreTitleMarkup ... 311
set-accidental-style ... 18
set-octavation ... 16
sf ... 83
sff ... 83
sfz ... 83
shiftOff ... 112
shiftOn ... 112
shiftOnn ... 112
shiftOnnn ... 112
short-indent ... 140, 335
show-available-fonts ... 181
showFirstLength ... 322
showKeySignature ... 253
showLastLength ... 322
showStaffSwitch ... 204
skipTypesetting ... 322
slurDashed ... 89
slurDotted ... 89
slurDown ... 89
slurNeutral ... 89
slurSolid ... 89
slurUp ... 89
small ... 149
sostenutoOff ... 206
sostenutoOn ... 206
sp ... 83
spacing ... 360
spp ... 83
staff-padding ... 201
Staff.midiInstrument ... 323
start-repeat ... 101
startGroup ... 158
startStaff ... 129, 130
startTrillSpan ... 96
Stem ... 204
stem-spacing-correction ... 360
stemDown ... 155
stemLeftBeamCount ... 65
stemNeutral ... 155
stemRightBeamCount ... 65
stemUp ... 155
stopGroup ... 158
stopStaff ... 129, 130
stopTrillSpan ... 96
storePredefinedDiagram ... 233
stringTunings ... 230
StringTunings ... 220
subdivideBeams ... 59
subsubtitle ... 308
subtitle ... 308
suggestAccidentals ... 281
sus ... 259
sustainOff ... 206
sustainOn ... 206
system-count ... 337
system-separator-markup ... 336

T

TabStaff ... 217
TabVoice ... 217
tagline ... 308
taor ... 253
teaching ... 22
teeny ... 149
tempo ... 137
text ... 206
textSpannerDown ... 161
textSpannerNeutral ... 161
textSpannerUp ... 161
thumb ... 150
tiny ... 149
title ... 308
top-margin ... 333
transpose ... 3, 8, 11
transposedCueDuring ... 147
transposition ... 16, 143
treCorde ... 206
tremolo ... 106
tremoloFlags ... 106
trill ... 96
TupletNumber ... 32
tupletNumberFormatFunction ... 32
tupletSpannerDuration ... 32

U

unaCorda ... 206
unfold ... 103
unHideNotes ... 152

V

voice ... 18, 20
Voice ... 109
voiceOne ... 109

W

whichBar ... 70
with-color ... 153

X

x11-color ... 153
x11-color ... 154

Anhang F LilyPond-Index

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.

!

! 5

'

' 1

(

(begin * * * *) 57
(end * * * *) 57

,

, ... 1

-

- 83

.

. 32

/

/ 265
/+ 265

:

: 108

<

< 109
<...> 109

=

= 8

>

> 109

?

? 5

[

[..... 66

]

] 66

^

^ 264

_

_ 189, 193

\

\! 85
\(..... 92
\) 92
\< 85
\> 85
\abs-fontsize 367
\addChordShape 237
\addlyrics 190, 191
\aeolian 15
\afterGrace 78
\aikenHeads 28
\alternative 100
\AncientRemoveEmptyStaffContext 138
\arpeggio 96
\arpeggioArrowDown 96
\arpeggioArrowUp 96
\arpeggioBracket 96
\arpeggioNormal 96
\arpeggioParenthesis 96
\arrow-head 179, 389
\ascendens 294
\auctum 294
\augmentum 294
\autoBeamOff 65
\autoBeamOn 65
\autochange 206
\backslashed-digit 399
\balloonGrobText 159
\balloonLengthOff 159
\balloonLengthOn 159
\balloonText 159
\bar 69
\beam 389
\bendAfter 94
\bold 172, 367
\book 304, 305
\bookpart 305
\box 177, 367
\bracket 89, 177, 389
\breathe 93
\breve 31, 39
\cadenzaOff 48

\cadenzaOn ... 48
\caesura ... 287
\caps ... 368
\cavum ... 294
\center-align ... 174, 376
\center-column ... 176, 376
\change ... 206
\char ... 399
\chordmode ... 4, 12, 235
\chords ... 266
\circle ... 177, 389
\clef ... 12
\column ... 176, 376
\column-lines ... 403
\combine ... 179, 377
\compressFullBarRests ... 42
\concat ... 377
\cr ... 85
\crescHairpin ... 86
\crescTextCresc ... 86
\decr ... 85
\defaultTimeSignature ... 46
\deminutum ... 294
\descendens ... 294
\dimHairpin ... 86
\dimTextDecr ... 86
\dimTextDecresc ... 86
\dimTextDim ... 86
\dir-column ... 377
\displayLilyMusic ... 321
\divisioMaior ... 287
\divisioMaxima ... 287
\divisioMinima ... 287
\dorian ... 15
\dotsDown ... 32
\dotsNeutral ... 32
\dotsUp ... 32
\doubleflat ... 393
\doublesharp ... 393
\downbow ... 217
\draw-circle ... 179, 390
\draw-line ... 179, 390
\dynamic ... 89, 368
\dynamicDown ... 86
\dynamicNeutral ... 86
\dynamicUp ... 86
\easyHeadsOff ... 28
\easyHeadsOn ... 28
\epsfile ... 179, 390
\espressivo ... 86
\expandFullBarRests ... 42
\f ... 85
\featherDurations ... 68
\ff ... 85
\fff ... 85
\ffff ... 85
\fill-line ... 176, 378
\filled-box ... 179, 390
\finalis ... 287
\finger ... 153, 368
\flat ... 393
\flexa ... 294
\fontCaps ... 368
\fontsize ... 172, 368
\fp ... 85
\fraction ... 399
\frenchChords ... 269
\fret-diagram ... 227, 396
\fret-diagram-terse ... 229, 397
\fret-diagram-verbose ... 230, 397
\fromproperty ... 399
\general-align ... 175, 378
\germanChords ... 269
\glissando ... 95
\grace ... 77
\halign ... 174, 379
\harmonic ... 218
\harp-pedal ... 398
\hbracket ... 177, 391
\hcenter-in ... 380
\header ... 305
\hideKeySignature ... 258
\hideNotes ... 155
\hideStaffSwitch ... 208
\hspace ... 380
\huge ... 152, 174, 369
\improvisationOff ... 30
\improvisationOn ... 30
\inclinatum ... 294
\include ... 314
\ionian ... 15
\italianChords ... 269
\italic ... 172, 369
\justified-lines ... 403
\justify ... 176, 381
\justify-field ... 381
\justify-string ... 382
\keepWithTag ... 317
\key ... 15, 28
\label, ... 312
\laissezVibrer ... 37
\large ... 152, 174, 369
\larger ... 172, 174, 369
\layout ... 305
\left-align ... 174, 382
\left-column ... 383
\line ... 383
\linea ... 294
\locrian ... 15
\longa ... 31, 39
\lookup ... 400
\lower ... 175, 383
\lydian ... 15
\lyricmode ... 188, 191
\lyricsto ... 191
\magnify ... 172, 370
\major ... 15
\makeClusters ... 111
\mark ... 75, 165
\markalphabet ... 400
\markletter ... 400
\markup ... 169, 170
\markuplines ... 169, 183
\maxima ... 31, 39
\medium ... 370
\melisma ... 194
\melismaEnd ... 194
\mergeDifferentlyDottedOff ... 114
\mergeDifferentlyDottedOn ... 114
\mergeDifferentlyHeadedOff ... 114

\mergeDifferentlyHeadedOn ... 114
\mf ... 85
\midi ... 305
\minor ... 15
\mixolydian ... 15
\mp ... 85
\musicglyph ... 393
\natural ... 393
\normal-size-sub ... 370
\normal-size-super ... 370
\normal-text ... 371
\normalsize ... 152, 174, 371
\note ... 394
\note-by-number ... 394
\null ... 401
\number ... 371
\numericTimeSignature ... 46
\octaveCheck ... 8
\on-the-fly ... 401
\oneVoice ... 111
\open ... 217
\oriscus ... 294
\ottava ... 17
\override ... 401
\override-lines ... 403
\p ... 85
\pad-around ... 178, 383
\pad-markup ... 178, 384
\pad-to-box ... 178, 384
\pad-x ... 178, 384
\page-ref ... 401
\page-ref ... 312
\paper ... 305, 311
\parallelMusic ... 121
\parenthesize ... 157
\partcombine ... 118
\partial ... 47, 100
\partial ... 101
\pes ... 294
\phrasingSlurDashed ... 92
\phrasingSlurDotted ... 92
\phrasingSlurDown ... 92
\phrasingSlurNeutral ... 92
\phrasingSlurSolid ... 92
\phrasingSlurUp ... 92
\phrygian ... 15
\pitchedTrill ... 99
\postscript ... 179, 391
\pp ... 85
\ppp ... 85
\pppp ... 85
\ppppp ... 85
\predefinedFretboardsOff ... 242
\predefinedFretboardsOn ... 242
\property in \lyricmode ... 188
\put-adjacent ... 385
\quilisma ... 294
\raise ... 175, 385
\relative ... 2, 4, 12, 207
\RemoveEmptyRhythmicStaffContext ... 138
\RemoveEmptyStaffContext ... 137
\removeWithTag ... 317
\repeat ... 100
\repeat percent ... 106
\repeat tremolo ... 108
\repeatTie ... 37
\repeatTie ... 101
\rest ... 39
\rfz ... 85
\right-align ... 174, 385
\right-column ... 385
\rightHandFinger ... 245
\roman ... 372
\rotate ... 386
\rounded-box ... 177, 392
\sacredHarpHeads ... 28
\sans ... 372
\scaleDurations ... 35
\score ... 303, 305, 394
\semiflat ... 395
\semiGermanChords ... 269
\semisharp ... 395
\sesquiflat ... 396
\sesquisharp ... 396
\sf ... 85
\sff ... 85
\sfz ... 85
\sharp ... 396
\shiftOff ... 114
\shiftOn ... 114
\shiftOnn ... 114
\shiftOnnn ... 114
\showKeySignature ... 258
\showStaffSwitch ... 208
\simple ... 372
\skip ... 40
\slashed-digit ... 401
\slurDashed ... 91
\slurDotted ... 91
\slurDown ... 91
\slurNeutral ... 91
\slurSolid ... 91
\slurUp ... 91
\small ... 152, 174, 372
\smallCaps ... 373
\smaller ... 172, 174, 373
\sostenutoOff ... 210
\sostenutoOn ... 210
\sp ... 85
\spp ... 85
\startGroup ... 162
\startStaff ... 132
\startTrillSpan ... 98
\stemDown ... 158
\stemNeutral ... 158
\stemUp ... 158
\stencil ... 402
\stopGroup ... 162
\stopStaff ... 132
\stopTrillSpan ... 98
\storePredefinedDiagram ... 237
\stropha ... 294
\strut ... 402
\sub ... 173, 373
\super ... 173, 374
\sustainOff ... 210
\sustainOn ... 210
\table-of-contents ... 314
\tag ... 317
\taor ... 258

\teeny ... 152, 174, 374
\tempo ... 140
\text ... 374
\textLengthOff ... 164
\textLengthOn ... 164
\thumb ... 153
\tied-lyric ... 396
\tieDashed ... 37
\tieDotted ... 37
\tieDown ... 37
\tieNeutral ... 37
\tieSolid ... 37
\tieUp ... 37
\time ... 45
\times ... 32
\tiny ... 152, 174, 374
\tocItem ... 314
\translate ... 175, 386
\translate-scaled ... 175, 386
\transparent ... 402
\transpose ... 4, 9, 12
\transposition ... 17
\treCorde ... 210
\triangle ... 179, 392
\trill ... 98
\tupletDown ... 33
\tupletNeutral ... 33
\tupletUp ... 33
\typewriter ... 375
\unaCorda ... 210
\underline ... 172, 375
\unfoldRepeats ... 326
\unHideNotes ... 155
\upbow ... 217
\upright ... 375
\vcenter ... 387
\verbatim-file ... 402
\virga ... 294
\virgula ... 287
\voiceFourStyle ... 114
\voiceNeutralStyle ... 114
\voiceOne ... 111
\voiceOne ... \voiceFour ... 111
\voiceOneStyle ... 114
\voiceThreeStyle ... 114
\voiceTwoStyle ... 114
\whiteout ... 402
\with-color ... 156, 402
\with-dimensions ... 403
\with-url ... 392
\wordwrap ... 176, 387
\wordwrap-field ... 387
\wordwrap-internal ... 403
\wordwrap-lines ... 403
\wordwrap-string ... 388
\wordwrap-string-internal ... 404

|

| ... 74, 75

~

~ ... 36

1

15ma ... 17

8

8va ... 17
8ve ... 17

A

a due ... 121
a due-Stellen ... 118
Abbildungen im Text ... 177
Abschnitte markieren ... 75
absolute Lautstärke ... 85
Absolute Spezifikation von Oktaven ... 1
Absoluter Modus: Tonhöhen ... 1
Abstand vergrößern, Gesangstext ... 197
Abstand von Hilfslinien ... 131
Abstand zwischen Systemen in Klaviernoten ... 209
Abstrich ... 83, 404
acciaccatura ... 337, 429
Accidental ... 6, 25, 280
accidental-interface ... 6
accidental-suggestion-interface ... 25
Accidental_engraver ... 6, 25, 297
AccidentalCautionary ... 6
AccidentalPlacement ... 25
Accidentals and key signatures ... 4, 6, 15, 16, 299
AccidentalSuggestion ... 25, 297
add ChordShape ... 237
addChordShape ... 337, 429
adding a white background to text ... 402
Adding and removing engravers ... 52
addInstrumentDefinition ... 337, 429
Additionen in Akkorden ... 263
addQuote ... 337, 429
aeolian ... 15
Aeolisch ... 15
afterGrace ... 337, 430
agestrichelter Legatobogen ... 91
Aiken-Notenköpfe ... 28
aikenHeads ... 28
Akkolade ... 125
Akkolade ... 129
Akkord ... 110, 261, 268
Akkord, gebrochen ... 96
Akkord-Diagramme ... 234
Akkordbezeichnungen ... 260, 266
Akkordbezeichnungen und Bunddiagramme ... 235
Akkorddiagramm ... 226
Akkorddiagramme, automatisch ... 242
Akkorde ... 109, 260
Akkorde über zwei Systeme ... 209
Akkorde und relativer Modus ... 3
Akkorde und Überbindungen ... 36
Akkorde, Entfernen von Tönen ... 264
Akkorde, zwischen Systemen mit \autochange ... 208
Akkorde: farbige Noten ... 157
Akkorde: Fingersatz ... 153
Akkorde: Versetzungszeichen ... 25
Akkordeigenschaften ... 261
Akkordeon ... 211
Akkordeon, Diskant-Symbole ... 211

Akkordeon, Register ... 211
Akkordformen für Bundinstrumente ... 237
Akkordmodi ... 261
Akkordmodus ... 260
Akkordstufen, Alteration ... 264
Akkordstufen, Veränderung ... 264
Akkordsymbole ... 266
Akkordsymbole in MIDI ... 326
Akkordsymbole, anpassen ... 268
Akkordtabulatur ... 226
Akzent ... 83
`Akzent` ... 85
Akzent ... 404
Akzidentien ... 4, 19
al niente ... 87
`al niente` ... 88
`allowPageTurn` ... 337, 430
Alte Schlüssel ... 12
alternative Schlüsse in ausgeschriebenen Wiederholungen ... 106
Alternative Schlüsse mit Bindebogen ... 101
alternativer Schluss ... 100
Altschlüssel ... 12
Ambitus ... 25
`Ambitus` ... 27
`ambitus-interface` ... 27
`Ambitus_engraver` ... 27
`AmbitusAccidental` ... 27
`AmbitusLine` ... 27
`AmbitusNoteHead` ... 27
Analyse ... 162
andere Stimmen zitieren ... 149
Ändern von Instrumentenbezeichnungn ... 144
Anfänger, Notenlernen ... 28
Anführungsstriche im Text ... 171
Anführungszeichen, Gesangstext ... 188
Angabe der Oktave: absolut ... 1
Anmerkung, Blase ... 159
Anpassen von Akkordsymbolen ... 268
Anpassen von Bunddiagrammen ... 232
Anstrich ... 83
Anzahl der Notenlinien einstellen ... 131
Anzahl der Wiederholung, ändern ... 103
Äolisch ... 15
`applyContext` ... 337, 430
`applyMusic` ... 337, 430
`applyOutput` ... 337, 430
`appoggiatura` ... 337, 430
arabische Musik ... 297
arabische Musik, Beispiel ... 301
arabische Notenbezeichnungen ... 297
Arabische Taktarten ... 300
arabische Tonarten ... 298
arabische Vorzeichen ... 298
arabisches Halb-B Versetzungszeichen ... 298
`arpeggio` ... 96
Arpeggio ... 96
`Arpeggio` ... 98
Arpeggio über Systeme im Klammernstil ... 98
Arpeggio-Symbole, besondere ... 96
`arpeggioArrowDown` ... 96
`arpeggioArrowUp` ... 96
`arpeggioBracket` ... 96
`arpeggioNormal` ... 96
`arpeggioParenthesis` ... 96
`arranger` ... 307
Art der Übungszeichen ... 76
Arten von Notenköpfen ... 366
`Articulation and dynamics` ... 88
articulation-event ... 148
articulations ... 285
Artikulationszeichen ... 83
`assertBeamQuant` ... 337, 430
`assertBeamSlope` ... 337, 430
Atemzeichen ... 93
Aufführungsanweisung: Tempo ... 140
Aufklappen von wiederholten Noten ... 106
Auflösungszeichen ... 4
Aufstrich ... 404
Auftakt ... 47
`Auftakt` ... 48
Auftakt in Wiederholung ... 101
Aufteilen von Noten ... 52
`aug` ... 262
Ausgabe von Akkordbezeichnungen ... 266
ausgeschriebene Wiederholungen ... 106
Ausklingen lassen ... 37
Ausklingen lassen, Bögen ... 37
Ausnahmen, Akkordsymbole ... 269
Ausrichten an Kadenz ... 81
Ausrichtung von Gesangstext ... 191
Ausrichtung von Taktlinien ... 73
Ausrichtung von Text ... 174
Aussehen von Taktnummern ... 72
Auswahl von Schriftgröße (Notation) ... 152
auto-knee-gap ... 56
autobeam ... 57
`autoBeaming` ... 57
autoBeamOff ... 55
autoBeamOn ... 55
`autoBeamSettings` ... 57
`autochange` ... 206
`autochange` ... 337, 430
`AutoChangeMusic` ... 208
automatische Ausrichtung von Silben ... 191
Automatische Balken, einstellen ... 57
automatische Bunddiagramme ... 242
automatische Kombination von Stimmen ... 118
Automatische Versetzungszeichen ... 19
Automatischer Systemwechsel ... 206
automatischer Systemwechsel und relativer Modus ... 207
automatisches Aufteilen von Noten ... 52

B

B ... 4
`B` ... 6
backslashed digits ... 399
Balken mit Knie ... 56
Balken und Zeilenumbrüche ... 56
Balken zwischen Systemen ... 206
Balken, gespreizt ... 68
Balken, manuell ... 55, 66
Balkenpausen, mehrtaktig ... 43
Ballon ... 159
`balloon-interface` ... 160
`Balloon_engraver` ... 159, 160
`balloonGrobText` ... 159
`balloonGrobText` ... 338, 430

balloonLengthOff ... 159
balloonLengthOn ... 159
balloonText ... 159
balloonText ... 338, 430
BalloonTextItem ... 160
banjo-c-tuning ... 247
banjo-modal-tuning ... 247
banjo-open-d-tuning ... 247
banjo-open-dm-tuning ... 247
Banjo-Stimmung ... 247
Banjo-Tabulatur ... 220
Banjo-Tabulaturen ... 247
bar ... 338, 430
bar-line-interface ... 416
Bar_engraver ... 268
barCheckSynchronize ... 74
Baritonschlüssel ... 12
BarLine ... 71
BarNumber ... 74
barNumberCheck ... 338, 430
barNumberVisibility ... 72
Barre, Gitarre ... 227
Barret, anzeigen für Bundinstrumente ... 246
Bartók-Pizzicato ... 218
BassFigure ... 276, 277
BassFigureAlignment ... 276, 277
BassFigureBracket ... 276, 277
BassFigureContinuation ... 276, 277
BassFigureLine ... 276, 277
Bassnote in Akkorden ... 265
Basso continuo ... 272
Bassschlüssel ... 12
Beam ... 57, 206, 224
beats, grouping ... 60
Bebalkung, automatisch, Einstellungen ... 57
Beenden eines Notensystems ... 131
Beenden eines Systems ... 132
Beginn eines Notensystems ... 124
Beginn von Wiederholung ... 103
Beginnen eines Notensystems ... 131
Beispiel der arabischen Musik ... 301
bendAfter ... 94
bendAfter ... 338, 430
Beschriftung ... 83
Beschriftung über Mehrtaktpausen ... 42
Beschriftung, Text ... 170
besondere Arpeggio-Symbole ... 96
besondere Notenköpfe ... 27
besondere Zeichen, Text ... 171
Bezifferter Bass ... 272
Bilder einbinden ... 179
Bindebogen ... 36
Bindebogen ... 38, 52
Bindebogen in alternativem Schluss ... 101
Bindebogen in Wiederholung ... 101
Bindebögen und Akkorde ... 36
Bindebogen und Wiederholung ... 103
Bindebögen wiederholen ... 37
Bindebögen, Ausehen ... 37
Bindebögen, durchgehend ... 37
Bindebögen, gepunktet ... 37
Bindebogen, Gesangstext ... 193
Bindebögen, gestrichelt ... 37
Bindestriche, Gesangstext ... 189, 195
Blase ... 159
Blasinstrumente ... 257
Blöcke, Text ... 176
Blocksatz, Text ... 176
Bogen zur Phrasierung ... 92
Bogen, Anzeige ... 217
Bögen, gleichzeitig ... 91
Bögen, gleichzeitige Phrasierung ... 92
Bögen, laissez vibrer ... 37
Bögen, manuelle Platzierung ... 91
Bögen, mehrfach ... 91
Bögen, Phrasierung ... 91
Bögen, über Noten ... 91
Bögen, unter Noten ... 91
bookTitleMarkup ... 310
bracket ... 89, 211
Bratschenschlüssel ... 12
breakable ... 56
breakbefore ... 308
breathe ... 93
breathe ... 338, 430
BreathingSign ... 94, 287
Brevis ... 32
Brevis ... 39
Bund ... 222
Bunddiagramme ... 226, 234
Bunddiagramme und Akkordbezeichnungen ... 235
Bunddiagramme, anpassen ... 232
Bunddiagramme, ausführlicher Stil ... 230
Bunddiagramme, automatisch ... 242
Bunddiagramme, eigene ... 226
Bunddiagramme, eigene definieren ... 236
Bunddiagramme, Fingersatz ... 243
Bunddiagramme, knapper Stil ... 229
Bunddiagramme, normaler Stil ... 227
Bunddiagramme, Transposition ... 235
Bundinstrumente, Akkordformen ... 237
Bundinstrumente, Fingersatz der rechten Hand ... 245
Bundinstrumente, Flageolett ... 247
Bundinstrumente, gedämpfte Noten ... 247
Bundinstrumente, Position und Barret anzeigen ... 246
Bundinstrumente, Saitenstimmung ... 225

C

C-Schlüssel ... 12
caesura ... 93
centering a column of text ... 376
change ... 206
changing direction of text columns ... 377
ChoirStaff ... 129
ChoirStaff ... 130
Chor-Tenorschlüssel ... 13
choral score ... 194
chord-Akkorde ... 260
Chord_name_engraver ... 268
chordmode ... 4, 12, 235
ChordName ... 268
chordNameExceptions ... 269
ChordNames ... 140, 235, 266, 268
chordNameSeparator ... 269
chordNoteNamer ... 269
chordPrefixSpacer ... 269
chordRootNamer ... 268
Chorsystem ... 125

circling text ... 389
clef ... 12
clef ... 338, 430
Clef ... 15
clef-interface ... 15
Clef_engraver ... 15
Cluster ... 111
Cluster ... 265
Cluster_spanner_engraver ... 111
ClusterSpanner ... 111
ClusterSpannerBeacon ... 111
Coda ... 76, 83, 404
Coda an Taktlinie ... 165
Collisions of objects ... 118
color ... 156
coloring text ... 402
Combining notes into chords ... 110
Completion_heads_engraver ... 52
composer ... 307
compound time signatures ... 47
concatenating text ... 377
ContextChange ... 206
Contexts and engravers ... 113
Continuo, Generalbass ... 272
controlling general text alignment ... 378
controlpitch ... 8
copyright ... 308
Copyright ... 310
cr ... 85
creating empty text objects ... 401
creating horizontal spaces in text ... 380
creating text fractions ... 399
creating vertical spaces in text ... 402
Crescendo ... 85
Crescendo ... 88
Crescendo-Klammer ... 85
crescHairpin ... 86
crescTextCresc ... 86
cross ... 27
cross-staff ... 209
cueDuring ... 338, 430
CueVoice ... 151
currentBarNumber ... 71, 82
Custodes ... 286
Custos ... 286
Custos_engraver ... 286

D

D'al Segno ... 83
D.S al Fine ... 76
Dal Segno ... 76
Dämpfung, Bundinstrumente ... 247
Dateien einfügen ... 314
Dateistruktur ... 305
Dauer ... 31
Dauerbezeichnungen für Noten und Pausen ... 32
Dauern skalieren ... 35
Daumenbezeichnung ... 83, 404
decr ... 85
Decrescendo ... 85
Decrescendo ... 88
dedication ... 307
default ... 19
Default_bar_line_engraver ... 51
defaultBarType ... 71
Definieren von eigenen Bunddiagrammen ... 236
Devnull-Kontext ... 197
Dicke der Notenlinien einstellen ... 131
didaktischer Versetzungszeichenstil ... 23
Die Feta-Schriftart ... 393
dim ... 262
dimHairpin ... 86
Diminuendo ... 85
dimTextDecr ... 86
dimTextDecresc ... 86
dimTextDim ... 86
Diskantsymbole, Akkordeon ... 211
dispatcher ... 433
displayLilyMusic ... 338, 430
displayMusic ... 338, 430
divisio ... 286
divisiones ... 286
dodecaphonic ... 23
dodekaphoner Versetzungszeichenstil ... 23
doit ... 94
doits ... 94
Doppel-B ... 4
Doppel-B ... 6
Doppelkreuz ... 4
Doppelkreuz ... 6
Doppellinie ... 69
Doppelpraller ... 404
Doppelschlag ... 83
doppelte Taktartensymbole ... 49
dorian ... 15
Dorisch ... 15
DotColumn ... 32
Dots ... 32
DoublePercentRepeat ... 108
DoublePercentRepeatCounter ... 108
drawing beams within text ... 389
drawing boxes with rounded corners ... 390
drawing boxes with rounded corners around text ... 392
drawing circles within text ... 390
drawing lines within text ... 390
drawing solid boxes within text ... 390
drawing triangles within text ... 392
Dreiklänge ... 261
Drums ... 248
DrumStaff ... 125, 255
DrumVoice ... 255
Dudelsack ... 258
Dur ... 15
durchgehender Legatobogen ... 91
durchgestrichener Hals ... 79
durchsichtige Noten ... 155
dynamic ... 89
dynamic-event ... 148
dynamicDown ... 86
DynamicLineSpanner ... 86, 89
dynamicNeutral ... 86
DynamicText ... 89
dynamicUp ... 86
Dynamik ... 85
Dynamik, mehrere Zeichen an einer Note ... 85
Dynamik, vertikale Position ... 86
Dynamik, zentriert für Tasteninstrumente ... 205

Dynamikzeichen, Anmerkung ... 89
Dynamikzeichen, eigene ... 89
Dynamikzeichen, Klammer ... 89

E

easyHeadsOff ... 28
easyHeadsOn ... 28
editorische Dynamikzeichen ... 89
editorische Noten ... 157
eigene Bunddiagramme ... 226, 232
Eigene Bunddiagramme definieren ... 236
eigene Dynamikzeichen ... 89
eigene Tabulaturen ... 225
ein System, Mehrstimmigkeit ... 111
Einbinden von Graphik ... 179
Einfärben von Objekten ... 156
Einfärben von Stimmen ... 114
einfügen von Dateien ... 314
Einfügen von Notationsobjekten ... 180
Eingabe von Noten parallel ... 121
Eingabedatei, Struktur ... 305
Einstellung von automatischen Balken ... 407
Einstellung von Hilfslinien ... 131
Einstellungen der Bebalkung ... 57
einzelnes Notensystem ... 124
Einzug ... 143
enclosing text in a box with rounded corners ... 392
enclosing text within a box ... 367
Ende von Wiederholung ... 103
endSpanners ... 338, 430
Engravers explained ... 52
Entfernen von Stufen in Akkorden ... 264
Entfernen von Tönen aus Akkorden ... 263
Erinnerungsvorzeichen ... 5
Erklärungsblase ... 159
erste Klammer ... 100
erweiterte Akkorde ... 263
espressivo ... 86
Espressivo ... 83, 404
Espressivo-Artikulation ... 86
evenFooterMarkup ... 311
evenHeaderMarkup ... 311
Explicitly instantiating voices ... 113
Explicitly instantiating voices ... 114

F

f ... 85
F-Schlüssel ... 12
Fähnchen, Alte Musik ... 283
falls ... 94
Farbe ... 156
Farbe, RGB ... 157
Färben von Stimmen ... 114
Farben, Liste ... 351
farbige Noten ... 156
farbige Noten in Akkorden ... 157
FDL, GNU Free Documentation License ... 456
featherDurations ... 338, 430
Fermate ... 83, 404
Fermate an Taktlinie ... 165
Fermate über Mehrtaktpausen ... 42
Feta font ... 352
ff ... 85
fff ... 85
ffff ... 85
FiguredBass ... 140, 276, 277
finalis ... 286
finger ... 153
Fingering ... 155, 222
fingering-event ... 155
Fingering_engraver ... 155
FingeringEvent ... 155
Fingersatz ... 153, 404
Fingersatz der rechten Hand, Bundinstrumente ... 245
Fingersatz in Bunddiagrammen ... 243
Fingersatz versus Saitenzahl ... 220
Fingersatz: Akkorde ... 153
Fingersatz: Daumen-Zeichen ... 153
Fingerwechsel ... 153
Fixing overlapping notation ... 206
flag-style ... 209
Flageolet ... 83, 404
Flageolett ... 218
Flageolett in Tabulaturen ... 222
Flageolett, Bundinstrumente ... 247
Flageolett, künstliches ... 218
Flageolett-Notenköpfe ... 27
Folgen einer Stimmen in anderes System ... 208
followVoice ... 208
Font, Feta ... 352
Font, Größe ändern für Notation ... 152
font-interface ... 153, 183, 401
font-size ... 152, 153
fontSize ... 152
Forbid_line_break_engraver ... 52
forget ... 24
forget-Versetzungszeichenstil ... 24
Form-Notenköpfe ... 28
Formatierung von Triolen ... 33
four-string-banjo ... 247
fp ... 85
Fragmente ... 146, 149
Französischer Violinschlüssel ... 12
fret (Bunddiagramme) ... 227
Fret (Bunddiagramme) ... 226
fret-diagram ... 227
fret-diagram-interface ... 232, 234, 238, 242
fret-diagram-interface ... 244
fret-diagram-terse ... 229
fret-diagram-terse-Markup ... 229
fret-diagram-verbose ... 230
fret-diagram-verbose-Markup ... 230
FretBoards ... 234
Fülllinie ... 195
Füllung um Text ... 178
Fußbezeichnung ... 404
Fußzeile ... 311

G

G-Schlüssel ... 12
Gabel ... 88
Ganztaktpausen ... 39, 41
Gebrochene Akkorde ... 96
gedämpft ... 83
Gedämpft ... 404
gedämpfte Noten, Bundinstrumente ... 247

Gedanktenstriche, Gesangstext ... 189
Geisternoten ... 157
Generalbass ... 272
`Generalbass` ... 273
Generalbass Fortsetzungslinie ... 275
gepunkteter Legatobogen ... 91
gerundeter Kasten, Graphik ... 177
Gesangstext ... 188
Gesangstext und Balken ... 57
Gesangstext, Ausrichtung ... 191
Gesangstext, einer Stimme zugewiesen ... 111
Gesangstext, Platz zwischen Silben ... 197
Gesangstext, überspringen ... 41
Gesangstext, Variablen ... 191
geschweifte Klammer ... 125
geschweifte Klammern, Schachteln ... 129
gespreizte Balken ... 68
gestopft ... 83
Gitarrengriffsymbole ... 226
Gitarrennotenköpfe ... 27
Gitarrentabulatur ... 220
Gitterlinien ... 160
`gleichartig` ... 108
gleichzeitige Bögen ... 91
gleichzeitige Noten: Versetzungszeichen ... 25
gleichzeitige Phrasierungsbögen ... 92
Gleiten in Tabulaturen ... 222
Gleiten nach oben/unten ... 94
`glissando` ... 95
`Glissando` ... 95
`Glissando nach unten` ... 94
Glissando, nach oben ... 94
Glissando, nach unten ... 94
Glissando, unbestimmt ... 94
`grace` ... 338, 431
grace notes within tuplet brackets ... 35
`GraceMusic` ... 80
`GrandStaff` ... 25
`GrandStaff` ... 129
Graphik einbinden ... 179
Graphik, eingebunden ... 177
Graphische Notation ... 179
Gregorianische quadratische Neumenligaturen ... 289
Gregorianischer Choral, Transkription ... 124
`GregorianTranscriptionStaff` ... 125
`grid-line-interface` ... 162
`grid-point-interface` ... 162
`Grid_line_span_engraver` ... 160, 162
`Grid_point_engraver` ... 160, 162
`gridInterval` ... 160
`GridLine` ... 162
`GridPoint` ... 162
Griff: Fingersatz ... 153
Griffsymbole, Bundinstrumente ... 226
Größe der Schriftart ... 172
Größe von Notensystem verändern ... 133
grouping beats ... 60
Grundton eines Akkordes ... 263
Grundton eines Akkords ... 261

H

`Hairpin` ... 89
Hal Leonard ... 28
Halb-B ... 5, 7
Halb-B-Versetzungszeichen, arabische Musik ... 298
halber Takt ... 47
Halbkreuz ... 5, 7
Hals ... 158
Hals nach oben ... 158
Hals nach unten ... 158
Hals neutral ... 158
Hals, mit Schrägstrich ... 79
Hals, Richtung von ... 158
Hals, unsichtbar ... 158
Hälse über zwei Systeme ... 209
Haltepedal, Stile ... 211
Harfe ... 215
Harfenpedal ... 215
`harmonic` ... 218
`hideKeySignature` ... 258
`hideNotes` ... 155
`hideStaffSwitch` ... 208
Hilfe, Blase ... 159
`Hilfslinie` ... 133
Hilfslinien, Abstände ... 131
Hilfslinien, Einstellungen ... 131
Hinzufügen von Tönen in Akkorden ... 263
hochgestellt ... 173
`horizontal-bracket-interface` ... 162
`Horizontal_bracket_engraver` ... 162
`HorizontalBracket` ... 162
horizontale Ausrichtung von Text ... 174
horizontale Kklammer ... 162
horizontally centering text ... 376
`How LilyPond input files work` ... 307
Hufnagel ... 279
`huge` ... 152

I

`I'm hearing Voices` ... 114, 250
Illustrationen im Text ... 177
importing stencils into text ... 402
Improvisation ... 30
`improvisationOff` ... 30
`improvisationOn` ... 30
`includePageLayoutFile` ... 338, 431
indent ... 143
inlining an Encapsulated PostScript image ... 390
inserting music into text ... 394
inserting PostScript directly into text ... 391
inserting URL links into text ... 392
`instrument` ... 307
`instrument-specific-markup-interface` ... 401
Instrumentbezeichnungen ... 323
Instrumente, transponierende ... 10
Instrumentenbezeichnung, Notation ... 143
Instrumentenbezeichnungen ... 142
Instrumentenbezeichnungen, wechseln ... 144
Instrumentengruppe ... 125
Instrumentenwechsel ... 145
`InstrumentName` ... 146
`instrumentSwitch` ... 338, 431
`Intervalle` ... 4
`ionian` ... 15
Ionisch ... 15

J

Justierung von Notensystemen ... 131
justifying lines of text ... 403
justifying text ... 381

K

Kadenz ... 48
Kadenz ... 49, 82
Kadenz, Ausrichten an ... 81
Kammerton ... 19
Kasten, Graphik ... 177
keepWithTag ... 317
keepWithTag ... 338, 431
key ... 15, 28
key-cancellation-interface ... 16
key-signature-interface ... 16
Key_engraver ... 16
Key_performer ... 16
KeyCancellation ... 16
KeyChangeEvent ... 16
KeySignature ... 16, 280, 281, 299
killCues ... 338, 431
Kirchenpausen ... 43
Kirchentonart ... 16
Kirchentonarten ... 15
Klammer ... 129
Klammer, Crescendo ... 85
Klammer, erste (Wiederholung) ... 100
Klammer, geschweift ... 125
Klammer, vertikal ... 125
Klammer, Wiederholung ... 103
Klammer, Wiederholung mit Text ... 104
Klammer-Arpeggio über Systeme ... 98
Klammern ... 162
Klammern um Noten ... 157
Klammern um Vorzeichen ... 5
Klammern, Analyse ... 162
Klammern, Graphik ... 177
Klammern, spitze ... 109
Klammern, Verschachteln ... 129
Klang ... 322
Klavier, Pedalbezeichnung ... 210
Klavier-Versetzungszeichenstil ... 22
Klavier: Warnungsversetzungszeichen ... 23
Klaviermusik, Dynamik zentrierten ... 205
Klaviersystem ... 125, 205
Knall-Pizzicato ... 218
Kombinieren von Stimmen ... 118
Komprimieren von Noten ... 35
Kontroll-Tonhöhe ... 8
Kopfzeile ... 311
Kreuz ... 4
Kreuz ... 6
Kreuznotenköpfe ... 27
künstliches Flageolett ... 218
kurze Instrumentenbezeichnungen ... 142

L

label ... 338, 431
Laissez vibrer ... 37
Laissez-vibrer-Bögen ... 38
LaissezVibrerTie ... 38
LaissezVibrerTieColumn ... 38
large ... 152
Lautstärke ... 85
Layout der Seite ... 311
ledger-line-spanner-interface ... 27
Ledger_line_engraver ... 27
LedgerLineSpanner ... 27
leere Systeme verstecken ... 137
Leerzeichen, Gesangstext ... 188, 189
left aligning text ... 382
Legatobögen ... 90
Legatobogen ... 92
Legatobogen zur Phrasierung ... 92
Legatobogen, gepunktet ... 91
Legatobogen, gestrichelt ... 91
Legatobögen, manuelle Platzierung ... 91
Legatobogen, massiv ... 91
Legatobogen-Stil ... 91
length ... 209
Length and thickness of objects ... 137
Ligature_bracket_engraver ... 288
LigatureBracket ... 287
Ligaturen ... 287
Ligaturen der quadratischen Neumennotation ... 289
Ligaturen, weiße Mensuralnotation ... 288
ligatures in text ... 377
lilypond aufrufen ... 322
Linie ... 133
Linien zwischen Systemen ... 160
Linien, Gitter ... 160
Liste der Farben ... 351
listener ... 433
locrian ... 15
Lokrisch ... 15
Longa ... 32
Longa ... 39
lowering text ... 383
ly:add-file-name-alist ... 433
ly:add-interface ... 433
ly:add-listener ... 433
ly:add-option ... 433
ly:all-grob-interfaces ... 433
ly:all-options ... 433
ly:all-stencil-expressions ... 433
ly:assoc-get ... 433
ly:book-add-bookpart! ... 433
ly:book-add-score! ... 433
ly:book-process ... 433
ly:book-process-to-systems ... 433
ly:box? ... 434
ly:bp ... 434
ly:bracket ... 434
ly:broadcast ... 434
ly:camel-case->lisp-identifier ... 434
ly:chain-assoc-get ... 434
ly:clear-anonymous-modules ... 434
ly:cm ... 434
ly:command-line-code ... 434
ly:command-line-options ... 434
ly:command-line-verbose? ... 434
ly:connect-dispatchers ... 434
ly:context-event-source ... 434
ly:context-events-below ... 434
ly:context-find ... 434
ly:context-grob-definition ... 434

ly:context-id ... 434
ly:context-name ... 434
ly:context-now ... 435
ly:context-parent ... 435
ly:context-property ... 435
ly:context-property-where-defined ... 435
ly:context-pushpop-property ... 435
ly:context-set-property! ... 435
ly:context-unset-property ... 435
ly:context? ... 435
ly:default-scale ... 435
ly:dimension? ... 435
ly:dir? ... 435
ly:duration->string ... 435
ly:duration-dot-count ... 435
ly:duration-factor ... 435
ly:duration-length ... 435
ly:duration-log ... 435
ly:duration<? ... 435
ly:duration? ... 435
ly:effective-prefix ... 435
ly:error ... 436
ly:eval-simple-closure ... 436
ly:event-deep-copy ... 436
ly:event-property ... 436
ly:event-set-property! ... 436
ly:expand-environment ... 436
ly:export ... 436
ly:find-accidentals-simple ... 436
ly:find-file ... 436
ly:font-config-add-directory ... 436
ly:font-config-add-font ... 436
ly:font-config-display-fonts ... 436
ly:font-config-get-font-file ... 436
ly:font-design-size ... 436
ly:font-file-name ... 436
ly:font-get-glyph ... 436
ly:font-glyph-name-to-charcode ... 437
ly:font-glyph-name-to-index ... 437
ly:font-index-to-charcode ... 437
ly:font-magnification ... 437
ly:font-metric? ... 437
ly:font-name ... 437
ly:font-sub-fonts ... 437
ly:format ... 437
ly:format-output ... 437
ly:get-all-function-documentation ... 437
ly:get-all-translators ... 437
ly:get-glyph ... 437
ly:get-listened-event-classes ... 437
ly:get-option ... 437
ly:gettext ... 438
ly:grob-alist-chain ... 438
ly:grob-array-length ... 438
ly:grob-array-ref ... 438
ly:grob-array? ... 438
ly:grob-basic-properties ... 438
ly:grob-common-refpoint ... 438
ly:grob-common-refpoint-of-array ... 438
ly:grob-default-font ... 438
ly:grob-extent ... 438
ly:grob-interfaces ... 438
ly:grob-layout ... 438
ly:grob-object ... 438
ly:grob-original ... 438
ly:grob-parent ... 438
ly:grob-pq<? ... 438
ly:grob-properties ... 438
ly:grob-property ... 438
ly:grob-property-data ... 439
ly:grob-relative-coordinate ... 439
ly:grob-robust-relative-extent ... 439
ly:grob-script-priority-less ... 439
ly:grob-set-property! ... 439
ly:grob-staff-position ... 439
ly:grob-suicide! ... 439
ly:grob-system ... 439
ly:grob-translate-axis! ... 439
ly:grob? ... 439
ly:gulp-file ... 439
ly:hash-table-keys ... 439
ly:inch ... 439
ly:input-both-locations ... 439
ly:input-file-line-char-column ... 439
ly:input-location? ... 439
ly:input-message ... 439
ly:interpret-music-expression ... 439
ly:interpret-stencil-expression ... 440
ly:intlog2 ... 440
ly:is-listened-event-class ... 440
ly:item-break-dir ... 440
ly:item? ... 440
ly:iterator? ... 440
ly:lexer-keywords ... 440
ly:lily-lexer? ... 440
ly:lily-parser? ... 440
ly:make-book ... 440
ly:make-book-part ... 440
ly:make-dispatcher ... 440
ly:make-duration ... 440
ly:make-global-context ... 440
ly:make-global-translator ... 440
ly:make-listener ... 440
ly:make-moment ... 441
ly:make-music ... 441
ly:make-music-function ... 441
ly:make-output-def ... 441
ly:make-page-label-marker ... 441
ly:make-page-permission-marker ... 441
ly:make-pango-description-string ... 441
ly:make-paper-outputter ... 441
ly:make-pitch ... 441
ly:make-prob ... 441
ly:make-scale ... 441
ly:make-score ... 441
ly:make-simple-closure ... 441
ly:make-stencil ... 441
ly:make-stream-event ... 442
ly:message ... 442
ly:minimal-breaking ... 442
ly:mm ... 442
ly:module->alist ... 442
ly:module-copy ... 442
ly:modules-lookup ... 442
ly:moment-add ... 442
ly:moment-div ... 442
ly:moment-grace-denominator ... 442
ly:moment-grace-numerator ... 442
ly:moment-main-denominator ... 442
ly:moment-main-numerator ... 442

ly:moment-mod ... 442
ly:moment-mul ... 442
ly:moment-sub ... 442
ly:moment<? ... 442
ly:moment? ... 442
ly:music-compress ... 443
ly:music-deep-copy ... 443
ly:music-duration-compress ... 443
ly:music-duration-length ... 443
ly:music-function-extract ... 443
ly:music-function? ... 443
ly:music-length ... 443
ly:music-list? ... 443
ly:music-mutable-properties ... 443
ly:music-output? ... 443
ly:music-property ... 443
ly:music-set-property! ... 443
ly:music-transpose ... 443
ly:music? ... 443
ly:note-head::stem-attachment ... 443
ly:number->string ... 443
ly:optimal-breaking ... 443
ly:option-usage ... 443
ly:otf->cff ... 443
ly:otf-font-glyph-info ... 444
ly:otf-font-table-data ... 444
ly:otf-font? ... 444
ly:otf-glyph-list ... 444
ly:output-def-clone ... 444
ly:output-def-lookup ... 444
ly:output-def-parent ... 444
ly:output-def-scope ... 444
ly:output-def-set-variable! ... 444
ly:output-def? ... 444
ly:output-description ... 444
ly:output-formats ... 444
ly:outputter-close ... 444
ly:outputter-dump-stencil ... 444
ly:outputter-dump-string ... 444
ly:outputter-output-scheme ... 444
ly:outputter-port ... 444
ly:page-marker? ... 444
ly:page-turn-breaking ... 444
ly:pango-font-physical-fonts ... 445
ly:pango-font? ... 445
ly:paper-book-pages ... 445
ly:paper-book-paper ... 445
ly:paper-book-performances ... 445
ly:paper-book-scopes ... 445
ly:paper-book-systems ... 445
ly:paper-book? ... 445
ly:paper-fonts ... 445
ly:paper-get-font ... 445
ly:paper-get-number ... 445
ly:paper-outputscale ... 445
ly:paper-score-paper-systems ... 445
ly:paper-system-minimum-distance ... 445
ly:paper-system? ... 445
ly:parse-file ... 445
ly:parser-clear-error ... 445
ly:parser-clone ... 445
ly:parser-define! ... 445
ly:parser-error ... 446
ly:parser-has-error? ... 446
ly:parser-lexer ... 446
ly:parser-lookup ... 446
ly:parser-output-name ... 446
ly:parser-parse-string ... 446
ly:parser-set-note-names ... 446
ly:performance-write ... 446
ly:pfb->pfa ... 446
ly:pitch-alteration ... 446
ly:pitch-diff ... 446
ly:pitch-negate ... 446
ly:pitch-notename ... 446
ly:pitch-octave ... 446
ly:pitch-quartertones ... 446
ly:pitch-semitones ... 446
ly:pitch-steps ... 446
ly:pitch-transpose ... 446
ly:pitch<? ... 446
ly:pitch? ... 447
ly:position-on-line? ... 447
ly:prob-immutable-properties ... 447
ly:prob-mutable-properties ... 447
ly:prob-property ... 447
ly:prob-property? ... 447
ly:prob-set-property! ... 447
ly:prob-type? ... 447
ly:prob? ... 447
ly:programming-error ... 447
ly:progress ... 447
ly:property-lookup-stats ... 447
ly:protects ... 447
ly:pt ... 447
ly:register-stencil-expression ... 447
ly:relative-group-extent ... 447
ly:reset-all-fonts ... 447
ly:round-filled-box ... 447
ly:round-filled-polygon ... 448
ly:run-translator ... 448
ly:score-add-output-def! ... 448
ly:score-embedded-format ... 448
ly:score-error? ... 448
ly:score-header ... 448
ly:score-music ... 448
ly:score-output-defs ... 448
ly:score-set-header! ... 448
ly:score? ... 448
ly:set-default-scale ... 448
ly:set-grob-creation-callback ... 448
ly:set-grob-modification-callback ... 448
ly:set-middle-C! ... 448
ly:set-option ... 448
ly:set-point-and-click ... 449
ly:set-property-cache-callback ... 449
ly:simple-closure? ... 449
ly:skyline-pair? ... 449
ly:skyline? ... 449
ly:smob-protects ... 449
ly:solve-spring-rod-problem ... 449
ly:source-file? ... 449
ly:spanner-bound ... 449
ly:spanner-broken-into ... 449
ly:spanner? ... 449
ly:staff-symbol-line-thickness ... 449
ly:start-environment ... 449
ly:stderr-redirect ... 449
ly:stencil-add ... 449
ly:stencil-aligned-to ... 450

ly:stencil-combine-at-edge ... 450
ly:stencil-empty? ... 450
ly:stencil-expr ... 450
ly:stencil-extent ... 450
ly:stencil-fonts ... 450
ly:stencil-in-color ... 450
ly:stencil-rotate ... 450
ly:stencil-rotate-absolute ... 450
ly:stencil-translate ... 450
ly:stencil-translate-axis ... 450
ly:stencil? ... 450
ly:stream-event? ... 450
ly:string-substitute ... 450
ly:system-font-load ... 450
ly:system-print ... 450
ly:system-stretch ... 451
ly:text-dimension ... 451
ly:text-interface::interpret-markup ... 451
ly:translator-description ... 451
ly:translator-group? ... 451
ly:translator-name ... 451
ly:translator? ... 451
ly:transpose-key-alist ... 451
ly:truncate-list! ... 451
ly:ttf->pfa ... 451
ly:ttf-ps-name ... 451
ly:unit ... 451
ly:usage ... 451
ly:version ... 451
ly:warning ... 451
ly:wide-char->utf-8 ... 451
lydian ... 15
Lydisch ... 15
LyricCombineMusic ... 191, 193
LyricExtender ... 195
LyricHyphen ... 195
Lyrics ... 140, 191, 193, 410
LyricSpace ... 190
LyricText ... 190, 204

M

m ... 262
magnifying text ... 370
magstep ... 152
maj ... 262
major ... 15
major seven symbols ... 269
majorSevenSymbol ... 268
make-dynamic-script ... 89
make-pango-font-tree ... 186
makeClusters ... 111
makeClusters ... 338, 431
manuelle Balken ... 55
manuelle Systemwechsel ... 206
manuelle Wiederholungszeichen ... 103
Maqam ... 297
Marcato ... 83, 404
Marke ... 317
markierte Noten behalten ... 317
markierte Noten entfernen ... 317
markup ... 170
markup, Syntax ... 170
massiver Legatobogen ... 91
Maxima ... 39
measure groupings ... 60
measure sub-grouping ... 60
measureLength ... 82
measurePosition ... 47, 82
Medicaea, Editio ... 279
mehre Dynamikzeichen an einer Note ... 85
mehrere Phrasierungsbögen ... 92
mehrere Stimmen ... 114
mehrfache Bögen ... 91
Mehrstimmigkeit ... 111
Mehrstimmigkeit, ein System ... 111
Mehrtaktpause mit Fermate ... 42
Mehrtaktpausen ... 39, 41
Mehrtaktpausen, ausschreiben ... 42
Mehrtaktpausen, Beschriftung ... 42
Mehrtaktpausen, komprimieren ... 42
Mehrtaktpausen, Positionierung ... 44
Mehrtaktpausen, Text hinzufügen ... 42
mehrzeiliger Text ... 176
Melisma ... 193, 195
Mensur ... 284
Mensural_ligature_engraver ... 280, 288
Mensuralligaturen ... 288
Mensuralmusik, Transkription ... 128
Mensuralnotation ... 279
MensuralStaff ... 125
MensuralStaffContext ... 295
MensuralVoiceContext ... 295
Mensurstriche ... 128
mergeDifferentlyDottedOff ... 114
mergeDifferentlyDottedOn ... 114
mergeDifferentlyHeadedOff ... 114
mergeDifferentlyHeadedOn ... 114
merging text ... 377
meter ... 307
Metronomangabe ... 142
Metronombezeichnung ... 140
Metronombezeichnung ... 142
MetronomeMark ... 142
Metrum ... 45
Metrum ... 51
Metrum, Noten ohne ... 48, 82
Metrum, polymetrisch ... 49
Mezzosopranschlüssel ... 12
mf ... 85
MIDI ... 17, 322
MIDI und Wiederholungen ... 326
MIDI, Akkordsymbole ... 326
MIDI, Mikrotöne ... 326
MIDI, Rhythmen ... 326
MIDI, Tonhöhen ... 326
MIDI, Vierteltöne ... 326
MIDI-Instrumentenbezeichnungen ... 350
MIDI-Kontextdefinitionen ... 325
MIDI-Transposition ... 17
MIDI-Umgebung ... 325
Mikrotöne ... 5, 7
Mikrotöne in MIDI ... 326
minimumFret ... 222
minor ... 15
mixed ... 211
mixolydian ... 15
Mixolydisch ... 15
modern ... 21

`modern-cautionary` ... 21
`modern-voice` ... 22
`modern-voice-cautionary` ... 22
modern-Warnung-Versetzungszeichenstil ... 21
moderne Versetzungszeichen ... 22
Moderner Stil, Versetzungszeichen ... 21
moderner Versetzungszeichensitl mit Warnungen für Stimmen ... 22
moderner Versetzungszeichenstil ... 21, 22
moderner Verseztungszeichenstil mit Warnungen ... 21
Modi, in Akkorden ... 261
Modifikatoren, Akkorde ... 261
Modus ... 15
Moll ... 15
Mordent ... 83, 404
`Moving objects` ... 174, 177, 205
`mp` ... 85
`MultiMeasureRest` ... 45
`MultiMeasureRestNumber` ... 45
`MultiMeasureRestText` ... 45
`Multiple notes at once` ... 118
`Music expressions explained` ... 304
Musica ficta ... 296
`musicMap` ... 338, 431
Musik komprimieren ... 35
Musikanalyse ... 162
Musikobjekte, Einfügen ... 180
musikwissenschaftliche Analyse ... 162

N

`N-tole` ... 35
N-tole, Formatierung ... 33
N-tolen ... 32
Nachschlag ... 78
Name von Sänger ... 200
`neo-modern` ... 23
`neo-modern-cautionary` ... 23
neo-modern-cautionary-Versetzungszeichenstil ... 23
neo-moderner Versetzungszeichenstil ... 23
`Nesting music expressions` ... 133, 137
neue Dynamikzeichen ... 89
neues Notensystem ... 124
`New_fingering_engraver` ... 155
niente, al ... 87
`no-reset` ... 24
`noPageBreak` ... 338, 431
`noPageTurn` ... 339, 431
normale Wiederholung ... 100
`normalsize` ... 152
Notation für Streicher ... 216
Notation, Aiken ... 28
Notation, Erklärungen ... 159
Notationsobjekte, Einfügen ... 180
`note-collision-interface` ... 420, 422, 424
`note-event` ... 27, 28, 30
note-event ... 148
`note-head-interface` ... 27, 28, 30
`Note_head_line_engraver` ... 208
`Note_heads_engraver` ... 27, 28, 30, 52
`Note_spacing_engraver` ... 156
`NoteCollision` ... 118
`NoteColumn` ... 118
`NoteHead` ... 27, 28, 30, 279
Noteknöpfe, einfache Notation ... 28
Noten in Klammern ... 157
Noten komprimieren ... 35
Noten ohne Metrum ... 82
Noten ohne Takt ... 48, 82
Noten verschmelzen ... 114
Noten verstecken ... 155
Noten wiederholt schreiben ... 106
Noten, aufteilen ... 52
Noten, durchsichtig ... 155
Noten, farbig ... 156
Noten, farbige in Akkorden ... 157
Noten, parlato ... 27
Noten, punktiert ... 32
Noten, Schriftgröße ... 152
Noten, Stichnoten ... 149
Noten, transponieren ... 9
Noten, unsichtbar ... 155
Noten, Wechsel zwischen Systemen ... 206
Notenbezeichnungen, arabisch ... 297
Notenbezeichnungen, Deutsch ... 4
Notenbezeichnungen, Holländisch ... 4
Notenbezeichnungen, Standard ... 4
Notenbezeichungen, andere Sprachen ... 7
Notencluster ... 111
Noteneingabe: relative Oktavbestimmung ... 2
Notengruppenklammer ... 162
Notenhals, durchgestrichen ... 79
Notenhals, Richtung von ... 158
Notenhals, unsichtbar ... 158
Notenhälse über zwei Systeme ... 209
Notenkopfarten ... 366
Notenköpfe ... 152
Notenköpfe für Anfänger ... 28
Notenköpfe zum Lernen ... 28
Notenköpfe, Alte Musik ... 279
Notenköpfe, besondere ... 27
Notenköpfe, Flageolett ... 27
Notenköpfe, Formen ... 28
Notenköpfe, Gitarre ... 27
Notenköpfe, Improvisation ... 30
Notenköpfe, Kreuz ... 27
Notenköpfe, Raute ... 27
Notenköpfe, sacred harp ... 28
Notenköpfe, Übung ... 28
Notenlänge ... 31
Notenlinien, Anzahl ... 131
Notenlinien, Dicke ... 131
Notenlinien, Einstellungen ... 131
Notenschlüssel ... 12
Notensystem beginnen ... 131
Notensystem stoppen ... 131
Notensystem, beenden ... 132
Notensystem, Größe verändern ... 133
Notensystem, Klavier ... 205
Notensystem, neu ... 124
Notensystem, Tasteninstrumente ... 205
Notensysteme, gruppieren ... 125
Notensysteme, mehrere ... 125
Notensysteme, Modifikation ... 131
Notensystemgruppe ... 125
`Notenwert` ... 32
Notenzusammenstöße ... 114
notes within text by log and dot-count ... 394
notes within text by string ... 394

`NoteSpacing` ... 156
Nummerierung von Saite ... 220
Nummerierung, Strophen ... 199
Nummierung von Takten ... 71
nur Text ... 169

O

Objekte, farbig ... 156
`OctavateEight` ... 15
`octaveCheck` ... 8
`octaveCheck` ... 339, 431
`oddFooterMarkup` ... 311
`oddHeaderMarkup` ... 311
offen ... 83
Offen ... 404
Offene Saite, anzeigen ... 217
Oktavbestimmung, relativ ... 2
Oktavenmodus (relativ) und Akkorde ... 3
Oktavenüberprüfung ... 8
`Oktavierung` ... 17
Oktavierungskorrektur ... 8
Oktavtransposition ... 13
Oktavwechsel: Tonhöhe ... 1
`oldaddlyrics` ... 339, 431
`On the un-nestedness of brackets and ties` ... 92
`On the un-nestedness of brackets and ties` ... 93
`oneVoice` ... 111
`opus` ... 307
Orcherster, Streicher ... 216
`Organizing pieces with variables` ... 124, 315, 320
Orgelpedal-Bezeichnung ... 83
Orgelpedalbezeichnung ... 404
Ornament ... 83
Ornamente ... 77
ossia ... 138
Ossia ... 133
`Ossia` ... 137
Ossia-Systeme ... 133
`Other sources of information` ... 57, 61, 114, 315, 325, 327
`Other uses for tweaks` ... 205
`ottava` ... 17
`ottava` ... 339, 431
`ottava-bracket-interface` ... 17
`Ottava_spanner_engraver` ... 17
`OttavaBracket` ... 17
`overrideProperty` ... 339, 431
overriding properties within text markup ... 401

P

`p` ... 85
pädagogische Notenköpfe ... 28
padding text ... 384
padding text horizontally ... 384
`pageBreak` ... 339, 431
`pageTurn` ... 339, 431
Pango ... 183
Parallele Notation, Eingabe ... 121
`parallelMusic` ... 121
`parallelMusic` ... 339, 431
`parentheses-interface` ... 158
`ParenthesesItem` ... 158
`Parenthesis_engraver` ... 158
`parenthesize` ... 157
`parenthesize` ... 339, 432
Parlato ... 188
Parlato-Notenköpfe ... 27
`partcombine` ... 118
`partcombine` ... 339, 432
`PartCombineMusic` ... 121
partieller Takt ... 47
Partitur ... 125
Pausen ... 39
`Pausen über mehrere Takte` ... 45
Pausen verschieben, automatisch ... 114
Pausen, Alte Musik ... 281
Pausen, Ganztakt- ... 41
Pausen, Kirchenstil ... 43
Pausen, mehrere Takte ausschreiben ... 42
Pausen, mehrere Takte komprimieren ... 42
Pausen, Mehrtakt- ... 41
Pausen, mehrtaktig ... 39
Pausen, unsichtbar ... 40
Pausen, Zusammenfalten ... 45
Pausen, Zusammenstöße ... 45
Pausendauern ... 39
Pausenzeichen ... 93
Pedal, Harfe ... 215
Pedal, sostenuto ... 210
Pedal-Bezeichnung ... 83
Pedalbezeichnung ... 210
Pedalbezeichnung, Klammer ... 211
Pedalbezeichnung, Stile ... 211
Pedalbezeichnung, Text ... 211
Pedaldiagramme, Harfe ... 215
`pedalSustainStyle` ... 211
`percent` ... 106
`PercentRepeat` ... 108
`PercentRepeatCounter` ... 108
`PercentRepeatedMusic` ... 108
Percussionsnotensystem ... 124
Perkussion ... 248, 250
Perkussionsnotensystem ... 124
Petrucci ... 279
Phrasierung, Gesang ... 193
Phrasierungsbögen ... 91, 92
Phrasierungsbögen, gleichzeitig ... 92
Phrasierungsbögen, mehrfach ... 92
Phrasierungsklammern ... 162
Phrasierungszeichen ... 92
`PhrasingSlur` ... 93
`phrasingSlurDashed` ... 92
`phrasingSlurDotted` ... 92
`phrasingSlurDown` ... 92
`phrasingSlurNeutral` ... 92
`phrasingSlurSolid` ... 92
`phrasingSlurUp` ... 92
`phrygian` ... 15
Phrygisch ... 15
`piano` ... 22
`Piano templates` ... 205
Piano, Pedalbezeichnung ... 210
`piano-cautionary` ... 23
Piano-System ... 205
Piano-Versetzungszeichenstil ... 22
`Piano_pedal_engraver` ... 211
`PianoPedalBracket` ... 211

`PianoStaff` ... 25, 98
`PianoStaff` ... 129
`PianoStaff` ... 146, 205, 206
`piece` ... 307
`pipeSymbol` ... 75
`Pitch_squash_engraver` ... 30, 55, 414
`pitchedTrill` ... 99
`pitchedTrill` ... 339, 432
Pizzicato, Bartók ... 218
Pizzicato, Knall- ... 218
placing horizontal brackets around text ... 391
placing vertical brackets around text ... 389
Platz um Text ... 178
Platzhalternoten ... 40
`poet` ... 307
`pointAndClickOff` ... 339, 432
`pointAndClickOn` ... 339, 432
`Polymetrie` ... 35, 51
`Polymetrische Taktangabe` ... 51
polymetrische Taktarten ... 49
Polyphonie ... 111, 114
`Polyphonie` ... 118
Polyphonie, ein System ... 111
Portato ... 83
`Portato` ... 85
Portato ... 404
Position und Barret für Bundinstrumente ... 246
Position von Mehrtaktpausen ... 44
Postscript, Graphik ... 179
`pp` ... 85
`ppp` ... 85
`pppp` ... 85
`ppppp` ... 85
Praller ... 83, 404
Prallermordent ... 404
`predefinedFretboardsOff` ... 242
`predefinedFretboardsOn` ... 242
Prima volta ... 100
`print-all-headers` ... 310
`Prozent-Wiederholung` ... 108
Prozent-Wiederholungen ... 106
Punktierung ... 32
putting space around text ... 384

Q

Quadratische Neumenligaturen ... 289
Quelldatei, Struktur ... 305
`Quinte` ... 4
`quotedEventTypes` ... 148
`quoteDuring` ... 339, 432
`QuoteMusic` ... 149

R

`r` ... 39
`R` ... 41
Rahmen, Text ... 177
railroad tracks ... 93
raising text ... 385
Rand um Text ... 178
Rautennotenköpfe ... 27
`Real music example` ... 205
rechte Hand, Fingersatz für Bundinstrumente ... 245
referencing page numbers in text ... 401
`RehearsalMark` ... 77, 168
Relativ ... 2
`relative` ... 2, 4, 12, 207
Relative Oktavbestimmung ... 2
relative Tonhöhe in Akkorden ... 109
`RelativeOctaveCheck` ... 9
`RelativeOctaveMusic` ... 4
relativer Modus und Akkorde ... 3
relativer Modus und automatischer Systemwechsel ... 207
Relativer Oktavenmodus und Transposition ... 4
`removeWithTag` ... 317
`removeWithTag` ... 339, 432
Renaissancemusik ... 128
`repeatCommands` ... 103
`RepeatedMusic` ... 103, 105, 106
`RepeatSlash` ... 108
repitetive Musik ... 106
`resetRelativeOctave` ... 340, 432
`Rest` ... 40, 281
rest-event ... 148
`RestCollision` ... 118
`rfz` ... 85
`rgb-color` ... 157
RGB-Farbe ... 157
Rhythmen in MIDI ... 326
`RhythmicStaff` ... 30, 55, 125
Rhythmische Aufteilungen ... 32
rhythmisches Notensystem ... 124
right aligning text ... 385
`rightHandFinger` ... 245
`rightHandFinger` ... 340, 432
rotating text ... 386

S

`s` ... 40
Sackpfeife ... 258
sacred harp-Notenköpfe ... 28
`sacredHarpHeads` ... 28
Saite, offen ... 217
Saitenstimmung für Bundinstrumente ... 225
Saitenzahl ... 220
Sängername ... 200
SATB ... 194
Sätze, mehrere ... 304
Satzzeichen ... 188
`scaleDurations` ... 340, 432
scaling text ... 386
Schachtelung von Systemen ... 129
Schlagzeug ... 248, 250
Schluss, alternativer in Wiederholung ... 100
Schlüssel ... 4, 12
Schlüssel Alter Musik ... 12
Schlüssel, Alte Musik ... 281
Schlüssel, C ... 12
Schlüssel, F ... 12
Schlüssel, G ... 12
Schlüssel, transponierend ... 13
Schottischer Dudelsack ... 258
schräge Notenköpfe ... 30
Schriftarten, Hintergrundinformation ... 183
Schriftartenfamilien, Definieren ... 186

Schriftfamilien ... 173
Schriftgröße ... 172
Schriftgröße (Notation) ändern ... 152
Schriftgröße (Notation), Standard ... 153
Schriftschnitt verändern ... 172
Schriftschnitte ... 173
`Score` ... 83, 407, 409
`Score is a (single) compound musical expression` ... 304
`Scores and parts` ... 315
`scoreTitleMarkup` ... 310
`scoreTweak` ... 340, 432
`Script` ... 85
Seconda volta ... 100
Segno ... 76, 83, 404
Segno an Taktlinie ... 165
Seitenlayout ... 311
Seitenumbruch, erzwingen ... 308
Semai-Form ... 300
separater Text ... 169
Septakkorde ... 261
sesqui-B ... 7
sesqui-Kreuz ... 7
`set-accidental-style` ... 19
`set-octavation` ... 17
setting extent of text objects ... 403
setting horizontal text alignment ... 379
`Setting simple songs` ... 187
setting subscript in standard font size ... 370
setting superscript in standard font size ... 370
Setzen von Text ... 170
`sf` ... 85
`sff` ... 85
`sfz` ... 85
`shiftDurations` ... 340, 432
`shiftOff` ... 114
`shiftOn` ... 114
`shiftOnn` ... 114
`shiftOnnn` ... 114
short-indent ... 143
`show-available-fonts` ... 185
`showFirstLength` ... 322
`showKeySignature` ... 258
`showLastLength` ... 322
`showStaffSwitch` ... 208
Silben spreizen ... 197
simple text strings ... 372
simple text strings with tie characters ... 396
simultane Noten und Versetzungszeichen ... 25
`Size of objects` ... 137
Skalieren von Dauern ... 35
Skip ... 40
`SkipMusic` ... 41
`skipTypesetting` ... 322
`Skordatur` ... 16
slashed digits ... 401
Slide in Tabulaturen ... 222
`Slur` ... 92
`slurDashed` ... 91
`slurDotted` ... 91
`slurDown` ... 91
`slurNeutral` ... 91
`slurSolid` ... 91
`slurUp` ... 91
`small` ... 152
solo-Stellen ... 118
Sonderzeichen in Textbeschriftungen ... 171
Sopranschlüssel ... 12
Sopranschlüssel in C ... 12
sos. ... 210
sostenuto-Pedal ... 210
`SostenutoEvent` ... 211
`sostenutoOff` ... 210
`sostenutoOn` ... 210
`SostenutoPedal` ... 211
`SostenutoPedalLineSpanner` ... 211
`sp` ... 85
`spacing-spanner-interface` ... 427
`spacingTweaks` ... 340, 432
`SpanBar` ... 71
spitze Klammern ... 109
`spp` ... 85
Sprache, Tonhöhenbezeichnungn in anderer ... 7
Sprechgesang ... 188
Spreizen von Silben ... 197
Springen zwischen Systemen ... 206
Staccatissimo ... 83, 404
Staccato ... 83
`Staccato` ... 85
Staccato ... 404
stacking text in a column ... 376
`Staff` ... 25, 27, 51, 71, 125
`Staff` ... 129
`Staff` ... 140, 146, 162, 286, 407
`staff-padding` ... 205
`staff-symbol-interface` ... 132, 133
`Staff.midiInstrument` ... 323
`Staff_symbol_engraver` ... 140
`StaffGroup` ... 129
`StaffGroup` ... 130
`StaffSymbol` ... 125, 133, 137
Standard-Schriftgröße (Notation) ... 153
Standard-Versetzungszeichenstil ... 19, 20
Standardnotenbezeichnungen ... 4
`StanzaNumber` ... 204
`start-repeat` ... 103
`startGroup` ... 162
`startTrillSpan` ... 98
`Stem` ... 158, 209, 210, 283
`stem-interface` ... 158
`Stem_engraver` ... 158
`stemDown` ... 158
`stemLeftBeamCount` ... 66
`stemNeutral` ... 158
`stemRightBeamCount` ... 66
`stemUp` ... 158
Stichnoten ... 146, 149
Stichnoten, Formatierung ... 149
Stil von Legatobögen ... 91
Stil von Taktangaben ... 46
Stil von Übungszeichen ... 76
Stile, Notenköpfe ... 27, 366
Stile, Stimmen ... 114
Stimme ... 111
`Stimme` ... 121
Stimme folgen ... 208
Stimme-Versetzungszeichenstil ... 21
Stimmen kombinieren ... 118
Stimmen verschieben ... 114
Stimmen, farbige Unterscheidung ... 114

Stimmen, mehrere ... 114
Stimmen, Stile ... 114
Stimmen, Versetzungszeichen für ... 22
Stimmen, Versetzungszeichenstil mit Warnung für Stimmen ... 22
Stimmen, zitieren ... 146
Stimmfolgestriche ... 208
Stimmgruppe ... 125
Stimmkreuzung ... 208
Stimmumfang ... 25
Stimmung, Banjo ... 247
`stopGroup` ... 162
`stopTrillSpan` ... 98
`storePredefinedDiagram` ... 237
`storePredefinedDiagram` ... 340, 432
Strecker, Text ... 164
Streicher ... 216
Streicher, Bogenanzeige ... 217
Striche zur Stimmverfolgung ... 208
Striche: Notenköpfe ... 30
Strichnotenköpfe ... 30
`String quartet` ... 217
`StringNumber` ... 222
`stringTunings` ... 234
`StringTunings` ... 225
`StrokeFinger` ... 246
Strophennummer ... 199
Struktur, Datei ... 305
Subbassschlüssel ... 12
`subdivideBeams` ... 60
subscript text ... 373
`subsubtitle` ... 307
`subtitle` ... 307
Subtraktion in Akkorden ... 263
`suggestAccidentals` ... 296
superscript text ... 374
`sus` ... 264
`SustainEvent` ... 211
`sustainOff` ... 210
`sustainOn` ... 210
`SustainPedal` ... 211
`SustainPedalLineSpanner` ... 211
Symbole auf der Taktlinie ... 165
Symbole, Akkord- ... 266
Symbole, Akkordeon ... 211
Symbole, nicht Musik- ... 179
`System` ... 125, 133, 137
System querende Hälse ... 209
System, beenden ... 132
System, Chor ... 125
System, geschachtelt ... 129
System, Größe verändern ... 133
SystemBeginnBegrenzer, geschachtelt ... 129
`Systeme` ... 125
Systeme verstecken ... 137
Systeme, leere ... 137
Systeme, mehrere ... 125
Systeme, Tremolo zwischen ... 109
Systemgruppe ... 125
Systemgruppen, Verschachtelung ... 129
`SystemStartBar` ... 129
`SystemStartBar` ... 130
`SystemStartBrace` ... 129
`SystemStartBrace` ... 130
`SystemStartBracket` ... 129
`SystemStartBracket` ... 130
`SystemStartSquare` ... 129
`SystemStartSquare` ... 130
Systemwechsel von Stimmen ... 208
Systemwechsel, automatisch ... 206
Systemwechsel, manuell ... 206

T

`Tab_note_heads_engraver` ... 226
`TabNoteHead` ... 224
`TabStaff` ... 125, 222, 224
Tabulatur ... 124, 220
Tabulatur und Flageolett ... 222
Tabulatur, Banjo ... 225, 247
Tabulatur, Bassgitarre ... 225
Tabulatur, Grundlegendes ... 222
Tabulatur, Mandoline ... 225
Tabulatur, Saitenstimmung ... 225
Tabulaturen und Gleiten ... 222
Tabulaturen, eigen ... 225
Tabulatursystem ... 124
`TabVoice` ... 222, 224
`tag` ... 317
`tag` ... 340, 432
Tag ... 317
`tagline` ... 308
Tagline ... 310
Takt, Noten ohne ... 82
Taktangabe ... 45
`Taktangabe` ... 47
Taktangabe, Sichtbarkeit ... 45
Taktangaben-Stile ... 46
Taktart, Alte Musik ... 284
Taktart, Noten ohne ... 48
Taktarten, arabisch ... 300
Taktarten, polymetrisch ... 49
Takte verkürzen ... 47
Taktlänge ändern ... 47
Taktlinen, ausschalten ... 48
Taktlinie, Symbole anfügen ... 165
Taktlinie, Wiederholung ... 103
Taktlinien ... 69
Taktlinien, Ausrichtung ... 73
Taktlinien, unsichtbar ... 69
Taktnummer ... 82
Taktnummer, Form ... 72
Taktnummern ... 71
Taktnummern, ausschalten ... 48
Taktnummern, regelmäßiger Abstand ... 72
Taktposition und Wiederholung ... 103
Taktüberprüfung ... 74
Taktweise Wiederholungen ... 106
`taor` ... 258
taqasim ... 300
Tasteninstrumente, Notensystem ... 205
Tasteninstrumente, zentrierte Dynamik ... 205
`teaching` ... 23
teaching-Versetzungszeichenstil ... 23
`teeny` ... 152
Tempo ... 140
Tempobezeichnung ... 140
`Tempobezeichnung` ... 142
Tenorschlüssel ... 12

Tenorschlüssel, Chor ... 13
Tenuto ... 83
`Tenuto` ... 85
Tenuto ... 404
`text` ... 211
Text alleine ... 169
Text auf der Seite zentrieren ... 176
text columns, left-aligned ... 383
text columns, right-aligned ... 385
Text einrahmen ... 177
`Text formatieren` ... 424, 428
Text in Voltaklammer ... 104
Text mit Sonderzeichen ... 171
Text über Mehrtaktpausen ... 42
Text und Balken ... 57
Text verzieren ... 177
Text, andere Sprachen ... 163
Text, Ausrichtung ... 174
Text, Blocksatz ... 176
Text, horizontale Ausrichtung ... 174
Text, mehrere Zeilen ... 176
Text, Rand außen ... 178
Text, Syntax ... 170
Text, vertikale Ausrichtung ... 175
`text-interface` ... 401
Textarten ... 163
Textbeschriftung ... 170
Textbeschriftung, Sonderzeichen ... 171
Textbeschriftungs-Ausdrücke ... 170
Textblasen ... 159
Textblöcke ... 176
Textelemente, nicht leer ... 163
Textgröße ... 172
`TextScript` ... 85, 164, 170, 174, 177, 180, 182, 183
`TextSpanner` ... 165
`textSpannerDown` ... 165
`textSpannerNeutral` ... 165
`textSpannerUp` ... 165
Textstrecker ... 164
`thumb` ... 153
thumb-script ... 153
`Tie` ... 38
`TieColumn` ... 38
tiefergestellt ... 173
ties, placement ... 37
time signature, compound ... 47
`TimeScaledMusic` ... 35
`TimeSignature` ... 47, 51, 284
`Timing_translator` ... 47, 48, 51, 71, 83, 409
`tiny` ... 152
Titel ... 311
`title` ... 307
`tocItem` ... 340, 432
Tonart ... 4, 15
Tonhöhe: Wechsel der Oktave ... 1
Tonhöhen in MIDI ... 326
Tonhöhen, transponieren ... 9
Tonhöhenbezeichnungen ... 1
`Tonhöhenbezeichnungen` ... 2, 4, 6, 8
Tonhöhenbezeichnungen, andere Sprachen ... 7
`Tonumfang` ... 27
Transkription von Mensuralmusik ... 128
translating text ... 386
transparent, Noten ... 155
Transponieren ... 9
transponierende Instrumente ... 10
`Transponierende Instrumente` ... 19
transponierende Schlüssel ... 13
Transponierendes Instrument ... 17
`transpose` ... 4, 9, 12
`transposedCueDuring` ... 340, 432
`TransposedMusic` ... 12
`transposition` ... 17
`transposition` ... 340, 433
Transposition ... 9
Transposition und relativer Modus ... 4
Transposition von Bunddiagrammen ... 235
Transposition, Instrumente ... 17
Transposition, MIDI ... 17
tre corde ... 210
`treCorde` ... 210
`tremolo` ... 108
Tremolo ... 108
Tremolo über Systeme ... 109
Tremolobalken ... 108
`tremoloFlags` ... 108
Tremolozeichen ... 108
Trennstriche, Gesangstext ... 195
`trill` ... 98
Triller ... 83, 98
`Triller` ... 99
Triller ... 404
Triller mit Tonhöhe ... 99
Triller mit Tonhöhe und erzwungenem Versetzungszeichen ... 99
`TrillSpanner` ... 99
`Triole` ... 35
Triole, Formatierung ... 33
Triolen ... 32
`TupletBracket` ... 35
`TupletNumber` ... 33, 35
`tupletNumberFormatFunction` ... 33
`tupletSpannerDuration` ... 33
`tweak` ... 340, 433
`Tweaking methods` ... 35

U

U.C. ... 210
Überbindung ... 36
Überbindung in Wiederholung ... 101
Überbindung und Wiederholungen ... 37
Überbindung, Versetzungszeichen ... 5
Überbindungen und Akkorde ... 36
Überschriften ... 311
Überspringen von Zeichen ... 40
Übungszeichen ... 75
Übungszeichen formatieren ... 76
Übungszeichenstil ... 76
Übungszwecke, Notenköpfe ... 28
Umbruch von Text ... 176
Umgekehrungen ... 265
Umkehrungen ... 261
una corda ... 210
`unaCorda` ... 210
`UnaCordaEvent` ... 211
`UnaCordaPedal` ... 211
`UnaCordaPedalLineSpanner` ... 211
underlining text ... 375

`unfold` ... 106
`UnfoldedRepeatedMusic` ... 103, 106
`unfoldRepeats` ... 340, 433
`unHideNotes` ... 155
unsichtbare Noten ... 155
Unsichtbare Pausen ... 40
unsichtbare Taktlinien ... 69
unsichtbarer Notenhals ... 158
Unterteilte Taktarten ... 49

V

Varcoda ... 83, 404
Variablen ... 306
Variablen, Benutzung ... 315
Variablen, Gesangstext ... 191
Vaticana, Editio ... 279
`Vaticana_ligature_engraver` ... 280
`VaticanaStaff` ... 125
VaticanaStaffContext ... 295
VaticanaVoiceContext ... 295
Verändern von automatischer Bebalkung ... 57
veränderte Akkorde ... 263
`Verkleinerte Systeme` ... 137, 140
Verschachtelte Musik ... 121
verschachtelte Systemklammern ... 129
verschachtelte Wiederholung ... 103
Verschachtelung von Systemen ... 129
Verschieben von Noten ... 114
Verschieben von Pausen, automatisch ... 114
Verschmelzen von Noten ... 114
Verschwinden von leeren Systemen ... 137
Versetzungszeichen ... 4
Versetzungszeichen an übergebundener Note ... 5
Versetzungszeichen für Klavier ... 22
Versetzungszeichen in Akkorden ... 25
Versetzungszeichen pro Stimme ... 22
Versetzungszeichen und gleichzeitige Noten ... 25
Versetzungszeichen, automatisch ... 19
Versetzungszeichen, Deutsch ... 4
Versetzungszeichen, Erinnerung ... 5
Versetzungszeichen, erzwungen für Triller ... 99
Versetzungszeichen, moderne Stile ... 21
Versetzungszeichen, moderner Stil mit Warnungen ... 21
Versetzungszeichen, musica ficta ... 296
Versetzungszeichen, piano cautionary ... 23
Versetzungszeichen, Standard ... 19
Versetzungszeichen, Viertelton ... 6
Versetzungszeichen, Vierteltöne ... 5
Versetzungszeichen, Warnung ... 5
Versetzungszeichenstil ... 19
Versetzungszeichenstil forget ... 24
Versetzungszeichenstil Klavier mit Warnungen ... 23
Versetzungszeichenstil modern ... 21
Versetzungszeichenstil neo-modern mit Warnungen ... 23
Versetzungszeichenstil teaching ... 23
Versetzungszeichenstil Vergessen ... 24
Versetzungszeichenstil, modern ... 22
Versetzungszeichenstil, modern mit Warnung für Stimmen ... 22
Versetzungszeichenstil, modern-cautionary ... 21
Versetzungszeichenstil, neo-modern ... 23
Versetzungszeichenstil, no reset ... 24
Versetzungszeichenstil, piano ... 22
Versetzungszeichenstil, Standard ... 20
Versetzungszeichenstil, Stimme ... 21
Versetzungszeichenstil, Zwölftonmusik ... 23
Versetzungszeichenstil: nicht zurücksetzen ... 24
Verstecken von Noten ... 155
Verstecken von Rhythmus-Systemen ... 138
Verstecken von Systemen ... 137
Verstecken von Systemen der Alten Musik ... 138
versteckte Notensysteme ... 133
`VerticalAxisGroup` ... 140
vertically centering text ... 387
vertickale Ausrichtung von Text ... 175
vertikale Linien zwischen Systemen ... 160
vertikale Position von Dynamik ... 86
Verwaltung der Zeiteinheiten ... 82
Verzierung, danach ... 78
Verzierungen ... 77
`Verzierungen` ... 80
viele Stimmen ... 114
`Viertelton` ... 6
Vierteltöne ... 5
Vierteltöne in MIDI ... 326
Vierteltonversetzungszeichen ... 6
Violinschlüssel ... 12
`Vocal ensembles` ... 194
`voice` ... 19, 21
`Voice` ... 27, 30, 111, 121, 149, 151, 288
Voice-Stile ... 114
Voice-Versetzungszeichenstil ... 21
`VoiceFollower` ... 208
`voiceOne` ... 111
`Voices contain music` ... 114, 118
Volta ... 100
Volta und Überbindung ... 37
`Volta-Klammer` ... 103
Volta-Klammer mit Text ... 104
Volta-Klammern und Wiederholungen ... 37
`Volta_engraver` ... 268
`VoltaBracket` ... 103, 105
Voltaklammer, ändern ... 103
`VoltaRepeatedMusic` ... 103, 105
Vorhalt ... 77
`Vorhalt` ... 80
Vorlage, arabische Musik ... 301
Vorschlag ... 77
`Vorschlag` ... 80
Vorzeichen ... 15
Vorzeichen in Klammern ... 5
Vorzeichen, Alte Musik ... 280
Vorzeichen, Erinnerung ... 5
Vorzeichen, Vierteltöne ... 5

W

Warnungsversetzungszeichen für Klavier ... 23
Warnungsversetzungszeichen, neo-modern ... 23
Warnungsvorzeichen ... 5
Wechsel der Oktave ... 1
Wechsel des Systems, automatisch ... 206
Wechsel des Systems, manuell ... 206
Wechsel von Instrument ... 145
Wechsel zwischen Systemen ... 208
Wechseln von Instrumentenbezeichnungen ... 144

Weiße Mensuralligaturen ... 288
weit außeinander liegende Balken ... 56
`whichBar` ... 71
wiederholte Musik ... 106
Wiederholung mit alternativem Schluss ... 100
Wiederholung mit Auftakt ... 101
Wiederholung und Bindebögen ... 37
Wiederholung und Bindebogen ... 103
Wiederholung und Zählzeit ... 103
Wiederholung, alternative Schlüsse ... 103
Wiederholung, aufklappen ... 106
Wiederholung, Beginn ... 103
Wiederholung, Ende ... 103
Wiederholung, kurz ... 106
Wiederholung, manuell ... 103
Wiederholung, mehrdeutig ... 103
Wiederholung, Prozent ... 106
Wiederholung, taktweise ... 106
Wiederholung, Tremolo ... 108
Wiederholung, verschachtelt ... 103
Wiederholung, Voltaklammer ... 103
Wiederholungen ... 70, 100
`Wiederholungen` ... 103
Wiederholungen in MIDI ... 326
Wiederholungen mit Überbindung ... 101
Wiederholungen, ausgeschrieben ... 106
Wiederholungsklammer mit Text ... 104
Wiederholungstaktlinie ... 103
Wiederholungszeichen ... 69
wirkliche Tonhöhe ... 4
`with-color` ... 156
`withMusicProperty` ... 340, 433
`Working on input files` ... 304

X

`x11-color` ... 156, 157
x11-Farbe ... 157
X11-Farben ... 156

Z

Zahl der Notenlinien einstellen ... 131
Zahl von Saite ... 220
Zählzeit und Wiederholung ... 103
`Zäsur` ... 94
Zeichen ... 83
Zeichen, Übung: Formatierung ... 76
Zeichnen im Text ... 177
Zeilenumbruch, Balken ... 56
Zeilenumbrüche ... 69
Zeit (in der Partitur) ... 82
Zentrieren von Text auf der Seite ... 176
zentrierte Musik für Tasteninstrumente ... 205
Ziernoten ... 77
Zitieren von anderen Stimmen ... 146, 149
zitierter Text ... 163
Zusammenfalten von Pausen ... 45
Zusammenstöße ... 114
zweite Klammer ... 100
Zwischensystem-Tremolo ... 109
Zwischensysteme-Klammer-Arpeggio ... 98
Zwölftonmusik, Versetzungszeichenstil ... 23