# Table of Contents

1 Music definitions ......................................................... 2
  1.1 Music expressions .................................................. 2
    1.1.1 AbsoluteDynamicEvent ........................................ 2
    1.1.2 AnnotateOutputEvent ......................................... 2
    1.1.3 ApplyContext .................................................. 2
    1.1.4 ArpeggioEvent ................................................ 3
    1.1.5 ArticulationEvent ............................................. 3
    1.1.6 AutoChangeMusic ............................................... 4
    1.1.7 BarCheck ...................................................... 4
    1.1.8 BassFigureEvent .............................................. 5
    1.1.9 BeamEvent .................................................... 5
    1.1.10 BeamChangeEvent ............................................. 5
    1.1.11 BeamForbidEvent ............................................ 6
    1.1.12 BendAfterEvent .............................................. 6
    1.1.13 BreathingEvent ............................................... 6
    1.1.14 ClusterNoteEvent ............................................ 6
    1.1.15 ContextChangeEvent ......................................... 7
    1.1.16 ContextSpeccedMusic ....................................... 7
    1.1.17 CrescendoEvent .............................................. 7
    1.1.18 DecrescendoEvent .......................................... 8
    1.1.19 Event ........................................................ 8
    1.1.20 EventChord ................................................... 8
    1.1.21 ExtenderEvent ............................................... 9
    1.1.22 FingeringEvent .............................................. 9
    1.1.23 GlissandoEvent .............................................. 10
    1.1.24 GraceMusic .................................................. 10
    1.1.25 HarmonicEvent ............................................. 10
    1.1.26 HyphenEvent ................................................. 11
    1.1.27 KeyChangeEvent ............................................. 11
    1.1.28 LabelEvent .................................................. 11
    1.1.29 LaissezVibrerEvent ....................................... 12
    1.1.30 LigatureEvent ................................................ 12
    1.1.31 LineBreakEvent ............................................. 13
    1.1.32 LyricCombineMusic ......................................... 13
    1.1.33 LyricEvent .................................................. 13
    1.1.34 MarkEvent ................................................... 14
    1.1.35 MultiMeasureRestEvent .................................... 14
    1.1.36 MultiMeasureRestMusic .................................... 14
    1.1.37 MultiMeasureTextEvent .................................... 15
    1.1.38 Music ......................................................... 15
    1.1.39 NoteEvent ................................................... 15
    1.1.40 NoteGroupingEvent ........................................ 16
    1.1.41 OverrideProperty .......................................... 16
    1.1.42 PageBreakEvent ............................................. 16
    1.1.43 PageTurnEvent .............................................. 17
    1.1.44 PartCombineMusic .......................................... 17
    1.1.45 PercentEvent ............................................... 18
    1.1.46 PercentRepeatedMusic ...................................... 18
1.2 Music classes
   1.2.1 StreamEvent .......................................... 34
   1.2.2 absolute-dynamic-event .............................. 35
   1.2.3 annotate-output-event ................................ 35
   1.2.4 apply-output-event ................................... 35
   1.2.5 arpeggio-event ....................................... 35
   1.2.6 articulation-event ................................... 36
   1.2.7 bass-figure-event .................................... 36
   1.2.8 beam-event ............................................ 36
   1.2.9 beam-forbid-event .................................... 36
   1.2.10 bend-after-event .................................... 36
   1.2.11 break-event ......................................... 36
1.2.12 breathing-event .................................................. 36
1.2.13 cluster-note-event ............................................... 36
1.2.14 crescendo-event .................................................. 36
1.2.15 decrescendo-event ............................................... 36
1.2.16 dynamic-event .................................................... 37
1.2.17 extender-event ................................................... 37
1.2.18 fingering-event ................................................... 37
1.2.19 glissando-event ................................................... 37
1.2.20 harmonic-event ................................................... 37
1.2.21 hyphen-event ...................................................... 37
1.2.22 key-change-event ............................................... 37
1.2.23 label-event ....................................................... 37
1.2.24 laissez-vibier-event ............................................. 37
1.2.25 layout-instruction-event ...................................... 37
1.2.26 ligature-event .................................................... 38
1.2.27 line-break-event .................................................. 38
1.2.28 lyric-event ........................................................ 38
1.2.29 mark-event ....................................................... 38
1.2.30 melodic-event ...................................................... 38
1.2.31 multi-measure-rest-event ..................................... 38
1.2.32 multi-measure-text-event ..................................... 38
1.2.33 music-event ........................................................ 38
1.2.34 note-event ........................................................ 39
1.2.35 note-grouping-event ............................................ 39
1.2.36 page-break-event .................................................. 39
1.2.37 page-turn-event .................................................... 39
1.2.38 part-combine-event ............................................. 39
1.2.39 pedal-event ........................................................ 39
1.2.40 percent-event ...................................................... 40
1.2.41 pes-or-flexa-event .............................................. 40
1.2.42 phrasing-slur-event .............................................. 40
1.2.43 repeat-tie-event ................................................... 40
1.2.44 rest-event .......................................................... 40
1.2.45 rhythmic-event .................................................... 40
1.2.46 script-event ........................................................ 40
1.2.47 skip-event .......................................................... 40
1.2.48 slur-event .......................................................... 40
1.2.49 solo-one-event ..................................................... 41
1.2.50 solo-two-event ..................................................... 41
1.2.51 sostenuto-event ................................................... 41
1.2.52 spacing-section-event ......................................... 41
1.2.53 span-dynamic-event ............................................ 41
1.2.54 span-event ........................................................ 41
1.2.55 staff-span-event .................................................. 41
1.2.56 string-number-event .......................................... 41
1.2.57 stroke-finger-event .............................................. 42
1.2.58 sustain-event ..................................................... 42
1.2.59 text-script-event ................................................. 42
1.2.60 text-span-event ................................................... 42
1.2.61 tie-event ............................................................ 42
1.2.62 tremolo-event ...................................................... 42
1.2.63 tremolo-span-event .............................................. 42
1.2.64 trill-span-event .................................................... 42
1.2.65 tuplet-span-event ................................................. 42
2 Translation ......................................................... 48
  2.1 Contexts ....................................................... 48
    2.1.1 ChoirStaff ............................................. 48
    2.1.2 ChordNames ............................................ 48
    2.1.3 CueVoice .............................................. 50
    2.1.4 Devnull ................................................. 62
    2.1.5 DrumStaff .............................................. 62
    2.1.6 DrumVoice ............................................. 67
    2.1.7 FiguredBass ........................................... 78
    2.1.8 FretBoards ............................................ 80
    2.1.9 Global ................................................. 82
    2.1.10 GrandStaff ........................................... 82
    2.1.11 GregorianTranscriptionStaff ......................... 83
    2.1.12 GregorianTranscriptionVoice ......................... 92
    2.1.13 Lyrics ................................................ 104
    2.1.14 MensuralStaff ....................................... 106
    2.1.15 MensuralVoice ..................................... 116
    2.1.16 NoteNames .......................................... 127
    2.1.17 PianoStaff .......................................... 129
    2.1.18 RhythmicStaff ....................................... 130
    2.1.19 Score ................................................. 133
    2.1.20 Staff ................................................. 145
    2.1.21 StaffGroup .......................................... 154
    2.1.22 TabStaff ............................................. 155
    2.1.23 TabVoice .............................................. 161
    2.1.24 VaticanaStaff ...................................... 173
    2.1.25 VaticanaVoice ...................................... 182
    2.1.26 Voice ............................................... 194
  2.2 Engravers and Performers .................................. 205
    2.2.1 Accidental engraver .................................. 206
    2.2.2 Ambitus engraver ..................................... 207
    2.2.3 Arpeggio engraver .................................... 207
    2.2.4 Auto beam engraver ................................... 207
    2.2.5 Axis group engraver .................................. 208
    2.2.6 Balloon engraver ..................................... 208
    2.2.7 Bar engraver .......................................... 208
    2.2.8 Bar number engraver ................................ 209
    2.2.9 Beam engraver ........................................ 209
    2.2.10 Beam performer ....................................... 210
    2.2.11 Bend engraver ........................................ 210
    2.2.12 Break align engraver ................................ 210
    2.2.13 Breathing sign engraver .............................. 210
    2.2.14 Chord name engraver ................................ 211
    2.2.15 Chord tremolo engraver .............................. 211
    2.2.16 Clef engraver ........................................ 212
    2.2.17 Cluster spanner engraver ............................ 212
    2.2.18 Collision engraver ................................... 212
    2.2.19 Completion heads engraver ......................... 213
    2.2.20 Control track performer ............................. 213
    2.2.21 Custos engraver ...................................... 213
2.2.22 Default_bar_line_engraver .................................................. 213
2.2.23 Dot_column_engraver ....................................................... 214
2.2.24 Dots_engraver ............................................................... 214
2.2.25 Drum_note_performer .................................................... 215
2.2.26 Drum_notes_engraver ..................................................... 215
2.2.27 Dynamic_align_engraver ................................................ 215
2.2.28 Dynamic_engraver ....................................................... 215
2.2.29 Dynamic_performer ..................................................... 216
2.2.30 Engraver ................................................................. 216
2.2.31 Extender_engraver ...................................................... 216
2.2.32 Figured_bass_engraver ................................................ 216
2.2.33 Figured_bass_position_engraver ..................................... 217
2.2.34 Fingering_engraver ..................................................... 217
2.2.35 Font_size_engraver ...................................................... 217
2.2.36 Forbid_line_break_engraver ......................................... 218
2.2.37 Fretboard_engraver ..................................................... 218
2.2.38 Glissando_engraver ..................................................... 219
2.2.39 Grace_beam_engraver ................................................ 219
2.2.40 Grace_engraver .......................................................... 219
2.2.41 Grace_spacing_engraver ............................................... 220
2.2.42 Grid_line_span_engraver ............................................... 220
2.2.43 Grid_point_engraver .................................................... 220
2.2.44 Grob_pq_engraver ...................................................... 220
2.2.45 Hara_kiri_engraver .................................................... 221
2.2.46 Horizontal_bracket_engraver ....................................... 221
2.2.47 Hyphen_engraver ........................................................ 221
2.2.48 Instrument_name_engraver ........................................... 221
2.2.49 Instrument_switch_engraver ........................................ 222
2.2.50 Key_engraver ............................................................. 222
2.2.51 Key_performer ........................................................... 223
2.2.52 Laissez_vibrier_engraver ............................................ 223
2.2.53 Ledger_line_engraver .................................................. 223
2.2.54 Ligature_bracket_engraver ........................................... 224
2.2.55 Lyric_engraver ........................................................... 224
2.2.56 Lyric_performer ........................................................ 224
2.2.57 Mark_engraver ........................................................... 224
2.2.58 Measure_grouping_engraver ....................................... 225
2.2.59 Melody_engraver ........................................................ 225
2.2.60 Mensural_ligature_engraver ........................................ 225
2.2.61 Metronome_mark_engraver ........................................... 225
2.2.62 Multi_measure_rest_engraver ....................................... 226
2.2.63 New_dynamic_engraver ................................................ 227
2.2.64 New_fingering_engraver .............................................. 227
2.2.65 Note_head_line_engraver ............................................. 228
2.2.66 Note_heads_engraver .................................................. 228
2.2.67 Note_name_engraver ................................................... 228
2.2.68 Note_performer .......................................................... 229
2.2.69 Note_spacing_engraver ................................................. 229
2.2.70 Note_swallow_translator .............................................. 229
2.2.71 Ottava_spanner_engraver ............................................. 229
2.2.72 Output_property_engraver ............................................ 229
2.2.73 Page_turn_engraver .................................................... 230
2.2.74 Paper_column_engraver ............................................... 230
2.2.75 Parenthesis_engraver .................................................. 230
3 Backend .................................................. 257

3.1 All layout objects ........................................ 257
  3.1.1 Accidental ........................................ 257
  3.1.2 AccidentalCautionary ................................ 257
  3.1.3 AccidentalPlacement ................................ 258
  3.1.4 AccidentalSuggestion ................................ 259
  3.1.5 Ambitus ........................................... 260
  3.1.6 AmbitusAccidental ................................ 261
  3.1.7 AmbitusLine ....................................... 262
  3.1.8 AmbitusNoteHead ................................ 262
  3.1.9 Arpeggio ......................................... 263
  3.1.10 BalloonTextItem ................................ 264
  3.1.11 BarLine .......................................... 264
  3.1.12 BarNumber ....................................... 266
  3.1.13 BassFigure ...................................... 267
  3.1.14 BassFigureAlignment ................................ 268
  3.1.15 BassFigureAlignmentPositioning .................. 268
  3.1.16 BassFigureBracket ................................ 269
  3.1.17 BassFigureContinuation .......................... 269
  3.1.18 BassFigureLine .................................. 270
  3.1.19 Beam ............................................ 270
  3.1.20 BendAfter ....................................... 271
  3.1.21 BreakAlignGroup ................................ 272
  3.1.22 BreakAlignment .................................. 272
  3.1.23 BreathingSign ................................... 273
  3.1.24 ChordName ...................................... 274
  3.1.25 Clef ............................................. 275
  3.1.26 ClusterSpanner .................................. 276
  3.1.27 ClusterSpannerBeacon ............................ 276
  3.1.28 CombineTextScript ................................ 277
  3.1.29 Custos .......................................... 278
  3.1.30 DotColumn ...................................... 279
  3.1.31 Dots ............................................ 279
  3.1.32 DoublePercentRepeat ............................ 280
  3.1.33 DoublePercentRepeatCounter ..................... 281
  3.1.34 DynamicLineSpanner ............................. 282
  3.1.35 DynamicText ...................................... 283
  3.1.36 DynamicTextSpanner ............................. 284
  3.1.37 Fingering ........................................ 285
  3.1.38 FretBoard ....................................... 287
  3.1.39 Glissando ........................................ 288
  3.1.40 GraceSpacing .................................... 289
  3.1.41 GridLine .......................................... 289
  3.1.42 GridPoint ........................................ 290
  3.1.43 Hairpin ........................................... 290
  3.1.44 HarmonicParenthesesItem ......................... 291
  3.1.45 HorizontalBracket ................................ 292
  3.1.46 InstrumentName .................................. 293
  3.1.47 InstrumentSwitch ................................ 294
  3.1.48 KeyCancellation .................................. 295
3.2 Graphical Object Interfaces

3.2.1 accidental-interface ........................................... 358
3.2.2 accidental-placement-interface ............................... 359
3.2.3 accidental-suggestion-interface .............................. 360
3.2.4 align-interface .................................................. 360
3.2.5 ambitus-interface ............................................... 360
3.2.6 arpeggio-interface .............................................. 361
3.2.7 axis-group-interface .......................................... 361
3.2.8 balloon-interface ................................------------ 362
3.2.9 bar-line-interface ............................................. 362
3.2.10 bass-figure-alignment-interface ............................. 363
3.2.11 bass-figure-interface ........................................ 363
3.2.12 beam-interface ............................................... 363
3.2.13 bend-after-interface .......................................... 365
3.2.14 break-alignable-interface .................................... 366
3.2.15 break-aligned-interface ...................................... 366
3.2.16 break-alignment-interface .................................. 367
3.2.17 breathing-sign-interface .................................... 367
3.2.18 chord-name-interface ........................................ 367
3.2.19 clef-interface ................................................ 368
3.2.20 cluster-beacon-interface ..................................... 368
3.2.21 cluster-interface ............................................. 368
3.2.22 custos-interface ............................................. 369
3.2.23 dot-column-interface ........................................ 369
3.2.24 dots-interface ................................................. 369
3.2.25 dynamic-interface ............................................ 370
3.2.26 dynamic-line-spanner-interface ............................ 370
3.2.27 dynamic-text-spanner-interface ............................. 370
3.1.103 SustainPedalLineSpanner .................................... 335
3.1.104 System .......................................................... 337
3.1.105 SystemStartBar .............................................. 337
3.1.106 SystemStartBrace ............................................ 338
3.1.107 SystemStartBracket ......................................... 339
3.1.108 SystemStartSquare ........................................... 340
3.1.109 TabNoteHead .................................................. 340
3.1.110 TextScript .................................................... 341
3.1.111 TextSpanner .................................................. 343
3.1.112 Tie ............................................................ 344
3.1.113 TieColumn .................................................... 345
3.1.114 TimeSignature ................................................ 345
3.1.115 TrillPitchAccidental ........................................ 346
3.1.116 TrillPitchGroup ............................................. 347
3.1.117 TrillPitchHead .............................................. 348
3.1.118 TrillSpanner .................................................. 349
3.1.119 TupletBracket ............................................... 350
3.1.120 TupletNumber ............................................... 351
3.1.121 UnaCordaPedal .............................................. 352
3.1.122 UnaCordaPedalLineSpanner ................................. 353
3.1.123 VaticanaLigature .......................................... 354
3.1.124 VerticalAlignment ........................................... 354
3.1.125 VerticalAxisGroup .......................................... 355
3.1.126 VoiceFollower ................................................ 356
3.1.127 VoltaBracket ................................................ 356
3.1.128 VoltaBracketSpanner ....................................... 357
3.2.82 script-interface ........................................... 398
3.2.83 self-alignment-interface .................................. 398
3.2.84 semi-tie-column-interface .................................. 399
3.2.85 semi-tie-interface ........................................... 399
3.2.86 separation-item-interface ................................... 400
3.2.87 side-position-interface ..................................... 400
3.2.88 slur-interface ............................................... 402
3.2.89 spaceable-grob-interface ................................... 403
3.2.90 spacing-interface ............................................ 404
3.2.91 spacing-options-interface ................................... 404
3.2.92 spacing-spanner-interface ................................... 404
3.2.93 span-bar-interface .......................................... 405
3.2.94 spanner-interface ............................................ 405
3.2.95 staff-spacing-interface ...................................... 406
3.2.96 staff-symbol-interface ...................................... 406
3.2.97 staff-symbol-reference-interface .......................... 407
3.2.98 stanza-number-interface .................................... 407
3.2.99 stem-interface ................................................. 407
3.2.100 stem-tremolo-interface ..................................... 410
3.2.101 string-number-interface .................................... 410
3.2.102 stroke-finger-interface ..................................... 410
3.2.103 system-interface ............................................. 410
3.2.104 system-start-delimiter-interface ......................... 411
3.2.105 system-start-text-interface ................................ 412
3.2.106 tablature-interface ......................................... 412
3.2.107 text-interface ................................................ 412
3.2.108 text-script-interface ....................................... 413
3.2.109 tie-column-interface ....................................... 413
3.2.110 tie-interface ................................................ 414
3.2.111 time-signature-interface ................................... 415
3.2.112 trill-pitch-accidental-interface .......................... 415
3.2.113 trill-spanner-interface ..................................... 415
3.2.114 tuplet-bracket-interface ................................... 415
3.2.115 tuplet-number-interface .................................... 417
3.2.116 unbreakable-spanner-interface .............................. 417
3.2.117 vaticana-ligature-interface ................................ 417
3.2.118 vertically-spaceable-interface .............................. 418
3.2.119 volta-bracket-interface ..................................... 418
3.3 User backend properties ........................................ 419
3.4 Internal backend properties ..................................... 433

4 Scheme functions ................................................. 439

Appendix A Indices ................................................ 458
A.1 Concept index .................................................. 458
A.2 Function index .................................................. 458
This is the Internals Reference (IR) for version 2.12.3 of LilyPond, the GNU music typesetter.
Chapter 1: Music definitions

1 Music definitions

1.1 Music expressions

1.1.1 AbsoluteDynamicEvent

Create a dynamic mark.

Syntax: \note\x, where \x is a dynamic mark like \ppp or \sfz. A complete list is in file 'ly/dynamic-scripts-init.ly'.

Event classes: Section 1.2.2 [absolute-dynamic-event], page 35, Section 1.2.16 [dynamic-event], page 37, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.28 [Dynamic_engraver], page 215, Section 2.2.29 [Dynamic_performer], page 216 and Section 2.2.63 [New_dynamic_engraver], page 227.

Properties:

name (symbol):
'AbsoluteDynamicEvent
Name of this music object.

types (list):
'(general-music event dynamic-event absolute-dynamic-event)
The types of this music object; determines by what engraver this music expression is processed.

1.1.2 AnnotateOutputEvent

Print an annotation of an output element.

Event classes: Section 1.2.3 [annotate-output-event], page 35, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.6 [Balloon_engraver], page 208.

Properties:

name (symbol):
'AnnotateOutputEvent
Name of this music object.

types (list):
'(general-music event annotate-output-event)
The types of this music object; determines by what engraver this music expression is processed.

1.1.3 ApplyContext

Call the argument with the current context during interpreting phase.

Properties:

name (symbol):
'ApplyContext
Name of this music object.

types (list):
'(general-music apply-context)
The types of this music object; determines by what engraver this music expression is processed.
iterator-ctor (procedure):

\ly:apply-context-iterator::constructor

Function to construct a music-event-iterator object for this music.

### 1.1.4 ApplyOutputEvent

Call the argument with all current grobs during interpreting phase.

Syntax: `\applyOutput #'context func`

Arguments to `func` are 1. the grob, 2. the originating context, and 3. the context where `func` is called.

Event classes: Section 1.2.4 [apply-output-event], page 35, Section 1.2.25 [layout-instruction-event], page 37, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.72 [Output property engraver], page 229.

Properties:

- **name (symbol):**
  - `'ApplyOutputEvent

  Name of this music object.

- **types (list):**
  - '(general-music event apply-output-event)

  The types of this music object; determines by what engraver this music expression is processed.

### 1.1.5 ArpeggioEvent

Make an arpeggio on this note.

Syntax: `note-\arpeggio`

Event classes: Section 1.2.5 [arpeggio-event], page 35, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.3 [Arpeggio engraver], page 207.

Properties:

- **name (symbol):**
  - `'ArpeggioEvent

  Name of this music object.

- **types (list):**
  - '(general-music arpeggio-event event)

  The types of this music object; determines by what engraver this music expression is processed.

### 1.1.6 ArticulationEvent

Add an articulation marking to a note.

Syntax: `notexy`, where `x` is a direction (`^` for up or `_` for down), or LilyPond's choice (no direction specified), and where `y` is an articulation (such as `-`, `->`, `\tenuto`, `\downbow`). See the Notation Reference for details.

Event classes: Section 1.2.6 [articulation-event], page 36, Section 1.2.46 [script-event], page 40, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.91 [Script engraver], page 235.

Properties:
name (symbol):

'ArticulationEvent

Name of this music object.

types (list):

'(general-music event articulation-event script-event)

The types of this music object; determines by what engraver this music expression is processed.

1.1.7 AutoChangeMusic

Used for making voices that switch between piano staves automatically.

Properties:

name (symbol):

'AutoChangeMusic

Name of this music object.

iterator-ctor (procedure):

ly:auto-change-iterator::constructor

Function to construct a music-event-iterator object for this music.

start-callback (procedure):

ly:music-wrapper::start-callback

Function to compute the negative length of starting grace notes. This property can only be defined as initializer in 'scm/define-music-types.scm'.

length-callback (procedure):

ly:music-wrapper::length-callback

How to compute the duration of this music. This property can only be defined as initializer in 'scm/define-music-types.scm'.

types (list):

'(general-music music-wraper-music auto-change-instruction)

The types of this music object; determines by what engraver this music expression is processed.

1.1.8 BarCheck

Check whether this music coincides with the start of the measure.

Properties:

name (symbol):

'BarCheck

Name of this music object.

types (list):

'(general-music bar-check)

The types of this music object; determines by what engraver this music expression is processed.

iterator-ctor (procedure):

ly:bar-check-iterator::constructor

Function to construct a music-event-iterator object for this music.
1.1.9 BassFigureEvent

Print a bass-figure text.

Event classes: Section 1.2.7 [bass-figure-event], page 36, Section 1.2.45 [rhythmic-event], page 40, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.32 [Figured_bass_engraver], page 216.

Properties:

name (symbol):
'BassFigureEvent
Name of this music object.

types (list):
'(general-music event rhythmic-event bass-figure-event)
The types of this music object; determines by what engraver this music expression is processed.

1.1.10 BeamEvent

Start or stop a beam.

Syntax for manual control: c8- [ c c-] c8

Event classes: Section 1.2.8 [beam-event], page 36, Section 1.2.54 [span-event], page 41, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.9 [Beam_engraver], page 209, Section 2.2.10 [Beam_performer], page 210 and Section 2.2.39 [Grace_beam_engraver], page 219.

Properties:

name (symbol):
'BeamEvent
Name of this music object.

types (list):
'(general-music event beam-event span-event)
The types of this music object; determines by what engraver this music expression is processed.

1.1.11 BeamForbidEvent

Specify that a note may not auto-beamed.

Event classes: Section 1.2.9 [beam-forbid-event], page 36, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.4 [Auto_beam_engraver], page 207.

Properties:

name (symbol):
'BeamForbidEvent
Name of this music object.

types (list):
'(general-music event beam-forbid-event)
The types of this music object; determines by what engraver this music expression is processed.
1.1.12 BendAfterEvent

A drop/fall/doit jazz articulation.

Event classes: Section 1.2.10 [bend-after-event], page 36, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.11 [Bend engraver], page 210.

Properties:

name (symbol):
  "BendAfterEvent"
  Name of this music object.

types (list):
  '(general-music bend-after-event event)
  The types of this music object; determines by what engraver this music expression is processed.

1.1.13 BreathingEvent

Create a 'breath mark’ or ‘comma’.

Syntax: note\breathe

Event classes: Section 1.2.12 [breathing-event], page 36, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.13 [Breathing sign engraver], page 210.

Properties:

name (symbol):
  "BreathingEvent"
  Name of this music object.

types (list):
  '(general-music event breathing-event)
  The types of this music object; determines by what engraver this music expression is processed.

1.1.14 ClusterNoteEvent

A note that is part of a cluster.

Event classes: Section 1.2.13 [cluster-note-event], page 36, Section 1.2.30 [melodic-event], page 38, Section 1.2.45 [rhythmic-event], page 40, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.17 [Cluster spanner engraver], page 212.

Properties:

name (symbol):
  "ClusterNoteEvent"
  Name of this music object.

types (list):
  '(general-music cluster-note-event melodic-event rhythmic-event event)
  The types of this music object; determines by what engraver this music expression is processed.
### 1.1.15 ContextChange

Change staves in Piano staff.

**Syntax:** \change Staff = new-id

**Properties:**

- `name (symbol):`
  - 'ContextChange
  - Name of this music object.

- `iterator-ctor (procedure):`
  - ly:change-iterator::constructor
  - Function to construct a music-event-iterator object for this music.

- `types (list):`
  - '(general-music translator-change-instruction)
  - The types of this music object; determines by what engraver this music expression is processed.

### 1.1.16 ContextSpeccedMusic

Interpret the argument music within a specific context.

**Properties:**

- `name (symbol):`
  - 'ContextSpeccedMusic
  - Name of this music object.

- `iterator-ctor (procedure):`
  - ly:context-specced-music-iterator::constructor
  - Function to construct a music-event-iterator object for this music.

- `length-callback (procedure):`
  - ly:music-wrapper::length-callback
  - How to compute the duration of this music. This property can only be defined as initializer in `scm/define-music-types.scm`.

- `start-callback (procedure):`
  - ly:music-wrapper::start-callback
  - Function to compute the negative length of starting grace notes. This property can only be defined as initializer in `scm/define-music-types.scm`.

- `types (list):`
  - '(context-specification general-music music-wrapper-music)
  - The types of this music object; determines by what engraver this music expression is processed.

### 1.1.17 CrescendoEvent

Begin or end a crescendo.

**Syntax:** note\cr ... note\rc

You can also use \<, \!, \cresc, and \endcresc. See the Notation Reference for details.

**Event classes:** Section 1.2.14 [crescendo-event], page 36, Section 1.2.53 [span-dynamic-event], page 41, Section 1.2.54 [span-event], page 41, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.
Chapter 1: Music definitions

Accepted by: Section 2.2.28 [Dynamic engraver], page 215, Section 2.2.29 [Dynamic performer], page 216 and Section 2.2.63 [New dynamic engraver], page 227.

Properties:

name (symbol): 'CrescendoEvent
Name of this music object.

types (list):
'(general-music span-event span-dynamic-event crescendo-event event)
The types of this music object; determines by what engraver this music expression is processed.

1.1.18 DecrescendoEvent
See Section 1.1.17 [CrescendoEvent], page 7.

Event classes: Section 1.2.15 [decrescendo-event], page 36, Section 1.2.53 [span-dynamic-event], page 41, Section 1.2.54 [span-event], page 41, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.28 [Dynamic engraver], page 215, Section 2.2.29 [Dynamic performer], page 216 and Section 2.2.63 [New dynamic engraver], page 227.

Properties:

name (symbol): 'DecrescendoEvent
Name of this music object.

types (list):
'(general-music span-event span-dynamic-event decrescendo-event event)
The types of this music object; determines by what engraver this music expression is processed.

1.1.19 Event
Atomic music event.

Properties:

name (symbol): 'Event
Name of this music object.

types (list):
'(general-music event)
The types of this music object; determines by what engraver this music expression is processed.

1.1.20 EventChord
Internally used to group a set of events.

Properties:

name (symbol): 'EventChord
Name of this music object.
iterator-ctor (procedure):
   ly:event-chord-iterator::constructor
   Function to construct a music-event-iterator object for this music.

length-callback (procedure):
   ly:music-sequence::maximum-length-callback
   How to compute the duration of this music. This property can only be
defined as initializer in 'scm/define-music-types.scm'.

to-relative-callback (procedure):
   ly:music-sequence::event-chord-relative-callback
   How to transform a piece of music to relative pitches.

types (list):
   '(general-music event-chord simultaneous-music)
   The types of this music object; determines by what engraver this music
   expression is processed.

1.1.21 ExtenderEvent

Extend lyrics.

   Event classes: Section 1.2.17 [extender-event], page 37, Section 1.2.33 [music-event], page 38
   and Section 1.2.1 [StreamEvent], page 34.

   Accepted by: Section 2.2.31 [Extender engraver], page 216.

   Properties:

   name (symbol):
      'ExtenderEvent
      Name of this music object.

   types (list):
      '(general-music extender-event event)
      The types of this music object; determines by what engraver this music
      expression is processed.

1.1.22 FingeringEvent

Specify what finger to use for this note.

   Event classes: Section 1.2.18 [fingering-event], page 37, Section 1.2.33 [music-event], page 38
   and Section 1.2.1 [StreamEvent], page 34.

   Accepted by: Section 2.2.34 [Fingering engraver], page 217.

   Properties:

   name (symbol):
      'FingeringEvent
      Name of this music object.

   types (list):
      '(general-music fingering-event event)
      The types of this music object; determines by what engraver this music
      expression is processed.
1.1.23 GlissandoEvent

Start a glissando on this note.

Event classes: Section 1.2.19 [glissando-event], page 37, Section 1.2.33 [music-event], page 38
and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.38 [Glissando engraver], page 219.

Properties:

- **name** (symbol):
  - 'GlissandoEvent
    - Name of this music object.

- **types** (list):
  - '(general-music glissando-event event)
    - The types of this music object; determines by what engraver this music expression is processed.

1.1.24 GraceMusic

Interpret the argument as grace notes.

Properties:

- **name** (symbol):
  - 'GraceMusic
    - Name of this music object.

- **start-callback** (procedure):
  - ly:grace-music::start-callback
    - Function to compute the negative length of starting grace notes. This property can only be defined as initializer in 'scm/define-music-types.scm'.

- **length** (moment):
  - #'<Mom 0>
    - The duration of this music.

- **iterator-ctor** (procedure):
  - ly:grace-iterator::constructor
    - Function to construct a music-event-iterator object for this music.

- **types** (list):
  - '(grace-music music-wrapper-music general-music)
    - The types of this music object; determines by what engraver this music expression is processed.

1.1.25 HarmonicEvent

Mark a note as harmonic.

Event classes: Section 1.2.20 [harmonic-event], page 37, Section 1.2.33 [music-event], page 38
and Section 1.2.1 [StreamEvent], page 34.

Not accepted by any engraver or performer.

Properties:

- **name** (symbol):
  - 'HarmonicEvent
    - Name of this music object.
Chapter 1: Music definitions

1.1.26 HyphenEvent
A hyphen between lyric syllables.

Event classes: Section 1.2.21 [hyphen-event], page 37, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.47 [Hyphen_engraver], page 221.

Properties:

name (symbol):
   'HyphenEvent
   Name of this music object.

types (list):
   '(general-music hyphen-event event)
   The types of this music object; determines by what engraver this music expression is processed.

1.1.27 KeyChangeEvent
Change the key signature.

Syntax: \key name scale

Event classes: Section 1.2.22 [key-change-event], page 37, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.50 [Key_engraver], page 222 and Section 2.2.51 [Key_performer], page 223.

Properties:

name (symbol):
   'KeyChangeEvent
   Name of this music object.

to-relative-callback (procedure):
   #<procedure #f (x p)>
   How to transform a piece of music to relative pitches.

types (list):
   '(general-music key-change-event event)
   The types of this music object; determines by what engraver this music expression is processed.

1.1.28 LabelEvent
Place a bookmarking label.

Event classes: Section 1.2.23 [label-event], page 37, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.74 [Paper_column_engraver], page 230.

Properties:
name (symbol):  
'LabelEvent  
Name of this music object.

types (list):  
'$(general-music label-event event)$  
The types of this music object; determines by what engraver this music expression is processed.

1.1.29 LaissezVibrerEvent

Don’t damp this chord.

Syntax: note$laissezVibrer$

Event classes: Section 1.2.24 [laissez-vibrer-event], page 37, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.52 [Laissez_vibrer_engraver], page 223.

Properties:

name (symbol):  
'LaissezVibrerEvent  
Name of this music object.

types (list):  
'$(general-music event laissez-vibrer-event)$  
The types of this music object; determines by what engraver this music expression is processed.

1.1.30 LigatureEvent

Start or end a ligature.

Syntax: note$ligature$

Event classes: Section 1.2.26 [ligature-event], page 38, Section 1.2.54 [span-event], page 41, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.54 [Ligature_bracket_engraver], page 224, Section 2.2.60 [Mensural_ligature_engraver], page 225 and Section 2.2.127 [Vaticana_ligature_engraver], page 245.

Properties:

name (symbol):  
'LigatureEvent  
Name of this music object.

span-type (string):  
'ligature  
What kind of spanner should be created?  
TODO: Consider making type into symbol.

types (list):  
'$(general-music span-event ligature-event event)$  
The types of this music object; determines by what engraver this music expression is processed.
1.1.31 **LineBreakEvent**
Allow, forbid or force a line break.

Event classes: Section 1.2.27 [line-break-event], page 38, Section 1.2.11 [break-event], page 36, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.73 [Page_turn_engraver], page 230 and Section 2.2.74 [Paper_column_engraver], page 230.

Properties:

- **name (symbol):**
  
  `'LineBreakEvent`

  Name of this music object.

- **types (list):**

  `'(general-music line-break-event break-event event)`

  The types of this music object; determines by what engraver this music expression is processed.

1.1.32 **LyricCombineMusic**
Align lyrics to the start of notes.

Syntax: \lyricsto voicename lyrics

Properties:

- **name (symbol):**

  `'LyricCombineMusic`

  Name of this music object.

- **length (moment):**

  `#<Mom 0>`

  The duration of this music.

- **types (list):**

  `'(general-music lyric-combine-music)`

  The types of this music object; determines by what engraver this music expression is processed.

- **iterator-ctor (procedure):**

  `ly:lyric-combine-music-iterator::constructor`

  Function to construct a music-event-iterator object for this music.

1.1.33 **LyricEvent**
A lyric syllable. Must be entered in lyrics mode, i.e., \lyrics { twinkle4 twinkle4 }.

Event classes: Section 1.2.28 [lyric-event], page 38, Section 1.2.45 [rhythmic-event], page 40, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.55 [Lyric_engraver], page 224 and Section 2.2.56 [Lyric_performer], page 224.

Properties:

- **name (symbol):**

  `'LyricEvent`

  Name of this music object.
Chapter 1: Music definitions

1.1.34 MarkEvent
Insert a rehearsal mark.
Syntax: \mark marker
Example: \mark "A"
Event classes: Section 1.2.29 [mark-event], page 38, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.
Accepted by: Section 2.2.57 [Mark_engraver], page 224.
Properties:
  name (symbol):
    'MarkEvent
    Name of this music object.
  types (list):
    '(general-music mark-event event)
    The types of this music object; determines by what engraver this music expression is processed.

1.1.35 MultiMeasureRestEvent
Used internally by MultiMeasureRestMusic to signal rests.
Event classes: Section 1.2.31 [multi-measure-rest-event], page 38, Section 1.2.45 [rhythmic-event], page 40, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.
Accepted by: Section 2.2.62 [Multi_measure_rest_engraver], page 226.
Properties:
  name (symbol):
    'MultiMeasureRestEvent
    Name of this music object.
  types (list):
    '(general-music event rhythmic-event multi-measure-rest-event)
    The types of this music object; determines by what engraver this music expression is processed.

1.1.36 MultiMeasureRestMusic
Rests that may be compressed into Multi rests.
Syntax: R2.*4 for 4 measures in 3/4 time.
Properties:
  name (symbol):
    'MultiMeasureRestMusic
    Name of this music object.
  iterator-ctor (procedure):
    ly:sequential-iterator::constructor
    Function to construct a music-event-iterator object for this music.
elements-callback (procedure):
    mm-rest-child-list
    Return a list of children, for use by a sequential iterator. Takes a single
    music parameter.

types (list):
    '(general-music multi-measure-rest)
    The types of this music object; determines by what engraver this music
    expression is processed.

1.1.37 MultiMeasureTextEvent
Texts on multi measure rests.
Syntax: R-\markup { \roman "bla" }  
Note the explicit font switch.
Event classes: Section 1.2.32 [multi-measure-text-event], page 38, Section 1.2.33 [music-
    event], page 38 and Section 1.2.1 [StreamEvent], page 34.
Accepted by: Section 2.2.62 [Multi_measure_rest_engraver], page 226.
Properties:
    name (symbol):
        'MultiMeasureTextEvent
        Name of this music object.
    types (list):
        '(general-music event multi-measure-text-event)
        The types of this music object; determines by what engraver this music
        expression is processed.

1.1.38 Music
Generic type for music expressions.
Properties:
    name (symbol):
        'Music
        Name of this music object.
    types (list):
        '(general-music)
        The types of this music object; determines by what engraver this music
        expression is processed.

1.1.39 NoteEvent
A note.
Event classes: Section 1.2.34 [note-event], page 39, Section 1.2.30 [melodic-event], page 38,
    Section 1.2.45 [rhythmic-event], page 40, Section 1.2.33 [music-event], page 38 and Section 1.2.1
    [StreamEvent], page 34.
Accepted by: Section 2.2.14 [Chord_name_engraver], page 211, Section 2.2.19 [Comple-
    tion_heads_engraver], page 213, Section 2.2.25 [Drum_note_performer], page 215, Section 2.2.26
    [Drum_notes_engraver], page 215, Section 2.2.37 [Fretboard_engraver], page 218, Section 2.2.66
    [Note_heads_engraver], page 228, Section 2.2.67 [Note_name_engraver], page 228, Section 2.2.68
    [Note_performer], page 229 and Section 2.2.113 [Tab_note_heads_engraver], page 240.
Properties:
name (symbol):
  'NoteEvent
  Name of this music object.

types (list):
  '(general-music event note-event rhythmic-event melodic-event)
  The types of this music object; determines by what engraver this music expression is processed.

1.1.40 NoteGroupingEvent
Start or stop grouping brackets.

Event classes: Section 1.2.35 [note-grouping-event], page 39, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.46 [Horizontal_bracket_engraver], page 221.

Properties:
  name (symbol):
    'NoteGroupingEvent
    Name of this music object.

  types (list):
    '(general-music event note-grouping-event)
    The types of this music object; determines by what engraver this music expression is processed.

1.1.41 OverrideProperty
Extend the definition of a graphical object.

Syntax: \override [ context . ] object property = value

Properties:
  name (symbol):
    'OverrideProperty
    Name of this music object.

  types (list):
    '(general-music layout-instruction-event override-property-event)
    The types of this music object; determines by what engraver this music expression is processed.

  iterator-ctor (procedure):
    ly:push-property-iterator::constructor
    Function to construct a music-event-iterator object for this music.

1.1.42 PageBreakEvent
Allow, forbid or force a page break.

Event classes: Section 1.2.36 [page-break-event], page 39, Section 1.2.11 [break-event], page 36, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.73 [Page_turn_engraver], page 230 and Section 2.2.74 [Paper_column_engraver], page 230.

Properties:
name (symbol):
   'PageBreakEvent
   Name of this music object.

types (list):
   '(general-music break-event page-break-event event)
   The types of this music object; determines by what engraver this music expression is processed.

1.1.43 PageTurnEvent
Allow, forbid or force a page turn.

   Event classes: Section 1.2.37 [page-turn-event], page 39, Section 1.2.11 [break-event], page 36, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.
   Accepted by: Section 2.2.73 [Page_turn_engraver], page 230 and Section 2.2.74 [Paper_column_engraver], page 230.

   Properties:
   name (symbol):
      'PageTurnEvent
      Name of this music object.

types (list):
   '(general-music break-event page-turn-event event)
   The types of this music object; determines by what engraver this music expression is processed.

1.1.44 PartCombineMusic
Combine two parts on a staff, either merged or as separate voices.

   Properties:
   name (symbol):
      'PartCombineMusic
      Name of this music object.

length-callback (procedure):
   ly:music-sequence::maximum-length-callback
   How to compute the duration of this music. This property can only be defined as initializer in ‘scm/define-music-types.scm’.

start-callback (procedure):
   ly:music-sequence::minimum-start-callback
   Function to compute the negative length of starting grace notes. This property can only be defined as initializer in ‘scm/define-music-types.scm’.

types (list):
   '(general-music part-combine-music)
   The types of this music object; determines by what engraver this music expression is processed.

iterator ctor (procedure):
   ly:part-combine-iterator::constructor
   Function to construct a music-event-iterator object for this music.
1.1.45 **PercentEvent**

Used internally to signal percent repeats.

Event classes: Section 1.2.40 [percent-event], page 40, Section 1.2.45 [rhythmic-event], page 40, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.77 [Percent_repeat_engraver], page 231 and Section 2.2.95 [Slash_repeat_engraver], page 236.

Properties:

- **name** (symbol): `'PercentEvent`
  Name of this music object.

- **types** (list):
  `'(general-music event percent-event rhythmic-event)`
  The types of this music object; determines by what engraver this music expression is processed.

1.1.46 **PercentRepeatedMusic**

Repeats encoded by percents.

Properties:

- **name** (symbol): `'PercentRepeatedMusic`
  Name of this music object.

- **iterator-ctor** (procedure):
  `ly:percent-repeat-iterator::constructor`
  Function to construct a music-event-iterator object for this music.

- **start-callback** (procedure):
  `ly:repeated-music::first-start`
  Function to compute the negative length of starting grace notes. This property can only be defined as initializer in `scm/define-music-types.scm`.

- **length-callback** (procedure):
  `ly:repeated-music::unfolded-music-length`
  How to compute the duration of this music. This property can only be defined as initializer in `scm/define-music-types.scm`.

- **types** (list):
  `'(general-music repeated-music percent-repeated-music)`
  The types of this music object; determines by what engraver this music expression is processed.

1.1.47 **PesOrFlexaEvent**

Within a ligature, mark the previous and the following note to form a pes (if melody goes up) or a flexa (if melody goes down).

Event classes: Section 1.2.41 [pes-or-flexa-event], page 40, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.127 [Vaticana_ligature_engraver], page 245.

Properties:
name (symbol):
  'PesOrFlexaEvent
  Name of this music object.

types (list):
  '(general-music pes-or-flexa-event event)
  The types of this music object; determines by what engraver this music expression is processed.

1.1.48 PhrasingSlurEvent
Start or end phrasing slur.

Syntax:  note\( ( \text{and} \) note\)

Event classes:  Section 1.2.42 [phrasing-slur-event], page 40, Section 1.2.54 [span-event], page 41, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by:  Section 2.2.78 [Phrasing_slur_engraver], page 232.

Properties:
  name (symbol):
    'PhrasingSlurEvent
    Name of this music object.
  types (list):
    '(general-music span-event event phrasing-slur-event)
    The types of this music object; determines by what engraver this music expression is processed.

1.1.49 PropertySet
Set a context property.

Syntax:  \property context.prop = scheme-val

Properties:
  name (symbol):
    'PropertySet
    Name of this music object.
  types (list):
    '(layout-instruction-event general-music)
    The types of this music object; determines by what engraver this music expression is processed.
  iterator-ctor (procedure):
    ly:property-iterator::constructor
    Function to construct a music-event-iterator object for this music.

1.1.50 PropertyUnset
Remove the definition of a context \property.

Properties:
  name (symbol):
    'PropertyUnset
    Name of this music object.
Chapter 1: Music definitions

1.1.51 QuoteMusic
Quote preprocessed snippets of music.
Properties:

- **name** (symbol):
  `'QuoteMusic`
  Name of this music object.

- **iterator-ctor** (procedure):
  `ly:music-wrapper-iterator::constructor`
  Function to construct a music-event-iterator object for this music.

- **length-callback** (procedure):
  `ly:music-wrapper::length-callback`
  How to compute the duration of this music. This property can only be defined as initializer in `scm/define-music-types.scm`.

- **start-callback** (procedure):
  `ly:music-wrapper::start-callback`
  Function to compute the negative length of starting grace notes. This property can only be defined as initializer in `scm/define-music-types.scm`.

- **types** (list):
  `'(general-music music-wrapper-music)`
  The types of this music object; determines by what engraver this music expression is processed.

1.1.52 RelativeOctaveCheck
Check if a pitch is in the correct octave.
Properties:

- **name** (symbol):
  `'RelativeOctaveCheck`
  Name of this music object.

- **to-relative-callback** (procedure):
  `ly:relative-octave-check::relative-callback`
  How to transform a piece of music to relative pitches.

- **types** (list):
  `'(general-music relative-octave-check)`
  The types of this music object; determines by what engraver this music expression is processed.
1.1.53 RelativeOctaveMusic
Music that was entered in relative octave notation.

Properties:

- name (symbol):
  'RelativeOctaveMusic
  Name of this music object.
- to-relative-callback (procedure):
  ly:relative-octave-music::relative-callback
  How to transform a piece of music to relative pitches.
- iterator-ctor (procedure):
  ly:iterator::iterator::constructor
  Function to construct a music-event-iterator object for this music.
- length-callback (procedure):
  ly:iterator::length-callback
  How to compute the duration of this music. This property can only be defined as initializer in 'scm/define-music-types.scm'.
- start-callback (procedure):
  ly:iterator::start-callback
  Function to compute the negative length of starting grace notes. This property can only be defined as initializer in 'scm/define-music-types.scm'.
- types (list):
  '(general-music event repeat-tie-event)
  The types of this music object; determines by what engraver this music expression is processed.

1.1.54 RepeatTieEvent
Ties for starting a second volta bracket.

Event classes: Section 1.2.43 [repeat-tie-event], page 40, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.85 [Repeat_tie_engraver], page 234.

Properties:

- name (symbol):
  'RepeatTieEvent
  Name of this music object.
- types (list):
  '(general-music event repeat-tie-event)
  The types of this music object; determines by what engraver this music expression is processed.

1.1.55 RepeatedMusic
Repeat music in different ways.

Properties:
name (symbol):
  'RepeatedMusic
  Name of this music object.

types (list):
  '(general-music repeated-music)
  The types of this music object; determines by what engraver this music
  expression is processed.

1.1.56 RestEvent
A Rest.
Syntax: r4 for a quarter rest.
Event classes: Section 1.2.44 [rest-event], page 40, Section 1.2.45 [rhythmic-event], page 40,
  Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.
Accepted by: Section 2.2.32 [Figured_bass_engraver], page 216 and Section 2.2.87
  [Rest_engraver], page 234.
Properties:
  name (symbol):
    'RestEvent
    Name of this music object.

types (list):
    '(general-music event rhythmic-event rest-event)
    The types of this music object; determines by what engraver this music
    expression is processed.

1.1.57 RevertProperty
The opposite of Section 1.1.41 [OverrideProperty], page 16: remove a previously added property
  from a graphical object definition.
Properties:
  name (symbol):
    'RevertProperty
    Name of this music object.

types (list):
    '(general-music layout-instruction-event)
    The types of this music object; determines by what engraver this music
    expression is processed.

iterator-ctor (procedure):
  ly:pop-property-iterator::constructor
  Function to construct a music-event-iterator object for this music.

1.1.58 ScriptEvent
Add an articulation mark to a note.
Event classes: Section 1.2.46 [script-event], page 40, Section 1.2.33 [music-event], page 38
  and Section 1.2.1 [StreamEvent], page 34.
Not accepted by any engraver or performer.
Properties:
name (symbol):  
'ScriptEvent  
Name of this music object.

types (list):  
'(general-music event)  
The types of this music object; determines by what engraver this music expression is processed.

1.1.59 SequentialMusic  
Music expressions concatenated.

Syntax: \sequential { ... } or simply { ... }
Properties:

name (symbol):  
'SequentalMusic  
Name of this music object.

length-callback (procedure):
ly:music-sequence::cumulative-length-callback  
How to compute the duration of this music. This property can only be defined as initializer in 'scm/define-music-types.scm'.

start-callback (procedure):
ly:music-sequence::first-start-callback  
Function to compute the negative length of starting grace notes. This property can only be defined as initializer in 'scm/define-music-types.scm'.

elements-callback (procedure):
#<procedure #f (m)>  
Return a list of children, for use by a sequential iterator. Takes a single music parameter.

iterator-ctor (procedure):
ly:sequential-iterator::constructor  
Function to construct a music-event-iterator object for this music.

types (list):
'(general-music sequential-music)  
The types of this music object; determines by what engraver this music expression is processed.

1.1.60 SimultaneousMusic  
Music playing together.

Syntax: \simultaneous { ... } or << ... >>
Properties:

name (symbol):  
'SimultaneousMusic  
Name of this music object.

iterator-ctor (procedure):
ly:simultaneous-music-iterator::constructor  
Function to construct a music-event-iterator object for this music.
start-callback (procedure):
  ly:music-sequence::minimum-start-callback
  Function to compute the negative length of starting grace notes. This property can only be defined as initializer in 'scm/define-music-types.scm'.

length-callback (procedure):
  ly:music-sequence::maximum-length-callback
  How to compute the duration of this music. This property can only be defined as initializer in 'scm/define-music-types.scm'.

to-relative-callback (procedure):
  ly:music-sequence::simultaneous-relative-callback
  How to transform a piece of music to relative pitches.

types (list):
  '(general-music simultaneous-music)
  The types of this music object; determines by what engraver this music expression is processed.

1.1.61 SkipEvent
Filler that takes up duration, but does not print anything.

Syntax: s4 for a skip equivalent to a quarter rest.

Event classes: Section 1.2.47 [skip-event], page 40, Section 1.2.45 [rhythmic-event], page 40, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Not accepted by any engraver or performer.

Properties:
  name (symbol):
    'SkipEvent
    Name of this music object.

  types (list):
    '(general-music event rhythmic-event skip-event)
    The types of this music object; determines by what engraver this music expression is processed.

1.1.62 SkipMusic
Filler that takes up duration, does not print anything, and also does not create staves or voices implicitly.

Syntax: \skip duration

Properties:
  name (symbol):
    'SkipMusic
    Name of this music object.

  length-callback (procedure):
    ly:music-duration-length
    How to compute the duration of this music. This property can only be defined as initializer in 'scm/define-music-types.scm'.
iterator-ctor (procedure):
  ly:simple-music-iterator::constructor
  Function to construct a music-event-iterator object for this music.

types (list):
  '(general-music event rhythmic-event skip-event)
  The types of this music object; determines by what engraver this music expression is processed.

1.1.63 SlurEvent
Start or end slur.

Syntax: note ( and note)

Event classes: Section 1.2.48 [slur-event], page 40, Section 1.2.54 [span-event], page 41, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.96 [Slur engraver], page 237 and Section 2.2.97 [Slur performer], page 237.

Properties:
  name (symbol):
    'SlurEvent
    Name of this music object.

types (list):
  '(general-music span-event event slur-event)
  The types of this music object; determines by what engraver this music expression is processed.

1.1.64 SoloOneEvent
Print ‘Solo 1’.

Event classes: Section 1.2.49 [solo-one-event], page 41, Section 1.2.38 [part-combine-event], page 39, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.76 [Part combine engraver], page 231.

Properties:
  name (symbol):
    'SoloOneEvent
    Name of this music object.

  part-combine-status (symbol):
    'solo1
    Change to what kind of state? Options are solo1, solo2 and unisono.

types (list):
  '(general-music event part-combine-event solo-one-event)
  The types of this music object; determines by what engraver this music expression is processed.

1.1.65 SoloTwoEvent
Print ‘Solo 2’.

Event classes: Section 1.2.50 [solo-two-event], page 41, Section 1.2.38 [part-combine-event], page 39, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.76 [Part combine engraver], page 231.

Properties:
name (symbol):
  'SoloTwoEvent
Name of this music object.

part-combine-status (symbol):
  'solo2
Change to what kind of state? Options are solo1, solo2 and unisono.

types (list):
  '(general-music event part-combine-event solo-two-event)
The types of this music object; determines by what engraver this music expression is processed.

1.1.66 SostenutoEvent
Depress or release sostenuto pedal.
  Event classes: Section 1.2.51 [sostenuto-event], page 41, Section 1.2.39 [pedal-event], page 39, Section 1.2.54 [span-event], page 41, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.
  Accepted by: Section 2.2.80 [Piano_pedal_engraver], page 232 and Section 2.2.81 [Piano_pedal_performer], page 233.
  Properties:
  name (symbol):
    'SostenutoEvent
Name of this music object.

types (list):
  '(general-music event pedal-event sostenuto-event)
The types of this music object; determines by what engraver this music expression is processed.

1.1.67 SpacingSectionEvent
Start a new spacing section.
  Event classes: Section 1.2.52 [spacing-section-event], page 41, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.
  Accepted by: Section 2.2.98 [Spacing_engraver], page 237.
  Properties:
  name (symbol):
    'SpacingSectionEvent
Name of this music object.

types (list):
  '(general-music event spacing-section-event)
The types of this music object; determines by what engraver this music expression is processed.

1.1.68 SpanEvent
Event for anything that is started at a different time than stopped.
  Event classes: Section 1.2.54 [span-event], page 41, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.
  Not accepted by any engraver or performer.
  Properties:
name (symbol):
   'SpanEvent
   Name of this music object.

types (list):
   '(general-music event)
   The types of this music object; determines by what engraver this music expression is processed.

1.1.69 StaffSpanEvent
Start or stop a staff symbol.

Event classes: Section 1.2.55 [staff-span-event], page 41, Section 1.2.54 [span-event], page 41, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.104 [Staff_symbol_engraver], page 238.

Properties:

   name (symbol):
      'StaffSpanEvent
      Name of this music object.

   types (list):
      '(general-music event span-event staff-span-event)
      The types of this music object; determines by what engraver this music expression is processed.

1.1.70 StringNumberEvent
Specify on which string to play this note.

Syntax: \number

Event classes: Section 1.2.56 [string-number-event], page 41, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.37 [Fretboard_engraver], page 218 and Section 2.2.113 [Tab_note_heads_engraver], page 240.

Properties:

   name (symbol):
      'StringNumberEvent
      Name of this music object.

   types (list):
      '(general-music string-number-event event)
      The types of this music object; determines by what engraver this music expression is processed.

1.1.71 StrokeFingerEvent
Specify with which finger to pluck a string.

Syntax: \rightHandFinger text

Event classes: Section 1.2.57 [stroke-finger-event], page 42, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.34 [Fingering_engraver], page 217.

Properties:
name (symbol):
   'StrokeFingerEvent
   Name of this music object.

types (list):
   '(general-music stroke-finger-event event)
   The types of this music object; determines by what engraver this music expression is processed.

1.1.72 SustainEvent
Depress or release sustain pedal.

Event classes: Section 1.2.58 [sustain-event], page 42, Section 1.2.39 [pedal-event], page 39, Section 1.2.54 [span-event], page 41, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.80 [Piano_pedal_ engraver], page 232 and Section 2.2.81 [Piano_pedal_performer], page 233.

Properties:

   name (symbol):
      'SustainEvent
      Name of this music object.

   types (list):
      '(general-music event pedal-event sustain-event)
      The types of this music object; determines by what engraver this music expression is processed.

1.1.73 TextScriptEvent
Print text.

Event classes: Section 1.2.59 [text-script-event], page 42, Section 1.2.46 [script-event], page 40, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.116 [Text_ engraver], page 241.

Properties:

   name (symbol):
      'TextScriptEvent
      Name of this music object.

   types (list):
      '(general-music script-event text-script-event event)
      The types of this music object; determines by what engraver this music expression is processed.

1.1.74 TextSpanEvent
Start a text spanner, e.g., 8va.....|

Event classes: Section 1.2.60 [text-span-event], page 42, Section 1.2.54 [span-event], page 41, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.117 [Text_spanner_ engraver], page 242.

Properties:
name (symbol):
  'TextSpanEvent
  Name of this music object.

types (list):
  '(general-music span-event event text-span-event)
  The types of this music object; determines by what engraver this music
  expression is processed.

1.1.75 TieEvent

A tie.

Syntax: note-`

Event classes: Section 1.2.61 [tie-event], page 42, Section 1.2.33 [music-event], page 38 and
Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.19 [Completion_heads_ engraver], page 213, Section 2.2.118
[Tie_ engraver], page 242 and Section 2.2.119 [Tie_performer], page 242.

Properties:

name (symbol):
  'TieEvent
  Name of this music object.

types (list):
  '(general-music tie-event event)
  The types of this music object; determines by what engraver this music
  expression is processed.

1.1.76 TimeScaledMusic

Multiply durations, as in tuplets.

Syntax: \times fraction music, e.g. \times 2/3 { ... } for triplets.

Properties:

name (symbol):
  'TimeScaledMusic
  Name of this music object.

length-callback (procedure):
  ly:music-wrapper::length-callback
  How to compute the duration of this music. This property can only be
defined as initializer in ‘scm/define-music-types.scm’.

start-callback (procedure):
  ly:music-wrapper::start-callback
  Function to compute the negative length of starting grace
  notes. This property can only be defined as initializer in ‘scm/
derine-music-types.scm’.

iterator-ctor (procedure):
  ly:time-scaled-music-iterator::constructor
  Function to construct a music-event-iterator object for this music.

types (list):
  '(time-scaled-music music-wrapper-music general-music)
  The types of this music object; determines by what engraver this music
  expression is processed.
1.1.77 TransposedMusic
Music that has been transposed.

Properties:

- **name (symbol):**
  - 'TransposedMusic
    - Name of this music object.
- **iterator-ctor (procedure):**
  - ly:music-wrapper-iterator::constructor
    - Function to construct a music-event-iterator object for this music.
- **start-callback (procedure):**
  - ly:music-wrapper::start-callback
    - Function to compute the negative length of starting grace notes. This property can only be defined as initializer in `scm/define-music-types.scm`.
- **length-callback (procedure):**
  - ly:music-wrapper::length-callback
    - How to compute the duration of this music. This property can only be defined as initializer in `scm/define-music-types.scm`.
- **to-relative-callback (procedure):**
  - ly:relative-octave-music::no-relative-callback
    - How to transform a piece of music to relative pitches.
- **types (list):**
  - '(music-wrapper-music general-music transposed-music)
    - The types of this music object; determines by what engraver this music expression is processed.

1.1.78 TremoloEvent
Unmeasured tremolo.

Event classes: Section 1.2.62 [tremolo-event], page 42, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.107 [Stem_engraver], page 239.

Properties:

- **name (symbol):**
  - 'TremoloEvent
    - Name of this music object.
- **types (list):**
  - '(general-music event tremolo-event)
    - The types of this music object; determines by what engraver this music expression is processed.

1.1.79 TremoloRepeatedMusic
Repeated notes denoted by tremolo beams.

Properties:
name (symbol):
   'TremoloRepeatedMusic
   Name of this music object.

iterator-ctor (procedure):
   ly:chord-tremolo-iterator::constructor
   Function to construct a music-event-iterator object for this music.

start-callback (procedure):
   ly:repeated-music::first-start
   Function to compute the negative length of starting grace notes. This property can only be defined as initializer in 'scm/define-music-types.scm'.

length-callback (procedure):
   ly:repeated-music::folded-music-length
   How to compute the duration of this music. This property can only be defined as initializer in 'scm/define-music-types.scm'.

types (list):
   '(general-music repeated-music tremolo-repeated-music)
   The types of this music object; determines by what engraver this music expression is processed.

1.1.80 TremoloSpanEvent

Tremolo over two stems

Event classes: Section 1.2.63 [tremolo-span-event], page 42, Section 1.2.54 [span-event], page 41, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.15 [Chord_tremolo_ engraver], page 211.

Properties:

name (symbol):
   'TremoloSpanEvent
   Name of this music object.

types (list):
   '(general-music event span-event tremolo-span-event)
   The types of this music object; determines by what engraver this music expression is processed.

1.1.81 TrillSpanEvent

Start a trill spanner tr~~~

Event classes: Section 1.2.64 [trill-span-event], page 42, Section 1.2.54 [span-event], page 41, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.124 [Trill_spanner_ engraver], page 244.

Properties:

name (symbol):
   'TrillSpanEvent
   Name of this music object.

types (list):
   '(general-music span-event event trill-span-event)
   The types of this music object; determines by what engraver this music expression is processed.
1.1.82 TupletSpanEvent

Used internally to signal where tuplet brackets start and stop.

Event classes: Section 1.2.65 [tuplet-span-event], page 42, Section 1.2.54 [span-event], page 41, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.125 [Tuplet_engraver], page 244.

Properties:

- **name** (symbol):
  - ‘TupletSpanEvent
    Name of this music object.

- **types** (list):
  - ‘(tuplet-span-event span-event event general-music)
    The types of this music object; determines by what engraver this music expression is processed.

1.1.83 UnaCordaEvent

Depress or release una-corda pedal.

Event classes: Section 1.2.66 [una-corda-event], page 43, Section 1.2.39 [pedal-event], page 39, Section 1.2.54 [span-event], page 41, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.80 [Piano_pedal_ engraver], page 232 and Section 2.2.81 [Piano_pedal_performer], page 233.

Properties:

- **name** (symbol):
  - ‘UnaCordaEvent
    Name of this music object.

- **types** (list):
  - ‘(general-music event pedal-event una-corda-event)
    The types of this music object; determines by what engraver this music expression is processed.

1.1.84 UnfoldedRepeatedMusic

Repeated music which is fully written (and played) out.

Properties:

- **name** (symbol):
  - ‘UnfoldedRepeatedMusic
    Name of this music object.

- **iterator-ctor** (procedure):
  - ly:unfolded-repeat-iterator::constructor
    Function to construct a music-event-iterator object for this music.

- **start-callback** (procedure):
  - ly:repeated-music::first-start
    Function to compute the negative length of starting grace notes. This property can only be defined as initializer in ‘scm/define-music-types.scm’.
Chapter 1: Music definitions

1.1.85 UnisonoEvent

Print ‘a 2’.

Event classes: Section 1.2.67 [unisono-event], page 43, Section 1.2.38 [part-combine-event], page 39, Section 1.2.33 [music-event], page 38 and Section 1.2.1 [StreamEvent], page 34.

Accepted by: Section 2.2.76 [Part combine engraver], page 231.

Properties:

name (symbol):
 'UnisonoEvent
 Name of this music object.

part-combine-status (symbol):
 'unisono
 Change to what kind of state? Options are solo1, solo2 and unisono.

types (list):
 '(general-music event part-combine-event unisono-event)
 The types of this music object; determines by what engraver this music expression is processed.

1.1.86 UnrelativableMusic

Music that cannot be converted from relative to absolute notation. For example, transposed music.

Properties:

name (symbol):
 'UnrelativableMusic
 Name of this music object.

to-relative-callback (procedure):
 ly:relative-octave-music::no-relative-callback
 How to transform a piece of music to relative pitches.

iterator-ctor (procedure):
 ly:.music-wrapper-iterator::constructor
 Function to construct a music-event-iterator object for this music.

length-callback (procedure):
 ly:music-wrapper::length-callback
 How to compute the duration of this music. This property can only be defined as initializer in ‘scm/define-music-types.scm’. 
types (list):
'\{(music-wrapper-music general-music unrelativable-music)\}
The types of this music object; determines by what engraver this music
expression is processed.

1.1.87 VoiceSeparator
Separate polyphonic voices in simultaneous music.
Syntax: \ \ \\
Properties:

name (symbol):
'VoiceSeparator
Name of this music object.

types (list):
'\{(separator general-music)\}
The types of this music object; determines by what engraver this music
expression is processed.

1.1.88 VoltaRepeatedMusic
Repeats with alternatives placed sequentially.
Properties:

name (symbol):
'VoltaRepeatedMusic
Name of this music object.

iterator-ctor (procedure):
ly:volta-repeat-iterator::constructor
Function to construct a music-event-iterator object for this music.

start-callback (procedure):
ly:repeated-music::first-start
Function to compute the negative length of starting grace
notes. This property can only be defined as initializer in 'scm/
define-music-types.scm'.

length-callback (procedure):
ly:repeated-music::volta-music-length
How to compute the duration of this music. This property can only be
declared as initializer in ‘scm/define-music-types.scm’.

types (list):
'\{(general-music repeated-music volta-repeated-music)\}
The types of this music object; determines by what engraver this music
expression is processed.

1.2 Music classes

1.2.1 StreamEvent
Music event type StreamEvent is in music objects of type Section 1.1.1 [Absolute-
DynamicEvent], page 2, Section 1.1.2 [AnnotateOutputEvent], page 2, Section 1.1.4
Chapter 1: Music definitions

[ApplyOutputEvent], page 3, Section 1.1.5 [ArpeggioEvent], page 3, Section 1.1.6 [ArticulationEvent], page 3, Section 1.1.9 [BassFigureEvent], page 5, Section 1.1.10 [BeamEvent], page 5, Section 1.1.11 [BeamForbidEvent], page 5, Section 1.1.12 [BendAfterEvent], page 6, Section 1.1.13 [BreathingEvent], page 6, Section 1.1.14 [ClusterNoteEvent], page 6, Section 1.1.17 [CrescendoEvent], page 7, Section 1.1.18 [DecrescendoEvent], page 8, Section 1.1.21 [ExtenderEvent], page 9, Section 1.1.22 [FingeringEvent], page 9, Section 1.1.23 [GlissandoEvent], page 10, Section 1.1.25 [HarmonicEvent], page 10, Section 1.1.26 [HyphenEvent], page 11, Section 1.1.27 [KeyChangeEvent], page 11, Section 1.1.28 [LabelEvent], page 11, Section 1.1.29 [LaissezVibrerEvent], page 12, Section 1.1.30 [LigatureEvent], page 12, Section 1.1.31 [LineBreakEvent], page 13, Section 1.1.33 [LyricEvent], page 13, Section 1.1.34 [MarkEvent], page 14, Section 1.1.35 [MultiMeasureRestEvent], page 14, Section 1.1.37 [MultiMeasureTextEvent], page 15, Section 1.1.39 [NoteEvent], page 15, Section 1.1.40 [NoteGroupingEvent], page 16, Section 1.1.42 [PageBreakEvent], page 16, Section 1.1.43 [PageTurnEvent], page 17, Section 1.1.45 [PercentEvent], page 18, Section 1.1.47 [PesOrFlexaEvent], page 18, Section 1.1.48 [PhrasingSlurEvent], page 19, Section 1.1.54 [RepeatTieEvent], page 21, Section 1.1.56 [RestEvent], page 22, Section 1.1.58 [ScriptEvent], page 22, Section 1.1.61 [SkipEvent], page 24, Section 1.1.63 [SlurEvent], page 25, Section 1.1.64 [SoloOneEvent], page 25, Section 1.1.65 [SoloTwoEvent], page 25, Section 1.1.66 [SostenutoEvent], page 26, Section 1.1.67 [SpacingSectionEvent], page 26, Section 1.1.68 [SpanEvent], page 26, Section 1.1.69 [StaffSpanEvent], page 27, Section 1.1.70 [StringNumberEvent], page 27, Section 1.1.71 [StrokeFingerEvent], page 27, Section 1.1.72 [SustainEvent], page 28, Section 1.1.73 [TextScriptEvent], page 28, Section 1.1.74 [TextSpanEvent], page 28, Section 1.1.75 [TieEvent], page 29, Section 1.1.78 [TremoloEvent], page 30, Section 1.1.80 [TremoloSpanEvent], page 31, Section 1.1.81 [TrillSpanEvent], page 31, Section 1.1.82 [TupletSpanEvent], page 32, Section 1.1.83 [UnaCordaEvent], page 32 and Section 1.1.85 [UnisonoEvent], page 33.

Not accepted by any engraver or performer.

1.2.2 absolute-dynamic-event
Music event type absolute-dynamic-event is in music objects of type Section 1.1.1 [AbsoluteDynamicEvent], page 2.

Accepted by: Section 2.2.28 [Dynamic_engraver], page 215, Section 2.2.29 [Dynamic_performer], page 216 and Section 2.2.63 [New_dynamic_engraver], page 227.

1.2.3 annotate-output-event
Music event type annotate-output-event is in music objects of type Section 1.1.2 [AnnotateOutputEvent], page 2.

Accepted by: Section 2.2.6 [Balloon_engraver], page 208.

1.2.4 apply-output-event
Music event type apply-output-event is in music objects of type Section 1.1.4 [ApplyOutputEvent], page 3.

Accepted by: Section 2.2.72 [Output_property_engraver], page 229.

1.2.5 arpeggio-event
Music event type arpeggio-event is in music objects of type Section 1.1.5 [ArpeggioEvent], page 3.

Accepted by: Section 2.2.3 [Arpeggio_engraver], page 207.
1.2.6 articulation-event
Music event type articulation-event is in music objects of type Section 1.1.6 [Articulation-Event], page 3.
Accepted by: Section 2.2.91 [Script_engraver], page 235.

1.2.7 bass-figure-event
Music event type bass-figure-event is in music objects of type Section 1.1.9 [BassFigureEvent], page 5.
Accepted by: Section 2.2.32 [Figured_bass_engraver], page 216.

1.2.8 beam-event
Music event type beam-event is in music objects of type Section 1.1.10 [BeamEvent], page 5.
Accepted by: Section 2.2.9 [Beam_engraver], page 209, Section 2.2.10 [Beam_performer], page 210 and Section 2.2.39 [Grace_beam_engraver], page 219.

1.2.9 beam-forbid-event
Music event type beam-forbid-event is in music objects of type Section 1.1.11 [BeamForbidEvent], page 5.
Accepted by: Section 2.2.4 [Auto_beam_engraver], page 207.

1.2.10 bend-after-event
Music event type bend-after-event is in music objects of type Section 1.1.12 [BendAfterEvent], page 6.
Accepted by: Section 2.2.11 [Bend_engraver], page 210.

1.2.11 break-event
Music event type break-event is in music objects of type Section 1.1.31 [LineBreakEvent], page 13, Section 1.1.42 [PageBreakEvent], page 16 and Section 1.1.43 [PageTurnEvent], page 17.
Accepted by: Section 2.2.73 [Page_turn_engraver], page 230 and Section 2.2.74 [Paper_column_engraver], page 230.

1.2.12 breathing-event
Music event type breathing-event is in music objects of type Section 1.1.13 [BreathingEvent], page 6.
Accepted by: Section 2.2.13 [Breathing_sign_engraver], page 210.

1.2.13 cluster-note-event
Music event type cluster-note-event is in music objects of type Section 1.1.14 [ClusterNoteEvent], page 6.
Accepted by: Section 2.2.17 [Cluster_spanner_engraver], page 212.

1.2.14 crescendo-event
Music event type crescendo-event is in music objects of type Section 1.1.17 [CrescendoEvent], page 7.
Accepted by: Section 2.2.29 [Dynamic_performer], page 216.

1.2.15 decrescendo-event
Music event type decrescendo-event is in music objects of type Section 1.1.18 [DecrescendoEvent], page 8.
Accepted by: Section 2.2.29 [Dynamic_performer], page 216.
1.2.16 dynamic-event
Music event type dynamic-event is in music objects of type Section 1.1.1 [AbsoluteDynamicEvent], page 2.
Not accepted by any engraver or performer.

1.2.17 extender-event
Music event type extender-event is in music objects of type Section 1.1.21 [ExtenderEvent], page 9.
Accepted by: Section 2.2.31 [Extender_engraver], page 216.

1.2.18 fingering-event
Music event type fingering-event is in music objects of type Section 1.1.22 [FingeringEvent], page 9.
Accepted by: Section 2.2.34 [Fingering_engraver], page 217.

1.2.19 glissando-event
Music event type glissando-event is in music objects of type Section 1.1.23 [GlissandoEvent], page 10.
Accepted by: Section 2.2.38 [Glissando_engraver], page 219.

1.2.20 harmonic-event
Music event type harmonic-event is in music objects of type Section 1.1.25 [HarmonicEvent], page 10.
Not accepted by any engraver or performer.

1.2.21 hyphen-event
Music event type hyphen-event is in music objects of type Section 1.1.26 [HyphenEvent], page 11.
Accepted by: Section 2.2.47 [Hyphen_engraver], page 221.

1.2.22 key-change-event
Music event type key-change-event is in music objects of type Section 1.1.27 [KeyChangeEvent], page 11.
Accepted by: Section 2.2.50 [Key_engraver], page 222 and Section 2.2.51 [Key_performer], page 223.

1.2.23 label-event
Music event type label-event is in music objects of type Section 1.1.28 [LabelEvent], page 11.
Accepted by: Section 2.2.74 [Paper_column_engraver], page 230.

1.2.24 laissez-vibrer-event
Music event type laissez-vibrer-event is in music objects of type Section 1.1.29 [LaissezVibrerEvent], page 12.
Accepted by: Section 2.2.52 [Laissez_vibrer_engraver], page 223.

1.2.25 layout-instruction-event
Music event type layout-instruction-event is in music objects of type Section 1.1.4 [ApplyOutputEvent], page 3.
Not accepted by any engraver or performer.
1.2.26 ligature-event
Music event type ligature-event is in music objects of type Section 1.1.30 [LigatureEvent], page 12.
   Accepted by: Section 2.2.54 [Ligature_bracket_engraver], page 224, Section 2.2.60 [Mensural_ligature_engraver], page 225 and Section 2.2.127 [Vaticana_ligature_engraver], page 245.

1.2.27 line-break-event
Music event type line-break-event is in music objects of type Section 1.1.31 [LineBreakEvent], page 13.
   Not accepted by any engraver or performer.

1.2.28 lyric-event
Music event type lyric-event is in music objects of type Section 1.1.33 [LyricEvent], page 13.
   Accepted by: Section 2.2.55 [Lyric_engraver], page 224 and Section 2.2.56 [Lyric_performer], page 224.

1.2.29 mark-event
Music event type mark-event is in music objects of type Section 1.1.34 [MarkEvent], page 14.
   Accepted by: Section 2.2.57 [Mark_engraver], page 224.

1.2.30 melodic-event
Music event type melodic-event is in music objects of type Section 1.1.14 [ClusterNoteEvent], page 6 and Section 1.1.39 [NoteEvent], page 15.
   Not accepted by any engraver or performer.

1.2.31 multi-measure-rest-event
Music event type multi-measure-rest-event is in music objects of type Section 1.1.35 [MultiMeasureRestEvent], page 14.
   Accepted by: Section 2.2.62 [Multi_measure_rest_engraver], page 226.

1.2.32 multi-measure-text-event
Music event type multi-measure-text-event is in music objects of type Section 1.1.37 [MultiMeasureTextEvent], page 15.
   Accepted by: Section 2.2.62 [Multi_measure_rest_engraver], page 226.

1.2.33 music-event
Music event type music-event is in music objects of type Section 1.1.1 [Absolute-DynamicEvent], page 2, Section 1.1.2 [AnnotateOutputEvent], page 2, Section 1.1.4 [ApplyOutputEvent], page 3, Section 1.1.5 [ArpeggioEvent], page 3, Section 1.1.6 [ArticulationEvent], page 3, Section 1.1.9 [BassFigureEvent], page 5, Section 1.1.10 [BeamEvent], page 5, Section 1.1.11 [BeamForbidEvent], page 5, Section 1.1.12 [BendAfterEvent], page 6, Section 1.1.13 [BreathingEvent], page 6, Section 1.1.14 [ClusterNoteEvent], page 6, Section 1.1.17 [CrescendoEvent], page 7, Section 1.1.18 [DecrescendoEvent], page 8, Section 1.1.21 [ExtenderEvent], page 9, Section 1.1.22 [FingeringEvent], page 9, Section 1.1.23 [GlissandoEvent], page 10, Section 1.1.25 [HarmonicEvent], page 10, Section 1.1.26 [Hyphen-Event], page 11, Section 1.1.27 [KeyChangeEvent], page 11, Section 1.1.28 [LabelEvent], page 11, Section 1.1.29 [LaissezVibrierEvent], page 12, Section 1.1.30 [LigatureEvent], page 12, Section 1.1.31 [LineBreakEvent], page 13, Section 1.1.33 [LyricEvent], page 13, Section 1.1.34 [MarkEvent], page 14, Section 1.1.35 [MultiMeasureRestEvent], page 14,
Section 1.1.37 [MultiMeasureTextEvent], page 15, Section 1.1.39 [NoteEvent], page 15, Section 1.1.40 [NoteGroupingEvent], page 16, Section 1.1.42 [PageBreakEvent], page 16, Section 1.1.43 [PageTurnEvent], page 17, Section 1.1.45 [PercentEvent], page 18, Section 1.1.47 [PesOrFlexaEvent], page 18, Section 1.1.48 [PhrasingSlurEvent], page 19, Section 1.1.54 [RepeatTieEvent], page 21, Section 1.1.56 [RestEvent], page 22, Section 1.1.58 [ScriptEvent], page 22, Section 1.1.61 [SkipEvent], page 24, Section 1.1.63 [SlurEvent], page 25, Section 1.1.64 [SoloOneEvent], page 25, Section 1.1.65 [SoloTwoEvent], page 25, Section 1.1.66 [SostenutoEvent], page 26, Section 1.1.67 [SpacingSectionEvent], page 26, Section 1.1.68 [SpanEvent], page 26, Section 1.1.69 [StaffSpanEvent], page 27, Section 1.1.70 [StringNumberEvent], page 27, Section 1.1.71 [StrokeFingerEvent], page 27, Section 1.1.72 [SustainEvent], page 28, Section 1.1.73 [TextScriptEvent], page 28, Section 1.1.74 [TextSpanEvent], page 28, Section 1.1.75 [TieEvent], page 29, Section 1.1.78 [TremoloEvent], page 30, Section 1.1.80 [TremoloSpanEvent], page 31, Section 1.1.81 [TrillSpanEvent], page 31, Section 1.1.82 [TupletSpanEvent], page 32, Section 1.1.83 [UnaCordaEvent], page 32 and Section 1.1.85 [UnisonoEvent], page 33.

Not accepted by any engraver or performer.

1.2.34 note-event

Music event type note-event is in music objects of type Section 1.1.39 [NoteEvent], page 15.

Accepted by: Section 2.2.14 [Chord_name_engraver], page 211, Section 2.2.19 [Completion_heads_engraver], page 213, Section 2.2.25 [Drum_note_performer], page 215, Section 2.2.26 [Drum_notes_engraver], page 215, Section 2.2.37 [Fretboard_engraver], page 218, Section 2.2.66 [Note_heads_engraver], page 228, Section 2.2.67 [Note_name_engraver], page 228, Section 2.2.68 [Note_performer], page 229 and Section 2.2.113 [Tab_note_heads_engraver], page 240.

1.2.35 note-grouping-event

Music event type note-grouping-event is in music objects of type Section 1.1.40 [NoteGroupingEvent], page 16.

Accepted by: Section 2.2.46 [Horizontal_bracket_engraver], page 221.

1.2.36 page-break-event

Music event type page-break-event is in music objects of type Section 1.1.42 [PageBreakEvent], page 16.

Not accepted by any engraver or performer.

1.2.37 page-turn-event

Music event type page-turn-event is in music objects of type Section 1.1.43 [PageTurnEvent], page 17.

Not accepted by any engraver or performer.

1.2.38 part-combine-event

Music event type part-combine-event is in music objects of type Section 1.1.64 [SoloOneEvent], page 25, Section 1.1.65 [SoloTwoEvent], page 25 and Section 1.1.85 [UnisonoEvent], page 33.

Accepted by: Section 2.2.76 [Part_combine_engraver], page 231.

1.2.39 pedal-event

Music event type pedal-event is in music objects of type Section 1.1.66 [SostenutoEvent], page 26, Section 1.1.72 [SustainEvent], page 28 and Section 1.1.83 [UnaCordaEvent], page 32.

Not accepted by any engraver or performer.
1.2.40 percent-event
Music event type percent-event is in music objects of type Section 1.1.45 [PercentEvent], page 18.
   Accepted by: Section 2.2.77 [Percent_repeat_engraver], page 231 and Section 2.2.95 [Slash_repeat_engraver], page 236.

1.2.41 pes-or-flexa-event
Music event type pes-or-flexa-event is in music objects of type Section 1.1.47 [PesOrFlexaEvent], page 18.
   Accepted by: Section 2.2.127 [Vaticana_ligature_engraver], page 245.

1.2.42 phrasing-slur-event
Music event type phrasing-slur-event is in music objects of type Section 1.1.48 [PhrasingSlurEvent], page 19.
   Accepted by: Section 2.2.78 [Phrasing_slur_engraver], page 232.

1.2.43 repeat-tie-event
Music event type repeat-tie-event is in music objects of type Section 1.1.54 [RepeatTieEvent], page 21.
   Accepted by: Section 2.2.85 [Repeat_tie_engraver], page 234.

1.2.44 rest-event
Music event type rest-event is in music objects of type Section 1.1.56 [RestEvent], page 22.
   Accepted by: Section 2.2.32 [Figured_bass_engraver], page 216 and Section 2.2.87 [Rest_engraver], page 234.

1.2.45 rhythmic-event
Music event type rhythmic-event is in music objects of type Section 1.1.9 [BassFigureEvent], page 5, Section 1.1.14 [ClusterNoteEvent], page 6, Section 1.1.33 [LyricEvent], page 13, Section 1.1.35 [MultiMeasureRestEvent], page 14, Section 1.1.39 [NoteEvent], page 15, Section 1.1.45 [PercentEvent], page 18, Section 1.1.56 [RestEvent], page 22 and Section 1.1.61 [SkipEvent], page 24.
   Not accepted by any engraver or performer.

1.2.46 script-event
Music event type script-event is in music objects of type Section 1.1.6 [ArticulationEvent], page 3, Section 1.1.58 [ScriptEvent], page 22 and Section 1.1.73 [TextScriptEvent], page 28.
   Not accepted by any engraver or performer.

1.2.47 skip-event
Music event type skip-event is in music objects of type Section 1.1.61 [SkipEvent], page 24.
   Not accepted by any engraver or performer.

1.2.48 slur-event
Music event type slur-event is in music objects of type Section 1.1.63 [SlurEvent], page 25.
   Accepted by: Section 2.2.96 [Slur_engraver], page 237 and Section 2.2.97 [Slur_performer], page 237.
1.2.49 solo-one-event
Music event type **solo-one-event** is in music objects of type Section 1.1.64 [SoloOneEvent], page 25.
Not accepted by any engraver or performer.

1.2.50 solo-two-event
Music event type **solo-two-event** is in music objects of type Section 1.1.65 [SoloTwoEvent], page 25.
Not accepted by any engraver or performer.

1.2.51 sostenuto-event
Music event type **sostenuto-event** is in music objects of type Section 1.1.66 [SostenutoEvent], page 26.
Accepted by: Section 2.2.80 [Piano_pedal_engraver], page 232 and Section 2.2.81 [Piano_pedal_performer], page 233.

1.2.52 spacing-section-event
Music event type **spacing-section-event** is in music objects of type Section 1.1.67 [SpacingSectionEvent], page 26.
Accepted by: Section 2.2.98 [Spacing_engraver], page 237.

1.2.53 span-dynamic-event
Music event type **span-dynamic-event** is in music objects of type Section 1.1.17 [CrescendoEvent], page 7 and Section 1.1.18 [DecrescendoEvent], page 8.
Accepted by: Section 2.2.28 [Dynamic_engraver], page 215 and Section 2.2.63 [New_dynamic_engraver], page 227.

1.2.54 span-event
Music event type **span-event** is in music objects of type Section 1.1.10 [BeamEvent], page 5, Section 1.1.17 [CrescendoEvent], page 7, Section 1.1.18 [DecrescendoEvent], page 8, Section 1.1.30 [LigatureEvent], page 12, Section 1.1.48 [PhrasingSlurEvent], page 19, Section 1.1.63 [SlurEvent], page 25, Section 1.1.66 [SostenutoEvent], page 26, Section 1.1.68 [SpanEvent], page 26, Section 1.1.69 [StaffSpanEvent], page 27, Section 1.1.72 [SustainEvent], page 28, Section 1.1.74 [TextSpanEvent], page 28, Section 1.1.80 [TremoloSpanEvent], page 31, Section 1.1.81 [TrillSpanEvent], page 31, Section 1.1.82 [TupletSpanEvent], page 32 and Section 1.1.83 [UnaCordaEvent], page 32.
Not accepted by any engraver or performer.

1.2.55 staff-span-event
Music event type **staff-span-event** is in music objects of type Section 1.1.69 [StaffSpanEvent], page 27.
Accepted by: Section 2.2.104 [Staff_symbol_engraver], page 238.

1.2.56 string-number-event
Music event type **string-number-event** is in music objects of type Section 1.1.70 [StringNumberEvent], page 27.
Accepted by: Section 2.2.37 [Fretboard_engraver], page 218 and Section 2.2.113 [Tab_note_heads_engraver], page 240.
1.2.57 stroke-finger-event
Music event type stroke-finger-event is in music objects of type Section 1.1.71 [StrokeFingerEvent], page 27.

Accepted by: Section 2.2.34 [Fingering_engraver], page 217.

1.2.58 sustain-event
Music event type sustain-event is in music objects of type Section 1.1.72 [SustainEvent], page 28.

Accepted by: Section 2.2.80 [Piano_pedal_engraver], page 232 and Section 2.2.81 [Piano_pedal_performer], page 233.

1.2.59 text-script-event
Music event type text-script-event is in music objects of type Section 1.1.73 [TextScriptEvent], page 28.

Accepted by: Section 2.2.116 [Text_engraver], page 241.

1.2.60 text-span-event
Music event type text-span-event is in music objects of type Section 1.1.74 [TextSpanEvent], page 28.

Accepted by: Section 2.2.117 [Text_spanner_engraver], page 242.

1.2.61 tie-event
Music event type tie-event is in music objects of type Section 1.1.75 [TieEvent], page 29.

Accepted by: Section 2.2.19 [Completion_heads_engraver], page 213, Section 2.2.118 [Tie_engraver], page 242 and Section 2.2.119 [Tie_performer], page 242.

1.2.62 tremolo-event
Music event type tremolo-event is in music objects of type Section 1.1.78 [TremoloEvent], page 30.

Accepted by: Section 2.2.107 [Stem_engraver], page 239.

1.2.63 tremolo-span-event
Music event type tremolo-span-event is in music objects of type Section 1.1.80 [TremoloSpanEvent], page 31.

Accepted by: Section 2.2.15 [Chord_tremolo_engraver], page 211.

1.2.64 trill-span-event
Music event type trill-span-event is in music objects of type Section 1.1.81 [TrillSpanEvent], page 31.

Accepted by: Section 2.2.124 [Trill_spanner_engraver], page 244.

1.2.65 tuplet-span-event
Music event type tuplet-span-event is in music objects of type Section 1.1.82 [TupletSpanEvent], page 32.

Accepted by: Section 2.2.125 [Tuplet_engraver], page 244.
1.2.66 una-corda-event
Music event type una-corda-event is in music objects of type Section 1.1.83 [UnaCordaEvent], page 32.

Accepted by: Section 2.2.80 [Piano_pedal_ engraver], page 232 and Section 2.2.81 [Piano_pedal_performer], page 233.

1.2.67 unisono-event
Music event type unisono-event is in music objects of type Section 1.1.85 [UnisonoEvent], page 33.

Not accepted by any engraver or performer.

1.3 Music properties

X-offset (number)
Offset of resulting grob; only used for balloon texts.

Y-offset (number)
Offset of resulting grob; only used for balloon texts.

absolute-octave (integer)
The absolute octave for a octave check note.

alteration (number)
Alteration for figured bass.

articulation-type (string)
Key for script definitions alist.
TODO: Consider making type into symbol.

articulations (list of music)
Articulation events specifically for this note.

associated-context (string)
Name of the Voice context associated with this newaddlyrics section.

augmented (boolean)
This figure is for an augmented figured bass (with + sign).

augmented-slash (boolean)
This figure is for an augmented figured bass (back-slashed number).

bass (boolean)
Set if this note is a bass note in a chord.

bracket-start (boolean)
Start a bracket here.
TODO: Use SpanEvents?

bracket-stop (boolean)
Stop a bracket here.

break-penalty (number)
Penalty for line break hint.

break-permission (symbol)
Whether to allow, forbid or force a line break.

cautonary (boolean)
If set, this alteration needs a cautionary accidental.
change-to-id (string)
    Name of the context to change to.

change-to-type (symbol)
    Type of the context to change to.

compress-procedure (procedure)
    Compress this music expression. Arg 1: the music, arg 2: factor.

context-id (string)
    Name of context.

context-type (symbol)
    Type of context.

create-new (boolean)
    Create a fresh context.

delta-step (number)
    How much should a fall change pitch?

denominator (integer)
    Denominator in a time signature.

descend-only (boolean)
    If set, this \context only descends in the context tree.

digit (integer)
    Digit for fingering.

diminished (boolean)
    This bass figure should be slashed.

direction (direction)
    Print this up or down?

drum-type (symbol)
    Which percussion instrument to play this note on.

duration (duration)
    Duration of this note or lyric.

element (music)
    The single child of a Music_wrapper music object, or the body of a repeat.

elements (list of music)
    A list of elements for sequential of simultaneous music, or the alternatives of repeated music.

elements-callback (procedure)
    Return a list of children, for use by a sequential iterator. Takes a single music parameter.

error-found (boolean)
    If true, a parsing error was found in this expression.

figure (integer)
    A bass figure.

force-accidental (boolean)
    If set, a cautionary accidental should always be printed on this note.

grob-property (symbol)
    The symbol of the grob property to set.
grob-property-path (list)
   A list of symbols, locating a nested grob property, e.g., (beamed-lengths details).

grob-value (any type)
   The value of the grob property to set.

input-tag (any type)
   Arbitrary marker to relate input and output.

inversion (boolean)
   If set, this chord note is inverted.

iterator-ctor (procedure)
   Function to construct a music-event-iterator object for this music.

label (markup)
   Label of a mark.

last-pitch (pitch)
   The last pitch after relativization.

length (moment)
   The duration of this music.

length-callback (procedure)
   How to compute the duration of this music. This property can only be defined as
   initializer in ‘scm/define-music-types.scm’.

line-break-permission (symbol)
   When the music is at top-level, whether to allow, forbid or force a line break.

metronome-count (number)
   How many beats in a minute?

name (symbol)
   Name of this music object.

no-continuation (boolean)
   If set, disallow continuation lines.

numerator (integer)
   Numerator of a time signature.

octavation (integer)
   This pitch was octavated by how many octaves? For chord inversions, this is nega-
   tive.

once (boolean)
   Apply this operation only during one time step?

origin (input location)
   Where was this piece of music defined?

page-break-permission (symbol)
   When the music is at top-level, whether to allow, forbid or force a page break.

page-label (symbol)
   The label of a page marker.

page-marker (boolean)
   If true, and the music expression is found at top-level, a page marker object is
   instanciated instead of a score.
**page-turn-permission** (symbol)
   When the music is at top-level, whether to allow, forbid or force a page turn.

**parenthesize** (boolean)
   Enclose resulting objects in parentheses?

**part-combine-status** (symbol)
   Change to what kind of state? Options are solo1, solo2 and unisono.

**pitch** (pitch)
   The pitch of this note.

**pitch-alist** (list)
   A list of pitches jointly forming the scale of a key signature.

**pop-first** (boolean)
   Do a revert before we try to do a override on some grob property.

**prob-property** (symbol)
   The symbol of the prob property to set.

**procedure** (procedure)
   The function to run with \applycontext. It must take a single argument, being the context.

**property-operations** (list)
   Do these operations for instantiating the context.

**quoted-context-id** (string)
   The ID of the context to direct quotes to, e.g., cue.

**quoted-context-type** (symbol)
   The name of the context to direct quotes to, e.g., Voice.

**quoted-events** (vector)
   A vector of with moment and event-list entries.

**quoted-music-name** (string)
   The name of the voice to quote.

**quoted-transposition** (pitch)
   The pitch used for the quote, overriding \transposition.

**quoted-voice-direction** (direction)
   Should the quoted voice be up-stem or down-stem?

**repeat-count** (integer)
   Do a \repeat how often?

**span-direction** (direction)
   Does this start or stop a spanner?

**span-type** (string)
   What kind of spanner should be created?
   TODO: Consider making type into symbol.

**split-list** (list)
   Splitting moments for part combiner.

**start-callback** (procedure)
   Function to compute the negative length of starting grace notes. This property can only be defined as initializer in ‘scm/define-music-types.scm’.
**string-number** (integer)

The number of the string in a `StringNumberEvent`.

**symbol** (symbol)

Grob name to perform an override or revert on.

**tags** (list)

List of symbols that for denoting extra details, e.g., \tag #'part ... could tag a piece of music as only being active in a part.

**tempo-unit** (duration)

The unit for the metronome count.

**text** (markup)

Markup expression to be printed.

**text-type** (symbol)

Particular type of text script (e.g., finger, dynamic).

**to-relative-callback** (procedure)

How to transform a piece of music to relative pitches.

**tonic** (pitch)

Base of the scale.

**tremolo-type** (integer)

Speed of tremolo, e.g., 16 for c4:16.

**trill-pitch** (pitch)

Pitch of other note of the trill.

**tweaks** (list)

An alist of properties to override in the backend for the grob made of this event.

**type** (symbol)

The type of this music object. Determines iteration in some cases.

**types** (list)

The types of this music object; determines by what engraver this music expression is processed.

**untransposable** (boolean)

If set, this music is not transposed.

**value** (any type)

Assignment value for a translation property.

**void** (boolean)

If this property is #t, then the music expression is to be discarded by the toplevel music handler.

**what** (symbol)

What to change for auto-change.

FIXME: Naming.
2 Translation

2.1 Contexts

2.1.1 ChoirStaff

Identical to StaffGroup except that the contained staves are not connected vertically.

This context creates the following layout object(s):

- Section 3.1.105 [SystemStartBar], page 337
- Section 3.1.106 [SystemStartBrace], page 338
- Section 3.1.107 [SystemStartBracket], page 339
- Section 3.1.108 [SystemStartSquare], page 340

This context sets the following properties:

- Set translator property shortVocalName to '().
- Set translator property vocalName to '().
- Set translator property systemStartDelimiter to 'SystemStartBracket.

Context ChoirStaff can contain:

- Section 2.1.20 [Staff], page 145
- Section 2.1.21 [StaffGroup], page 154
- Section 2.1.1 [ChoirStaff], page 48
- Section 2.1.2 [ChordNames], page 48
- Section 2.1.13 [Lyrics], page 104
- Section 2.1.17 [PianoStaff], page 129
- Section 2.1.10 [GrandStaff], page 82
- Section 2.1.18 [RhythmicStaff], page 130
- Section 2.1.5 [DrumStaff], page 62

This context is built from the following engraver(s):

- Section 2.2.111 [System_start_delimiter_engraver], page 240

Create a system start delimiter (i.e., a SystemStartBar, SystemStartBrace, SystemStartBracket or SystemStartSquare spanner).

Properties (read)

- 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.

- currentCommandColumn (layout object)
  - Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

This engraver creates the following layout object(s):

- Section 3.1.105 [SystemStartBar], page 337
- Section 3.1.106 [SystemStartBrace], page 338
- Section 3.1.107 [SystemStartBracket], page 339
- Section 3.1.108 [SystemStartSquare], page 340

2.1.2 ChordNames

Typesets chord names.

This context creates the following layout object(s):

- Section 3.1.24 [ChordName], page 274
- Section 3.1.95 [StaffSpacing], page 329
- Section 3.1.125 [VerticalAxisGroup], page 355

This context sets the following properties:
• Set grob-property `remove-empty` in Section 3.1.125 [VerticalAxisGroup], page 355 to `#t`.
• Set grob-property `remove-first` in Section 3.1.125 [VerticalAxisGroup], page 355 to `#t`.
• Set grob-property `minimum-Y-extent` in Section 3.1.125 [VerticalAxisGroup], page 355 to `'(0 . 2)`.

This context is a ‘bottom’ context; it cannot contain other contexts.

This context is built from the following engraver(s):

**Section 2.2.45 [Hara_kiri_engraver], page 221**

Like `Axis_group_engraver`, but make a hara-kiri spanner, and add interesting items (i.e., note heads, lyric syllables, and normal rests).

Properties (read)

`keepAliveInterfaces` (list)
A list of symbols, signifying grob interfaces that are worth keeping a staff with `remove-empty` set around for.

This engraver creates the following layout object(s):

Section 3.1.125 [VerticalAxisGroup], page 355.

**Section 2.2.94 [Skip_event_swallow_translator], page 236**

Swallow `\skip`.

**Section 2.2.14 [Chord_name_engraver], page 211**

Catch note events and generate the appropriate chordname.

Music types accepted:

Section 1.2.34 [note-event], page 39

Properties (read)

`chordChanges` (boolean)
Only show changes in chords scheme?

`chordNameExceptions` (list)
An alist of chord exceptions. Contains `(chord . markup)` entries.

`chordNameFunction` (procedure)
The function that converts lists of pitches to chord names.

`chordNoteNamer` (procedure)
A function that converts from a pitch object to a text markup. Used for single pitches.

`chordRootNamer` (procedure)
A function that converts from a pitch object to a text markup. Used for chords.

`chordNameExceptions` (list)
An alist of chord exceptions. Contains `(chord . markup)` entries.

`majorSevenSymbol` (markup)
How should the major 7th be formatted in a chord name?
This engraver creates the following layout object(s):
Section 3.1.24 [ChordName], page 274.

Section 2.2.93 [Separating_line_group_engraver], page 236
Generate objects for computing spacing parameters.
Properties (read)

createSpacing (boolean)
Create StaffSpacing objects? Should be set for staves.

Properties (write)

hasStaffSpacing (boolean)
True if the current CommandColumn contains items that will affect spacing.

This engraver creates the following layout object(s):
Section 3.1.95 [StaffSpacing], page 329.

Section 2.2.72 [Output_property_engraver], page 229
Apply a procedure to any grob acknowledged.
Music types accepted:
Section 1.2.4 [apply-output-event], page 35

Section 2.2.88 [Rest_swallow_translator], page 235
Swallow rest.

2.1.3 CueVoice
Corresponds to a voice on a staff. This context handles the conversion of dynamic signs, stems, beams, super- and subscripts, slurs, ties, and rests.

You have to instantiate this explicitly if you want to have multiple voices on the same staff.
This context also accepts commands for the following context(s):
Voice.
This context creates the following layout object(s):
Section 3.1.9 [Arpeggio], page 263, Section 3.1.19 [Beam], page 270, Section 3.1.20 [BendAfter], page 271, Section 3.1.23 [BreathingSign], page 273, Section 3.1.27 [ClusterSpannerBeacon], page 276, Section 3.1.26 [ClusterSpanner], page 276, Section 3.1.28 [CombineTextScript], page 277, Section 3.1.31 [Dots], page 279, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.34 [DynamicLineSpanner], page 282, Section 3.1.36 [DynamicTextSpanner], page 284, Section 3.1.35 [DynamicText], page 283, Section 3.1.37 [Fingering], page 285, Section 3.1.39 [Glissando], page 288, Section 3.1.43 [Hairpin], page 290, Section 3.1.47 [InstrumentSwitch], page 294, Section 3.1.51 [LaissezVibrerTieColumn], page 297, Section 3.1.50 [LaissezVibrerTie], page 297, Section 3.1.54 [LigatureBracket], page 299, Section 3.1.64 [MultiMeasureRestNumber], page 306, Section 3.1.65 [MultiMeasureRestText], page 307, Section 3.1.63 [MultiMeasureRest], page 305, Section 3.1.68 [NoteColumn], page 310, Section 3.1.69 [NoteHead], page 310, Section 3.1.71 [NoteSpacing], page 311, Section 3.1.77 [PercentRepeatCounter], page 315, Section 3.1.76 [PercentRepeat], page 315, Section 3.1.78 [PhrasingSlur], page 317, Section 3.1.81 [RepeatSlash], page 320, Section 3.1.83 [RepeatTieColumn], page 321, Section 3.1.82 [RepeatTie], page 320, Section 3.1.84 [Rest], page 321, Section 3.1.87 [ScriptColumn], page 323, Section 3.1.86 [Script], page 322, Section 3.1.90 [Slur], page 324, Section 3.1.99 [StemTremolo], page 332, Section 3.1.98 [Stem], page 330, Section 3.1.100 [StringNumber], page 332,
Section 3.1.101 [StrokeFinger], page 334, Section 3.1.110 [TextScript], page 341, Section 3.1.111 [TextSpanner], page 343, Section 3.1.113 [TieColumn], page 345, Section 3.1.112 [Tie], page 344, Section 3.1.115 [TrillPitchAccidental], page 346, Section 3.1.116 [TrillPitchGroup], page 347, Section 3.1.117 [TrillPitchHead], page 348, Section 3.1.118 [TrillSpanner], page 349, Section 3.1.119 [TupleBracket], page 350, Section 3.1.120 [TupleNumber], page 351 and Section 3.1.126 [VoiceFollower], page 356.

This context sets the following properties:

- Set grob-property thickness in Section 3.1.19 [Beam], page 270 to 0.35.
- Set grob-property length-fraction in Section 3.1.19 [Beam], page 270 to 0.629960524947437.
- Set grob-property length-fraction in Section 3.1.98 [Stem], page 330 to 0.629960524947437.
- Set translator property fontSize to -4.
- Set translator property localKeySignature to '()'.

This context is a ‘bottom’ context; it cannot contain other contexts.

This context is built from the following engraver(s):

Section 2.2.94 [Skip_event_swallow_translator], page 236
Swallow \skip.

Section 2.2.49 [Instrument_switch_engraver], page 222
Create a cue text for taking instrument.
Properties (read)

instrumentCueName (markup)
The name to print if another instrument is to be taken.

This engraver creates the following layout object(s):
Section 3.1.47 [InstrumentSwitch], page 294.

Section 2.2.40 [Grace_engraver], page 219
Set font size and other properties for grace notes.
Properties (read)

graceSettings (list)
Overrides for grace notes. This property should be manipulated through the add-grace-property function.

Section 2.2.125 [Tuplet_engraver], page 244
Catch tuplet events and generate appropriate bracket.
Music types accepted:
Section 1.2.65 [tuplet-span-event], page 42
Properties (read)

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.
This engraver creates the following layout object(s):
Section 3.1.119 [TupletBracket], page 350 and Section 3.1.120 [Tuplet-Number], page 351.

Section 2.2.118 [Tie_engraver], page 242
Generate ties between note heads of equal pitch.
Music types accepted:
Section 1.2.61 [tie-event], page 42
Properties (read)

  tieWaitForNote (boolean)
  If true, tied notes do not have to follow each other directly. This can be used for writing out arpeggios.

Properties (write)

  tieMelismaBusy (boolean)
  Signal whether a tie is present.

This engraver creates the following layout object(s):
Section 3.1.112 [Tie], page 344 and Section 3.1.113 [TieColumn], page 345.

Section 2.2.96 [Slur_engraver], page 237
Build slur grobs from slur events.
Music types accepted:
Section 1.2.48 [slur-event], page 40
Properties (read)

  slurMelismaBusy (boolean)
  Signal if a slur is present.

  doubleSlurs (boolean)
  If set, two slurs are created for every slurred note, one above and one below the chord.

This engraver creates the following layout object(s):
Section 3.1.90 [Slur], page 324.

Section 2.2.17 [Cluster_spanner_engraver], page 212
Engrave a cluster using Spanner notation.
Music types accepted:
Section 1.2.13 [cluster-note-event], page 36
This engraver creates the following layout object(s):
Section 3.1.26 [ClusterSpanner], page 276 and Section 3.1.27 [ClusterSpannerBeacon], page 276.

Section 2.2.78 [Phrasing_slur_engraver], page 232
Print phrasing slurs. Similar to Section 2.2.96 [Slur_engraver], page 237.
Music types accepted:
Section 1.2.42 [phrasing-slur-event], page 40
This engraver creates the following layout object(s):
Section 3.1.78 [PhrasingSlur], page 317.
Section 2.2.101 [Spanner_break_forbid_ engraver], page 238
Forbid breaks in certain spanners.

Section 2.2.69 [Note_spacing_ engraver], page 229
Generate NoteSpacing, an object linking horizontal lines for use in spacing.

This engraver creates the following layout object(s):
Section 3.1.71 [NoteSpacing], page 311.

Section 2.2.89 [Rhythmic_column_ engraver], page 235
Generate NoteColumn, an object that groups stems, note heads, and rests.

This engraver creates the following layout object(s):
Section 3.1.68 [NoteColumn], page 310.

Section 2.2.90 [Script_column_ engraver], page 235
Find potentially colliding scripts and put them into a ScriptColumn object; that will fix the collisions.

This engraver creates the following layout object(s):
Section 3.1.87 [ScriptColumn], page 323.

Section 2.2.91 [Script_ engraver], page 235
Handle note scripted articulations.

Music types accepted:
Section 1.2.6 [articulation-event], page 36

Properties (read)

\begin{verbatim}
scriptDefinitions (list)
The description of scripts. This is used by the Script_ engraver for typesetting note-superscripts and subscripts. See `scm/script.scm` for more information.
\end{verbatim}

This engraver creates the following layout object(s):
Section 3.1.86 [Script], page 322.

Section 2.2.11 [Bend_ engraver], page 210
Create fall spanners.

Music types accepted:
Section 1.2.10 [bend-after-event], page 36

This engraver creates the following layout object(s):
Section 3.1.20 [BendAfter], page 271.

Section 2.2.34 [Fingering_ engraver], page 217
Create fingering scripts.

Music types accepted:
Section 1.2.57 [stroke-finger-event], page 42 and Section 1.2.18 [fingering-event], page 37

This engraver creates the following layout object(s):
Section 3.1.37 [Fingering], page 285.

Section 2.2.27 [Dynamic_align_ engraver], page 215
Align hairpins and dynamic texts on a horizontal line

Properties (read)
currentMusicalColumn (layout object)
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):
Section 3.1.34 [DynamicLineSpanner], page 282.

Section 2.2.63 [New_dynamic_engraver], page 227
Create hairpins, dynamic texts, and their vertical alignments. The symbols are collected onto a DynamicLineSpanner grob which takes care of vertical positioning.
Music types accepted:
Section 1.2.53 [span-dynamic-event], page 41 and Section 1.2.2 [absolute-dynamic-event], page 35
Properties (read)
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.’.
currentMusicalColumn (layout object)
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).
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.’.

This engraver creates the following layout object(s):
Section 3.1.35 [DynamicText], page 283, Section 3.1.36 [DynamicTextSpanner], page 284 and Section 3.1.43 [Hairpin], page 290.

Section 2.2.116 [Text_engraver], page 241
Create text scripts.
Music types accepted:
Section 1.2.59 [text-script-event], page 42
This engraver creates the following layout object(s):
Section 3.1.110 [TextScript], page 341.

Section 2.2.76 [Part_combine_engraver], page 231
Part combine engraver for orchestral scores: Print markings ‘a2’, ‘Solo’, ‘Solo II’, and ‘unisono’.
Music types accepted:
Section 1.2.38 [part-combine-event], page 39
Properties (read)
printPartCombineTexts (boolean)
   Set ‘Solo’ and ‘A due’ texts in the part combiner?

soloText (markup)
   The text for the start of a solo when part-combining.

soloIIIText (markup)
   The text for the start of a solo for voice ‘two’ when part-combining.

aDueText (markup)
   Text to print at a unisono passage.

This engraver creates the following layout object(s):
Section 3.1.28 [CombineTextScript], page 277.

Section 2.2.95 [Slash_repeat_engraver], page 236
Make beat repeats.
Music types accepted:
Section 1.2.40 [percent-event], page 40
Properties (read)

measureLength (moment)
   Length of one measure in the current time signature.

This engraver creates the following layout object(s):
Section 3.1.81 [RepeatSlash], page 320.

Section 2.2.77 [Percent_repeat_engraver], page 231
Make whole bar and double bar repeats.
Music types accepted:
Section 1.2.40 [percent-event], page 40
Properties (read)

countPercentRepeats (boolean)
   If set, produce counters for percent repeats.

currentCommandColumn (layout object)
   Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

measureLength (moment)
   Length of one measure in the current time signature.

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.

Properties (write)

forbidBreak (boolean)
   If set to ##t, prevent a line break at this point.
This engraver creates the following layout object(s):
Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.76 [PercentRepeat], page 315 and Section 3.1.77 [PercentRepeatCounter], page 315.

**Section 2.2.15 [Chord_tremolo_engraver], page 211**
Generate beams for tremolo repeats.
Music types accepted:
Section 1.2.63 [tremolo-span-event], page 42
This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

**Section 2.2.64 [New_fingering_engraver], page 227**
Create fingering scripts for notes in a new chord. This engraver is ill-named, since it also takes care of articulations and harmonic note heads.
Properties (read)

  - **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.

  - **harmonicDots** (boolean)
    If set, harmonic notes in dotted chords get dots.

  - **strokeFingerOrientations** (list)
    See fingeringOrientations.

  - **stringNumberOrientations** (list)
    See fingeringOrientations.

This engraver creates the following layout object(s):
Section 3.1.37 [Fingering], page 285, Section 3.1.86 [Script], page 322, Section 3.1.100 [StringNumber], page 332 and Section 3.1.101 [StrokeFinger], page 334.

**Section 2.2.4 [Auto_beam_engraver], page 207**
Generate beams based on measure characteristics and observed Stems. Uses beatLength, measureLength, and measurePosition to decide when to start and stop a beam. Overriding beaming is done through Section 2.2.107 [Stem_engraver], page 239 properties stemLeftBeamCount and stemRightBeamCount.
Music types accepted:
Section 1.2.9 [beam-forbid-event], page 36
Properties (read)

  - **autoBeaming** (boolean)
    If set to true then beams are generated automatically.

  - **autoBeamSettings** (list)
    Specifies when automatically generated beams should begin and end. See Section “Setting automatic beam behavior” in Notation Reference for more information.
beatLength (moment)
The length of one beat in this time signature.

subdivideBeams (boolean)
If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.39 [Grace_beam_engraver], page 219
Handle Beam events by engraving beams. If omitted, then notes are printed with flags instead of beams. Only engravess beams when we are at grace points in time.
Music types accepted:
Section 1.2.8 [beam-event], page 36
Properties (read)

beamMelismaBusy (boolean)
Signal if a beam is present.

beatLength (moment)
The length of one beat in this time signature.

subdivideBeams (boolean)
If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.9 [Beam_engraver], page 209
Handle Beam events by engraving beams. If omitted, then notes are printed with flags instead of beams.
Music types accepted:
Section 1.2.8 [beam-event], page 36
Properties (read)

beamMelismaBusy (boolean)
Signal if a beam is present.

beatLength (moment)
The length of one beat in this time signature.

subdivideBeams (boolean)
If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

Properties (write)

forbidBreak (boolean)
If set to #t, prevent a line break at this point.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.
Section 2.2.107 [Stem_engraver], page 239

Create stems and single-stem tremolos. It also works together with the beam engraver for overriding beaming.

Music types accepted:
Section 1.2.62 [tremolo-event], page 42

Properties (read)

- **tremoloFlags** (integer)
  The number of tremolo flags to add if no number is specified.

- **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**.

This engraver creates the following layout object(s):
Section 3.1.98 [Stem], page 330 and Section 3.1.99 [StemTremolo], page 332.

Section 2.2.126 [Tweak_engraver], page 244

Read the **tweaks** property from the originating event, and set properties.

Section 2.2.87 [Rest_engraver], page 234

Engrave rests.

Music types accepted:
Section 1.2.44 [rest-event], page 40

Properties (read)

- **middleCPosition** (number)
  The place of the middle C, measured in half staff-spaces. Usually determined by looking at **middleCClefPosition** and **middleCOffset**.

This engraver creates the following layout object(s):
Section 3.1.84 [Rest], page 321.

Section 2.2.24 [Dots_engraver], page 214

Create Section 3.1.31 [Dots], page 279 objects for Section 3.2.80 [rhythmic-head-interface], page 397s.

This engraver creates the following layout object(s):
Section 3.1.31 [Dots], page 279.

Section 2.2.66 [Note_heads_engraver], page 228

Generate note heads.

Music types accepted:
Section 1.2.34 [note-event], page 39

Properties (read)

- **middleCPosition** (number)
  The place of the middle C, measured in half staff-spaces. Usually determined by looking at **middleCClefPosition** and **middleCOffset**.
staffLineLayoutFunction (procedure)
    Layout of staff lines, traditional, or semitone.

This engraver creates the following layout object(s):
Section 3.1.69 [NoteHead], page 310.

Section 2.2.13 [Breathing_sign_engraver], page 210
Create a breathing sign.
Music types accepted:
Section 1.2.12 [breathing-event], page 36
This engraver creates the following layout object(s):
Section 3.1.23 [BreathingSign], page 273.

Section 2.2.54 [Ligature_bracket_engraver], page 224
Handle Ligature_events by engraving Ligature brackets.
Music types accepted:
Section 1.2.26 [ligature-event], page 38
This engraver creates the following layout object(s):
Section 3.1.54 [LigatureBracket], page 299.

Section 2.2.38 [Glissando_engraver], page 219
Engrave glissandi.
Music types accepted:
Section 1.2.19 [glissando-event], page 37
This engraver creates the following layout object(s):
Section 3.1.39 [Glissando], page 288.

Section 2.2.65 [Note_head_line_engraver], page 228
Engrave a line between two note heads, for example a glissando. If followVoice is set, staff switches also generate a line.
Properties (read)

    followVoice (boolean)
        If set, note heads are tracked across staff switches by a thin line.

This engraver creates the following layout object(s):
Section 3.1.39 [Glissando], page 288 and Section 3.1.126 [VoiceFollower], page 356.

Section 2.2.85 [Repeat_tie_engraver], page 234
Create repeat ties.
Music types accepted:
Section 1.2.43 [repeat-tie-event], page 40
This engraver creates the following layout object(s):
Section 3.1.82 [RepeatTie], page 320 and Section 3.1.83 [RepeatTieColumn], page 321.

Section 2.2.52 [Laissez_vibrer_engraver], page 223
Create laissez vibrer items.
Music types accepted:
Section 2.2.36 [Forbid_line_break_engraver], page 218
Forbid line breaks when note heads are still playing at some point.

Properties (read)

\texttt{busyGrobs} (list)
A queue of (\texttt{end-moment} . \GROB) cons cells.
This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Properties (write)

\texttt{forbidBreak} (boolean)
If set to \texttt{##t}, prevent a line break at this point.

Section 2.2.124 [Trill_spanner_engraver], page 244
Create trill spanner from an event.
Music types accepted:

Section 2.2.117 [Text_spanner_engraver], page 242
Create text spanner from an event.
Music types accepted:

Section 1.2.60 [text-span-event], page 42
This engraver creates the following layout object(s):

Section 3.1.111 [TextSpanner], page 343.

Section 2.2.62 [Multi_measure_rest_engraver], page 226
Engrave multi-measure rests that are produced with ‘R’. It reads measurePosition and internalBarNumber to determine what number to print over the Section 3.1.63 [MultiMeasureRest], page 305. Reads measureLength to determine whether it should use a whole rest or a breve rest to represent one measure.

Music types accepted:

Section 1.2.32 [multi-measure-text-event], page 38 and Section 1.2.31 [multi-measure-rest-event], page 38

Properties (read)

- internalBarNumber (integer)
  Contains the current bar number. This property is used for internal timekeeping, among others by the Accidental_engraver.

- restNumberThreshold (number)
  If a multimeasure rest has more measures than this, a number is printed.

- breakableSeparationItem (layout object)
  The breakable items in this time step, for this staff.

- currentCommandColumn (layout object)
  Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

- measurePosition (moment)
  How much of the current measure have we had. This can be set manually to create incomplete measures.

- measureLength (moment)
  Length of one measure in the current time signature.

This engraver creates the following layout object(s):

Section 3.1.63 [MultiMeasureRest], page 305, Section 3.1.64 [MultiMeasureRestNumber], page 306 and Section 3.1.65 [MultiMeasureRestText], page 307.

Section 2.2.3 [Arpeggio_engraver], page 207
Generate an Arpeggio symbol.

Music types accepted:

Section 1.2.5 [arpeggio-event], page 35
This engraver creates the following layout object(s):

Section 3.1.9 [Arpeggio], page 263.
Section 2.2.72 [Output_property_engraver], page 229
Apply a procedure to any grob acknowledged.

Music types accepted:
Section 1.2.4 [apply-output-event], page 35

Section 2.2.83 [Pitched_trill_engraver], page 233
Print the bracketed note head after a note head with trill.
This engraver creates the following layout object(s):
Section 3.1.115 [TrillPitchAccidental], page 346, Section 3.1.116 [Trill-PitchGroup], page 347 and Section 3.1.117 [TrillPitchHead], page 348.

Section 2.2.35 [Font_size_engraver], page 217
Put fontSize into font-size grob property.
Properties (read)

\[
\text{fontSize (number)}
\]
The relative size of all grobs in a context.

2.1.4 Devnull
Silently discards all musical information given to this context.
This context also accepts commands for the following context(s):
Voice and Staff.
This context creates the following layout object(s):
none.
This context is a ‘bottom’ context; it cannot contain other contexts.
This context is built from the following engraver(s):

Section 2.2.109 [Swallow_engraver], page 240
This engraver swallows everything given to it silently. The purpose of this is to prevent spurious ‘event junked’ warnings.

2.1.5 DrumStaff
Handles typesetting for percussion.
This context also accepts commands for the following context(s):
Staff.
This context creates the following layout object(s):
Section 3.1.11 [BarLine], page 264, Section 3.1.15 [BassFigureAlignmentPositioning], page 268, Section 3.1.14 [BassFigureAlignment], page 268, Section 3.1.16 [BassFigureBracket], page 269, Section 3.1.17 [BassFigureContinuation], page 269, Section 3.1.18 [BassFigureLine], page 270, Section 3.1.13 [BassFigure], page 267, Section 3.1.25 [Clef], page 275, Section 3.1.30 [DotColumn], page 279, Section 3.1.46 [InstrumentName], page 293, Section 3.1.52 [Ledger-LineSpanner], page 298, Section 3.1.67 [NoteCollision], page 309, Section 3.1.72 [OctavateEight], page 311, Section 3.1.85 [RestCollision], page 322, Section 3.1.88 [ScriptRow], page 323, Section 3.1.92 [SostenutoPedalLineSpanner], page 326, Section 3.1.95 [StaffSpacing], page 329, Section 3.1.96 [StaffSymbol], page 329, Section 3.1.103 [SustainPedalLineSpanner], page 335, Section 3.1.114 [TimeSignature], page 345, Section 3.1.122 [UnaCordaPedalLineSpanner], page 353 and Section 3.1.125 [VerticalAxisGroup], page 355.

This context sets the following properties:
• Set grob-property staff-padding in Section 3.1.86 [Script], page 322 to 0.75.
• Set translator property clefPosition to 0.
• Set translator property `clefGlyph` to "clefs.percussion".
• Set translator property `shortInstrumentName` to '()'.
• Set translator property `instrumentName` to '()'.
• Set grob-property `minimum-Y-extent` in Section 3.1.125 [VerticalAxisGroup], page 355 to '(-4 . 4).
• Set translator property `ignoreFiguredBassRest` to #t.
• Set translator property `createSpacing` to #t.
• Set translator property `localKeySignature` to '()'.

Context DrumStaff can contain Section 2.1.6 [DrumVoice], page 67 and Section 2.1.3 [CueVoice], page 50.

This context is built from the following engraver(s):

Section 2.2.92 [Script_row_engraver], page 236
Determine order in horizontal side position elements.
This engraver creates the following layout object(s):
Section 3.1.88 [ScriptRow], page 323.

Section 2.2.33 [Figured_bass_position_engraver], page 217
Position figured bass alignments over notes.
This engraver creates the following layout object(s):
Section 3.1.15 [BassFigureAlignmentPositioning], page 268.

Section 2.2.32 [Figured_bass_engraver], page 216
Make figured bass numbers.
Music types accepted:
Section 1.2.7 [bass-figure-event], page 36 and Section 1.2.44 [rest-event], page 40
Properties (read)

`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.

`implicitBassFigures` (list)
A list of bass figures that are not printed as numbers, but only as extender lines.

`useBassFigureExtenders` (boolean)
Whether to use extender lines for repeated bass figures.

`ignoreFiguredBassRest` (boolean)
Don’t swallow rest events.
This engraver creates the following layout object(s):
Section 3.1.13 [BassFigure], page 267, Section 3.1.14 [BassFigure-Alignment], page 268, Section 3.1.16 [BassFigureBracket], page 269, Section 3.1.17 [BassFigureContinuation], page 269 and Section 3.1.18 [BassFigureLine], page 270.

Section 2.2.5 [Axis_group_engraver], page 208
Group all objects created in this context in a VerticalAxisGroup spanner.

Properties (read)

\texttt{currentCommandColumn (layout object)}
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

This engraver creates the following layout object(s):
Section 3.1.125 [VerticalAxisGroup], page 355.

Section 2.2.48 [Instrument_name_engraver], page 221
Create a system start text for instrument or vocal names.

Properties (read)

\texttt{currentCommandColumn (layout object)}
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

\texttt{shortInstrumentName (markup)}
See \texttt{instrument}.

\texttt{instrumentName (markup)}
The name to print left of a staff. The \texttt{instrument} property labels the staff in the first system, and the \texttt{instr} property labels following lines.

\texttt{shortVocalName (markup)}
Name of a vocal line, short version.

\texttt{vocalName (markup)}
Name of a vocal line.

This engraver creates the following layout object(s):
Section 3.1.46 [InstrumentName], page 293.

Section 2.2.79 [Piano_pedal_align_engraver], page 232
Align piano pedal symbols and brackets.

Properties (read)

\texttt{currentCommandColumn (layout object)}
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

This engraver creates the following layout object(s):
Section 3.1.92 [SostenutoPedalLineSpanner], page 326, Section 3.1.103 [SustainPedalLineSpanner], page 335 and Section 3.1.122 [UnaCordaPedalLineSpanner], page 353.
Section 2.2.86 [Rest_collision_engraver], page 234
Handle collisions of rests.
Properties (read)

`busyGros (list)`
A queue of `(end-moment . GROB)` cons cells. This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

This engraver creates the following layout object(s):
Section 3.1.85 [RestCollision], page 322.

Section 2.2.44 [Grob_pq_engraver], page 220
Administrate when certain grobs (e.g., note heads) stop playing.
Properties (read)

`busyGros (list)`
A queue of `(end-moment . GROB)` cons cells. This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Properties (write)

`busyGros (list)`
A queue of `(end-moment . GROB)` cons cells. This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Section 2.2.18 [Collision_engraver], page 212
Collect NoteColumns, and as soon as there are two or more, put them in a NoteCollision object.
This engraver creates the following layout object(s):
Section 3.1.67 [NoteCollision], page 309.

Section 2.2.104 [Staff_symbol_engraver], page 238
Create the constellation of five (default) staff lines.
Music types accepted:
Section 1.2.55 [staff-span-event], page 41
This engraver creates the following layout object(s):
Section 3.1.96 [StaffSymbol], page 329.

Section 2.2.53 [Ledger_line_engraver], page 223
Create the spanner to draw ledger lines, and notices objects that need ledger lines.
This engraver creates the following layout object(s):
Section 3.1.52 [LedgerLineSpanner], page 298.

Section 2.2.120 [Time_signature_engraver], page 243
Create a Section 3.1.114 [TimeSignature], page 345 whenever timeSignatureFraction changes.
Properties (read)
Chapter 2: Translation

implicitTimeSignatureVisibility (vector)
break visibility for the default time signature.

timeSignatureFraction (pair of numbers)
A pair of numbers, signifying the time signature. For example, #'(4 . 4) is a 4/4 time signature.

This engraver creates the following layout object(s):
Section 3.1.114 [TimeSignature], page 345.

Section 2.2.16 [Clef_engraver], page 212
Determine and set reference point for pitches.
Properties (read)

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.

explicitClefVisibility (vector)
‘break-visibility’ function for clef changes.

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.

This engraver creates the following layout object(s):
Section 3.1.25 [Clef], page 275 and Section 3.1.72 [OctavateEight], page 311.

Section 2.2.102 [Staff_collecting_engraver], page 238
Maintain the stavesFound variable.
Properties (read)

stavesFound (list of grobs)
A list of all staff-symbols found.

Properties (write)

stavesFound (list of grobs)
A list of all staff-symbols found.

Section 2.2.23 [Dot_column_engraver], page 214
Engrave dots on dotted notes shifted to the right of the note. If omitted, then dots appear on top of the notes.
This engraver creates the following layout object(s):
Section 3.1.30 [DotColumn], page 279.

Section 2.2.93 [Separating_line_group_engraver], page 236
Generate objects for computing spacing parameters.
Properties (read)
createSpacing (boolean)
Create StaffSpacing objects? Should be set for staves.

Properties (write)

hasStaffSpacing (boolean)
True if the current CommandColumn contains items that will affect spacing.

This engraver creates the following layout object(s):
Section 3.1.95 [StaffSpacing], page 329.

Section 2.2.35 [Font_size_ engraver], page 217
Put fontSize into font-size grob property.

Properties (read)

fontSize (number)
The relative size of all grobs in a context.

Section 2.2.7 [Bar_ engraver], page 208
Create barlines. This engraver is controlled through the whichBar property. If it has no bar line to create, it will forbid a linebreak at this point.

Properties (read)

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 Section “barline-interface” in Internals Reference.

Properties (write)

forbidBreak (boolean)
If set to ##t, prevent a line break at this point.

This engraver creates the following layout object(s):
Section 3.1.11 [BarLine], page 264.

Section 2.2.72 [Output_property_ engraver], page 229
Apply a procedure to any grob acknowledged.
Music types accepted:
Section 1.2.4 [apply-output-event], page 35

2.1.6 DrumVoice
A voice on a percussion staff.
This context also accepts commands for the following context(s):
Voice.
This context creates the following layout object(s):
Section 3.1.19 [Beam], page 270, Section 3.1.20 [BendAfter], page 271, Section 3.1.23 [BreathingSign], page 273, Section 3.1.28 [CombineTextScript], page 277, Section 3.1.31 [Dots],
This context sets the following properties:

- Set translator property `localKeySignature` to `()`. This context is a ‘bottom’ context; it cannot contain other contexts.

This context is built from the following engraver(s):

- **Section 2.2.94** [Skip_event_swallow_translator], page 236
  
  Swallow \skip.

- **Section 2.2.26** [Drum_notes_engraver], page 215

  Generate drum note heads.

  Music types accepted:
  
  Section 1.2.34 [note-event], page 39

  Properties (read)

  `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.

  This engraver creates the following layout object(s):

  Section 3.1.69 [NoteHead], page 310 and Section 3.1.86 [Script], page 322.

- **Section 2.2.44** [Grob_pq_engraver], page 220

  Administrate when certain grobs (e.g., note heads) stop playing.

  Properties (read)

  `busyGrobs` (list)

  A queue of `end-moment . GROB` cons cells. This is for internal (C++) use only. This prop-
Chapter 2: Translation

Property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Properties (write)

`busyGrobs` (list)
A queue of (end-moment . GROB) cons cells. This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Section 2.2.94 [Skip_event_swallow_translator], page 236
Swallow \skip.

Section 2.2.49 [Instrument_switch-engraver], page 222
Create a cue text for taking instrument.

Properties (read)

`instrumentCueName` (markup)
The name to print if another instrument is to be taken.

This engraver creates the following layout object(s):
Section 3.1.47 [InstrumentSwitch], page 294.

Section 2.2.40 [Grace_engraver], page 219
Set font size and other properties for grace notes.

Properties (read)

`graceSettings` (list)
Overrides for grace notes. This property should be manipulated through the add-grace-property function.

Section 2.2.125 [Tuplet_engraver], page 244
Catch tuplet events and generate appropriate bracket.

Music types accepted:
Section 1.2.65 [tuplet-span-event], page 42

Properties (read)

`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.

This engraver creates the following layout object(s):
Section 3.1.119 [TupletBracket], page 350 and Section 3.1.120 [Tuplet-Number], page 351.

Section 2.2.118 [Tie_engraver], page 242
Generate ties between note heads of equal pitch.

Music types accepted:
Section 1.2.61 [tie-event], page 42

Properties (read)
tieWaitForNote (boolean)
If true, tied notes do not have to follow each
other directly. This can be used for writing out
arpeggios.

Properties (write)

tieMelismaBusy (boolean)
Signal whether a tie is present.

This engraver creates the following layout object(s):
Section 3.1.112 [Tie], page 344 and Section 3.1.113 [TieColumn],
page 345.

Section 2.2.96 [Slur_engraver], page 237
Build slur grobs from slur events.
Music types accepted:
Section 1.2.48 [slur-event], page 40
Properties (read)

slurMelismaBusy (boolean)
Signal if a slur is present.

doubleSlurs (boolean)
If set, two slurs are created for every slurred
note, one above and one below the chord.

This engraver creates the following layout object(s):
Section 3.1.90 [Slur], page 324.

Section 2.2.78 [Phrasing_slur_engraver], page 232
Print phrasing slurs. Similar to Section 2.2.96 [Slur_engraver], page 237.
Music types accepted:
Section 1.2.42 [phrasing-slur-event], page 40
This engraver creates the following layout object(s):
Section 3.1.78 [PhrasingSlur], page 317.

Section 2.2.101 [Spanner_break_forbid_engraver], page 238
Forbid breaks in certain spanners.

Section 2.2.69 [Note_spacing_engraver], page 229
Generate NoteSpacing, an object linking horizontal lines for use in
spacing.
This engraver creates the following layout object(s):
Section 3.1.71 [NoteSpacing], page 311.

Section 2.2.89 [Rhythmic_column_engraver], page 235
Generate NoteColumn, an object that groups stems, note heads, and
rests.
This engraver creates the following layout object(s):
Section 3.1.68 [NoteColumn], page 310.

Section 2.2.90 [Script_column_engraver], page 235
Find potentially colliding scripts and put them into a ScriptColumn
object; that will fix the collisions.
This engraver creates the following layout object(s):
Section 3.1.87 [ScriptColumn], page 323.
Section 2.2.91 [Script-engraver], page 235
Handle note scripted articulations.

Music types accepted:
Section 1.2.6 [articulation-event], page 36
Properties (read)

scriptDefinitions (list)
The description of scripts. This is used by the Script-engraver for typesetting note-superscripts and subscripts. See ‘scm/script.scm’ for more information.

This engraver creates the following layout object(s):
Section 3.1.86 [Script], page 322.

Section 2.2.11 [Bend-engraver], page 210
Create fall spanners.
Music types accepted:
Section 1.2.10 [bend-after-event], page 36
This engraver creates the following layout object(s):
Section 3.1.20 [BendAfter], page 271.

Section 2.2.27 [Dynamic-align-engraver], page 215
Align hairpins and dynamic texts on a horizontal line
Properties (read)

currentMusicalColumn (layout object)
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):
Section 3.1.34 [DynamicLineSpanner], page 282.

Section 2.2.63 [New-dynamic-engraver], page 227
Create hairpins, dynamic texts, and their vertical alignments. The symbols are collected onto a DynamicLineSpanner grob which takes care of vertical positioning.

Music types accepted:
Section 1.2.53 [span-dynamic-event], page 41 and Section 1.2.2 [absolute-dynamic-event], page 35
Properties (read)

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.’.

currentMusicalColumn (layout object)
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).
**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.’.

This engraver creates the following layout object(s):
- Section 3.1.35 [DynamicText], page 283
- Section 3.1.36 [DynamicTextSpanner], page 284
- Section 3.1.43 [Hairpin], page 290

Section 2.2.116 [Text_engraver], page 241
Create text scripts.
Music types accepted:
- Section 1.2.59 [text-script-event], page 42
This engraver creates the following layout object(s):
- Section 3.1.110 [TextScript], page 341

Section 2.2.76 [Part_combine_engraver], page 231
Part combine engraver for orchestral scores: Print markings ‘a2’, ‘Solo’, ‘Solo II’, and ‘unisono’.
Music types accepted:
- Section 1.2.38 [part-combine-event], page 39
Properties (read)

  **printPartCombineTexts** (boolean)
  Set ‘Solo’ and ‘A due’ texts in the part combiner?

  **soloText** (markup)
The text for the start of a solo when part-combining.

  **soloIIText** (markup)
The text for the start of a solo for voice ‘two’ when part-combining.

  **aDueText** (markup)
Text to print at a unisono passage.

This engraver creates the following layout object(s):
- Section 3.1.28 [CombineTextScript], page 277

Section 2.2.95 [Slash_repeat_engraver], page 236
Make beat repeats.
Music types accepted:
- Section 1.2.40 [percent-event], page 40
Properties (read)

  **measureLength** (moment)
Length of one measure in the current time signature.
This engraver creates the following layout object(s):
Section 3.1.81 [RepeatSlash], page 320.

**Section 2.2.77 [Percent_repeat_engraver], page 231**
Make whole bar and double bar repeats.
Music types accepted:
Section 1.2.40 [percent-event], page 40
Properties (read)

- **countPercentRepeats** (boolean)
  If set, produce counters for percent repeats.
- **currentCommandColumn** (layout object)
  Grob that is X-parent to all current breakable (clef, key signature, etc.) items.
- **measureLength** (moment)
  Length of one measure in the current time signature.
- **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.

Properties (write)

- **forbidBreak** (boolean)
  If set to \$$\texttt{\#\#t}\$$, prevent a line break at this point.

This engraver creates the following layout object(s):
Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.76 [PercentRepeat], page 315 and Section 3.1.77 [PercentRepeatCounter], page 315.

**Section 2.2.15 [Chord_tremolo_engraver], page 211**
Generate beams for tremolo repeats.
Music types accepted:
Section 1.2.63 [tremolo-span-event], page 42
This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

**Section 2.2.4 [Auto_beam_engraver], page 207**
Generate beams based on measure characteristics and observed stems. Uses **beatLength**, **measureLength**, and **measurePosition** to decide when to start and stop a beam. Overriding beaming is done through Section 2.2.107 [Stem_engraver], page 239 properties **stemLeftBeamCount** and **stemRightBeamCount**.
Music types accepted:
Section 1.2.9 [beam-forbid-event], page 36
Properties (read)

- **autoBeaming** (boolean)
  If set to true then beams are generated automatically.
autoBeamSettings (list)
   Specifies when automatically generated beams should begin and end. See Section “Setting automatic beam behavior” in Notation Reference for more information.

beatLength (moment)
   The length of one beat in this time signature.

subdivideBeams (boolean)
   If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.39 [Grace_beam_engraver], page 219
Handle Beam events by engraving beams. If omitted, then notes are printed with flags instead of beams. Only engraves beams when we are at grace points in time.
Music types accepted:
Section 1.2.8 [beam-event], page 36
Properties (read)
   beamMelismaBusy (boolean)
      Signal if a beam is present.
   beatLength (moment)
      The length of one beat in this time signature.
   subdivideBeams (boolean)
      If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.9 [Beam_engraver], page 209
Handle Beam events by engraving beams. If omitted, then notes are printed with flags instead of beams.
Music types accepted:
Section 1.2.8 [beam-event], page 36
Properties (read)
   beamMelismaBusy (boolean)
      Signal if a beam is present.
   beatLength (moment)
      The length of one beat in this time signature.
   subdivideBeams (boolean)
      If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

Properties (write)
forbidBreak (boolean)
    If set to ##t, prevent a line break at this point.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.107 [Stem_ engraver], page 239
Create stems and single-stem tremolos. It also works together with the
beam engraver for overriding beaming.
Music types accepted:
Section 1.2.62 [tremolo-event], page 42
Properties (read)

tremoloFlags (integer)
    The number of tremolo flags to add if no num-
    ber is specified.

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.

This engraver creates the following layout object(s):
Section 3.1.98 [Stem], page 330 and Section 3.1.99 [StemTremolo],
page 332.

Section 2.2.126 [Tweak_ engraver], page 244
Read the tweaks property from the originating event, and set properties.

Section 2.2.87 [Rest_ engraver], page 234
Engrave rests.
Music types accepted:
Section 1.2.44 [rest-event], page 40
Properties (read)

middleCPosition (number)
    The place of the middle C, measured in half
    staff-spaces. Usually determined by looking at
    middleCClefPosition and middleCOffset.

This engraver creates the following layout object(s):
Section 3.1.84 [Rest], page 321.

Section 2.2.24 [Dots_ engraver], page 214
Create Section 3.1.31 [Dots], page 279 objects for Section 3.2.80
[rhythmic-head-interface], page 397s.
This engraver creates the following layout object(s):
Section 3.1.31 [Dots], page 279.

Section 2.2.13 [Breathing_sign_ engraver], page 210
Create a breathing sign.
Music types accepted:
Section 1.2.12 [breathing-event], page 36
This engraver creates the following layout object(s):
Section 3.1.23 [BreathingSign], page 273.

Section 2.2.85 [Repeat_tie_engraver], page 234
Create repeat ties.
Music types accepted:
Section 1.2.43 [repeat-tie-event], page 40
This engraver creates the following layout object(s):
Section 3.1.82 [RepeatTie], page 320 and Section 3.1.83 [RepeatTieCol-
umn], page 321.

Section 2.2.52 [Laissez_vibrer_engraver], page 223
Create laissez vibrer items.
Music types accepted:
Section 1.2.24 [laissez-vibrer-event], page 37
This engraver creates the following layout object(s):
Section 3.1.50 [LaissezVibrerTie], page 297 and Section 3.1.51 [Lais-
sezVibrerTieColumn], page 297.

Section 2.2.36 [Forbid_line_break_engraver], page 218
Forbid line breaks when note heads are still playing at some point.
Properties (read)

\begin{itemize}
  \item \textbf{busyGrobs} (list)
    A queue of \texttt{(end-moment . GROB)} cons cells.
    This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).
\end{itemize}

Properties (write)

\begin{itemize}
  \item \textbf{forbidBreak} (boolean)
    If set to \#\#, prevent a line break at this point.
\end{itemize}

Section 2.2.44 [Grob_pq_engraver], page 220
Administrate when certain grobs (e.g., note heads) stop playing.
Properties (read)

\begin{itemize}
  \item \textbf{busyGrobs} (list)
    A queue of \texttt{(end-moment . GROB)} cons cells.
    This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).
\end{itemize}

Properties (write)

\begin{itemize}
  \item \textbf{busyGrobs} (list)
    A queue of \texttt{(end-moment . GROB)} cons cells.
    This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).
\end{itemize}

Section 2.2.124 [Trill_spanner_engraver], page 244
Create trill spanner from an event.
Music types accepted:
Section 1.2.64 [trill-span-event], page 42

Properties (read)

\texttt{currentCommandColumn} (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

\texttt{currentMusicalColumn} (layout object)
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):
Section 3.1.118 [TrillSpanner], page 349.

Section 2.2.117 [Text_speller], page 242
Create text spanner from an event.

Music types accepted:
Section 1.2.60 [text-span-event], page 42
This engraver creates the following layout object(s):
Section 3.1.111 [TextSpanner], page 343.

Section 2.2.62 [Multi_measure_rest_engraver], page 226
Engrave multi-measure rests that are produced with ‘R’. It reads \texttt{measurePosition} and \texttt{internalBarNumber} to determine what number to print over the Section 3.1.63 [MultiMeasureRest], page 305. Reads \texttt{measureLength} to determine whether it should use a whole rest or a breve rest to represent one measure.

Music types accepted:
Section 1.2.32 [multi-measure-text-event], page 38 and Section 1.2.31 [multi-measure-rest-event], page 38

Properties (read)

\texttt{internalBarNumber} (integer)
Contains the current bar number. This property is used for internal timekeeping, among others by the Accidental_engraver.

\texttt{restNumberThreshold} (number)
If a multimeasure rest has more measures than this, a number is printed.

\texttt{breakableSeparationItem} (layout object)
The breakable items in this time step, for this staff.

\texttt{currentCommandColumn} (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

\texttt{measurePosition} (moment)
How much of the current measure have we had. This can be set manually to create incomplete measures.
measureLength (moment)

Length of one measure in the current time signature.

This engraver creates the following layout object(s):
Section 3.1.63 [MultiMeasureRest], page 305, Section 3.1.64 [MultiMeasureRestNumber], page 306 and Section 3.1.65 [MultiMeasureRestText], page 307.

Section 2.2.72 [Output_property_engraver], page 229
Apply a procedure to any grob acknowledged.
Music types accepted:
Section 1.2.4 [apply-output-event], page 35
Section 2.2.83 [Pitched_trill_engraver], page 233
Print the bracketed note head after a note head with trill.
This engraver creates the following layout object(s):
Section 3.1.115 [TrillPitchAccidental], page 346, Section 3.1.116 [TrillPitchGroup], page 347 and Section 3.1.117 [TrillPitchHead], page 348.

Section 2.2.35 [Font_size_engraver], page 217
Put \texttt{fontSize} into \texttt{font-size} grob property.
Properties (read)

\begin{itemize}
  \item \texttt{fontSize (number)}
  \end{itemize}

\texttt{fontSize} (number)
The relative size of all grobs in a context.

2.1.7 FiguredBass

(not documented)

This context creates the following layout object(s):
Section 3.1.14 [BassFigureAlignment], page 268, Section 3.1.16 [BassFigureBracket], page 269, Section 3.1.17 [BassFigureContinuation], page 269, Section 3.1.18 [BassFigureLine], page 270, Section 3.1.13 [BassFigure], page 267, Section 3.1.95 [StaffSpacing], page 329 and Section 3.1.125 [VerticalAxisGroup], page 355.

This context sets the following properties:

\begin{itemize}
  \item Set grob-property \texttt{minimum-Y-extent} in Section 3.1.125 [VerticalAxisGroup], page 355 to `(0 . 2).
  \item Set grob-property \texttt{remove-first} in Section 3.1.125 [VerticalAxisGroup], page 355 to \texttt{#t}.
  \item Set grob-property \texttt{remove-empty} in Section 3.1.125 [VerticalAxisGroup], page 355 to \texttt{#t}.
\end{itemize}

This context is a ‘bottom’ context; it cannot contain other contexts.
This context is built from the following engraver(s):

Section 2.2.45 [Hara_kiri_engraver], page 221
Like \texttt{Axis_group_engraver}, but make a hara-kiri spanner, and add interesting items (i.e., note heads, lyric syllables, and normal rests).
Properties (read)

\begin{itemize}
  \item \texttt{keepAliveInterfaces (list)}
  \end{itemize}

\texttt{keepAliveInterfaces} (list)
A list of symbols, signifying grob interfaces that are worth keeping a staff with \texttt{remove-empty} set around for.
This engraver creates the following layout object(s):
Section 3.1.125 [VerticalAxisGroup], page 355.

Section 2.2.93 [Separating_line_group_engraver], page 236
Generate objects for computing spacing parameters.
Properties (read)

createSpacing (boolean)
Create StaffSpacing objects? Should be set for staves.

Properties (write)

hasStaffSpacing (boolean)
True if the current CommandColumn contains items that will affect spacing.

This engraver creates the following layout object(s):
Section 3.1.95 [StaffSpacing], page 329.

Section 2.2.94 [Skip_event_swallow_translator], page 236
Swallow \skip.

Section 2.2.70 [Note_swallow_translator], page 229
Swallow notes.

Section 2.2.32 [Figured_bass_engraver], page 216
Make figured bass numbers.
Music types accepted:
Section 1.2.7 [bass-figure-event], page 36 and Section 1.2.44 [rest-event], page 40
Properties (read)

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.

implicitBassFigures (list)
A list of bass figures that are not printed as numbers, but only as extender lines.

useBassFigureExtenders (boolean)
Whether to use extender lines for repeated bass figures.

ignoreFiguredBassRest (boolean)
Don’t swallow rest events.

This engraver creates the following layout object(s):
2.1.8 FretBoards

(not documented)

This context creates the following layout object(s):

Section 3.1.38 [FretBoard], page 287, Section 3.1.46 [InstrumentName], page 293, Section 3.1.95 [StaffSpacing], page 329 and Section 3.1.125 [VerticalAxisGroup], page 355.

This context sets the following properties:

• Set translator property predefinedDiagramTable to #<hash-table 0/113>.

This context is a ‘bottom’ context; it cannot contain other contexts.

This context is built from the following engraver(s):

Section 2.2.48 [Instrument_name_engraver], page 221

Create a system start text for instrument or vocal names.

Properties (read)

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

shortInstrumentName (markup)
See 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.

shortVocalName (markup)
Name of a vocal line, short version.

vocalName (markup)
Name of a vocal line.

This engraver creates the following layout object(s):

Section 3.1.46 [InstrumentName], page 293.

Section 2.2.35 [Font_size_engraver], page 217
Put fontSize into font-size grob property.

Properties (read)

fontSize (number)
The relative size of all grobs in a context.

Section 2.2.93 [Separating_line_group_engraver], page 236
Generate objects for computing spacing parameters.

Properties (read)

createSpacing (boolean)
Create StaffSpacing objects? Should be set for staves.
Properties (write)

hasStaffSpacing (boolean)
True if the current CommandColumn contains items that will affect spacing.

This engraver creates the following layout object(s):
Section 3.1.95 [StaffSpacing], page 329.

Section 2.2.37 [Fretboard_engraver], page 218
Generate one or more tablature noteheads from event of type NoteEvent.
Music types accepted:
Section 1.2.56 [string-number-event], page 41 and Section 1.2.34 [note-event], page 39

Properties (read)

stringTunings (list)
The tablature strings tuning. It is a list of the pitch (in semitones) of each string (starting with the lower one).

minimumFret (number)
The tablature auto string-selecting mechanism selects the highest string with a fret at least minimumFret.

maximumFretStretch (number)
Don’t allocate frets further than this from specified frets.

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.

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.

predefinedDiagramTable (hash table)
The hash table of predefined fret diagrams to use in FretBoards.

This engraver creates the following layout object(s):
Section 3.1.38 [FretBoard], page 287.

Section 2.2.5 [Axis_group_engraver], page 208
Group all objects created in this context in a VerticalAxisGroup spanner.

Properties (read)

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.
This engraver creates the following layout object(s):
Section 3.1.125 [VerticalAxisGroup], page 355.

Section 2.2.72 [Output_property_engraver], page 229
Apply a procedure to any grob acknowledged.
Music types accepted:
Section 1.2.4 [apply-output-event], page 35

2.1.9 Global
Hard coded entry point for LilyPond. Cannot be tuned.
This context creates the following layout object(s):
none.
Context Global can contain Section 2.1.19 [Score], page 133.

2.1.10 GrandStaff
A group of staves, with a brace on the left side, grouping the staves together. The bar lines of the contained staves are connected vertically.
This context creates the following layout object(s):
Section 3.1.9 [Arpeggio], page 263, Section 3.1.94 [SpanBar], page 328, Section 3.1.105 [SystemStartBar], page 337, Section 3.1.106 [SystemStartBracket], page 338, Section 3.1.107 [SystemStartBrace], page 339 and Section 3.1.108 [SystemStartSquare], page 340.
This context sets the following properties:
• Set translator property systemStartDelimiter to 'SystemStartBrace.
• Set translator property localKeySignature to '().
Context GrandStaff can contain Section 2.1.20 [Staff], page 145 and Section 2.1.7 [FiguredBass], page 78.
This context is built from the following engraver(s):
Section 2.2.111 [System_start_delimiter_engraver], page 240
Create a system start delimiter (i.e., a SystemStartBar, SystemStartBracket, SystemStartBrace or SystemStartSquare spanner).
Properties (read)

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.

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

This engraver creates the following layout object(s):
Section 3.1.105 [SystemStartBar], page 337, Section 3.1.106 [SystemStartBracket], page 338, Section 3.1.107 [SystemStartBracket], page 339 and Section 3.1.108 [SystemStartSquare], page 340.
Section 2.2.99 [Span_arpeggio_engraver], page 238
Make arpeggios that span multiple staves.

Properties (read)

connectArpeggios (boolean)
  If set, connect arpeggios across piano staff.

This engraver creates the following layout object(s):
Section 3.1.9 [Arpeggio], page 263.

Section 2.2.100 [Span_bar_engraver], page 238
Make cross-staff bar lines: It catches all normal bar lines and draws a single span bar across them.

This engraver creates the following layout object(s):
Section 3.1.94 [SpanBar], page 328.

2.1.11 GregorianTranscriptionStaff
Handles clefs, bar lines, keys, accidentals. It can contain Voice contexts.

This context also accepts commands for the following context(s):
Staff.

This context creates the following layout object(s):
Section 3.1.2 [AccidentalCautionary], page 257, Section 3.1.3 [AccidentalPlacement], page 258, Section 3.1.4 [AccidentalSuggestion], page 259, Section 3.1.1 [Accidental], page 257, Section 3.1.11 [BarLine], page 264, Section 3.1.15 [BassFigureAlignmentPositioning], page 268, Section 3.1.14 [BassFigureAlignment], page 268, Section 3.1.16 [BassFigureBracket], page 269, Section 3.1.17 [BassFigureContinuation], page 269, Section 3.1.18 [BassFigureLine], page 270, Section 3.1.13 [BassFigure], page 267, Section 3.1.25 [Clef], page 275, Section 3.1.30 [DotColumn], page 279, Section 3.1.46 [InstrumentName], page 293, Section 3.1.48 [KeyCancellation], page 295, Section 3.1.49 [KeySignature], page 296, Section 3.1.52 [LedgerLineSpanner], page 298, Section 3.1.67 [NoteCollision], page 309, Section 3.1.72 [OctavateEight], page 311, Section 3.1.73 [OttavaBracket], page 312, Section 3.1.79 [PianoPedalBracket], page 318, Section 3.1.85 [RestCollision], page 322, Section 3.1.88 [ScriptRow], page 323, Section 3.1.92 [SostenutoPedalLineSpanner], page 326, Section 3.1.91 [SostenutoPedal], page 325, Section 3.1.95 [StaffSpacing], page 329, Section 3.1.96 [StaffSymbol], page 329, Section 3.1.103 [SustainPedalLineSpanner], page 335, Section 3.1.102 [SustainPedal], page 335, Section 3.1.114 [TimeSignature], page 345, Section 3.1.122 [UnaCordaPedalLineSpanner], page 353, Section 3.1.121 [UnaCordaPedal], page 352 and Section 3.1.125 [VerticalAxisGroup], page 355.

This context sets the following properties:
- Set grob-property transparent in Section 3.1.11 [BarLine], page 264 to #t.
- Set translator property shortInstrumentName to '().
- Set translator property instrumentName to '().
- Set grob-property minimum-Y-extent in Section 3.1.125 [VerticalAxisGroup], page 355 to '(-4 . 4).
- Set translator property ignoreFiguredBassRest to #t.
- Set translator property createSpacing to #t.
- Set translator property localKeySignature to '().

Context GregorianTranscriptionStaff can contain Section 2.1.12 [GregorianTranscriptionVoice], page 92 and Section 2.1.3 [CueVoice], page 50.

This context is built from the following engraver(s):
Section 2.2.92 [Script_row_engraver], page 236
Determine order in horizontal side position elements.
This engraver creates the following layout object(s):
Section 3.1.88 [ScriptRow], page 323.

Section 2.2.33 [Figured_bass_position_engraver], page 217
Position figured bass alignments over notes.
This engraver creates the following layout object(s):
Section 3.1.15 [BassFigureAlignmentPositioning], page 268.

Section 2.2.32 [Figured_bass_engraver], page 216
Make figured bass numbers.
Music types accepted:
Section 1.2.7 [bass-figure-event], page 36 and Section 1.2.44 [rest-event], page 40
Properties (read)

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.

implicitBassFigures (list)
A list of bass figures that are not printed as numbers, but only as extender lines.

useBassFigureExtenders (boolean)
Whether to use extender lines for repeated bass figures.

ignoreFiguredBassRest (boolean)
Don’t swallow rest events.

This engraver creates the following layout object(s):
Section 3.1.13 [BassFigure], page 267, Section 3.1.14 [BassFigureAlignment], page 268, Section 3.1.16 [BassFigureBracket], page 269, Section 3.1.17 [BassFigureContinuation], page 269 and Section 3.1.18 [BassFigureLine], page 270.

Section 2.2.5 [Axis_group_engraver], page 208
Group all objects created in this context in a VerticalAxisGroup spanner.
Properties (read)

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.
This engraver creates the following layout object(s):
Section 3.1.125 [VerticalAxisGroup], page 355.

Section 2.2.108 [String_number_engraver], page 240
Swallow string number events. The purpose of this engraver is to process tablatures for normal notation. To prevent warnings for unprocessed string number events to obscure real error messages, this engraver swallows them all.

Section 2.2.48 [Instrument_name_engraver], page 221
Create a system start text for instrument or vocal names.
Properties (read)

  currentCommandColumn (layout object)
    Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

  shortInstrumentName (markup)
    See 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.

  shortVocalName (markup)
    Name of a vocal line, short version.

  vocalName (markup)
    Name of a vocal line.

This engraver creates the following layout object(s):
Section 3.1.46 [InstrumentName], page 293.

Section 2.2.79 [Piano_pedal_align_engraver], page 232
Align piano pedal symbols and brackets.
Properties (read)

  currentCommandColumn (layout object)
    Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

This engraver creates the following layout object(s):
Section 3.1.92 [SostenutoPedalLineSpanner], page 326, Section 3.1.103 [SustainPedalLineSpanner], page 335 and Section 3.1.122 [UnaCordaPedalLineSpanner], page 353.

Section 2.2.80 [Piano_pedal_engraver], page 232
Engrave piano pedal symbols and brackets.
Music types accepted:
Section 1.2.66 [una-corda-event], page 43, Section 1.2.58 [sustain-event], page 42 and Section 1.2.51 [sostenuto-event], page 41
Properties (read)

  currentCommandColumn (layout object)
    Grob that is X-parent to all current breakable (clef, key signature, etc.) items.
#### 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.

This engraver creates the following layout object(s):

- Section 3.1.79 [PianoPedalBracket], page 318
- Section 3.1.91 [SostenutoPedal], page 325
- Section 3.1.102 [SustainPedal], page 335
- Section 3.1.121 [UnaCordaPedal], page 352

**Section 2.2.1 [Accidental_engraver], page 206**

Make accidentals. Catch note heads, ties and notices key-change events. This engraver usually lives at Staff level, but reads the settings for Accidental at Voice level, so you can \override them at Voice.

**Properties (read)**

#### 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 Section “Score” in Internals Reference then all staves share accidentals, and if context is Section “Staff” in Internals Reference 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.

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.

internalBarNumber (integer)
Contains the current barnumber. This property is used for internal timekeeping, among others by the Accidental engraver.

extraNatural (boolean)
Whether to typeset an extra natural sign before accidentals changing from a non-natural to another non-natural.

harmonicAccidentals (boolean)
If set, harmonic notes in chords get accidentals.

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)).

localKeySignature (list)
The key signature at this point in the measure. The format is the same as for keySignature, but can also contain ((octave . name) . (alter barnumber . measureposition)) pairs.

Properties (write)

localKeySignature (list)
The key signature at this point in the measure. The format is the same as for keySignature,
but can also contain \((\text{octave \ name}) \cdot (\text{alter \ barnumber \ measureposition})\) pairs.

This engraver creates the following layout object(s):
Section 3.1.1 [Accidental], page 257, Section 3.1.2 [AccidentalCautionary], page 257, Section 3.1.3 [AccidentalPlacement], page 258 and Section 3.1.4 [AccidentalSuggestion], page 259.

Section 2.2.86 [Rest_collision_engraver], page 234
Handle collisions of rests.

Properties (read)

\textbf{busyGrobs} (list)
A queue of (end-moment \ GROB) cons cells.
This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

This engraver creates the following layout object(s):
Section 3.1.85 [RestCollision], page 322.

Section 2.2.44 [Grob_pq_engraver], page 220
Administrate when certain grobs (e.g., note heads) stop playing.

Properties (read)

\textbf{busyGrobs} (list)
A queue of (end-moment \ GROB) cons cells.
This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Properties (write)

\textbf{busyGrobs} (list)
A queue of (end-moment \ GROB) cons cells.
This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Section 2.2.18 [Collision_engraver], page 212
Collect \texttt{NoteColumns}, and as soon as there are two or more, put them in a \texttt{NoteCollision} object.

This engraver creates the following layout object(s):
Section 3.1.67 [NoteCollision], page 309.

Section 2.2.104 [Staff_symbol_engraver], page 238
Create the constellation of five (default) staff lines.

Music types accepted:
Section 1.2.55 [staff-span-event], page 41
This engraver creates the following layout object(s):
Section 3.1.96 [StaffSymbol], page 329.

Section 2.2.53 [Ledger_line_engraver], page 223
Create the spanner to draw ledger lines, and notices objects that need ledger lines.

This engraver creates the following layout object(s):
Section 3.1.52 [LedgerLineSpanner], page 298.
Section 2.2.120 [Time_signature_engraver], page 243
Create a Section 3.1.114 [TimeSignature], page 345 whenever timeSignatureFraction changes.

Properties (read)

*implicitTimeSignatureVisibility* (vector)
break visibility for the default time signature.

*timeSignatureFraction* (pair of numbers)
A pair of numbers, signifying the time signature. For example, #'(4 . 4) is a 4/4 time signature.

This engraver creates the following layout object(s):
Section 3.1.114 [TimeSignature], page 345.

Section 2.2.50 [Key_engraver], page 222
Engrave a key signature.

Music types accepted:
Section 1.2.22 [key-change-event], page 37

Properties (read)

*createKeyOnClefChange* (boolean)
Print a key signature whenever the clef is changed.

*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.

*extraNatural* (boolean)
Whether to typeset an extra natural sign before accidentals changing from a non-natural to another non-natural.

*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)).

*lastKeySignature* (list)
Last key signature before a key signature change.
**printKeyCancellation** (boolean)
Print restoration alterations before a key signature change.

Properties (write)

**keySignature** (list)
The current key signature. This is an alist containing \((step . alter)\) or \((\text{octave} . step) . alter\), where \(step\) is a number in the range 0 to 6 and \(alter\) a fraction, denoting alteration. For alterations, use symbols, e.g.
\[
\text{keySignature} = \#\'(6 , \text{FLAT})\).

**lastKeySignature** (list)
Last key signature before a key signature change.

**tonic** (pitch)
The tonic of the current scale.

This engraver creates the following layout object(s):
Section 3.1.48 [KeyCancellation], page 295 and Section 3.1.49 [KeySignature], page 296.

**Section 2.2.16 [Clef_engraver], page 212**
Determine and set reference point for pitches.

Properties (read)

**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.

**explicitClefVisibility** (vector)
‘break-visibility’ function for clef changes.

**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.

This engraver creates the following layout object(s):
Section 3.1.25 [Clef], page 275 and Section 3.1.72 [OctavateEight], page 311.

**Section 2.2.71 [Ottava_spanner_engraver], page 229**
Create a text spanner when the ottavation property changes.

Properties (read)

**ottavation** (markup)
If set, the text for an ottava spanner. Changing this creates a new text spanner.
originalMiddleCPosition (integer)
Used for temporary overriding middle C in octavation brackets.

currentMusicalColumn (layout object)
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):
Section 3.1.73 [OttavaBracket], page 312.

Section 2.2.102 [Staff_collecting_engraver], page 238
Maintain the stavesFound variable.

Properties (read)

stavesFound (list of grobs)
A list of all staff-symbols found.

Properties (write)

stavesFound (list of grobs)
A list of all staff-symbols found.

Section 2.2.23 [Dot_column_engraver], page 214
Engrave dots on dotted notes shifted to the right of the note. If omitted, then dots appear on top of the notes.
This engraver creates the following layout object(s):
Section 3.1.30 [DotColumn], page 279.

Section 2.2.93 [Separating_line_group_engraver], page 236
Generate objects for computing spacing parameters.

Properties (read)

createSpacing (boolean)
Create StaffSpacing objects? Should be set for staves.

Properties (write)

hasStaffSpacing (boolean)
True if the current CommandColumn contains items that will affect spacing.

This engraver creates the following layout object(s):
Section 3.1.95 [StaffSpacing], page 329.

Section 2.2.35 [Font_size_engraver], page 217
Put fontSize into font-size grob property.

Properties (read)

fontSize (number)
The relative size of all grobs in a context.

Section 2.2.7 [Bar_engraver], page 208
Create barlines. This engraver is controlled through the whichBar property. If it has no bar line to create, it will forbid a linebreak at this point.

Properties (read)
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 Section “bar-line-interface” in Internals Reference.

Properties (write)

forbidBreak (boolean)
If set to ##t, prevent a line break at this point.

This engraver creates the following layout object(s):
Section 3.1.11 [BarLine], page 264.

Section 2.2.72 [Output_property_engraver], page 229
Apply a procedure to any grob acknowledged.

Music types accepted:
Section 1.2.4 [apply-output-event], page 35

2.1.12 GregorianTranscriptionVoice
Corresponds to a voice on a staff. This context handles the conversion of dynamic signs, stems, beams, super- and subscripts, slurs, ties, and rests.

You have to instantiate this explicitly if you want to have multiple voices on the same staff.

This context also accepts commands for the following context(s):
Voice.

This context creates the following layout object(s):
Section 3.1.9 [Arpeggio], page 263, Section 3.1.19 [Beam], page 270, Section 3.1.20 [BendAfter], page 271, Section 3.1.23 [BreathingSign], page 273, Section 3.1.27 [ClusterSpannerBeacon], page 276, Section 3.1.26 [ClusterSpanner], page 276, Section 3.1.28 [CombineTextScript], page 277, Section 3.1.31 [Dots], page 279, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.34 [DynamicLineSpanner], page 282, Section 3.1.36 [DynamicTextSpanner], page 284, Section 3.1.35 [DynamicText], page 283, Section 3.1.37 [Fingering], page 285, Section 3.1.39 [Glissando], page 288, Section 3.1.43 [Hairpin], page 290, Section 3.1.47 [InstrumentSwitch], page 294, Section 3.1.51 [LaissezVibrerTieColumn], page 297, Section 3.1.50 [LaissezVibrerTie], page 297, Section 3.1.54 [LigatureBracket], page 299, Section 3.1.64 [MultiMeasureRestNumber], page 306, Section 3.1.65 [MultiMeasureRestText], page 307, Section 3.1.63 [MultiMeasureRest], page 305, Section 3.1.68 [NoteColumn], page 310, Section 3.1.69 [NoteHead], page 310, Section 3.1.71 [NoteSpacing], page 311, Section 3.1.77 [PercentRepeatCounter], page 315, Section 3.1.76 [PercentRepeat], page 315, Section 3.1.78 [PhrasingSlur], page 317, Section 3.1.81 [RepeatSlash], page 320, Section 3.1.83 [RepeatTieColumn], page 321, Section 3.1.82 [RepeatTie], page 320, Section 3.1.84 [Rest], page 321, Section 3.1.87 [ScriptColumn], page 323, Section 3.1.86 [Script], page 322, Section 3.1.90 [Slur], page 324, Section 3.1.99 [StemTremolo], page 332, Section 3.1.98 [Stem], page 330, Section 3.1.100 [StringNumber], page 332, Section 3.1.101 [StrokeFinger], page 334, Section 3.1.110 [TextScript], page 341, Section 3.1.111 [TextSpanner], page 343, Section 3.1.113 [TieColumn], page 345, Section 3.1.112 [Tie], page 344, Section 3.1.115 [TrillPitchAccidental], page 346, Section 3.1.116 [TrillPitchGroup], page 347, Section 3.1.117 [TrillPitchHead], page 348, Section 3.1.118 [TrillSpanner], page 349,
Section 3.1.119 [TupletBracket], page 350, Section 3.1.120 [TupletNumber], page 351 and Section 3.1.126 [VoiceFollower], page 356.

This context sets the following properties:

- Set grob-property `padding` in Section 3.1.111 [TextSpanner], page 343 to -0.1.
- Set grob-property `style` in Section 3.1.111 [TextSpanner], page 343 to 'line.'
- Set grob-property `dash-fraction` in Section 3.1.111 [TextSpanner], page 343 to '('.
- Set translator property `autoBeaming` to #f.
- Set grob-property `padding` in Section 3.1.86 [Script], page 322 to 0.5.
- Set grob-property `transparent` in Section 3.1.54 [LigatureBracket], page 299 to #t.
- Set translator property `localKeySignature` to '('.

This context is a 'bottom' context; it cannot contain other contexts.

This context is built from the following engraver(s):

**Section 2.2.94 [Skip_event_swallow_translator], page 236**

Swallow \skip.

**Section 2.2.49 [Instrument_switch_engraver], page 222**

Create a cue text for taking instrument.

Properties (read)

instrumentCueName (markup)

The name to print if another instrument is to be taken.

This engraver creates the following layout object(s):

**Section 3.1.47 [InstrumentSwitch], page 294**

**Section 2.2.40 [Grace_engraver], page 219**

Set font size and other properties for grace notes.

Properties (read)

graceSettings (list)

Overrides for grace notes. This property should be manipulated through the `add-grace-property` function.

**Section 2.2.125 [Tuplet_engraver], page 244**

Catch tuplet events and generate appropriate bracket.

Music types accepted:

**Section 1.2.65 [tuplet-span-event], page 42**

Properties (read)

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.

This engraver creates the following layout object(s):

**Section 3.1.119 [TupletBracket], page 350 and Section 3.1.120 [TupletNumber], page 351**.
Section 2.2.118 [Tie_engraver], page 242
Generate ties between note heads of equal pitch.
Music types accepted:
Section 1.2.61 [tie-event], page 42
Properties (read)

\texttt{tieWaitForNote} (boolean)
If true, tied notes do not have to follow each other directly. This can be used for writing out arpeggios.

Properties (write)

\texttt{tieMelismaBusy} (boolean)
Signal whether a tie is present.

This engraver creates the following layout object(s):
Section 3.1.112 [Tie], page 344 and Section 3.1.113 [TieColumn], page 345.

Section 2.2.96 [Slur_engraver], page 237
Build slur grobs from slur events.
Music types accepted:
Section 1.2.48 [slur-event], page 40
Properties (read)

\texttt{slurMelismaBusy} (boolean)
Signal if a slur is present.

\texttt{doubleSlurs} (boolean)
If set, two slurs are created for every slurred note, one above and one below the chord.

This engraver creates the following layout object(s):
Section 3.1.90 [Slur], page 324.

Section 2.2.17 [Cluster_spanner_engraver], page 212
Engrave a cluster using \texttt{Spanner} notation.
Music types accepted:
Section 1.2.13 [cluster-note-event], page 36
This engraver creates the following layout object(s):
Section 3.1.26 [ClusterSpanner], page 276 and Section 3.1.27 [ClusterSpannerBeacon], page 276.

Section 2.2.78 [Phrasing_slur_engraver], page 232
Print phrasing slurs. Similar to Section 2.2.96 [Slur_engraver], page 237.
Music types accepted:
Section 1.2.42 [phrasing-slur-event], page 40
This engraver creates the following layout object(s):
Section 3.1.78 [PhrasingSlur], page 317.

Section 2.2.101 [Spanner_break_forbid_engraver], page 238
Forbid breaks in certain spanners.
Section 2.2.69 [Note_spacing_engraver], page 229
Generate NoteSpacing, an object linking horizontal lines for use in spacing.
This engraver creates the following layout object(s):
Section 3.1.71 [NoteSpacing], page 311.

Section 2.2.89 [Rhythmic_column_engraver], page 235
Generate NoteColumn, an object that groups stems, note heads, and rests.
This engraver creates the following layout object(s):
Section 3.1.68 [NoteColumn], page 310.

Section 2.2.90 [Script_column_engraver], page 235
Find potentially colliding scripts and put them into a ScriptColumn object; that will fix the collisions.
This engraver creates the following layout object(s):
Section 3.1.87 [ScriptColumn], page 323.

Section 2.2.91 [Script_engraver], page 235
Handle note scripted articulations.
Music types accepted:
Section 1.2.6 [articulation-event], page 36
Properties (read)
scriptDefinitions (list)
The description of scripts. This is used by the Script_engraver for typesetting note-superscripts and subscripts. See 'scm/script.scm' for more information.
This engraver creates the following layout object(s):
Section 3.1.86 [Script], page 322.

Section 2.2.11 [Bend_engraver], page 210
Create fall spanners.
Music types accepted:
Section 1.2.10 [bend-after-event], page 36
This engraver creates the following layout object(s):
Section 3.1.20 [BendAfter], page 271.

Section 2.2.34 [Fingering_engraver], page 217
Create fingering scripts.
Music types accepted:
Section 1.2.57 [stroke-finger-event], page 42 and Section 1.2.18 [fingering-event], page 37
This engraver creates the following layout object(s):
Section 3.1.37 [Fingering], page 285.

Section 2.2.27 [Dynamic_align_engraver], page 215
Align hairpins and dynamic texts on a horizontal line
Properties (read)
currentMusicalColumn (layout object)
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):
Section 3.1.34 [DynamicLineSpanner], page 282.
Section 2.2.63 [New_dynamic_engraver], page 227
Create hairpins, dynamic texts, and their vertical alignments. The symbols are collected onto a DynamicLineSpanner grob which takes care of vertical positioning.
Music types accepted:
Section 1.2.53 [span-dynamic-event], page 41 and Section 1.2.2 [absolute-dynamic-event], page 35
Properties (read)
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.’.
currentMusicalColumn (layout object)
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).
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.’.
This engraver creates the following layout object(s):
Section 3.1.35 [DynamicText], page 283, Section 3.1.36 [DynamicTextSpanner], page 284 and Section 3.1.43 [Hairpin], page 290.
Section 2.2.116 [Text_engraver], page 241
Create text scripts.
Music types accepted:
Section 1.2.59 [text-script-event], page 42
This engraver creates the following layout object(s):
Section 3.1.110 [TextScript], page 341.
Section 2.2.76 [Part_combine_engraver], page 231
Part combine engraver for orchestral scores: Print markings ‘a2’, ‘Solo’, ‘Solo II’, and ‘unisono’.
Music types accepted:
Section 1.2.38 [part-combine-event], page 39
Properties (read)
printPartCombineTexts (boolean)
Set ‘Solo’ and ‘A due’ texts in the part combiner?

soloText (markup)
The text for the start of a solo when part-combining.

soloIIText (markup)
The text for the start of a solo for voice ‘two’ when part-combining.

aDueText (markup)
Text to print at a unisono passage.

This engraver creates the following layout object(s):
Section 3.1.28 [CombineTextScript], page 277.

Section 2.2.95 [Slash_repeat_engraver], page 236
Make beat repeats.
Music types accepted:
Section 1.2.40 [percent-event], page 40
Properties (read)

measureLength (moment)
Length of one measure in the current time signature.

This engraver creates the following layout object(s):
Section 3.1.81 [RepeatSlash], page 320.

Section 2.2.77 [Percent_repeat_engraver], page 231
Make whole bar and double bar repeats.
Music types accepted:
Section 1.2.40 [percent-event], page 40
Properties (read)

countPercentRepeats (boolean)
If set, produce counters for percent repeats.

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

measureLength (moment)
Length of one measure in the current time signature.

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.

Properties (write)

forbidBreak (boolean)
If set to ##t, prevent a line break at this point.
This engraver creates the following layout object(s):
Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.76 [PercentRepeat], page 315 and Section 3.1.77 [PercentRepeatCounter], page 315.

**Section 2.2.15 [Chord_tremolo_engraver], page 211**
Generate beams for tremolo repeats.
Music types accepted:
Section 1.2.63 [tremolo-span-event], page 42
This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

**Section 2.2.64 [New_fingering_engraver], page 227**
Create fingering scripts for notes in a new chord. This engraver is ill-named, since it also takes care of articulations and harmonic note heads.
Properties (read)

- **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.

- **harmonicDots (boolean)**
  If set, harmonic notes in dotted chords get dots.

- **strokeFingerOrientations (list)**
  See fingeringOrientations.

- **stringNumberOrientations (list)**
  See fingeringOrientations.

This engraver creates the following layout object(s):
Section 3.1.37 [Fingering], page 285, Section 3.1.86 [Script], page 322, Section 3.1.100 [StringNumber], page 332 and Section 3.1.101 [StrokeFinger], page 334.

**Section 2.2.4 [Auto_beam_engraver], page 207**
Generate beams based on measure characteristics and observed Stems. Uses beatLength, measureLength, and measurePosition to decide when to start and stop a beam. Overriding beaming is done through Section 2.2.107 [Stem_engraver], page 239 properties stemLeftBeamCount and stemRightBeamCount.
Music types accepted:
Section 1.2.9 [beam-forbid-event], page 36
Properties (read)

- **autoBeaming (boolean)**
  If set to true then beams are generated automatically.

- **autoBeamSettings (list)**
  Specifies when automatically generated beams should begin and end. See Section “Setting automatic beam behavior” in Notation Reference for more information.
beatLength (moment)
The length of one beat in this time signature.

subdivideBeams (boolean)
If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.39 [Grace_beam_engraver], page 219
Handle Beam events by engraving beams. If omitted, then notes are printed with flags instead of beams. Only engraves beams when we are at grace points in time.
Music types accepted:
Section 1.2.8 [beam-event], page 36
Properties (read)

beamMelismaBusy (boolean)
Signal if a beam is present.

beatLength (moment)
The length of one beat in this time signature.

subdivideBeams (boolean)
If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.9 [Beam_engraver], page 209
Handle Beam events by engraving beams. If omitted, then notes are printed with flags instead of beams.
Music types accepted:
Section 1.2.8 [beam-event], page 36
Properties (read)

beamMelismaBusy (boolean)
Signal if a beam is present.

beatLength (moment)
The length of one beat in this time signature.

subdivideBeams (boolean)
If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

Properties (write)

forbidBreak (boolean)
If set to ##t, prevent a line break at this point.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.
Section 2.2.107 [Stem_engraver], page 239
Create stems and single-stem tremolos. It also works together with the beam engraver for overriding beaming.
Music types accepted:
Section 1.2.62 [tremolo-event], page 42
Properties (read)

- **tremoloFlags** (integer)
  The number of tremolo flags to add if no number is specified.

- **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**.

This engraver creates the following layout object(s):
Section 3.1.98 [Stem], page 330 and Section 3.1.99 [StemTremolo], page 332.

Section 2.2.126 [Tweak_engraver], page 244
Read the **tweaks** property from the originating event, and set properties.

Section 2.2.87 [Rest_engraver], page 234
Engrave rests.
Music types accepted:
Section 1.2.44 [rest-event], page 40
Properties (read)

- **middleCPosition** (number)
  The place of the middle C, measured in half staff-spaces. Usually determined by looking at middleCClefPosition and middleCOffset.

This engraver creates the following layout object(s):
Section 3.1.84 [Rest], page 321.

Section 2.2.24 [Dots_engraver], page 214
Create Section 3.1.31 [Dots], page 279 objects for Section 3.2.80 [rhythmic-head-interface], page 397s.
This engraver creates the following layout object(s):
Section 3.1.31 [Dots], page 279.

Section 2.2.66 [Note_heads_engraver], page 228
Generate note heads.
Music types accepted:
Section 1.2.34 [note-event], page 39
Properties (read)

- **middleCPosition** (number)
  The place of the middle C, measured in half staff-spaces. Usually determined by looking at middleCClefPosition and middleCOffset.
staffLineLayoutFunction (procedure)
  Layout of staff lines, traditional, or semitone.

This engraver creates the following layout object(s):
Section 3.1.69 [NoteHead], page 310.

Section 2.2.13 [Breathing_sign_engraver], page 210
Create a breathing sign.
Music types accepted:
Section 1.2.12 [breathing-event], page 36
This engraver creates the following layout object(s):
Section 3.1.23 [BreathingSign], page 273.

Section 2.2.54 [Ligature_bracket_engraver], page 224
Handle Ligature_events by engraving Ligature brackets.
Music types accepted:
Section 1.2.26 [ligature-event], page 38
This engraver creates the following layout object(s):
Section 3.1.54 [LigatureBracket], page 299.

Section 2.2.38 [Glissando_engraver], page 219
Engrave glissandi.
Music types accepted:
Section 1.2.19 [glissando-event], page 37
This engraver creates the following layout object(s):
Section 3.1.39 [Glissando], page 288.

Section 2.2.85 [Repeat_tie_engraver], page 234
Create repeat ties.
Music types accepted:
Section 1.2.43 [repeat-tie-event], page 40
This engraver creates the following layout object(s):
Section 3.1.82 [RepeatTie], page 320 and Section 3.1.83 [RepeatTieColumn], page 321.

Section 2.2.52 [Laissez_vibrer_engraver], page 223
Create laissez vibrer items.
Music types accepted:
Section 1.2.24 [laissez-vibrer-event], page 37
This engraver creates the following layout object(s):
Section 3.1.50 [LaissezVibrerTie], page 297 and Section 3.1.51 [LaissezVibrerTieColumn], page 297.

Section 2.2.36 [Forbid_line_break_engraver], page 218
Forbid line breaks when note heads are still playing at some point.
Properties (read)

`busyGrobs` (list)
A queue of `(end-moment . GROB)` cons cells. This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Properties (write)

`forbidBreak` (boolean)
If set to `##t`, prevent a line break at this point.

Section 2.2.44 [Grob_pq_engraver], page 220
Administrate when certain grobs (e.g., note heads) stop playing.
Properties (read)

`busyGrobs` (list)
A queue of `(end-moment . GROB)` cons cells. This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Properties (write)

`busyGrobs` (list)
A queue of `(end-moment . GROB)` cons cells. This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Section 2.2.124 [Trill_spanner_engraver], page 244
Create trill spanner from an event.
Music types accepted:
Section 1.2.64 [trill-span-event], page 42
Properties (read)

`currentCommandColumn` (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

`currentMusicalColumn` (layout object)
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):
Section 3.1.118 [TrillSpanner], page 349.

Section 2.2.117 [Text_spanner_engraver], page 242
Create text spanner from an event.
Music types accepted:
Section 1.2.60 [text-span-event], page 42
This engraver creates the following layout object(s):
Section 3.1.111 [TextSpanner], page 343.

**Section 2.2.62 [Multi_measure_rest_engraver], page 226**
Engrave multi-measure rests that are produced with ‘R’. It reads `measurePosition` and `internalBarNumber` to determine what number to print over the Section 3.1.63 [MultiMeasureRest], page 305. Reads `measureLength` to determine whether it should use a whole rest or a breve rest to represent one measure.

Music types accepted:
Section 1.2.32 [multi-measure-text-event], page 38 and Section 1.2.31 [multi-measure-rest-event], page 38

Properties (read)

- `internalBarNumber` (integer)
  Contains the current bar number. This property is used for internal timekeeping, among others by the `Accidental_engraver`.

- `restNumberThreshold` (number)
  If a multimeasure rest has more measures than this, a number is printed.

- `breakableSeparationItem` (layout object)
  The breakable items in this time step, for this staff.

- `currentCommandColumn` (layout object)
  Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

- `measurePosition` (moment)
  How much of the current measure have we had. This can be set manually to create incomplete measures.

- `measureLength` (moment)
  Length of one measure in the current time signature.

This engraver creates the following layout object(s):
Section 3.1.63 [MultiMeasureRest], page 305, Section 3.1.64 [MultiMeasureRestNumber], page 306 and Section 3.1.65 [MultiMeasureRestText], page 307.

**Section 2.2.3 [Arpeggio_engraver], page 207**
Generate an Arpeggio symbol.
Music types accepted:
Section 1.2.5 [arpeggio-event], page 35
This engraver creates the following layout object(s):
Section 3.1.9 [Arpeggio], page 263.
Section 2.2.72 [Output_property_engraver], page 229
Apply a procedure to any grob acknowledged.
Music types accepted:
Section 1.2.4 [apply-output-event], page 35

Section 2.2.83 [Pitched_trill_engraver], page 233
Print the bracketed note head after a note head with trill.
This engraver creates the following layout object(s):
Section 3.1.115 [TrillPitchAccidental], page 346, Section 3.1.116 [TrillPitchGroup], page 347 and Section 3.1.117 [TrillPitchHead], page 348.

Section 2.2.35 [Font_size_engraver], page 217
Put fontSize into font-size grob property.
Properties (read)

fontSize (number)
The relative size of all grobs in a context.

2.1.13 Lyrics
Corresponds to a voice with lyrics. Handles the printing of a single line of lyrics.
This context creates the following layout object(s):
Section 3.1.46 [InstrumentName], page 293, Section 3.1.55 [LyricExtender], page 300, Section 3.1.56 [LyricHyphen], page 301, Section 3.1.57 [LyricSpace], page 302, Section 3.1.58 [LyricText], page 302, Section 3.1.97 [StanzaNumber], page 330 and Section 3.1.125 [VerticalAxisGroup], page 355.
This context sets the following properties:
• Set grob-property bar-size in Section 3.1.11 [BarLine], page 264 to 0.1.
• Set grob-property font-size in Section 3.1.46 [InstrumentName], page 293 to 1.0.
• Set grob-property self-alignment-Y in Section 3.1.46 [InstrumentName], page 293 to #f.
• Set grob-property padding in Section 3.1.89 [SeparationItem], page 323 to 0.2.
• Set grob-property keep-fixed-while-stretching in Section 3.1.125 [VerticalAxisGroup], page 355 to #t.
• Set grob-property remove-empty in Section 3.1.125 [VerticalAxisGroup], page 355 to #t.
• Set grob-property remove-first in Section 3.1.125 [VerticalAxisGroup], page 355 to #t.
• Set translator property shortInstrumentName to '() .
• Set translator property instrumentName to '() .
• Set grob-property minimum-Y-extent in Section 3.1.125 [VerticalAxisGroup], page 355 to '(-0.75 . 2.0).
This context is a ‘bottom’ context; it cannot contain other contexts.
This context is built from the following engraver(s):
Section 2.2.45 [Hara_kiri_engraver], page 221
Like Axis_group_engraver, but make a hara-kiri spanner, and add interesting items (i.e., note heads, lyric syllables, and normal rests).
Properties (read)

keepAliveInterfaces (list)
A list of symbols, signifying grob interfaces that are worth keeping a staff with remove-empty set around for.
This engraver creates the following layout object(s):
Section 3.1.125 [VerticalAxisGroup], page 355.

Section 2.2.35 [Font_size_engraver], page 217
Put fontSize into font-size grob property.
Properties (read)

  fontSize (number)
  The relative size of all grobs in a context.

Section 2.2.94 [Skip_event_swallow_translator], page 236
Swallow \skip.

Section 2.2.48 [Instrument_name_engraver], page 221
Create a system start text for instrument or vocal names.
Properties (read)

  currentCommandColumn (layout object)
  Grob that is X-parent to all current breakable
  (clef, key signature, etc.) items.

  shortInstrumentName (markup)
  See 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.

  shortVocalName (markup)
  Name of a vocal line, short version.

  vocalName (markup)
  Name of a vocal line.

This engraver creates the following layout object(s):
Section 3.1.46 [InstrumentName], page 293.

Section 2.2.106 [Stanza_number_engraver], page 239
Engrave stanza numbers.
Properties (read)

  stanza (markup)
  Stanza 'number' to print before the start of a
  verse. Use in Lyrics context.

This engraver creates the following layout object(s):
Section 3.1.97 [StanzaNumber], page 330.

Section 2.2.47 [Hyphen_engraver], page 221
Create lyric hyphens and distance constraints between words.
Music types accepted:
Section 1.2.21 [hyphen-event], page 37
This engraver creates the following layout object(s):
Section 3.1.56 [LyricHyphen], page 301 and Section 3.1.57 [LyricSpace], page 302.
Section 2.2.31 [Extender_engraver], page 216
Create lyric extenders.
Music types accepted:
Section 1.2.17 [extender-event], page 37
Properties (read)

`extendersOverRests` (boolean)
Whether to continue extenders as they cross a rest.

This engraver creates the following layout object(s):
Section 3.1.55 [LyricExtender], page 300.

Section 2.2.55 [Lyric_engraver], page 224
Engrave text for lyrics.
Music types accepted:
Section 1.2.28 [lyric-event], page 38
Properties (read)

`ignoreMelismata` (boolean)
Ignore melismata for this Section “Lyrics” in Internals Reference line.

`lyricMelismaAlignment` (direction)
Alignment to use for a melisma syllable.

This engraver creates the following layout object(s):
Section 3.1.58 [LyricText], page 302.

2.1.14 MensuralStaff
Same as Staff context, except that it is accommodated for typesetting a piece in mensural style.

This context also accepts commands for the following context(s):

Staff.

This context creates the following layout object(s):

Section 3.1.2 [AccidentalCautionary], page 257, Section 3.1.3 [AccidentalPlacement], page 258, Section 3.1.4 [AccidentalSuggestion], page 259, Section 3.1.1 [Accidental], page 257, Section 3.1.11 [BarLine], page 264, Section 3.1.15 [BassFigureAlignmentPositioning], page 268, Section 3.1.14 [BassFigureAlignment], page 268, Section 3.1.16 [BassFigureBracket], page 269, Section 3.1.17 [BassFigureContinuation], page 269, Section 3.1.18 [BassFigureLine], page 270, Section 3.1.13 [BassFigure], page 267, Section 3.1.25 [Clef], page 275, Section 3.1.29 [Custos], page 278, Section 3.1.30 [DotColumn], page 279, Section 3.1.46 [InstrumentName], page 293, Section 3.1.48 [KeyCancellation], page 295, Section 3.1.49 [KeySignature], page 296, Section 3.1.52 [LedgerLineSpanner], page 298, Section 3.1.67 [NoteCollision], page 309, Section 3.1.72 [OctavateEight], page 311, Section 3.1.73 [OttavaBracket], page 312, Section 3.1.79 [PianoPedalBracket], page 318, Section 3.1.85 [RestCollision], page 322, Section 3.1.88 [ScriptRow], page 323, Section 3.1.92 [SostenutoPedalLineSpanner], page 326, Section 3.1.91 [SostenutoPedal], page 325, Section 3.1.95 [StaffSpacing], page 329, Section 3.1.96 [StaffSymbol], page 329, Section 3.1.103 [SustainPedalLineSpanner], page 335, Section 3.1.102 [SustainPedal], page 335, Section 3.1.114 [TimeSignature], page 345, Section 3.1.122 [UnaCordaPedalLineSpanner], page 353, Section 3.1.121 [UnaCordaPedal], page 352 and Section 3.1.125 [VerticalAxisGroup], page 355.

This context sets the following properties:
• Set translator property `printKeyCancellation` to `#f`.

• Set translator property `autoCautionaries` to `'()`.

• Set translator property `autoAccidentals` to `(Staff #$<procedure #f (context pitch barnum measurepos)>)`.

• Set translator property `extraNatural` to `#f`.

• Set grob-property `neutral-direction` in Section 3.1.29 [Custos], page 278 to `-1`.

• Set grob-property `neutral-position` in Section 3.1.29 [Custos], page 278 to `3`.

• Set grob-property `style` in Section 3.1.29 [Custos], page 278 to `'mensural`.

• Set grob-property `glyph-name-alist` in Section 3.1.1 [Accidental], page 257 to `((-1/2 . accidentals.mensuralM1) (0 . accidentals.vaticana0) (1/2 . accidentals.mensural1))`.

• Set grob-property `glyph-name-alist` in Section 3.1.49 [KeySignature], page 296 to `((-1/2 . accidentals.mensuralM1) (0 . accidentals.vaticana0) (1/2 . accidentals.mensural1))`.

• Set grob-property `style` in Section 3.1.114 [TimeSignature], page 345 to `'mensural`.

• Set translator property `clefOctavation` to `0`.

• Set translator property `clefPosition` to `-2`.

• Set translator property `middleCPosition` to `-6`.

• Set translator property `middleCClefPosition` to `-6`.

• Set translator property `clefGlyph` to `"clefs.mensural.g"`.

• Set grob-property `thickness` in Section 3.1.96 [StaffSymbol], page 329 to `0.6`.

• Set grob-property `transparent` in Section 3.1.11 [BarLine], page 264 to `#t`.

• Set translator property `shortInstrumentName` to `()`.

• Set translator property `instrumentName` to `()`.

• Set grob-property `minimum-Y-extent` in Section 3.1.125 [VerticalAxisGroup], page 355 to `"(-4 . 4)"`.

• Set translator property `ignoreFiguredBassRest` to `#t`.

• Set translator property `createSpacing` to `#t`.

• Set translator property `localKeySignature` to `()`.

Context MensuralStaff can contain Section 2.1.15 [MensuralVoice], page 116 and Section 2.1.3 [CueVoice], page 50.

This context is built from the following engraver(s):

Section 2.2.21 [Custos_engraver], page 213
Engrave custodes.

This engraver creates the following layout object(s):
Section 3.1.29 [Custos], page 278.

Section 2.2.92 [Script_row_engraver], page 236
Determine order in horizontal side position elements.
This engraver creates the following layout object(s):
Section 3.1.88 [ScriptRow], page 323.

Section 2.2.33 [Figured_bass_position_engraver], page 217
Position figured bass alignments over notes.
This engraver creates the following layout object(s):
Section 3.1.15 [BassFigureAlignmentPositioning], page 268.
Section 2.2.32 [Figured_bass_engraver], page 216
Make figured bass numbers.
Music types accepted:
Section 1.2.7 [bass-figure-event], page 36 and Section 1.2.44 [rest-event], page 40
Properties (read)

- `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.

- `implicitBassFigures` (list)
  A list of bass figures that are not printed as numbers, but only as extender lines.

- `useBassFigureExtenders` (boolean)
  Whether to use extender lines for repeated bass figures.

- `ignoreFiguredBassRest` (boolean)
  Don’t swallow rest events.

This engraver creates the following layout object(s):
Section 3.1.13 [BassFigure], page 267, Section 3.1.14 [BassFigure-Alignment], page 268, Section 3.1.16 [BassFigureBracket], page 269, Section 3.1.17 [BassFigureContinuation], page 269 and Section 3.1.18 [BassFigureLine], page 270.

Section 2.2.5 [Axis_group_engraver], page 208
Group all objects created in this context in a `VerticalAxisGroup` spanner.
Properties (read)

- `currentCommandColumn` (layout object)
  Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

This engraver creates the following layout object(s):
Section 3.1.125 [VerticalAxisGroup], page 355.

Section 2.2.108 [String_number_engraver], page 240
Swallow string number events. The purpose of this engraver is to process tablatures for normal notation. To prevent warnings for unprocessed string number events to obscure real error messages, this engraver swallows them all.

Section 2.2.48 [Instrument_name_engraver], page 221
Create a system start text for instrument or vocal names.
Properties (read)
**currentCommandColumn** (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

**shortInstrumentName** (markup)
See `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.

**shortVocalName** (markup)
Name of a vocal line, short version.

**vocalName** (markup)
Name of a vocal line.

This engraver creates the following layout object(s):
Section 3.1.46 [InstrumentName], page 293.

**Section 2.2.79 [Piano_pedal_align_engraver], page 232**
Align piano pedal symbols and brackets.
Properties (read)

**currentCommandColumn** (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

This engraver creates the following layout object(s):
Section 3.1.92 [SostenutoPedalLineSpanner], page 326, Section 3.1.103 [SustainPedalLineSpanner], page 335 and Section 3.1.122 [UnaCordaPedalLineSpanner], page 353.

**Section 2.2.80 [Piano_pedal_engraver], page 232**
Engrave piano pedal symbols and brackets.
Music types accepted:
Section 1.2.66 [una-corda-event], page 43, Section 1.2.58 [sustain-event], page 42 and Section 1.2.51 [sostenuto-event], page 41
Properties (read)

**currentCommandColumn** (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

**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.

This engraver creates the following layout object(s):
Section 3.1.79 [PianoPedalBracket], page 318, Section 3.1.91 [SostenutoPedal], page 325, Section 3.1.102 [SustainPedal], page 335 and Section 3.1.121 [UnaCordaPedal], page 352.

Section 2.2.1 [Accidental_engraver], page 206
Make accidentals. Catch note heads, ties and notices key-change events. This engraver usually lives at Staff level, but reads the settings for Accidental at Voice level, so you can \override them at Voice.

Properties (read)

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 Section “Score” in Internals Reference then all staves share accidentals, and if context is Section “Staff” in Internals Reference 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. \((#t \cdot #f)\) does not make sense.

**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.

**internalBarNumber** (integer)
Contains the current barnumber. This property is used for internal timekeeping, among others by the **Accidental_engraver**.

**extraNatural** (boolean)
Whether to typeset an extra natural sign before accidentals changing from a non-natural to another non-natural.

**harmonicAccidentals** (boolean)
If set, harmonic notes in chords get accidentals.

**keySignature** (list)
The current key signature. This is an alist containing \((\text{step} \cdot \text{alter})\) or \(((\text{octave} \cdot \text{step}) \cdot \text{alter})\), where \text{step} is a number in the range 0 to 6 and \text{alter} a fraction, denoting alteration. For alterations, use symbols, e.g. \text{keySignature} = \#`\((6 \cdot ,\text{FLAT}))\).

**localKeySignature** (list)
The key signature at this point in the measure. The format is the same as for **keySignature**, but can also contain \(((\text{octave} \cdot \text{name}) \cdot (\text{alter barnumber} \cdot \text{measureposition}))\) pairs.

Properties (write)

**localKeySignature** (list)
The key signature at this point in the measure. The format is the same as for **keySignature**, but can also contain \(((\text{octave} \cdot \text{name}) \cdot (\text{alter barnumber} \cdot \text{measureposition}))\) pairs.

This engraver creates the following layout object(s):
Section 3.1.1 [Accidental], page 257, Section 3.1.2 [AccidentalCautionary], page 257, Section 3.1.3 [AccidentalPlacement], page 258 and Section 3.1.4 [AccidentalSuggestion], page 259.

Section 2.2.86 [Rest_collision_engraver], page 234
Handle collisions of rests.

Properties (read)
busyGrobs (list)
A queue of \texttt{(end-moment . GROB)} cons cells.
This is for internal (C++) use only. This property contains the grobs which are still busy (e.g.
note heads, spanners, etc.).

This engraver creates the following layout object(s):
Section 3.1.85 [RestCollision], page 322.

Section 2.2.44 [Grob_pq_engraver], page 220
Administrate when certain grobs (e.g., note heads) stop playing.
Properties (read)

\begin{verbatim}
busyGrobs (list)
A queue of \texttt{(end-moment . GROB)} cons cells.
This is for internal (C++) use only. This property contains the grobs which are still busy (e.g.
note heads, spanners, etc.).
\end{verbatim}

Properties (write)

\begin{verbatim}
busyGrobs (list)
A queue of \texttt{(end-moment . GROB)} cons cells.
This is for internal (C++) use only. This property contains the grobs which are still busy (e.g.
note heads, spanners, etc.).
\end{verbatim}

Section 2.2.18 [Collision_engraver], page 212
Collect \texttt{NoteColumns}, and as soon as there are two or more, put them
in a \texttt{NoteCollision} object.
This engraver creates the following layout object(s):
Section 3.1.67 [NoteCollision], page 309.

Section 2.2.104 [Staff_symbol_engraver], page 238
Create the constellation of five (default) staff lines.
Music types accepted:
Section 1.2.55 [staff-span-event], page 41
This engraver creates the following layout object(s):
Section 3.1.96 [StaffSymbol], page 329.

Section 2.2.53 [Ledger_line_engraver], page 223
Create the spanner to draw ledger lines, and notices objects that need
ledger lines.
This engraver creates the following layout object(s):
Section 3.1.52 [LedgerLineSpanner], page 298.

Section 2.2.120 [Time_signature_engraver], page 243
Create a Section 3.1.114 [TimeSignature], page 345 whenever
timeSignatureFraction changes.
Properties (read)

\begin{verbatim}
implicitTimeSignatureVisibility (vector)
break visibility for the default time signature.
\end{verbatim}
timeSignatureFraction (pair of numbers)
A pair of numbers, signifying the time signature. For example, #'(4 . 4) is a 4/4 time signature.

This engraver creates the following layout object(s):
Section 3.1.114 [TimeSignature], page 345.

Section 2.2.50 [Key_engraver], page 222
Engrave a key signature.

Music types accepted:
Section 1.2.22 [key-change-event], page 37

Properties (read)

createKeyOnClefChange (boolean)
Print a key signature whenever the clef is changed.

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.

extraNatural (boolean)
Whether to typeset an extra natural sign before accidentals changing from a non-natural to another non-natural.

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)).

lastKeySignature (list)
Last key signature before a key signature change.

printKeyCancellation (boolean)
Print restoration alterations before a key signature change.

Properties (write)

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)).'

lastKeySignature (list)
Last key signature before a key signature change.

tonic (pitch)
The tonic of the current scale.

This engraver creates the following layout object(s):
Section 3.1.48 [KeyCancellation], page 295 and Section 3.1.49 [KeySignature], page 296.

Section 2.2.16 [Clef_ engraver], page 212
Determine and set reference point for pitches.
Properties (read)
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.
explicitClefVisibility (vector)
‘break-visibility’ function for clef changes.
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.

This engraver creates the following layout object(s):
Section 3.1.25 [Clef], page 275 and Section 3.1.72 [OctavateEight], page 311.

Section 2.2.71 [Ottava_spanner_ engraver], page 229
Create a text spanner when the ottavation property changes.
Properties (read)
ottavation (markup)
If set, the text for an ottava spanner. Changing this creates a new text spanner.
originalMiddleCPosition (integer)
Used for temporary overriding middle C in octavation brackets.
currentMusicalColumn (layout object)
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).
This engraver creates the following layout object(s):
Section 3.1.73 [OttavaBracket], page 312.

Section 2.2.102 [Staff_collecting_engraver], page 238
Maintain the stavesFound variable.

Properties (read)

stavesFound (list of grobs)
A list of all staff-symbols found.

Properties (write)

stavesFound (list of grobs)
A list of all staff-symbols found.

Section 2.2.23 [Dot_column_engraver], page 214
Engrave dots on dotted notes shifted to the right of the note. If omitted, then dots appear on top of the notes.

This engraver creates the following layout object(s):
Section 3.1.30 [DotColumn], page 279.

Section 2.2.93 [Separating_line_group_engraver], page 236
Generate objects for computing spacing parameters.

Properties (read)

createSpacing (boolean)
Create StaffSpacing objects? Should be set for staves.

Properties (write)

hasStaffSpacing (boolean)
True if the current CommandColumn contains items that will affect spacing.

This engraver creates the following layout object(s):
Section 3.1.95 [StaffSpacing], page 329.

Section 2.2.35 [Font_size_engraver], page 217
Put fontSize into font-size grob property.

Properties (read)

fontSize (number)
The relative size of all grobs in a context.

Section 2.2.7 [Bar_engraver], page 208
Create barlines. This engraver is controlled through the whichBar property. If it has no bar line to create, it will forbid a linebreak at this point.

Properties (read)

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 Section “barline-interface” in Internals Reference.
Properties (write)

forbidBreak (boolean)

If set to #t, prevent a line break at this point.

This engraver creates the following layout object(s):

Section 3.1.11 [BarLine], page 264.

Section 2.2.72 [Output_property_engraver], page 229

Apply a procedure to any grob acknowledged.

Music types accepted:

Section 1.2.4 [apply-output-event], page 35

2.1.15 MensuralVoice

Same as Voice context, except that it is accommodated for typesetting a piece in mensural style.

This context also accepts commands for the following context(s):

Voice.

This context creates the following layout object(s):

Section 3.1.9 [Arpeggio], page 263, Section 3.1.19 [Beam], page 270, Section 3.1.20 [BendAfter], page 271, Section 3.1.23 [BreathingSign], page 273, Section 3.1.27 [ClusterSpannerBeacon], page 276, Section 3.1.26 [ClusterSpanner], page 276, Section 3.1.28 [CombineTextScript], page 277, Section 3.1.31 [Dots], page 279, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.34 [DynamicLineSpanner], page 282, Section 3.1.36 [DynamicTextSpanner], page 284, Section 3.1.35 [DynamicText], page 283, Section 3.1.37 [Fingering], page 285, Section 3.1.39 [Glissando], page 288, Section 3.1.43 [Hairpin], page 290, Section 3.1.47 [InstrumentSwitch], page 294, Section 3.1.51 [LaissezVibrerTieColumn], page 297, Section 3.1.50 [LaissezVibrerTie], page 297, Section 3.1.61 [MensuralLigature], page 304, Section 3.1.64 [MultiMeasureRestNumber], page 306, Section 3.1.65 [MultiMeasureRestText], page 307, Section 3.1.63 [MultiMeasureRest], page 305, Section 3.1.68 [NoteColumn], page 310, Section 3.1.69 [NoteHead], page 310, Section 3.1.71 [NoteSpacing], page 311, Section 3.1.77 [PercentRepeatCounter], page 315, Section 3.1.76 [PercentRepeat], page 315, Section 3.1.78 [PhrasingSlur], page 317, Section 3.1.81 [RepeatSlash], page 320, Section 3.1.83 [RepeatTieColumn], page 321, Section 3.1.82 [RepeatTie], page 320, Section 3.1.84 [Rest], page 321, Section 3.1.87 [ScriptColumn], page 323, Section 3.1.86 [Script], page 322, Section 3.1.99 [StemTremolo], page 332, Section 3.1.98 [Stem], page 330, Section 3.1.100 [StringNumber], page 332, Section 3.1.101 [StrokeFinger], page 334, Section 3.1.110 [TextScript], page 341, Section 3.1.111 [TextSpanner], page 343, Section 3.1.113 [TieColumn], page 345, Section 3.1.112 [Tie], page 344, Section 3.1.115 [TrillPitchAccidental], page 346, Section 3.1.116 [TrillPitchGroup], page 347, Section 3.1.117 [TrillPitchHead], page 348, Section 3.1.118 [TrillSpanner], page 349, Section 3.1.119 [TupletBracket], page 350, Section 3.1.120 [TupletNumber], page 351 and Section 3.1.126 [VoiceFollower], page 356.

This context sets the following properties:

- Set translator property autoBeaming to #f.
- Set grob-property style in Section 3.1.84 [Rest], page 321 to 'mensural.
- Set grob-property style in Section 3.1.69 [NoteHead], page 310 to 'mensural.
- Set translator property localKeySignature to '( ).

This context is a 'bottom' context; it cannot contain other contexts.

This context is built from the following engraver(s):
Section 2.2.60 [Mensural_ligature_engraver], page 225
Handle Mensural_ligature_events by gluing special ligature heads together.
Music types accepted:
Section 1.2.26 [ligature-event], page 38
This engraver creates the following layout object(s):
Section 3.1.61 [MensuralLigature], page 304.

Section 2.2.94 [Skip_event_swallow_translator], page 236
Swallow \skip.

Section 2.2.49 [Instrument_switch_engraver], page 222
Create a cue text for taking instrument.
Properties (read)

  instrumentCueName (markup)
  The name to print if another instrument is to be taken.

This engraver creates the following layout object(s):
Section 3.1.47 [InstrumentSwitch], page 294.

Section 2.2.40 [Grace_engraver], page 219
Set font size and other properties for grace notes.
Properties (read)

  graceSettings (list)
  Overrides for grace notes. This property should be manipulated through the add-grace-property function.

Section 2.2.125 [Tuplet_engraver], page 244
Catch tuplet events and generate appropriate bracket.
Music types accepted:
Section 1.2.65 [tuplet-span-event], page 42
Properties (read)

  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.

This engraver creates the following layout object(s):
Section 3.1.119 [TupletBracket], page 350 and Section 3.1.120 [TupletNumber], page 351.

Section 2.2.118 [Tie_engraver], page 242
Generate ties between note heads of equal pitch.
Music types accepted:
Section 1.2.61 [tie-event], page 42
Properties (read)
tieWaitForNote (boolean)
If true, tied notes do not have to follow each other directly. This can be used for writing out arpeggios.

Properties (write)

tieMelismaBusy (boolean)
Signal whether a tie is present.

This engraver creates the following layout object(s):
Section 3.1.112 [Tie], page 344 and Section 3.1.113 [TieColumn], page 345.

Section 2.2.17 [Cluster_spanner_engraver], page 212
Engrave a cluster using Spanner notation.
Music types accepted:
Section 1.2.13 [cluster-note-event], page 36
This engraver creates the following layout object(s):
Section 3.1.26 [ClusterSpanner], page 276 and Section 3.1.27 [ClusterSpannerBeacon], page 276.

Section 2.2.78 [Phrasing_slur_engraver], page 232
Print phrasing slurs. Similar to Section 2.2.96 [Slur_engraver], page 237.
Music types accepted:
Section 1.2.42 [phrasing-slur-event], page 40
This engraver creates the following layout object(s):
Section 3.1.78 [PhrasingSlur], page 317.

Section 2.2.101 [Spanner_break_forbid_engraver], page 238
Forbid breaks in certain spanners.

Section 2.2.69 [Note_spacing_engraver], page 229
Generate NoteSpacing, an object linking horizontal lines for use in spacing.

This engraver creates the following layout object(s):
Section 3.1.71 [NoteSpacing], page 311.

Section 2.2.89 [Rhythmic_column_engraver], page 235
Generate NoteColumn, an object that groups stems, note heads, and rests.

This engraver creates the following layout object(s):
Section 3.1.68 [NoteColumn], page 310.

Section 2.2.90 [Script_column_engraver], page 235
Find potentially colliding scripts and put them into a ScriptColumn object; that will fix the collisions.

This engraver creates the following layout object(s):
Section 3.1.87 [ScriptColumn], page 323.

Section 2.2.91 [Script_engraver], page 235
Handle note scripted articulations.

Music types accepted:
Section 1.2.6 [articulation-event], page 36

Properties (read)
The description of scripts. This is used by the Script_engraver for typesetting note-superscripts and subscripts. See 'scm/script.scm' for more information.

This engraver creates the following layout object(s):
Section 3.1.86 [Script], page 322.

Section 2.2.11 [Bend_engraver], page 210
Create fall spanners.
Music types accepted:
Section 1.2.10 [bend-after-event], page 36
This engraver creates the following layout object(s):
Section 3.1.20 [BendAfter], page 271.

Section 2.2.34 [Fingering_engraver], page 217
Create fingering scripts.
Music types accepted:
Section 1.2.57 [stroke-finger-event], page 42 and Section 1.2.18 [fingering-event], page 37
This engraver creates the following layout object(s):
Section 3.1.37 [Fingering], page 285.

Section 2.2.27 [Dynamic_align_engraver], page 215
Align hairpins and dynamic texts on a horizontal line
Properties (read)
  currentMusicalColumn (layout object)
    Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).
This engraver creates the following layout object(s):
Section 3.1.34 [DynamicLineSpanner], page 282.

Section 2.2.63 [New_dynamic_engraver], page 227
Create hairpins, dynamic texts, and their vertical alignments. The symbols are collected onto a DynamicLineSpanner grob which takes care of vertical positioning.
Music types accepted:
Section 1.2.53 [span-dynamic-event], page 41 and Section 1.2.2 [absolute-dynamic-event], page 35
Properties (read)
  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.’.
  currentMusicalColumn (layout object)
    Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).
**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.'.

This engraver creates the following layout object(s):
Section 3.1.35 [DynamicText], page 283, Section 3.1.36 [DynamicTextSpanner], page 284 and Section 3.1.43 [Hairpin], page 290.

**Section 2.2.116 [Text_engraver], page 241**
Create text scripts.
Music types accepted:
Section 1.2.59 [text-script-event], page 42
This engraver creates the following layout object(s):
Section 3.1.110 [TextScript], page 341.

**Section 2.2.76 [Part_combine_engraver], page 231**
Part combine engraver for orchestral scores: Print markings ‘a2’, ‘Solo’, ‘Solo II’, and ‘unisono’.
Music types accepted:
Section 1.2.38 [part-combine-event], page 39
Properties (read)

**printPartCombineTexts** (boolean)
Set ‘Solo’ and ‘A due’ texts in the part combiner?

**soloText** (markup)
The text for the start of a solo when part-combining.

**soloIIText** (markup)
The text for the start of a solo for voice ‘two’ when part-combining.

**aDueText** (markup)
Text to print at a unisono passage.

This engraver creates the following layout object(s):
Section 3.1.28 [CombineTextScript], page 277.

**Section 2.2.95 [Slash_repeat_engraver], page 236**
Make beat repeats.
Music types accepted:
Section 1.2.40 [percent-event], page 40
Properties (read)

**measureLength** (moment)
Length of one measure in the current time signature.
This engraver creates the following layout object(s):
Section 3.1.81 [RepeatSlash], page 320.

Section 2.2.77 [Percent_repeat_engraver], page 231
Make whole bar and double bar repeats.
Music types accepted:
Section 1.2.40 [percent-event], page 40
Properties (read)

- **countPercentRepeats** (boolean)
  If set, produce counters for percent repeats.

- **currentCommandColumn** (layout object)
  Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

- **measureLength** (moment)
  Length of one measure in the current time signature.

- **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.

Properties (write)

- **forbidBreak** (boolean)
  If set to `##t`, prevent a line break at this point.

This engraver creates the following layout object(s):
Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.76 [PercentRepeat], page 315 and Section 3.1.77 [PercentRepeatCounter], page 315.

Section 2.2.15 [Chord_tremolo_engraver], page 211
Generate beams for tremolo repeats.
Music types accepted:
Section 1.2.63 [tremolo-span-event], page 42
This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.64 [New_fingering_engraver], page 227
Create fingering scripts for notes in a new chord. This engraver is ill-named, since it also takes care of articulations and harmonic note heads.
Properties (read)

- **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.

- **harmonicDots** (boolean)
  If set, harmonic notes in dotted chords get dots.
strokeFingerOrientations (list)
See fingeringOrientations.

stringNumberOrientations (list)
See fingeringOrientations.

This engraver creates the following layout object(s):
Section 3.1.37 [Fingering], page 285, Section 3.1.86 [Script], page 322,
Section 3.1.100 [StringNumber], page 332 and Section 3.1.101 [StrokeFinger], page 334.

Section 2.2.4 [Auto_beam_engraver], page 207
Generate beams based on measure characteristics and observed Stems. Uses beatLength, measureLength, and measurePosition to decide when to start and stop a beam. Overriding beaming is done through Section 2.2.107 [Stem_engraver], page 239 properties stemLeftBeamCount and stemRightBeamCount.
Music types accepted:
Section 1.2.9 [beam-forbid-event], page 36
Properties (read)

autoBeaming (boolean)
If set to true then beams are generated automatically.

autoBeamSettings (list)
Specifies when automatically generated beams should begin and end. See Section “Setting automatic beam behavior” in Notation Reference for more information.

beatLength (moment)
The length of one beat in this time signature.

subdivideBeams (boolean)
If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.39 [Grace_beam_engraver], page 219
Handle Beam events by engraving beams. If omitted, then notes are printed with flags instead of beams. Only engraves beams when we are at grace points in time.
Music types accepted:
Section 1.2.8 [beam-event], page 36
Properties (read)

beamMelismaBusy (boolean)
Signal if a beam is present.

beatLength (moment)
The length of one beat in this time signature.
subdivideBeams (boolean)
If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.9 [Beam_engraver], page 209
Handle Beam events by engraving beams. If omitted, then notes are printed with flags instead of beams.
Music types accepted:
Section 1.2.8 [beam-event], page 36
Properties (read)
beamMelismaBusy (boolean)
Signal if a beam is present.

beatLength (moment)
The length of one beat in this time signature.

subdivideBeams (boolean)
If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

Properties (write)

forbidBreak (boolean)
If set to ##t, prevent a line break at this point.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.107 [Stem_engraver], page 239
Create stems and single-stem tremolos. It also works together with the beam engraver for overriding beaming.
Music types accepted:
Section 1.2.62 [tremolo-event], page 42
Properties (read)

tremoloFlags (integer)
The number of tremolo flags to add if no number is specified.

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.

This engraver creates the following layout object(s):
Section 3.1.98 [Stem], page 330 and Section 3.1.99 [StemTremolo], page 332.
Section 2.2.126 [Tweak_engraver], page 244
Read the \texttt{tweaks} property from the originating event, and set properties.

Section 2.2.87 [Rest_engraver], page 234
Engrave rests.
Music types accepted:
Section 1.2.44 [rest-event], page 40
Properties (read)
\begin{itemize}
  \item \texttt{middleCPosition} (number)
    The place of the middle C, measured in half staff-spaces. Usually determined by looking at \texttt{middleCClefPosition} and \texttt{middleCOffset}.
\end{itemize}
This engraver creates the following layout object(s):
Section 3.1.84 [Rest], page 321.

Section 2.2.24 [Dots_engraver], page 214
Create Section 3.1.31 [Dots], page 279 objects for Section 3.2.80 [rhythmic-head-interface], page 397s.
This engraver creates the following layout object(s):
Section 3.1.31 [Dots], page 279.

Section 2.2.66 [Note_heads_engraver], page 228
Generate note heads.
Music types accepted:
Section 1.2.34 [note-event], page 39
Properties (read)
\begin{itemize}
  \item \texttt{middleCPosition} (number)
    The place of the middle C, measured in half staff-spaces. Usually determined by looking at \texttt{middleCClefPosition} and \texttt{middleCOffset}.
  \item \texttt{staffLineLayoutFunction} (procedure)
    Layout of staff lines, \texttt{traditional}, or \texttt{semitone}.
\end{itemize}
This engraver creates the following layout object(s):
Section 3.1.69 [NoteHead], page 310.

Section 2.2.13 [Breathing_sign_engraver], page 210
Create a breathing sign.
Music types accepted:
Section 1.2.12 [breathing-event], page 36
This engraver creates the following layout object(s):
Section 3.1.23 [BreathingSign], page 273.

Section 2.2.38 [Glissando_engraver], page 219
Engrave glissandi.
Music types accepted:
Section 1.2.19 [glissando-event], page 37
This engraver creates the following layout object(s):
Section 3.1.39 [Glissando], page 288.
Section 2.2.65 [Note_head_line_engraver], page 228
Engrave a line between two note heads, for example a glissando. If followVoice is set, staff switches also generate a line.
Properties (read)

followVoice (boolean)
If set, note heads are tracked across staff switches by a thin line.

This engraver creates the following layout object(s):
Section 3.1.39 [Glissando], page 288 and Section 3.1.126 [VoiceFollower], page 356.

Section 2.2.85 [Repeat_tie_engraver], page 234
Create repeat ties.
Music types accepted:
Section 1.2.43 [repeat-tie-event], page 40
This engraver creates the following layout object(s):
Section 3.1.82 [RepeatTie], page 320 and Section 3.1.83 [RepeatTieColumn], page 321.

Section 2.2.52 [Laissez_vibrer_engraver], page 223
Create laissez vibrer items.
Music types accepted:
Section 1.2.24 [laissez-vibrer-event], page 37
This engraver creates the following layout object(s):
Section 3.1.50 [LaissezVibrerTie], page 297 and Section 3.1.51 [LaissezVibrerTieColumn], page 297.

Section 2.2.36 [Forbid_line_break_engraver], page 218
Forbid line breaks when note heads are still playing at some point.
Properties (read)

busyGrobs (list)
A queue of (end-moment . GROB) cons cells.
This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Properties (write)

forbidBreak (boolean)
If set to ##t, prevent a line break at this point.

Section 2.2.44 [Grob_pq_engraver], page 220
Administrate when certain grobs (e.g., note heads) stop playing.
Properties (read)

busyGrobs (list)
A queue of (end-moment . GROB) cons cells.
This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Properties (write)
busyGrobs (list)
A queue of (end-moment . GROB) cons cells. This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Section 2.2.124 [Trill_spanner_engraver], page 244
Create trill spanner from an event.
Music types accepted:
Section 1.2.64 [trill-span-event], page 42
Properties (read)

  currentCommandColumn (layout object)
  Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

  currentMusicalColumn (layout object)
  Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):
Section 3.1.118 [TrillSpanner], page 349.

Section 2.2.117 [Text_spanner_engraver], page 242
Create text spanner from an event.
Music types accepted:
Section 1.2.60 [text-span-event], page 42
This engraver creates the following layout object(s):
Section 3.1.111 [TextSpanner], page 343.

Section 2.2.62 [Multi_measure_rest_engraver], page 226
Engrave multi-measure rests that are produced with ‘R’. It reads measurePosition and internalBarNumber to determine what number to print over the Section 3.1.63 [MultiMeasureRest], page 305. Reads measureLength to determine whether it should use a whole rest or a breve rest to represent one measure.
Music types accepted:
Section 1.2.32 [multi-measure-text-event], page 38 and Section 1.2.31 [multi-measure-rest-event], page 38
Properties (read)

  internalBarNumber (integer)
  Contains the current barnumber. This property is used for internal timekeeping, among others by the Accidental_engraver.

  restNumberThreshold (number)
  If a multimeasure rest has more measures than this, a number is printed.

  breakableSeparationItem (layout object)
  The breakable items in this time step, for this staff.
currentCommandColumn (layout object)
Grob that is X-parent to all current breakable
( Clef, key signature, etc.) items.

measurePosition (moment)
How much of the current measure have we had.
This can be set manually to create incomplete
measures.

measureLength (moment)
Length of one measure in the current time sig-
nature.

This engraver creates the following layout object(s):
Section 3.1.63 [MultiMeasureRest], page 305, Section 3.1.64 [MultiMea-
ureRestNumber], page 306 and Section 3.1.65 [MultiMeasureRestText],
page 307.

Section 2.2.3 [Arpeggio_engraver], page 207
Generate an Arpeggio symbol.
Music types accepted:
Section 1.2.5 [arpeggio-event], page 35
This engraver creates the following layout object(s):
Section 3.1.9 [Arpeggio], page 263.

Section 2.2.72 [Output_property_engraver], page 229
Apply a procedure to any grob acknowledged.
Music types accepted:
Section 1.2.4 [apply-output-event], page 35

Section 2.2.83 [Pitched_trill_engraver], page 233
Print the bracketed note head after a note head with trill.
This engraver creates the following layout object(s):
Section 3.1.115 [TrillPitchAccidental], page 346, Section 3.1.116 [Trill-
PitchGroup], page 347 and Section 3.1.117 [TrillPitchHead], page 348.

Section 2.2.35 [Font_size_engraver], page 217
Put fontSize into font-size grob property.
Properties (read)

fontSize (number)
The relative size of all grobs in a context.

2.1.16 NoteNames
(not documented)

This context creates the following layout object(s):
Section 3.1.70 [NoteName], page 311, Section 3.1.95 [StaffSpacing], page 329, Section 3.1.113
[TieColumn], page 345, Section 3.1.112 [Tie], page 344 and Section 3.1.125 [VerticalAxisGroup],
page 355.

This context sets the following properties:
• Set grob-property minimum-Y-extent in Section 3.1.125 [VerticalAxisGroup], page 355 to
#f.
This context is a ‘bottom’ context; it cannot contain other contexts.
This context is built from the following engraver(s):

Section 2.2.93 [Separating_line_group_engraver], page 236
Generate objects for computing spacing parameters.
Properties (read)

createSpacing (boolean)
Create StaffSpacing objects? Should be set for staves.

Properties (write)

hasStaffSpacing (boolean)
True if the current CommandColumn contains items that will affect spacing.

This engraver creates the following layout object(s):
Section 3.1.95 [StaffSpacing], page 329.

Section 2.2.67 [Note_name_engraver], page 228
Print pitches as words.
Music types accepted:
Section 1.2.34 [note-event], page 39
Properties (read)

printOctaveNames (boolean)
Print octave marks for the NoteNames context.

This engraver creates the following layout object(s):
Section 3.1.70 [NoteName], page 311.

Section 2.2.118 [Tie_engraver], page 242
Generate ties between note heads of equal pitch.
Music types accepted:
Section 1.2.61 [tie-event], page 42
Properties (read)

tieWaitForNote (boolean)
If true, tied notes do not have to follow each other directly. This can be used for writing out arpeggios.

Properties (write)

tieMelismaBusy (boolean)
Signal whether a tie is present.

This engraver creates the following layout object(s):
Section 3.1.112 [Tie], page 344 and Section 3.1.113 [TieColumn], page 345.

Section 2.2.94 [Skip_event_swallow_translator], page 236
Swallow \skip.

Section 2.2.88 [Rest_swallow_translator], page 235
Swallow rest.
Section 2.2.5 [Axis_group_engraver], page 208

Group all objects created in this context in a VerticalAxisGroup spanner.

Properties (read)

currentCommandColumn (layout object)
   Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

This engraver creates the following layout object(s):

Section 3.1.125 [VerticalAxisGroup], page 355.

2.1.17 PianoStaff

Just like GrandStaff but with support for instrument names at the start of each system.

This context also accepts commands for the following context(s):

   GrandStaff.

This context creates the following layout object(s):

Section 3.1.9 [Arpeggio], page 263, Section 3.1.46 [InstrumentName], page 293, Section 3.1.94 [SpanBar], page 328, Section 3.1.105 [SystemStartBar], page 337, Section 3.1.106 [SystemStartBrace], page 338, Section 3.1.107 [SystemStartBracket], page 339 and Section 3.1.108 [SystemStartSquare], page 340.

This context sets the following properties:

• Set translator property shortInstrumentName to '().
• Set translator property instrumentName to '().
• Set translator property systemStartDelimiter to 'SystemStartBrace.
• Set translator property localKeySignature to '().

Context PianoStaff can contain Section 2.1.20 [Staff], page 145 and Section 2.1.7 [FiguredBass], page 78.

This context is built from the following engraver(s):

Section 2.2.48 [Instrument_name_engraver], page 221

Create a system start text for instrument or vocal names.

Properties (read)

currentCommandColumn (layout object)
   Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

shortInstrumentName (markup)
   See 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.

shortVocalName (markup)
   Name of a vocal line, short version.

vocalName (markup)
   Name of a vocal line.
This engraver creates the following layout object(s):
Section 3.1.46 [InstrumentName], page 293.

Section 2.2.111 [System_start_delimiter_engraver], page 240
Create a system start delimiter (i.e., a SystemStartBar, SystemStartBrace, SystemStartBracket or SystemStartSquare spanner).
Properties (read)

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.

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

This engraver creates the following layout object(s):
Section 3.1.105 [SystemStartBar], page 337, Section 3.1.106 [SystemStartBrace], page 338, Section 3.1.107 [SystemStartBracket], page 339 and Section 3.1.108 [SystemStartSquare], page 340.

Section 2.2.99 [Span_arpeggio_engraver], page 238
Make arpeggios that span multiple staves.
Properties (read)

connectArpeggios (boolean)
If set, connect arpeggios across piano staff.

This engraver creates the following layout object(s):
Section 3.1.9 [Arpeggio], page 263.

Section 2.2.100 [Span_bar_engraver], page 238
Make cross-staff bar lines: It catches all normal bar lines and draws a single span bar across them.
This engraver creates the following layout object(s):
Section 3.1.94 [SpanBar], page 328.

2.1.18 RhythmicStaff
A context like Staff but for printing rhythms. Pitches are ignored; the notes are printed on one line.

This context also accepts commands for the following context(s):
Staff.

This context creates the following layout object(s):
Section 3.1.11 [BarLine], page 264, Section 3.1.30 [DotColumn], page 279, Section 3.1.46 [InstrumentName], page 293, Section 3.1.52 [LedgerLineSpanner], page 298, Section 3.1.95 [StaffSpacing], page 329, Section 3.1.96 [StaffSymbol], page 329, Section 3.1.114 [TimeSignature], page 345 and Section 3.1.125 [VerticalAxisGroup], page 355.

This context sets the following properties:
• Set grob-property `neutral-direction` in Section 3.1.19 [Beam], page 270 to 1.
• Set grob-property `neutral-direction` in Section 3.1.98 [Stem], page 330 to 1.
• Set grob-property `line-count` in Section 3.1.96 [StaffSymbol], page 329 to 1.
• Set grob-property `staff-padding` in Section 3.1.127 [VoltaBracket], page 356 to 3.
• Set grob-property `bar-size` in Section 3.1.11 [BarLine], page 264 to 4.
• Set translator property `squashedPosition` to 0.
• Set translator property `createSpacing` to #t.
• Set translator property `localKeySignature` to '().
• Set grob-property `minimum-Y-extent` in Section 3.1.125 [VerticalAxisGroup], page 355 to #f.

Context RhythmicStaff can contain Section 2.1.26 [Voice], page 194 and Section 2.1.3 [CueVoice], page 50.

This context is built from the following engraver(s):

Section 2.2.53 [Ledger_line_engraver], page 223
Create the spanner to draw ledger lines, and notices objects that need ledger lines.
This engraver creates the following layout object(s):
Section 3.1.52 [LedgerLineSpanner], page 298.

Section 2.2.5 [Axis_group_engraver], page 208
Group all objects created in this context in a VerticalAxisGroup spanner.
Properties (read)

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.
This engraver creates the following layout object(s):
Section 3.1.125 [VerticalAxisGroup], page 355.

Section 2.2.48 [Instrument_name_engraver], page 221
Create a system start text for instrument or vocal names.
Properties (read)

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

shortInstrumentName (markup)
See 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.

shortVocalName (markup)
Name of a vocal line, short version.

vocalName (markup)
Name of a vocal line.
This engraver creates the following layout object(s):
Section 3.1.46 [InstrumentName], page 293.

Section 2.2.120 [Time_signature_engraver], page 243
Create a Section 3.1.114 [TimeSignature], page 345 whenever timeSignatureFraction changes.

Properties (read)

  implicitTimeSignatureVisibility (vector)
  break visibility for the default time signature.

  timeSignatureFraction (pair of numbers)
  A pair of numbers, signifying the time signature. For example, #'(4 . 4) is a 4/4 time signature.

This engraver creates the following layout object(s):
Section 3.1.114 [TimeSignature], page 345.

Section 2.2.82 [Pitch_squash_engraver], page 233
Set the vertical position of note heads to squashedPosition, if that property is set. This can be used to make a single-line staff demonstrating the rhythm of a melody.

Properties (read)

  squashedPosition (integer)
  Vertical position of squashing for Section “Pitch_squash_engraver” in Internals Reference.

Section 2.2.104 [Staff_symbol_engraver], page 238
Create the constellation of five (default) staff lines.

Music types accepted:
Section 1.2.55 [staff-span-event], page 41
This engraver creates the following layout object(s):
Section 3.1.96 [StaffSymbol], page 329.

Section 2.2.7 [Bar_engraver], page 208
Create barlines. This engraver is controlled through the whichBar property. If it has no bar line to create, it will forbid a linebreak at this point.

Properties (read)

  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 Section “barline-interface” in Internals Reference.

Properties (write)

  forbidBreak (boolean)
  If set to ##t, prevent a line break at this point.
This engraver creates the following layout object(s):
Section 3.1.11 [BarLine], page 264.

**Section 2.2.23 [Dot_column_engraver], page 214**
Engrave dots on dotted notes shifted to the right of the note. If omitted, then dots appear on top of the notes.
This engraver creates the following layout object(s):
Section 3.1.30 [DotColumn], page 279.

**Section 2.2.93 [Separating_line_group_engraver], page 236**
Generate objects for computing spacing parameters.
Properties (read)

\[createSpacing \text{(boolean)}\]
Create StaffSpacing objects? Should be set for staves.

Properties (write)

\[hasStaffSpacing \text{(boolean)}\]
True if the current CommandColumn contains items that will affect spacing.

This engraver creates the following layout object(s):
Section 3.1.95 [StaffSpacing], page 329.

**Section 2.2.35 [Font_size_engraver], page 217**
Put fontSize into font-size grob property.
Properties (read)

\[fontSize \text{(number)}\]
The relative size of all grobs in a context.

**Section 2.2.72 [Output_property_engraver], page 229**
Apply a procedure to any grob acknowledged.
Music types accepted:

- Section 1.2.4 [apply-output-event], page 35

### 2.1.19 Score

This is the top level notation context. No other context can contain a Score context. This context handles the administration of time signatures. It also makes sure that items such as clefs, time signatures, and key-signatures are aligned across staves.

You cannot explicitly instantiate a Score context (since it is not contained in any other context). It is instantiated automatically when an output definition (a `\score` or `\layout` block) is processed.

This context creates the following layout object(s):
Section 3.1.12 [BarNumber], page 266, Section 3.1.21 [BreakAlignGroup], page 272, Section 3.1.22 [BreakAlignment], page 272, Section 3.1.40 [GraceSpacing], page 289, Section 3.1.53 [LeftEdge], page 298, Section 3.1.62 [MetronomeMark], page 304, Section 3.1.66 [NonMusicalPaperColumn], page 308, Section 3.1.74 [PaperColumn], page 314, Section 3.1.75 [ParenthesesItem], page 314, Section 3.1.80 [RehearsalMark], page 318, Section 3.1.93 [SpacingSpanner], page 327, Section 3.1.105 [SystemStartBar], page 337, Section 3.1.106 [SystemStartBrace], page 338, Section 3.1.107 [SystemStartBracket], page 339, Section 3.1.108 [SystemStartSquare], page 340, Section 3.1.124 [VerticalAlignment], page 354, Section 3.1.128 [VoltaBracketSpanner], page 357 and Section 3.1.127 [VoltaBracket], page 356.
This context sets the following properties:

- Set translator property `timing` to #t.
- Set translator property `verticallySpacedContexts` to '(Staff).
- Set translator property `instrumentTransposition` to #<Pitch c'>.
- Set translator property `quotedEventTypes` to '(note-event rest-event tie-event beam-event tuplet-span-event).
- Set translator property `keepAliveInterfaces` to '(rhythmic-grob-interface lyric-interface percent-repeat-item-interface percent-repeat-interface stanza-number-interface).
- Set translator property `graceSettings` to '((Voice Stem direction 1) (Voice Stem font-size -3) (Voice NoteHead font-size -3) (Voice Dots font-size -3) (Voice Stem length-fraction 0.8) (Voice Stem no-stem-extend #t) (Voice Beam thickness 0.384) (Voice Beam length-fraction 0.8) (Voice Accidental font-size -4) (Voice Script font-size -3)).
- Set translator property `metronomeMarkFormatter` to format-metronome-markup.
- Set translator property `figuredBassFormatter` to format-bass-figure.
- Set translator property `tablatureFormat` to fret-number-tablature-format.
- Set translator property `stringTunings` to '(4 -1 -5 -10 -15 -20).
- Set translator property `highStringOne` to #t.
- Set translator property `bassStaffProperties` to '((assign clefGlyph clefs.F) (assign clefPosition 2) (assign middleCPosition 6) (assign middleCClefPosition 6)).
- Set translator property `chordNameExceptionsPartial` to '(((#<Pitch c'> #<Pitch d'>) (#<procedure line-markup (layout props args)> ((#<procedure normal-size-super-markup (layout props arg)> 2)))) ((#<Pitch c'> #<Pitch ees'>) (#<procedure line-markup (layout props args)> (m))) ((#<Pitch c'> #<Pitch f'>) (#<procedure line-markup (layout props args)> ((#<procedure normal-size-super-markup (layout props arg)> sus4)))(#<Pitch c'> #<Pitch g'>) (#<procedure line-markup (layout props args)> ((#<procedure normal-size-super-markup (layout props arg)> 5)))) ((#<Pitch c'> #<Pitch ees'> #<Pitch f'>) (#<procedure line-markup (layout props args)> (m)) (#<procedure line-markup (layout props args)> ((#<procedure normal-size-super-markup (layout props arg)> sus4))))((#<Pitch c'> #<Pitch d'> #<Pitch ees'>) (#<procedure line-markup (layout props args)> (m)) (#<procedure line-markup (layout props args)> ((#<procedure normal-size-super-markup (layout props arg)> sus2))))).
- Set translator property `chordNameExceptionsFull` to '(((#<Pitch c'> #<Pitch e'> #<Pitch ees'> #<Pitch gis'>) (#<procedure line-markup (layout props args)> (+))) ((#<Pitch c'> #<Pitch ees'> #<Pitch gis'>) (#<procedure line-markup (layout props args)> (o))) ((#<Pitch c'> #<Pitch ees'> #<Pitch ges'>) (#<procedure line-markup (layout props args)> (o7))) ((#<Pitch c'> #<Pitch gis'> #<Pitch ges'> #<Pitch beses'>) (#<procedure line-markup (layout props args)> (o7))).
- Set translator property `chordPrefixSpacer` to 0.
- Set translator property `chordRootNamer` to note-name->markup.
- Set translator property `chordNoteNamer` to '().
• Set translator property `chordNameExceptions` to `(((<Pitch e'> #<Pitch gis'>) #<procedure line-markup (layout props args)> (+)) ((<Pitch ees'> #<Pitch ges'>) #<procedure line-markup (layout props args)> ((<procedure super-markup (layout props arg)> o))) ((<Pitch ees'> #<Pitch ges'> #<Pitch bes'>) #<procedure line-markup (layout props args)> ((<procedure super-markup (layout props arg)> ))) ((<Pitch ees'> #<Pitch ges'> #<Pitch beses'>) #<procedure line-markup (layout props args)> ((<procedure super-markup (layout props arg)> o7)))).

• Set translator property `chordNameSeparator` to `(#<procedure simple-markup (layout props str)> /)

• Set translator property `majorSevenSymbol` to `(#<procedure line-markup (layout props args)> ((<procedure triangle-markup (layout props filled)> #f)))`

• Set translator property `chordNameFunction` to `ignatzek-chord-names`

• Set translator property `barCheckSynchronize` to `#f`

• Set translator property `keyAlterationOrder` to `((6 . -1/2) (2 . -1/2) (5 . -1/2) (1 . -1/2) (4 . -1/2) (0 . -1/2) (3 . -1/2) (3 . 1/2) (0 . 1/2) (4 . 1/2) (1 . 1/2) (5 . 1/2) (2 . 1/2) (6 . 1/2) (6 . -1) (2 . -1) (5 . -1) (1 . -1) (4 . -1) (0 . -1) (3 . -1) (3 . 1) (0 . 1) (4 . 1) (2 . 1) (5 . 1) (2 . 1) (6 . 1))`.

• Set translator property `printKeyCancellation` to `#t`

• Set translator property `autoCautionaries` to `()`

• Set translator property `extraNatural` to `#t`

• Set translator property `rehearsalMark` to `1`

• Set translator property `markFormatter` to `format-mark-letters`

• Set translator property `lyricMelismaAlignment` to `-1`

• Set translator property `strokeFingerOrientations` to `'(right)`

• Set translator property `stringNumberOrientations` to `'(up down)`

• Set translator property `fingeringOrientations` to `'(up down)`

• Set translator property `harmonicAccidentals` to `#t`

• Set translator property `pedalSostenutoStyle` to `'mixed`

• Set translator property `pedalSostenutoStrings` to `'(Sost. Ped. *Sost. Ped. *)`

• Set translator property `pedalUnaCordaStyle` to `'text`

• Set translator property `pedalUnaCordaStrings` to `'(una corda tre corde)`

• Set translator property `pedalSustainStyle` to `'text`

• Set translator property `pedalSustainStrings` to `(Ped. *Ped. *)`

• Set translator property `scriptDefinitions` to `(((thumb (script-stencil feta thumb . thumb) (avoid-slur . inside) (padding . 0.2) (direction . 1)) (accent (avoid-slur . around) (padding . 0.2) (quantize-position . #t) (script-stencil feta sforzato . sforzato) (side-relative-direction . -1)) (espressivo (avoid-slur . around) (padding . 0.2) (quantize-position . #t) (script-stencil feta espr . espr) (side-relative-direction . -1)) (marcato (script-stencil feta dmarcato . umarcato) (padding . 0.2) (avoid-slur . inside) (quantize-position . #t) (side-relative-direction . -1)) (staccatissimo (avoid-slur . inside) (script-stencil feta dstaccatissimo . ustaccatissimo) (padding . 0.2) (side-relative-direction . -1)) (portato (script-stencil feta uportato
Chapter 2: Translation

. dportato) (avoid-slur . around) (slur-padding . 0.3) (padding . 0.45)
(side-relative-direction . -1)) (accentus (script-stencil feta uaccentus
 . uaccentus) (side-relative-direction . -1) (avoid-slur . #f) (padding
 . 0.2) (quantize-position . #t) (script-priority . -100) (direction . 1))
(ictus (script-stencil feta ictus . ictus) (side-relative-direction . -1)
(quantize-position . #t) (avoid-slur . #f) (padding . 0.2) (script-priority
 . -100) (direction . -1)) (semicirculus (script-stencil feta dsemicircus
 . dsemicircus) (side-relative-direction . -1) (quantize-position . #t)
(avoid-slur . #f) (padding . 0.2) (script-priority . -100) (direction . 1))
circulus (script-stencil feta circulus . circulus) (side-relative-direction
 . -1) (avoid-slur . #f) (padding . 0.2) (quantize-position . #t) (script-
priority . -100) (direction . 1)) (signumcongruentiae (script-stencil feta
dsignumcongruentiae . usignumcongruentiae) (padding . 0.2) (avoid-slur
 . outside) (direction . 1)) (fermata (script-stencil feta dfermata . ufermata)
(padding . 0.2) (avoid-slur . around) (script-priority . 4000) (direction . 1))
(shortfermata (script-stencil feta dshortfermata . ushortfermata) (padding
 . 0.2) (avoid-slur . around) (direction . 1)) (longfermata (script-stencil
feta dlongfermata . ulongfermata) (padding . 0.2) (avoid-slur . around)
(direction . 1)) (verylongfermata (script-stencil feta dverylongfermata
 . uverylongfermata) (padding . 0.2) (avoid-slur . around) (direction . 1))
(stopped (script-stencil feta stopped . stopped) (avoid-slur . inside)
(padding . 0.2) (direction . 1)) (staccato (script-stencil feta staccato
 . staccato) (side-relative-direction . -1) (quantize-position . #t) (avoid-slur
 . inside) (toward-stem-shift . 0.5) (padding . 0.2) (script-priority . -100)
(tenuto (script-stencil feta tenuto . tenuto) quantize-position . #t)
(avoid-slur . inside) (padding . 0.2) (side-relative-direction . -1)) (comma
(script-stencil feta lcomma . rcomma) (quantize-position . #t) (padding
 . 0.2) (avoid-slur . #f) (direction . 1)) (varcomma (script-stencil feta
lvarcomma . rvarcomma) (quantize-position . #t) (padding . 0.2) (avoid-slur
 . #f) (direction . 1)) (upbow (script-stencil feta upbow . upbow) (avoid-slur
 . around) (padding . 0.2) (direction . 1)) (downbow (script-stencil feta downbow
 . downbow) (padding . 0.2) (avoid-slur . around) (direction . 1)) (lheel
(script-stencil feta upedalheel . upedalheel) (padding . 0.2) (avoid-slur
 . around) (direction . -1)) (rheel (script-stencil feta dpedalheel . dpedalheel)
(padding . 0.2) (avoid-slur . around) (direction . 1)) (ltoe (script-stencil
feta upedaltoe . upedaltoe) (padding . 0.2) (avoid-slur . around) (direction
 . -1)) (rtoe (script-stencil feta dpedaltoe . dpedaltoe) (padding . 0.2)
(avoid-slur . around) (direction . 1)) (turn (script-stencil feta turn
 . turn) (avoid-slur . inside) (padding . 0.2) (direction . 1)) (open (avoid-slur
 . outside) (padding . 0.2) (script-stencil feta open . open) (direction
 . 1)) (flageolet (script-stencil feta flageolet . flageolet) (padding . 0.2)
(avoid-slur . around) (direction . 1)) (reverseturn (script-stencil feta
reverseturn . reverseturn) (padding . 0.2) (avoid-slur . inside) (direction
 . 1)) (trill (script-stencil feta trill . trill) (direction . 1) (padding . 0.2)
(avoid-slur . outside) (script-priority . 2000)) (prall (script-stencil feta
prall . prall) (padding . 0.2) (avoid-slur . around) (direction . 1)) (mordent
(script-stencil feta mordent . mordent) (padding . 0.2) (avoid-slur . around)
(direction . 1)) (prallprall (script-stencil feta prallprall . prallprall)
(padding . 0.2) (avoid-slur . around) (direction . 1)) (prallmordent (script-
stencil feta prallmordent . prallmordent) (padding . 0.2) (avoid-slur . around)
(direction . 1)) (upprall (script-stencil feta upprall . upprall) (padding
 . 0.2) (avoid-slur . around) (direction . 1)) (downprall (script-stencil feta
downprall . downprall) (padding . 0.2) (avoid-slur . around) (direction . 1))
downprall . downprall) (padding . 0.2) (avoid-slur . around) (direction . 1)) (upmordent (script-stencil feta upmordent . upmordent) (padding . 0.2) (avoid-slur . around) (direction . 1)) (downmordent (script-stencil feta downmordent . downmordent) (padding . 0.2) (avoid-slur . around) (direction . 1)) (lineprall (script-stencil feta lineprall . lineprall) (padding . 0.2) (avoid-slur . around) (direction . 1)) (pralldown (script-stencil feta pralldown . pralldown) (padding . 0.2) (avoid-slur . around) (direction . 1)) (prallup (script-stencil feta prallup . prallup) (padding . 0.2) (avoid-slur . around) (direction . 1)) (segno (script-stencil feta segno . segno) (padding . 0.2) (avoid-slur . outside) (direction . 1)) (coda (script-stencil feta coda . coda) (padding . 0.2) (avoid-slur . outside) (direction . 1)) (varcoda (script-stencil feta varcoda . varcoda) (padding . 0.2) (avoid-slur . outside) (direction . 1))

- Set translator property autoBeamCheck to default-auto-beam-check.
- Set translator property autoBeamSettings to '(((end 1 32 2 2) . #<Mom 1/4>) ((end 1 32 2 2) . #<Mom 1/2>) ((end 1 32 2 2) . #<Mom 3/4>) ((end 1 32 3 2) . #<Mom 1/4>) ((end 1 32 3 2) . #<Mom 1/2>) ((end 1 32 3 2) . #<Mom 3/4>) ((end 1 32 3 2) . #<Mom 1/8>) ((end 1 32 3 2) . #<Mom 1/4>) ((end 1 32 3 2) . #<Mom 1/2>) ((end 1 32 3 2) . #<Mom 3/8>) ((end 1 32 3 2) . #<Mom 5/8>) ((end 1 32 3 2) . #<Mom 7/8>) ((end 1 32 3 2) . #<Mom 1>) ((end 1 32 3 2) . #<Mom 9/8>) ((end 1 32 3 2) . #<Mom 5/4>) ((end 1 32 3 2) . #<Mom 11/8>) ((end 1 32 3 2) . #<Mom 1/8>) ((end 1 32 3 2) . #<Mom 3/8>) ((end 1 32 3 2) . #<Mom 1/2>) ((end 1 32 3 2) . #<Mom 3/4>) ((end 1 32 3 2) . #<Mom 1/4>) ((end 1 32 3 2) . #<Mom 1/8>) ((end 1 32 3 2) . #<Mom 1/4>) ((end 1 32 3 2) . #<Mom 1/2>) ((end 1 32 3 2) . #<Mom 3/8>) ((end 1 32 3 2) . #<Mom 5/8>) ((end 1 32 3 2) . #<Mom 7/8>) ((end 1 32 3 2) . #<Mom 1>) ((end 1 32 3 2) . #<Mom 9/8>) ((end 1 32 3 2) . #<Mom 5/4>) ((end 1 32 3 2) . #<Mom 11/8>) ((end 1 32 3 2) . #<Mom 1/8>) ((end 1 32 3 2) . #<Mom 3/8>) ((end 1 32 3 2) . #<Mom 1/2>) ((end 1 32 3 2) . #<Mom 3/4>) ((end 1 32 3 2) . #<Mom 1/4>) ((end 1 32 3 2) . #<Mom 1/8>) ((end 1 32 3 2) . #<Mom 1/4>) ((end 1 32 3 2) . #<Mom 1/2>) ((end 1 32 3 2) . #<Mom 3/8>) ((end 1 32 3 2) . #<Mom 5/8>) ((end 1 32 3 2) . #<Mom 7/8>) ((end 1 32 3 2) . #<Mom 1>) ((end 1 32 3 2) . #<Mom 9/8>) ((end 1 32 3 2) . #<Mom 5/4>) ((end 1 32 3 2) . #<Mom 11/8>) ((end 1 32 3 2) . #<Mom 1/8>) ((end 1 32 3 2) . #<Mom 3/8>) ((end 1 32 3 2) . #<Mom 1/2>) ((end 1 32 3 2) . #<Mom 3/4>) ((end 1 32 3 2) . #<Mom 1/4>) ((end 1 32 3 2) . #<Mom 1/8>) ((end 1 32 3 2) . #<Mom 1/4>) ((end 1 32 3 2) . #<Mom 1/2>) ((end 1 32 3 2) . #<Mom 3/8>) ((end 1 32 3 2) . #<Mom 5/8>) ((end 1 32 3 2) . #<Mom 7/8>) ((end 1 32 3 2) . #<Mom 1>) ((end 1 32 3 2) . #<Mom 9/8>) ((end 1 32 3 2) . #<Mom 5/4>) ((end 1 32 3 2) . #<Mom 11/8>) ((end 1 32 3 2) . #<Mom 1/8>)

- Set translator property repeatCountVisibility to all-repeat-counts-visible.
- Set translator property implicitTimeSignatureVisibility to #(#f #t #t).
- Set translator property explicitKeySignatureVisibility to #(#t #t #t).
- Set translator property explicitClefVisibility to #(#t #t #t).
- Set translator property automaticBars to #t.
- Set translator property barNumberVisibility to first-bar-number-invisible.
- Set translator property doubleRepeatType to "::*:.
- Set translator property defaultBarType to "|".
• Set translator property `decrescendoSpanner` to 'hairpin'.
• Set translator property `crescendoSpanner` to 'hairpin'.
• Set translator property `firstClef` to #t.
• Set translator property `middleCPosition` to -6.
• Set translator property `middleCClefPosition` to -6.
• Set translator property `clefPosition` to -2.
• Set translator property `clefGlyph` to "clefs.G".
• Set translator property `tieWaitForNote` to #f.
• Set translator property `melismaBusyProperties` to '(melismaBusy slurMelismaBusy tieMelismaBusy beamMelismaBusy completionBusy)
• Set translator property `drumStyleTable` to #<hash-table 29/61>.
• Set translator property `systemStartDelimiter` to 'SystemStartBar.'
• Set translator property `printPartCombineTexts` to #t.
• Set translator property `aDueText` to "a2".
• Set translator property `soloIIText` to "Solo II".
• Set translator property `soloText` to "Solo".
• Set translator property `noteToFretFunction` to `determine-frets`. 

Context Score can contain Section 2.1.20 [Staff], page 145, Section 2.1.7 [FiguredBass], page 78, Section 2.1.16 [NoteNames], page 127, Section 2.1.4 [Devnull], page 62, Section 2.1.17 [PianoStaff], page 129, Section 2.1.1 [ChoirStaff], page 48, Section 2.1.10 [GrandStaff], page 82, Section 2.1.2 [ChordNames], page 48, Section 2.1.13 [Lyrics], page 104, Section 2.1.5 [DrumStaff], page 62, Section 2.1.21 [StaffGroup], page 154, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.24 [VaticanaStaff], page 173, Section 2.1.22 [TabStaff], page 155, Section 2.1.18 [RhythmicStaff], page 130 and Section 2.1.8 [FretBoards], page 80.

This context is built from the following engraver(s):

**Section 2.2.75 [Parenthesis_engraver], page 230**
Parenthesize objects whose music cause has the `parenthesize` property.
This engraver creates the following layout object(s):
Section 3.1.75 [ParenthesesItem], page 314.

**Section 2.2.8 [Bar_number_engraver], page 209**
A bar number is created whenever `measurePosition` is zero and when there is a bar line (i.e., when `whichBar` is set). It is put on top of all staves, and appears only at the left side of the staff. The staves are taken from `stavesFound`, which is maintained by Section 2.2.102 [Staff_collecting_engraver], page 238.

Properties (read)

`currentBarNumber` (integer)
Contains the current bar number. This property is incremented at every bar line.

`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 Section “bar-line-interface” in Internals Reference.

stavesFound (list of grobs)
A list of all staff-symbols found.

barNumberVisibility (procedure)
A Procedure that takes an integer and returns whether the corresponding bar number should be printed.

This engraver creates the following layout object(s):
Section 3.1.12 [BarNumber], page 266.

Section 2.2.105 [Stanza_number_align_engraver], page 239
This engraver ensures that stanza numbers are neatly aligned.

Section 2.2.128 [Vertical_align_engraver], page 245
Catch groups (staves, lyrics lines, etc.) and stack them vertically.
Properties (read)
alignAboveContext (string)
Where to insert newly created context in vertical alignment.

alignBelowContext (string)
Where to insert newly created context in vertical alignment.

This engraver creates the following layout object(s):
Section 3.1.124 [VerticalAlignment], page 354.

Section 2.2.41 [Grace_spacing_engraver], page 220
Bookkeeping of shortest starting and playing notes in grace note runs.
Properties (read)
currentMusicalColumn (layout object)
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):
Section 3.1.40 [GraceSpacing], page 289.

Section 2.2.98 [Spacing_engraver], page 237
Make a SpacingSpanner and do bookkeeping of shortest starting and playing notes.

Music types accepted:
Section 1.2.52 [spacing-section-event], page 41
Properties (read)
currentMusicalColumn (layout object)
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).
currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.
proportionalNotationDuration (moment)
   Global override for shortest-playing duration.
   This is used for switching on proportional notation.

This engraver creates the following layout object(s):
Section 3.1.93 [SpacingSpanner], page 327.

Section 2.2.12 [Break_align_engraver], page 210
Align grobs with corresponding break-align-symbols into groups, and
order the groups according to breakAlignOrder. The left edge of the
alignment gets a separate group, with a symbol left-edge.
This engraver creates the following layout object(s):
Section 3.1.21 [BreakAlignGroup], page 272, Section 3.1.22 [BreakAlignment], page 272 and Section 3.1.53 [LeftEdge], page 298.

Section 2.2.61 [Metronome_mark_engraver], page 225
Engrave metronome marking. This delegates the formatting work to
the function in the metronomeMarkFormatter property. The mark is
put over all staves. The staves are taken from the stavesFound prop-
erty, which is maintained by Section 2.2.102 [Staff_collecting_engraver],
page 238.
Properties (read)

stavesFound (list of grobs)
   A list of all staff-symbols found.

metronomeMarkFormatter (procedure)
   How to produce a metronome markup. Called
   with four arguments: text, duration, count and
   context.

tempoUnitDuration (duration)
   Unit for specifying tempo.

tempoUnitCount (number)
   Count for specifying tempo.

tempoText (markup)
   Text for tempo marks.

tempoHideNote (boolean)
   Hide the note=count in tempo marks.

This engraver creates the following layout object(s):
Section 3.1.62 [MetronomeMark], page 304.

Section 2.2.130 [Volta_engraver], page 245
Make volta brackets.
Properties (read)

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.

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.

**stavesFound** (list of grobs)
A list of all staff-symbols found.

This engraver creates the following layout object(s):
Section 3.1.127 [VoltaBracket], page 356 and Section 3.1.128 [VoltaBracketSpanner], page 357.

**Section 2.2.57 [Mark_engraver], page 224**
Create RehearsalMark objects. It puts them on top of all staves (which is taken from the property stavesFound). If moving this engraver to a different context, Section 2.2.102 [Staff_collecting_engraver], page 238 must move along, otherwise all marks end up on the same Y location.

Music types accepted:
Section 1.2.29 [mark-event], page 38

Properties (read)

**markFormatter** (procedure)
A procedure taking as arguments the context and the rehearsal mark. It should return the formatted mark as a markup object.

**rehearsalMark** (integer)
The last rehearsal mark printed.

**stavesFound** (list of grobs)
A list of all staff-symbols found.

This engraver creates the following layout object(s):
Section 3.1.80 [RehearsalMark], page 318.

**Section 2.2.111 [System_start_delimiter_engraver], page 240**
Create a system start delimiter (i.e., a SystemStartBar, SystemStartBrace, SystemStartBracket or SystemStartSquare spanner).

Properties (read)

**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.

**currentCommandColumn** (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

This engraver creates the following layout object(s):
Section 3.1.105 [SystemStartBar], page 337, Section 3.1.106 [SystemStartBrace], page 338, Section 3.1.107 [SystemStartBracket], page 339 and Section 3.1.108 [SystemStartSquare], page 340.
Section 2.2.72 [Output_property_engraver], page 229
Apply a procedure to any grob acknowledged.
Music types accepted:
Section 1.2.4 [apply-output-event], page 35

Section 2.2.22 [Default_bar_line_engraver], page 213
This engraver determines what kind of automatic bar lines should be produced, and sets whichBar accordingly. It should be at the same level as Section 2.2.122 [Timing_translator], page 243.
Properties (read)

**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.

**defaultBarType** (string)
Set the default type of bar line. See whichBar for information on available bar types.

This variable is read by Section “Timing_translator” in Internals Reference at Section “Score” in Internals Reference level.

**measureLength** (moment)
Length of one measure in the current time signature.

**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 Section “bar-line-interface” in Internals Reference.

**measurePosition** (moment)
How much of the current measure have we had. This can be set manually to create incomplete measures.

Properties (write)

**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.
Section 2.2.122 [Timing_translator], page 243
This engraver adds the alias Timing to its containing context. Responsible for synchronizing timing information from staves. Normally in Score. In order to create polyrhythmic music, this engraver should be removed from Score and placed in Staff.

Properties (read)

\begin{itemize}
\item \texttt{internalBarNumber} (integer)
  Contains the current bar number. This property is used for internal timekeeping, among others by the \texttt{Accidental_engraver}.
\item \texttt{currentBarNumber} (integer)
  Contains the current bar number. This property is incremented at every bar line.
\item \texttt{measureLength} (moment)
  Length of one measure in the current time signature.
\item \texttt{measurePosition} (moment)
  How much of the current measure have we had. This can be set manually to create incomplete measures.
\end{itemize}

Properties (write)

\begin{itemize}
\item \texttt{internalBarNumber} (integer)
  Contains the current bar number. This property is used for internal timekeeping, among others by the \texttt{Accidental_engraver}.
\item \texttt{currentBarNumber} (integer)
  Contains the current bar number. This property is incremented at every bar line.
\item \texttt{measurePosition} (moment)
  How much of the current measure have we had. This can be set manually to create incomplete measures.
\end{itemize}

Section 2.2.102 [Staff_collecting_engraver], page 238
Maintain the \texttt{stavesFound} variable.

Properties (read)

\begin{itemize}
\item \texttt{stavesFound} (list of grobs)
  A list of all staff-symbols found.
\end{itemize}

Properties (write)

\begin{itemize}
\item \texttt{stavesFound} (list of grobs)
  A list of all staff-symbols found.
\end{itemize}

Section 2.2.84 [Repeat_acknowledge_engraver], page 233
Acknowledge repeated music, and convert the contents of \texttt{repeatCommands} into an appropriate setting for \texttt{whichBar}.

Properties (read)

\begin{itemize}
\item \texttt{doubleRepeatType} (string)
  Set the default bar line for double repeats.
\end{itemize}
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.

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 Section “bar-line-interface” in Internals Reference.

Section 2.2.129 [Vertically_spaced_contexts_ engraver], page 245
Properties (read)
verticallySpacedContexts (list)
List of symbols, containing context names whose vertical axis groups should be taken into account for vertical spacing of systems.

Properties (write)
verticallySpacedContexts (list)
List of symbols, containing context names whose vertical axis groups should be taken into account for vertical spacing of systems.

Section 2.2.74 [Paper_column_ engraver], page 230
Take care of generating columns.
This engraver decides whether a column is breakable. The default is that a column is always breakable. However, every Bar_ engraver that does not have a barline at a certain point will set forbidBreaks in the score context to stop line breaks. In practice, this means that you can make a break point by creating a bar line (assuming that there are no beams or notes that prevent a break point).
Music types accepted:
Section 1.2.23 [label-event], page 37 and Section 1.2.11 [break-event], page 36
Properties (read)
forbidBreak (boolean)
If set to ##t, prevent a line break at this point.
Properties (write)
forbidBreak (boolean)
If set to ##t, prevent a line break at this point.
currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.
currentMusicalColumn (layout object)
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).
2.1.20 Staff

Handles clefs, bar lines, keys, accidentals. It can contain Voice contexts.

This context creates the following layout object(s):
Section 3.1.66 [NonMusicalPaperColumn], page 308 and Section 3.1.74 [PaperColumn], page 314.

This context creates the following layout object(s):
Section 3.1.14 [BassFigureAlignment], page 268, Section 3.1.16 [BassFigureBracket], page 269, Section 3.1.17 [BassFigureContinuation], page 269, Section 3.1.18 [BassFigureLine], page 270, Section 3.1.13 [BassFigure], page 267, Section 3.1.25 [Clef], page 275, Section 3.1.30 [DotColumn], page 279, Section 3.1.46 [InstrumentName], page 293, Section 3.1.48 [KeyCancellation], page 295, Section 3.1.49 [KeySignature], page 296, Section 3.1.52 [LedgerLineSpanner], page 298, Section 3.1.67 [NoteCollision], page 309, Section 3.1.72 [OctavateEight], page 311, Section 3.1.73 [OttavaBracket], page 312, Section 3.1.79 [PianoPedalBracket], page 318, Section 3.1.85 [RestCollision], page 322, Section 3.1.88 [ScriptRow], page 323, Section 3.1.92 [SostenutoPedalLineSpanner], page 326, Section 3.1.91 [SostenutoPedal], page 325, Section 3.1.95 [StaffSpacing], page 329, Section 3.1.96 [StaffSymbol], page 329, Section 3.1.103 [SustainPedalLineSpanner], page 335, Section 3.1.102 [SustainPedal], page 335, Section 3.1.114 [TimeSignature], page 345, Section 3.1.122 [UnaCordaPedalLineSpanner], page 353, Section 3.1.121 [UnaCordaPedal], page 352 and Section 3.1.125 [VerticalAxisGroup], page 355.

This context sets the following properties:
• Set translator property shortInstrumentName to '().
• Set translator property instrumentName to '().
• Set grob-property minimum-Y-extent in Section 3.1.125 [VerticalAxisGroup], page 355 to '(-4 . 4).
• Set translator property ignoreFiguredBassRest to #t.
• Set translator property createSpacing to #t.
• Set translator property localKeySignature to '().

Context Staff can contain Section 2.1.26 [Voice], page 194 and Section 2.1.3 [CueVoice], page 50.

This context is built from the following engraver(s):
Section 2.2.92 [Script_row_engraver], page 236
Determine order in horizontal side position elements.

This engraver creates the following layout object(s):
Section 3.1.88 [ScriptRow], page 323.

Section 2.2.33 [Figured_bass_position_engraver], page 217
Position figured bass alignments over notes.

This engraver creates the following layout object(s):
Section 3.1.15 [BassFigureAlignmentPositioning], page 268.

Section 2.2.32 [Figured_bass_engraver], page 216
Make figured bass numbers.

Music types accepted:
Section 1.2.7 [bass-figure-event], page 36 and Section 1.2.44 [rest-event], page 40

Properties (read)

- `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.

- `implicitBassFigures` (list)
  
  A list of bass figures that are not printed as numbers, but only as extender lines.

- `useBassFigureExtenders` (boolean)
  
  Whether to use extender lines for repeated bass figures.

- `ignoreFiguredBassRest` (boolean)
  
  Don’t swallow rest events.

This engraver creates the following layout object(s):

Section 3.1.13 [BassFigure], page 267, Section 3.1.14 [BassFigure-Alignment], page 268, Section 3.1.16 [BassFigureBracket], page 269, Section 3.1.17 [BassFigureContinuation], page 269 and Section 3.1.18 [BassFigureLine], page 270.

Section 2.2.5 [Axis_group_engraver], page 208

Group all objects created in this context in a VerticalAxisGroup spanner.

Properties (read)

- `currentCommandColumn` (layout object)
  
  Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

This engraver creates the following layout object(s):

Section 3.1.125 [VerticalAxisGroup], page 355.

Section 2.2.108 [String_number_engraver], page 240

Swallow string number events. The purpose of this engraver is to process tablatures for normal notation. To prevent warnings for unprocessed string number events to obscure real error messages, this engraver swallows them all.

Section 2.2.48 [Instrument_name_engraver], page 221

Create a system start text for instrument or vocal names.

Properties (read)

- `currentCommandColumn` (layout object)
  
  Grob that is X-parent to all current breakable (clef, key signature, etc.) items.
shortInstrumentName (markup)
See 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.

shortVocalName (markup)
Name of a vocal line, short version.

vocalName (markup)
Name of a vocal line.

This engraver creates the following layout object(s):
Section 3.1.46 [InstrumentName], page 293.

Section 2.2.79 [Piano_pedal_align_engraver], page 232
Align piano pedal symbols and brackets.
Properties (read)

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

This engraver creates the following layout object(s):
Section 3.1.92 [SostenutoPedalLineSpanner], page 326, Section 3.1.103 [SustainPedalLineSpanner], page 335 and Section 3.1.122 [UnaCordaPedalLineSpanner], page 353.

Section 2.2.80 [Piano_pedal_engraver], page 232
Engrave piano pedal symbols and brackets.
Music types accepted:
Section 1.2.66 [una-corda-event], page 43, Section 1.2.58 [sustain-event], page 42 and Section 1.2.51 [sostenuto-event], page 41
Properties (read)

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

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.

This engraver creates the following layout object(s):
Section 3.1.79 [PianoPedalBracket], page 318, Section 3.1.91
[SostenutoPedal], page 325, Section 3.1.102 [SustainPedal], page 335
and Section 3.1.121 [UnaCordaPedal], page 352.

Section 2.2.1 [Accidental_engraver], page 206
Make accidentals. Catch note heads, ties and notices key-change events. This engraver usually lives at Staff level, but reads the settings for Accidental at Voice level, so you can \override them at Voice.

Properties (read)

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 Section “Score” in Internals Reference then all staves share accidentals, and if context is Section “Staff” in Internals Reference 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.

**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.

**internalBarNumber** (integer)
Contains the current barnumber. This property is used for internal timekeeping, among others by the Accidental_engraver.

**extraNatural** (boolean)
Whether to typeset an extra natural sign before accidentals changing from a non-natural to another non-natural.

**harmonicAccidentals** (boolean)
If set, harmonic notes in chords get accidentals.

**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)).

**localKeySignature** (list)
The key signature at this point in the measure. The format is the same as for keySignature, but can also contain ((octave . name) . (alter barnumber . measureposition)) pairs.

This engraver creates the following layout object(s):
Section 3.1.1 [Accidental], page 257, Section 3.1.2 [AccidentalCautionary], page 257, Section 3.1.3 [AccidentalPlacement], page 258 and Section 3.1.4 [AccidentalSuggestion], page 259.

**Section 2.2.86 [Rest_collision_engraver], page 234**
Handle collisions of rests.

Properties (read)
busyGrobs (list)
A queue of (end-moment . GROB) cons cells.
This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

This engraver creates the following layout object(s):
Section 3.1.85 [RestCollision], page 322.

Section 2.2.44 [Grob_pq_engraver], page 220
Administrate when certain grobs (e.g., note heads) stop playing.
Properties (read)

busyGrobs (list)
A queue of (end-moment . GROB) cons cells.
This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Properties (write)

busyGrobs (list)
A queue of (end-moment . GROB) cons cells.
This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Section 2.2.18 [Collision_engraver], page 212
Collect NoteColumns, and as soon as there are two or more, put them in a NoteCollision object.
This engraver creates the following layout object(s):
Section 3.1.67 [NoteCollision], page 309.

Section 2.2.104 [Staff_symbol_engraver], page 238
Create the constellation of five (default) staff lines.
Music types accepted:
Section 1.2.55 [staff-span-event], page 41
This engraver creates the following layout object(s):
Section 3.1.96 [StaffSymbol], page 329.

Section 2.2.53 [Ledger_line_engraver], page 223
Create the spanner to draw ledger lines, and notices objects that need ledger lines.
This engraver creates the following layout object(s):
Section 3.1.52 [LedgerLineSpanner], page 298.

Section 2.2.120 [Time_signature_engraver], page 243
Create a Section 3.1.114 [TimeSignature], page 345 whenever timeSignatureFraction changes.
Properties (read)

implicitTimeSignatureVisibility (vector)
break visibility for the default time signature.
timeSignatureFraction (pair of numbers)
A pair of numbers, signifying the time signature. For example, #'(4 . 4) is a 4/4 time signature.

This engraver creates the following layout object(s):
Section 3.1.114 [TimeSignature], page 345.

Section 2.2.50 [Key_engraver], page 222
Engrave a key signature.
Music types accepted:
Section 1.2.22 [key-change-event], page 37
Properties (read)
createKeyOnClefChange (boolean)
Print a key signature whenever the clef is changed.

explicitKeySignatureVisibility (vector)
'b^reak-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.

extraNatural (boolean)
Whether to typeset an extra natural sign before accidentals changing from a non-natural to another non-natural.

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)).

lastKeySignature (list)
Last key signature before a key signature change.

printKeyCancellation (boolean)
Print restoration alterations before a key signature change.

Properties (write)
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)).
(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))

lastKeySignature (list)
Last key signature before a key signature change.

tonic (pitch)
The tonic of the current scale.

This engraver creates the following layout object(s):
Section 3.1.48 [KeyCancellation], page 295 and Section 3.1.49 [KeySignature], page 296.

Section 2.2.16 [Clef_engraver], page 212
Determine and set reference point for pitches.
Properties (read)

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.

explicitClefVisibility (vector)
‘break-visibility’ function for clef changes.

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.

This engraver creates the following layout object(s):
Section 3.1.25 [Clef], page 275 and Section 3.1.72 [OctavateEight], page 311.

Section 2.2.71 [Ottava_spanner_engraver], page 229
Create a text spanner when the ottavation property changes.
Properties (read)

ottavation (markup)
If set, the text for an ottava spanner. Changing this creates a new text spanner.

originalMiddleCPosition (integer)
Used for temporary overriding middle C in octavation brackets.

currentMusicalColumn (layout object)
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).
This engraver creates the following layout object(s):
Section 3.1.73 [OttavaBracket], page 312.

Section 2.2.102 [Staff_collecting_engraver], page 238
Maintain the stavesFound variable.
Properties (read)

stavesFound (list of grobs)
A list of all staff-symbols found.

Properties (write)

stavesFound (list of grobs)
A list of all staff-symbols found.

Section 2.2.23 [Dot_column_engraver], page 214
Engrave dots on dotted notes shifted to the right of the note. If omitted, then dots appear on top of the notes.
This engraver creates the following layout object(s):
Section 3.1.30 [DotColumn], page 279.

Section 2.2.93 [Separating_line_group_engraver], page 236
Generate objects for computing spacing parameters.
Properties (read)

createSpacing (boolean)
Create StaffSpacing objects? Should be set for staves.

Properties (write)

hasStaffSpacing (boolean)
True if the current CommandColumn contains items that will affect spacing.

This engraver creates the following layout object(s):
Section 3.1.95 [StaffSpacing], page 329.

Section 2.2.35 [Font_size_engraver], page 217
Put fontSize into font-size grob property.
Properties (read)

fontSize (number)
The relative size of all grobs in a context.

Section 2.2.7 [Bar_engraver], page 208
Create barlines. This engraver is controlled through the whichBar property. If it has no bar line to create, it will forbid a linebreak at this point.
Properties (read)

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 Section “bar-line-interface” in Internals Reference.
Properties (write)

forbidBreak (boolean)
If set to \#t, prevent a line break at this point.

This engraver creates the following layout object(s):
Section 3.1.11 [BarLine], page 264.

Section 2.2.72 [Output_property_engraver], page 229
Apply a procedure to any grob acknowledged.
Music types accepted:
Section 1.2.4 [apply-output-event], page 35

2.1.21 StaffGroup
Groups staves while adding a bracket on the left side, grouping the staves together. The bar
lines of the contained staves are connected vertically. StaffGroup only consists of a collection
of staves, with a bracket in front and spanning bar lines.

This context creates the following layout object(s):
Section 3.1.9 [Arpeggio], page 263, Section 3.1.94 [SpanBar], page 328, Section 3.1.105 [Sys-
temStartBar], page 337, Section 3.1.106 [SystemStartBracket], page 338, Section 3.1.107 [System-
StartBrace], page 339 and Section 3.1.108 [SystemStartSquare], page 340.

This context sets the following properties:
• Set translator property systemStartDelimiter to 'SystemStartBracket.'

Context StaffGroup can contain Section 2.1.20 [Staff], page 145, Section 2.1.21 [StaffGroup],
page 154, Section 2.1.1 [ChoirStaff], page 48, Section 2.1.7 [FiguredBass], page 78, Section 2.1.2
[ChordNames], page 48, Section 2.1.13 [Lyrics], page 104, Section 2.1.22 [TabStaff], page 155,
Section 2.1.17 [PianoStaff], page 129, Section 2.1.10 [GrandStaff], page 82, Section 2.1.5 [Drum-
Staff], page 62 and Section 2.1.18 [RhythmicStaff], page 130.

This context is built from the following engraver(s):
Section 2.2.111 [System_start_delimiter_engraver], page 240
Create a system start delimiter (i.e., a SystemStartBar,
SystemStartBrace, SystemStartBracket or SystemStartSquare
spanner).

Properties (read)

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.

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable
(clef, key signature, etc.) items.

This engraver creates the following layout object(s):
Section 3.1.105 [SystemStartBar], page 337, Section 3.1.106 [System-
StartBracket], page 338, Section 3.1.107 [SystemStartBracket], page 339
and Section 3.1.108 [SystemStartSquare], page 340.
Section 2.2.72 [Output_property_engraver], page 229
Apply a procedure to any grob acknowledged.
Music types accepted:
Section 1.2.4 [apply-output-event], page 35

Section 2.2.99 [Span_arpeggio_engraver], page 238
Make arpeggios that span multiple staves.
Properties (read)

\[\text{connectArpeggios} \text{ (boolean)}\]
If set, connect arpeggios across piano staff.
This engraver creates the following layout object(s):
Section 3.1.9 [Arpeggio], page 263.

Section 2.2.100 [Span_bar_engraver], page 238
Make cross-staff bar lines: It catches all normal bar lines and draws a
single span bar across them.
This engraver creates the following layout object(s):
Section 3.1.94 [SpanBar], page 328.

2.1.22 TabStaff
Context for generating tablature. [DOCME]
This context also accepts commands for the following context(s):
Staff.
This context creates the following layout object(s):
Section 3.1.11 [BarLine], page 264, Section 3.1.15 [BassFigureAlignmentPositioning],
page 268, Section 3.1.14 [BassFigureAlignment], page 268, Section 3.1.16 [BassFigureBracket],
page 269, Section 3.1.17 [BassFigureContinuation], page 269, Section 3.1.18 [BassFigureLine],
page 270, Section 3.1.13 [BassFigure], page 267, Section 3.1.25 [Clef], page 275, Section 3.1.30
[DotColumn], page 279, Section 3.1.46 [InstrumentName], page 293, Section 3.1.52 [LedgerLineSpanner],
page 298, Section 3.1.67 [NoteCollision], page 309, Section 3.1.72 [OctavateEight],
page 311, Section 3.1.73 [OttavaBracket], page 312, Section 3.1.79 [PianoPedalBracket],
page 318, Section 3.1.85 [RestCollision], page 322, Section 3.1.88 [ScriptRow], page 323,
Section 3.1.92 [SostenutoPedalLineSpanner], page 326, Section 3.1.91 [SostenutoPedal],
page 325, Section 3.1.95 [StaffSpacing], page 329, Section 3.1.96 [StaffSymbol], page 329,
Section 3.1.103 [SustainPedalLineSpanner], page 335, Section 3.1.102 [SustainPedal], page 335,
Section 3.1.114 [TimeSignature], page 345, Section 3.1.122 [UnaCordaPedalLineSpanner],
page 353, Section 3.1.121 [UnaCordaPedal], page 352 and Section 3.1.125 [VerticalAxisGroup],
page 355.
This context sets the following properties:
• Set translator property clefPosition to 0.
• Set translator property clefGlyph to "clefs.tab".
• Set grob-property avoid-note-head in Section 3.1.98 [Stem], page 330 to \#t.
• Set grob-property staff-space in Section 3.1.96 [StaffSymbol], page 329 to 1.5.
• Set translator property shortInstrumentName to '\(').
• Set translator property instrumentName to '\(').
• Set grob-property minimum-Y-extent in Section 3.1.125 [VerticalAxisGroup], page 355 to
  '\((-4 . 4)\).
• Set translator property ignoreFiguredBassRest to \#t.
• Set translator property `createSpacing` to `#t`.
• Set translator property `localKeySignature` to `()'.

Context TabStaff can contain Section 2.1.23 [TabVoice], page 161 and Section 2.1.3 [CueVoice], page 50.

This context is built from the following engraver(s):

**Section 2.2.114 [Tab_staff_symbol_engraver], page 241**
Create a tablature staff symbol, but look at `stringTunings` for the number of lines.

Properties (read)

- **stringTunings** (list)
  The tablature strings tuning. It is a list of the pitch (in semitones) of each string (starting with the lower one).

This engraver creates the following layout object(s):

- **Section 3.1.96 [StaffSymbol], page 329.**

**Section 2.2.92 [Script_row_engraver], page 236**
Determine order in horizontal side position elements.

This engraver creates the following layout object(s):

- **Section 3.1.88 [ScriptRow], page 323.**

**Section 2.2.33 [Figured_bass_position_engraver], page 217**
Position figured bass alignments over notes.

This engraver creates the following layout object(s):

- **Section 3.1.15 [BassFigureAlignmentPositioning], page 268.**

**Section 2.2.32 [Figured_bass_engraver], page 216**
Make figured bass numbers.

Music types accepted:

- **Section 1.2.7 [bass-figure-event], page 36** and **Section 1.2.44 [rest-event], page 40**

Properties (read)

- **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.

- **implicitBassFigures** (list)
  A list of bass figures that are not printed as numbers, but only as extender lines.

- **useBassFigureExtenders** (boolean)
  Whether to use extender lines for repeated bass figures.
**ignoreFiguredBassRest** (boolean)

Don’t swallow rest events.

This engraver creates the following layout object(s):

Section 3.1.13 [BassFigure], page 267, Section 3.1.14 [BassFigure-Alignment], page 268, Section 3.1.16 [BassFigureBracket], page 269, Section 3.1.17 [BassFigureContinuation], page 269 and Section 3.1.18 [BassFigureLine], page 270.

**Section 2.2.5 [Axis_group_engraver], page 208**

Group all objects created in this context in a `VerticalAxisGroup` spanner.

Properties (read)

- **currentCommandColumn** (layout object)
  
  Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

This engraver creates the following layout object(s):

Section 3.1.125 [VerticalAxisGroup], page 355.

**Section 2.2.48 [Instrument_name_engraver], page 221**

Create a system start text for instrument or vocal names.

Properties (read)

- **currentCommandColumn** (layout object)
  
  Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

- **shortInstrumentName** (markup)
  
  See 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.

- **shortVocalName** (markup)
  
  Name of a vocal line, short version.

- **vocalName** (markup)
  
  Name of a vocal line.

This engraver creates the following layout object(s):

Section 3.1.46 [InstrumentName], page 293.

**Section 2.2.79 [Piano_pedal_align_engraver], page 232**

Align piano pedal symbols and brackets.

Properties (read)

- **currentCommandColumn** (layout object)
  
  Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

This engraver creates the following layout object(s):

Section 3.1.92 [SostenutoPedalLineSpanner], page 326, Section 3.1.103 [SustainPedalLineSpanner], page 335 and Section 3.1.122 [UnaCordaPedalLineSpanner], page 353.
Section 2.2.80 [Piano Pedal Engraver], page 232
Engrave piano pedal symbols and brackets.
Music types accepted:
Section 1.2.66 [Una-Corda-Event], page 43, Section 1.2.58 [Sustain-Event],
page 42 and Section 1.2.51 [Sostenuto-Event], page 41
Properties (read)
  currentCommandColumn (layout object)
    Grob that is X-parent to all current breakable
    (clef, key signature, etc.) items.
  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.

This engraver creates the following layout object(s):
Section 3.1.79 [PianoPedalBracket], page 318, Section 3.1.91
[SostenutoPedal], page 325, Section 3.1.102 [SustainPedal], page 335
and Section 3.1.121 [UnaCordaPedal], page 352.

Section 2.2.86 [Rest Collision Engraver], page 234
Handle collisions of rests.
Properties (read)
  busyGrobs (list)
    A queue of (end-moment . GROB) cons cells.
    This is for internal (C++) use only. This property
    contains the grobs which are still busy (e.g.
    note heads, spanners, etc.).

This engraver creates the following layout object(s):
Section 3.1.85 [RestCollision], page 322.

Section 2.2.44 [Grob pq Engraver], page 220
Administrate when certain grobs (e.g., note heads) stop playing.
Properties (read)
  busyGrobs (list)
    A queue of (end-moment . GROB) cons cells.
    This is for internal (C++) use only. This property
    contains the grobs which are still busy (e.g.
    note heads, spanners, etc.).
Properties (write)

**busyGrobs** (list)
A queue of \texttt{(end\textunderscore moment . GROB)} cons cells. This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

**Section 2.2.18 [Collision\_engraver], page 212**
Collect NoteColumns, and as soon as there are two or more, put them in a NoteCollision object.

This engraver creates the following layout object(s):
Section 3.1.67 [NoteCollision], page 309.

**Section 2.2.104 [Staff\_symbol\_engraver], page 238**
Create the constellation of five (default) staff lines.

Music types accepted:
Section 1.2.55 [staff\-span\-event], page 41

This engraver creates the following layout object(s):
Section 3.1.96 [StaffSymbol], page 329.

**Section 2.2.53 [Ledger\_line\_engraver], page 223**
Create the spanner to draw ledger lines, and notices objects that need ledger lines.

This engraver creates the following layout object(s):
Section 3.1.52 [LedgerLineSpanner], page 298.

**Section 2.2.120 [Time\_signature\_engraver], page 243**
Create a \texttt{TimeSignature} whenever timeSignatureFraction changes.

Properties (read)

**implicitTimeSignatureVisibility** (vector)
break visibility for the default time signature.

**timeSignatureFraction** (pair of numbers)
A pair of numbers, signifying the time signature. For example, \texttt{#'(4 . 4)} is a 4/4 time signature.

This engraver creates the following layout object(s):
Section 3.1.114 [TimeSignature], page 345.

**Section 2.2.16 [Clef\_engraver], page 212**
Determine and set reference point for pitches.

Properties (read)

**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.
explicitClefVisibility (vector)
    'break-visibility' function for clef changes.

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.

This engraver creates the following layout object(s):
Section 3.1.25 [Clef], page 275 and Section 3.1.72 [OctavateEight],
page 311.

Section 2.2.71 [Ottava_spanner_engraver], page 229
Create a text spanner when the ottavation property changes.
Properties (read)

ottavation (markup)
    If set, the text for an ottava spanner. Changing
    this creates a new text spanner.

originalMiddleCPosition (integer)
    Used for temporary overriding middle C in oc-
tavation brackets.

currentMusicalColumn (layout object)
    Grob that is X-parent to all non-breakable
    items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):
Section 3.1.73 [OttavaBracket], page 312.

Section 2.2.102 [Staff_collecting_engraver], page 238
Maintain the stavesFound variable.
Properties (read)

stavesFound (list of grobs)
    A list of all staff-symbols found.

Properties (write)

stavesFound (list of grobs)
    A list of all staff-symbols found.

Section 2.2.23 [Dot_column_engraver], page 214
Engrave dots on dotted notes shifted to the right of the note. If omitted,
then dots appear on top of the notes.
This engraver creates the following layout object(s):
Section 3.1.30 [DotColumn], page 279.

Section 2.2.93 [Separating_line_group_engraver], page 236
Generate objects for computing spacing parameters.
Properties (read)

createSpacing (boolean)
    Create StaffSpacing objects? Should be set
    for staves.

Properties (write)
Chapter 2: Translation

hasStaffSpacing (boolean)
True if the current CommandColumn contains items that will affect spacing.

This engraver creates the following layout object(s):
Section 3.1.95 [StaffSpacing], page 329.

Section 2.2.35 [Font_size_ engraver], page 217
Put fontSize into font-size grob property.
Properties (read)

fontSize (number)
The relative size of all grobs in a context.

Section 2.2.7 [Bar_ engraver], page 208
Create barlines. This engraver is controlled through the whichBar property. If it has no bar line to create, it will forbid a linebreak at this point.
Properties (read)

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 Section "bar-line-interface" in Internals Reference.

Properties (write)

forbidBreak (boolean)
If set to ##t, prevent a line break at this point.

This engraver creates the following layout object(s):
Section 3.1.11 [BarLine], page 264.

Section 2.2.72 [Output_property_ engraver], page 229
Apply a procedure to any grob acknowledged.
Music types accepted:
Section 1.2.4 [apply-output-event], page 35

2.1.23 TabVoice
Context for drawing notes in a Tab staff.
This context also accepts commands for the following context(s):
Voice.

This context creates the following layout object(s):
Section 3.1.9 [Arpeggio], page 263, Section 3.1.19 [Beam], page 270, Section 3.1.20 [BendAfter], page 271, Section 3.1.23 [BreathingSign], page 273, Section 3.1.27 [ClusterSpannerBeacon], page 276, Section 3.1.26 [ClusterSpanner], page 276, Section 3.1.28 [CombineTextScript], page 277, Section 3.1.31 [Dots], page 279, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.34 [DynamicLineSpanner], page 282, Section 3.1.36 [DynamicTextSpanner], page 284, Section 3.1.35 [DynamicText], page 283, Section 3.1.39 [Glissando], page 288, Section 3.1.43 [Hairpin], page 290, Section 3.1.44 [HarmonicParenthesesItem], page 291,
Section 3.1.47 [InstrumentSwitch], page 294, Section 3.1.51 [LaissezVibrerTieColumn], page 297, Section 3.1.50 [LaissezVibrerTie], page 297, Section 3.1.54 [LigatureBracket], page 299, Section 3.1.64 [MultiMeasureRestNumber], page 306, Section 3.1.65 [MultiMeasureRestText], page 307, Section 3.1.63 [MultiMeasureRest], page 305, Section 3.1.68 [NoteColumn], page 310, Section 3.1.71 [NoteSpacing], page 311, Section 3.1.77 [PercentRepeatCounter], page 315, Section 3.1.76 [PercentRepeat], page 315, Section 3.1.78 [PhrasingSlur], page 317, Section 3.1.81 [RepeatSlash], page 320, Section 3.1.83 [RepeatTieColumn], page 321, Section 3.1.82 [RepeatTie], page 320, Section 3.1.84 [Rest], page 321, Section 3.1.87 [ScriptColumn], page 323, Section 3.1.86 [Script], page 322, Section 3.1.90 [Slur], page 324, Section 3.1.99 [StemTremolo], page 332, Section 3.1.98 [Stem], page 330, Section 3.1.109 [TabNoteHead], page 340, Section 3.1.110 [TextScript], page 341, Section 3.1.111 [TextSpanner], page 343, Section 3.1.113 [TieColumn], page 345, Section 3.1.112 [Tie], page 344, Section 3.1.115 [TrillPitchAccidental], page 346, Section 3.1.116 [TrillPitchGroup], page 347, Section 3.1.117 [TrillPitchHead], page 348, Section 3.1.118 [TrillSpanner], page 349, Section 3.1.119 [TupletBracket], page 350, Section 3.1.120 [TupletNumber], page 351 and Section 3.1.126 [VoiceFollower], page 356.

This context sets the following properties:

• Set grob-property gap in Section 3.1.39 [Glissando], page 288 to 0.2.
• Set grob-property extra-dy in Section 3.1.39 [Glissando], page 288 to 0.75.
• Set grob-property bound-details left in Section 3.1.39 [Glissando], page 288 to '((attach-dir . 1) (padding . 0.3)).
• Set grob-property bound-details right in Section 3.1.39 [Glissando], page 288 to '((attach-dir . -1) (padding . 0.3)).
• Set grob-property extra-dy in Section 3.1.39 [Glissando], page 288 to 0.75.
• Set grob-property length-fraction in Section 3.1.19 [Beam], page 270 to 0.62.
• Set grob-property thickness in Section 3.1.19 [Beam], page 270 to 0.32.
• Set translator property localKeySignature to ()

This context is a ‘bottom’ context; it cannot contain other contexts.

This context is built from the following engraver(s):

Section 2.2.112 [Tab_harmonic_engraver], page 240
In a tablature, parenthesize objects whose music cause has the parenthesize property.
This engraver creates the following layout object(s):
Section 3.1.44 [HarmonicParenthesesItem], page 291.

Section 2.2.113 [Tab_note_heads_engraver], page 240
Generate one or more tablature noteheads from event of type NoteEvent.
Music types accepted:
Section 1.2.56 [string-number-event], page 41 and Section 1.2.34 [note-event], page 39
Properties (read)

middleCPosition (number)
The place of the middle C, measured in half staff-spaces. Usually determined by looking at middleCClefPosition and middleCOffset.

stringTunings (list)
The tablature strings tuning. It is a list of the pitch (in semitones) of each string (starting with the lower one).
minimumFret (number)
The tablature auto string-selecting mechanism selects the highest string with a fret at least minimumFret.

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.

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.

stringOneTopmost (boolean)
Whether the first string is printed on the top line of the tablature.

This engraver creates the following layout object(s):
Section 3.1.109 [TabNoteHead], page 340.

Section 2.2.94 [Skip_event_swallow_translator], page 236
Swallow \skip.

Section 2.2.49 [Instrument_switch_engraver], page 222
Create a cue text for taking instrument.
Properties (read)

instrumentCueName (markup)
The name to print if another instrument is to be taken.

This engraver creates the following layout object(s):
Section 3.1.47 [InstrumentSwitch], page 294.

Section 2.2.40 [Grace_engraver], page 219
Set font size and other properties for grace notes.
Properties (read)

graceSettings (list)
Overrides for grace notes. This property should be manipulated through the add-grace-property function.

Section 2.2.125 [Tuplet_engraver], page 244
Catch tuplet events and generate appropriate bracket.
Music types accepted:
Section 1.2.65 [tuplet-span-event], page 42
Properties (read)

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.

This engraver creates the following layout object(s):
Section 3.1.119 [TupletBracket], page 350 and Section 3.1.120 [Tuplet-
Number], page 351.

Section 2.2.118 [Tie_engraver], page 242
Generate ties between note heads of equal pitch.
Music types accepted:
Section 1.2.61 [tie-event], page 42
Properties (read)
  tieWaitForNote (boolean)
  If true, tied notes do not have to follow each
  other directly. This can be used for writing out
  arpeggios.

Properties (write)
  tieMelismaBusy (boolean)
  Signal whether a tie is present.

This engraver creates the following layout object(s):
Section 3.1.112 [Tie], page 344 and Section 3.1.113 [TieColumn],
page 345.

Section 2.2.96 [Slur_engraver], page 237
Build slur grobs from slur events.
Music types accepted:
Section 1.2.48 [slur-event], page 40
Properties (read)
  slurMelismaBusy (boolean)
  Signal if a slur is present.
  doubleSlurs (boolean)
  If set, two slurs are created for every slurred
  note, one above and one below the chord.

This engraver creates the following layout object(s):
Section 3.1.90 [Slur], page 324.

Section 2.2.17 [Cluster_spanner_engraver], page 212
Engrave a cluster using Spanner notation.
Music types accepted:
Section 1.2.13 [cluster-note-event], page 36
This engraver creates the following layout object(s):
Section 3.1.26 [ClusterSpanner], page 276 and Section 3.1.27 [Clus-
terSpannerBeacon], page 276.

Section 2.2.78 [Phrasing_slur_engraver], page 232
Print phrasing slurs. Similar to Section 2.2.96 [Slur_engraver], page 237.
Music types accepted:
Chapter 2: Translation

Section 1.2.42 [phrasing-slur-event], page 40
This engraver creates the following layout object(s):

Section 3.1.78 [PhrasingSlur], page 317.

Section 2.2.101 [Spanner_break_forbid_engraver], page 238
Forbid breaks in certain spanners.

Section 2.2.69 [Note_spacing_engraver], page 229
Generate NoteSpacing, an object linking horizontal lines for use in spacing.
This engraver creates the following layout object(s):
Section 3.1.71 [NoteSpacing], page 311.

Section 2.2.89 [Rhythmic_column_engraver], page 235
Generate NoteColumn, an object that groups stems, note heads, and rests.
This engraver creates the following layout object(s):
Section 3.1.68 [NoteColumn], page 310.

Section 2.2.90 [Script_column_engraver], page 235
Find potentially colliding scripts and put them into a ScriptColumn object; that will fix the collisions.
This engraver creates the following layout object(s):
Section 3.1.87 [ScriptColumn], page 323.

Section 2.2.91 [Script_engraver], page 235
Handle note scripted articulations.
Music types accepted:
Section 1.2.6 [articulation-event], page 36
Properties (read)

\[\text{scriptDefinitions} \text{ (list)}\]
The description of scripts. This is used by the \text{Script_engraver} for typesetting note-superscripts and subscripts. See ‘scm/script.scm’ for more information.
This engraver creates the following layout object(s):
Section 3.1.86 [Script], page 322.

Section 2.2.11 [Bend_engraver], page 210
Create fall spanners.
Music types accepted:
Section 1.2.10 [bend-after-event], page 36
This engraver creates the following layout object(s):
Section 3.1.20 [BendAfter], page 271.

Section 2.2.27 [Dynamic_align_engraver], page 215
Align hairpins and dynamic texts on a horizontal line
Properties (read)

\[\text{currentMusicalColumn} \text{ (layout object)}\]
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).
This engraver creates the following layout object(s):
Section 3.1.34 [DynamicLineSpanner], page 282.

**Section 2.2.63 [New_dynamic_engraver], page 227**
Create hairpins, dynamic texts, and their vertical alignments. The symbols are collected onto a `DynamicLineSpanner` grob which takes care of vertical positioning.

Music types accepted:
Section 1.2.53 [span-dynamic-event], page 41 and Section 1.2.2 [absolute-dynamic-event], page 35

Properties (read)

- **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.`.

- **currentMusicalColumn** (layout object)
  Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

- **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.`.

This engraver creates the following layout object(s):
Section 3.1.35 [DynamicText], page 283, Section 3.1.36 [DynamicTextSpanner], page 284 and Section 3.1.43 [Hairpin], page 290.

**Section 2.2.116 [Text_engraver], page 241**
Create text scripts.

Music types accepted:
Section 1.2.59 [text-script-event], page 42

This engraver creates the following layout object(s):
Section 3.1.110 [TextScript], page 341.

**Section 2.2.76 [Part_combine_engraver], page 231**
Part combine engraver for orchestral scores: Print markings ‘a2’, ‘Solo’, ‘Solo II’, and ‘unisono’.

Music types accepted:
Section 1.2.38 [part-combine-event], page 39

Properties (read)

- **printPartCombineTexts** (boolean)
  Set ‘Solo’ and ‘A due’ texts in the part combiner?
soloText (markup)
The text for the start of a solo when part-combining.

soloIIText (markup)
The text for the start of a solo for voice ‘two’ when part-combining.

aDueText (markup)
Text to print at a unisono passage.

This engraver creates the following layout object(s):
Section 3.1.28 [CombineTextScript], page 277.

Section 2.2.95 [Slash_repeat_engraver], page 236
Make beat repeats.
Music types accepted:
Section 1.2.40 [percent-event], page 40
Properties (read)

measureLength (moment)
Length of one measure in the current time signature.

This engraver creates the following layout object(s):
Section 3.1.81 [RepeatSlash], page 320.

Section 2.2.77 [Percent_repeat_engraver], page 231
Make whole bar and double bar repeats.
Music types accepted:
Section 1.2.40 [percent-event], page 40
Properties (read)

countPercentRepeats (boolean)
If set, produce counters for percent repeats.

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

measureLength (moment)
Length of one measure in the current time signature.

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.

Properties (write)

forbidBreak (boolean)
If set to ##t, prevent a line break at this point.

This engraver creates the following layout object(s):
Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.76 [PercentRepeat], page 315 and Section 3.1.77 [PercentRepeatCounter], page 315.
Section 2.2.15 [Chord_tremolo_engraver], page 211
Generate beams for tremolo repeats.
Music types accepted:
Section 1.2.63 [tremolo-span-event], page 42
This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.4 [Auto_beam_engraver], page 207
Generate beams based on measure characteristics and observed Stems. Uses beatLength, measureLength, and measurePosition to decide when to start and stop a beam. Overriding beaming is done through Section 2.2.107 [Stem_engraver], page 239 properties stemLeftBeamCount and stemRightBeamCount.
Music types accepted:
Section 1.2.9 [beam-forbid-event], page 36
Properties (read)

autoBeaming (boolean)
If set to true then beams are generated automatically.

customBeamSettings (list)
Specifies when automatically generated beams should begin and end. See Section “Setting automatic beam behavior” in Notation Reference for more information.

beatLength (moment)
The length of one beat in this time signature.

subdivideBeams (boolean)
If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.39 [Grace_beam_engraver], page 219
Handle Beam events by engraving beams. If omitted, then notes are printed with flags instead of beams. Only engraves beams when we are at grace points in time.
Music types accepted:
Section 1.2.8 [beam-event], page 36
Properties (read)

beamMelismaBusy (boolean)
Signal if a beam is present.

beatLength (moment)
The length of one beat in this time signature.

subdivideBeams (boolean)
If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.
This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.9 [Beam_engraver], page 209
Handle Beam events by engraving beams. If omitted, then notes are printed with flags instead of beams.

Music types accepted:
Section 1.2.8 [beam-event], page 36

Properties (read)

beamMelismaBusy (boolean)
Signal if a beam is present.

beatLength (moment)
The length of one beat in this time signature.

subdivideBeams (boolean)
If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

Properties (write)

forbidBreak (boolean)
If set to ##t, prevent a line break at this point.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.107 [Stem_engraver], page 239
Create stems and single-stem tremolos. It also works together with the beam engraver for overriding beaming.

Music types accepted:
Section 1.2.62 [tremolo-event], page 42

Properties (read)

tremoloFlags (integer)
The number of tremolo flags to add if no number is specified.

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.

This engraver creates the following layout object(s):
Section 3.1.98 [Stem], page 330 and Section 3.1.99 [StemTremolo], page 332.

Section 2.2.126 [Tweak_engraver], page 244
Read the tweaks property from the originating event, and set properties.
Section 2.2.87 [Rest_engraver], page 234
Engrave rests.
Music types accepted:
Section 1.2.44 [rest-event], page 40
Properties (read)

middleCPosition (number)
The place of the middle C, measured in half staff-spaces. Usually determined by looking at middleCClefPosition and middleCOffset.

This engraver creates the following layout object(s):
Section 3.1.84 [Rest], page 321.

Section 2.2.24 [Dots_engraver], page 214
Create Section 3.1.31 [Dots], page 279 objects for Section 3.2.80 [rhythmic-head-interface], page 397s.
This engraver creates the following layout object(s):
Section 3.1.31 [Dots], page 279.

Section 2.2.13 [Breathing_sign_engraver], page 210
Create a breathing sign.
Music types accepted:
Section 1.2.12 [breathing-event], page 36
This engraver creates the following layout object(s):
Section 3.1.23 [BreathingSign], page 273.

Section 2.2.54 [Ligature_bracket_engraver], page 224
Handle Ligature_events by engraving Ligature brackets.
Music types accepted:
Section 1.2.26 [ligature-event], page 38
This engraver creates the following layout object(s):
Section 3.1.54 [LigatureBracket], page 299.

Section 2.2.38 [Glissando_engraver], page 219
Engrave glissandi.
Music types accepted:
Section 1.2.19 [glissando-event], page 37
This engraver creates the following layout object(s):
Section 3.1.39 [Glissando], page 288.

Section 2.2.65 [Note_head_line_engraver], page 228
Engrave a line between two note heads, for example a glissando. If followVoice is set, staff switches also generate a line.
Properties (read)

followVoice (boolean)
If set, note heads are tracked across staff switches by a thin line.

This engraver creates the following layout object(s):
Section 3.1.39 [Glissando], page 288 and Section 3.1.126 [VoiceFollower], page 356.
Section 2.2.85 [Repeat_tie_engraver], page 234
Create repeat ties.
Music types accepted:
Section 1.2.43 [repeat-tie-event], page 40
This engraver creates the following layout object(s):
Section 3.1.82 [RepeatTie], page 320 and Section 3.1.83 [RepeatTieColumn], page 321.

Section 2.2.52 [Laissez_vibrer_engraver], page 223
Create laissez vibrer items.
Music types accepted:
Section 1.2.24 [laissez-vibrer-event], page 37
This engraver creates the following layout object(s):
Section 3.1.50 [LaissezVibrerTie], page 297 and Section 3.1.51 [LaissezVibrerTieColumn], page 297.

Section 2.2.36 [Forbid_line_break_engraver], page 218
Forbid line breaks when note heads are still playing at some point.
Properties (read)

\textit{busyGrobs} (list)
A queue of \texttt{(end-moment . GROB)} cons cells.
This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Properties (write)

\textit{forbidBreak} (boolean)
If set to \#\#t, prevent a line break at this point.

Section 2.2.44 [Grob_pq_engraver], page 220
Administrate when certain grobs (e.g., note heads) stop playing.
Properties (read)

\textit{busyGrobs} (list)
A queue of \texttt{(end-moment . GROB)} cons cells.
This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Properties (write)

\textit{busyGrobs} (list)
A queue of \texttt{(end-moment . GROB)} cons cells.
This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Section 2.2.124 [Trill_spanner_engraver], page 244
Create trill spanner from an event.
Music types accepted:
Section 1.2.64 [trill-span-event], page 42
Properties (read)
currentCommandColumn (layout object)
Grob that is X-parent to all current breakable
(clef, key signature, etc.) items.

currentMusicalColumn (layout object)
Grob that is X-parent to all non-breakable
items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):
Section 3.1.118 [TrillSpanner], page 349.

Section 2.2.117 [Text_spanner_engraver], page 242
Create text spanner from an event.
Music types accepted:
Section 1.2.60 [text-span-event], page 42
This engraver creates the following layout object(s):
Section 3.1.111 [TextSpanner], page 343.

Section 2.2.62 [Multi_measure_rest_engraver], page 226
Engrave multi-measure rests that are produced with ‘R’. It reads
measurePosition and internalBarNumber to determine what number
to print over the Section 3.1.63 [MultiMeasureRest], page 305. Reads
measureLength to determine whether it should use a whole rest or a
breve rest to represent one measure.
Music types accepted:
Section 1.2.32 [multi-measure-text-event], page 38 and Section 1.2.31
[multi-measure-rest-event], page 38
Properties (read)

internalBarNumber (integer)
Contains the current bar number. This property
is used for internal timekeeping, among others
by the Accidental_engraver.

restNumberThreshold (number)
If a multimeasure rest has more measures than
this, a number is printed.

breakableSeparationItem (layout object)
The breakable items in this time step, for this
staff.

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable
(clef, key signature, etc.) items.

measurePosition (moment)
How much of the current measure have we had.
This can be set manually to create incomplete
measures.

measureLength (moment)
Length of one measure in the current time sig-
nature.
This engraver creates the following layout object(s):
Section 3.1.63 [MultiMeasureRest], page 305, Section 3.1.64 [MultiMeasureRestNumber], page 306 and Section 3.1.65 [MultiMeasureRestText], page 307.

**Section 2.2.3 [Arpeggio_engraver], page 207**
Generate an Arpeggio symbol.
Music types accepted:
Section 1.2.5 [arpeggio-event], page 35
This engraver creates the following layout object(s):
Section 3.1.9 [Arpeggio], page 263.

**Section 2.2.72 [Output_property_engraver], page 229**
Apply a procedure to any grob acknowledged.
Music types accepted:
Section 1.2.4 [apply-output-event], page 35

**Section 2.2.83 [Pitched_trill_engraver], page 233**
Print the bracketed note head after a note head with trill.
This engraver creates the following layout object(s):
Section 3.1.115 [TrillPitchAccidental], page 346, Section 3.1.116 [TrillPitchGroup], page 347 and Section 3.1.117 [TrillPitchHead], page 348.

**Section 2.2.35 [Font_size_engraver], page 217**
Put \texttt{fontSize} into \texttt{font-size} grob property.
Properties (read)

\begin{verbatim}
fontSize (number)
\end{verbatim}
The relative size of all grobs in a context.

### 2.1.24 VaticanaStaff

Same as Staff context, except that it is accommodated for typesetting Gregorian Chant in the notational style of Editio Vaticana.

This context also accepts commands for the following context(s):

Staff.

This context creates the following layout object(s):

Section 3.1.2 [AccidentalCautionary], page 257, Section 3.1.3 [AccidentalPlacement], page 258, Section 3.1.4 [AccidentalSuggestion], page 259, Section 3.1.1 [Accidental], page 257, Section 3.1.11 [BarLine], page 264, Section 3.1.15 [BassFigureAlignmentPositioning], page 268, Section 3.1.14 [BassFigureAlignment], page 268, Section 3.1.16 [BassFigureBracket], page 269, Section 3.1.17 [BassFigureContinuation], page 269, Section 3.1.18 [BassFigureLine], page 270, Section 3.1.13 [BassFigure], page 267, Section 3.1.25 [Clef], page 275, Section 3.1.29 [Custos], page 278, Section 3.1.30 [DotColumn], page 279, Section 3.1.46 [InstrumentName], page 293, Section 3.1.48 [KeyCancellation], page 295, Section 3.1.49 [KeySignature], page 296, Section 3.1.52 [LedgerLineSpanner], page 298, Section 3.1.67 [NoteCollision], page 309, Section 3.1.72 [OctavateEight], page 311, Section 3.1.73 [OttavaBracket], page 312, Section 3.1.79 [PianoPedalBracket], page 318, Section 3.1.85 [RestCollision], page 322, Section 3.1.88 [ScriptRow], page 323, Section 3.1.92 [SostenutoPedalLineSpanner], page 326, Section 3.1.91 [SostenutoPedal], page 325, Section 3.1.95 [StaffSpacing], page 329, Section 3.1.96 [StaffSymbol], page 329, Section 3.1.103 [SustainPedalLineSpanner], page 335, Section 3.1.102 [SustainPedal], page 335, Section 3.1.122 [UnaCordaPedalLineSpanner],
This context sets the following properties:

- Set grob-property `style` in Section 3.1.31 [Dots], page 279 to "vaticana."
- Set grob-property `neutral-direction` in Section 3.1.29 [Custos], page 278 to -1.
- Set grob-property `neutral-position` in Section 3.1.29 [Custos], page 278 to 3.
- Set grob-property `style` in Section 3.1.29 [Custos], page 278 to "vaticana."
- Set grob-property `glyph-name-alist` in Section 3.1.1 [Accidental], page 257 to 
  '(((-1/2 . accidentals.vaticanaM1) (0 . accidentals.vaticana0) (1/2 . accidentals.mensural1))).
- Set grob-property `glyph-name-alist` in Section 3.1.49 [KeySignature], page 296 to 
  '(((-1/2 . accidentals.vaticanaM1) (0 . accidentals.vaticana0) (1/2 . accidentals.mensural1))).
- Set translator property `clefOctavation` to 0.
- Set translator property `clefPosition` to 1.
- Set translator property `middleCClefPosition` to 1.
- Set translator property `middleCPosition` to 1.
- Set translator property `clefGlyph` to "clefs.vaticana.do".
- Set grob-property `thickness` in Section 3.1.96 [StaffSymbol], page 329 to 0.6.
- Set grob-property `line-count` in Section 3.1.96 [StaffSymbol], page 329 to 4.
- Set translator property `transparent` in Section 3.1.11 [BarLine], page 264 to #t.
- Set translator property `shortInstrumentName` to '().
- Set translator property `instrumentName` to '().
- Set grob-property `minimum-Y-extent` in Section 3.1.125 [VerticalAxisGroup], page 355 to 
  '(-4 . 4).
- Set translator property `ignoreFiguredBassRest` to #t.
- Set translator property `createSpacing` to #t.
- Set translator property `localKeySignature` to '().

Context `VaticanaStaff` can contain Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.3 [CueVoice], page 50.

This context is built from the following engraver(s):

**Section 2.2.21 [Custos_engraver], page 213**
Engrave custodes.

This engraver creates the following layout object(s):
Section 3.1.29 [Custos], page 278.

**Section 2.2.92 [Script_row_engraver], page 236**
Determine order in horizontal side position elements.
This engraver creates the following layout object(s):
Section 3.1.88 [ScriptRow], page 323.

**Section 2.2.33 [Figured_bass_position_engraver], page 217**
Position figured bass alignments over notes.
This engraver creates the following layout object(s):
Section 3.1.15 [BassFigureAlignmentPositioning], page 268.
Section 2.2.32 [Figured_bass_engraver], page 216
Make figured bass numbers.
Music types accepted:
Section 1.2.7 [bass-figure-event], page 36 and Section 1.2.44 [rest-event], page 40
Properties (read)

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.

implicitBassFigures (list)
A list of bass figures that are not printed as numbers, but only as extender lines.

useBassFigureExtenders (boolean)
Whether to use extender lines for repeated bass figures.

ignoreFiguredBassRest (boolean)
Don’t swallow rest events.

This engraver creates the following layout object(s):
Section 3.1.13 [BassFigure], page 267, Section 3.1.14 [BassFigure-Alignment], page 268, Section 3.1.16 [BassFigureBracket], page 269, Section 3.1.17 [BassFigureContinuation], page 269 and Section 3.1.18 [BassFigureLine], page 270.

Section 2.2.5 [Axis_group_engraver], page 208
Group all objects created in this context in a VerticalAxisGroup spanner.
Properties (read)

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

This engraver creates the following layout object(s):
Section 3.1.125 [VerticalAxisGroup], page 355.

Section 2.2.108 [String_number_engraver], page 240
Swallow string number events. The purpose of this engraver is to process tablatures for normal notation. To prevent warnings for unprocessed string number events to obscure real error messages, this engraver swallows them all.

Section 2.2.48 [Instrument_name_engraver], page 221
Create a system start text for instrument or vocal names.
Properties (read)
currentCommandColumn (layout object)
   Grob that is X-parent to all current breakable
   (clef, key signature, etc.) items.

shortInstrumentName (markup)
   See 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.

shortVocalName (markup)
   Name of a vocal line, short version.

vocalName (markup)
   Name of a vocal line.

This engraver creates the following layout object(s):
Section 3.1.46 [InstrumentName], page 293.

Section 2.2.79 [Piano_pedal_align_engraver], page 232
Align piano pedal symbols and brackets.
Properties (read)

   currentCommandColumn (layout object)
      Grob that is X-parent to all current breakable
      (clef, key signature, etc.) items.

This engraver creates the following layout object(s):
Section 3.1.92 [SostenutoPedalLineSpanner], page 326, Section 3.1.103
[SustainPedalLineSpanner], page 335 and Section 3.1.122 [UnaCor-
daPedalLineSpanner], page 353.

Section 2.2.80 [Piano_pedal_engraver], page 232
Engrave piano pedal symbols and brackets.
Music types accepted:
Section 1.2.66 [una-corda-event], page 43, Section 1.2.58 [sustain-event],
page 42 and Section 1.2.51 [sostenuto-event], page 41
Properties (read)

   currentCommandColumn (layout object)
      Grob that is X-parent to all current breakable
      (clef, key signature, etc.) items.

   pedalSostenutoStrings (list)
      See pedalSustainStrings.

   pedalSostenutoStyle (symbol)
      See pedalSustainStyle.

   pedalSustainStrings (list)
      A list of strings to print for sustain-pedal. For-
      mat 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.

This engraver creates the following layout object(s):
Section 3.1.79 [PianoPedalBracket], page 318, Section 3.1.91 [SostenutoPedal], page 325, Section 3.1.102 [SustainPedal], page 335 and Section 3.1.121 [UnaCordaPedal], page 352.

Section 2.2.1 [Accidental_engraver], page 206
Make accidentals. Catch note heads, ties and notices key-change events. This engraver usually lives at Staff level, but reads the settings for Accidental at Voice level, so you can \override them at Voice.

Properties (read)
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 Section “Score” in Internals Reference then all staves share accidentals, and if context is Section “Staff” in Internals Reference 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.

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.

internalBarNumber (integer)
Contains the current barnumber. This property is used for internal timekeeping, among others by the Accidental_engraver.

extraNatural (boolean)
Whether to typeset an extra natural sign before accidentals changing from a non-natural to another non-natural.

harmonicAccidentals (boolean)
If set, harmonic notes in chords get accidentals.

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)).

localKeySignature (list)
The key signature at this point in the measure. The format is the same as for keySignature, but can also contain ((octave . name) . (alter barnumber . measureposition)) pairs.

This engraver creates the following layout object(s):
Section 3.1.1 [Accidental], page 257, Section 3.1.2 [AccidentalCautionary], page 257, Section 3.1.3 [AccidentalPlacement], page 258 and Section 3.1.4 [AccidentalSuggestion], page 259.

Section 2.2.86 [Rest_collision_engraver], page 234
Handle collisions of rests.
Properties (read)
busyGrobs (list)
   A queue of (end-moment . GROB) cons cells.
   This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

This engraver creates the following layout object(s):
Section 3.1.85 [RestCollision], page 322.

Section 2.2.44 [Grob_pq_engraver], page 220
   Administrate when certain grobs (e.g., note heads) stop playing.
Properties (read)

busyGrobs (list)
   A queue of (end-moment . GROB) cons cells.
   This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Properties (write)

busyGrobs (list)
   A queue of (end-moment . GROB) cons cells.
   This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Section 2.2.18 [Collision_engraver], page 212
   Collect NoteColumns, and as soon as there are two or more, put them in a NoteCollision object.
   This engraver creates the following layout object(s):
Section 3.1.67 [NoteCollision], page 309.

Section 2.2.104 [Staff_symbol_engraver], page 238
   Create the constellation of five (default) staff lines.
   Music types accepted:
Section 1.2.55 [staff-span-event], page 41
   This engraver creates the following layout object(s):
Section 3.1.96 [StaffSymbol], page 329.

Section 2.2.53 [Ledger_line_engraver], page 223
   Create the spanner to draw ledger lines, and notices objects that need ledger lines.
   This engraver creates the following layout object(s):
Section 3.1.52 [LedgerLineSpanner], page 298.

Section 2.2.50 [Key_engraver], page 222
   Engrave a key signature.
   Music types accepted:
Section 1.2.22 [key-change-event], page 37
Properties (read)

createKeyOnClefChange (boolean)
   Print a key signature whenever the clef is changed.
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.

extraNatural (boolean)

Whether to typeset an extra natural sign before accidentals changing from a non-natural to another non-natural.

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)).

lastKeySignature (list)

Last key signature before a key signature change.

printKeyCancellation (boolean)

Print restoration alterations before a key signature change.

Properties (write)

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)).

lastKeySignature (list)

Last key signature before a key signature change.

tonic (pitch)

The tonic of the current scale.

This engraver creates the following layout object(s):

Section 3.1.48 [KeyCancellation], page 295 and Section 3.1.49 [KeySignature], page 296.

Section 2.2.16 [Clef_engraver], page 212

Determine and set reference point for pitches.

Properties (read)
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.

explicitClefVisibility (vector)
  'break-visibility' function for clef changes.

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.

This engraver creates the following layout object(s):
Section 3.1.25 [Clef], page 275 and Section 3.1.72 [OctavateEight], page 311.

Section 2.2.71 [Ottava_spanner_engraver], page 229
Create a text spanner when the ottavation property changes.
Properties (read)

  ottavation (markup)
    If set, the text for an ottava spanner. Changing this creates a new text spanner.

originalMiddleCPosition (integer)
  Used for temporary overriding middle C in octavation brackets.

currentMusicalColumn (layout object)
  Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):
Section 3.1.73 [OttavaBracket], page 312.

Section 2.2.102 [Staff_collecting_engraver], page 238
Maintain the stavesFound variable.
Properties (read)

  stavesFound (list of grobs)
    A list of all staff-symbols found.

Properties (write)

  stavesFound (list of grobs)
    A list of all staff-symbols found.

Section 2.2.23 [Dot_column_engraver], page 214
Engrave dots on dotted notes shifted to the right of the note. If omitted, then dots appear on top of the notes.
This engraver creates the following layout object(s):
Section 3.1.30 [DotColumn], page 279.
Section 2.2.93 [Separating_line_group_engraver], page 236
Generate objects for computing spacing parameters.

Properties (read)

createSpacing (boolean)
Create StaffSpacing objects? Should be set for staves.

Properties (write)

hasStaffSpacing (boolean)
True if the current CommandColumn contains items that will affect spacing.

This engraver creates the following layout object(s):
Section 3.1.95 [StaffSpacing], page 329.

Section 2.2.35 [Font_size_engraver], page 217
Put fontSize into font-size grob property.

Properties (read)

fontSize (number)
The relative size of all grobs in a context.

Section 2.2.7 [Bar_engraver], page 208
Create barlines. This engraver is controlled through the whichBar property. If it has no bar line to create, it will forbid a linebreak at this point.

Properties (read)

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 Section “barline-interface” in Internals Reference.

Properties (write)

forbidBreak (boolean)
If set to ##t, prevent a line break at this point.

This engraver creates the following layout object(s):
Section 3.1.11 [BarLine], page 264.

Section 2.2.72 [Output_property_engraver], page 229
Apply a procedure to any grob acknowledged.
Music types accepted:
Section 1.2.4 [apply-output-event], page 35

2.1.25 VaticanaVoice

Same as Voice context, except that it is accommodated for typesetting Gregorian Chant in the notational style of Editio Vaticana.

This context also accepts commands for the following context(s):
Voice.
This context creates the following layout object(s):

Section 3.1.9 [Arpeggio], page 263, Section 3.1.19 [Beam], page 270, Section 3.1.20 [BendAfter], page 271, Section 3.1.23 [BreathingSign], page 273, Section 3.1.27 [ClusterSpannerBeacon], page 276, Section 3.1.26 [ClusterSpanner], page 276, Section 3.1.28 [CombineTextScript], page 277, Section 3.1.30 [DotColumn], page 279, Section 3.1.31 [Dots], page 279, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.34 [DynamicLineSpanner], page 282, Section 3.1.36 [DynamicTextSpanner], page 284, Section 3.1.35 [DynamicText], page 283, Section 3.1.37 [Fingering], page 285, Section 3.1.39 [Glissando], page 288, Section 3.1.43 [Hairpin], page 290, Section 3.1.47 [InstrumentSwitch], page 294, Section 3.1.51 [LaissezVibrerTieColumn], page 297, Section 3.1.50 [LaissezVibrerTie], page 297, Section 3.1.64 [MultiMeasureRestNumber], page 306, Section 3.1.65 [MultiMeasureRestText], page 307, Section 3.1.63 [MultiMeasureRest], page 305, Section 3.1.68 [NoteColumn], page 310, Section 3.1.69 [NoteHead], page 310, Section 3.1.71 [NoteSpacing], page 311, Section 3.1.77 [PercentRepeatCounter], page 315, Section 3.1.76 [PercentRepeat], page 315, Section 3.1.78 [PhrasingSlur], page 317, Section 3.1.81 [RepeatSlash], page 320, Section 3.1.83 [RepeatTieColumn], page 321, Section 3.1.82 [RepeatTie], page 320, Section 3.1.84 [Rest], page 321, Section 3.1.87 [ScriptColumn], page 323, Section 3.1.86 [Script], page 322, Section 3.1.100 [StringNumber], page 332, Section 3.1.101 [StrokeFinger], page 334, Section 3.1.110 [TextScript], page 341, Section 3.1.111 [TextSpanner], page 343, Section 3.1.113 [ TieColumn], page 345, Section 3.1.112 [ Tie], page 344, Section 3.1.115 [TrillPitchAccidental], page 346, Section 3.1.116 [TrillPitchGroup], page 347, Section 3.1.117 [TrillPitchHead], page 348, Section 3.1.118 [TrillSpanner], page 349, Section 3.1.119 [TupletBracket], page 350, Section 3.1.120 [TupletNumber], page 351, Section 3.1.123 [VaticanaLigature], page 354 and Section 3.1.126 [VoiceFollower], page 356.

This context sets the following properties:

• Set grob-property padding in Section 3.1.111 [TextSpanner], page 343 to -0.1.
• Set grob-property style in Section 3.1.111 [TextSpanner], page 343 to ’line.
• Set translator property autoBeaming to #f.
• Set grob-property padding in Section 3.1.86 [Script], page 322 to 0.5.
• Set grob-property style in Section 3.1.69 [NoteHead], page 310 to ’vaticana.punctum.
• Set translator property localKeySignature to ’().

This context is a ‘bottom’ context; it cannot contain other contexts.

This context is built from the following engraver(s):

Section 2.2.127 [Vaticana_ligature_engraver], page 245
Handle ligatures by glueing special ligature heads together.

Music types accepted:
Section 1.2.26 [ligature-event], page 38 and Section 1.2.41 [pes-or-flexaevent], page 40
This engraver creates the following layout object(s):
Section 3.1.30 [DotColumn], page 279 and Section 3.1.123 [VaticanaLigature], page 354.

Section 2.2.94 [Skip_event_swallow_translator], page 236
Swallow \skip.

Section 2.2.49 [Instrument_switch_engraver], page 222
Create a cue text for taking instrument.
Properties (read)
**instrumentCueName** (markup)

The name to print if another instrument is to be taken.

This engraver creates the following layout object(s):
Section 3.1.47 [InstrumentSwitch], page 294.

**Section 2.2.40 [Grace_engraver], page 219**
Set font size and other properties for grace notes.
Properties (read)

**graceSettings** (list)
Overrides for grace notes. This property should be manipulated through the `add-grace-property` function.

**Section 2.2.125 [Tuplet_engraver], page 244**
Catch tuplet events and generate appropriate bracket.
Music types accepted:
Section 1.2.65 [tuplet-span-event], page 42
Properties (read)

**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.

This engraver creates the following layout object(s):
Section 3.1.119 [TupletBracket], page 350 and Section 3.1.120 [Tuplet-Number], page 351.

**Section 2.2.118 [Tie_engraver], page 242**
Generate ties between note heads of equal pitch.
Music types accepted:
Section 1.2.61 [tie-event], page 42
Properties (read)

**tieWaitForNote** (boolean)
If true, tied notes do not have to follow each other directly. This can be used for writing out arpeggios.

Properties (write)

**tieMelismaBusy** (boolean)
Signal whether a tie is present.

This engraver creates the following layout object(s):
Section 3.1.112 [Tie], page 344 and Section 3.1.113 [TieColumn], page 345.

**Section 2.2.17 [Cluster_spanner_engraver], page 212**
Engrave a cluster using Spanner notation.
Music types accepted:
Section 1.2.13 [cluster-note-event], page 36
This engraver creates the following layout object(s):
Section 3.1.26 [ClusterSpanner], page 276 and Section 3.1.27 [ClusterSpannerBeacon], page 276.

Section 2.2.78 [Phrasing_slur_engraver], page 232
Print phrasing slurs. Similar to Section 2.2.96 [Slur_engraver], page 237.
Music types accepted:
Section 1.2.42 [phrasing-slur-event], page 40
This engraver creates the following layout object(s):
Section 3.1.78 [PhrasingSlur], page 317.

Section 2.2.101 [Spanner_break_forbid_engraver], page 238
Forbid breaks in certain spanners.

Section 2.2.69 [Note_spacing_engraver], page 229
Generate NoteSpacing, an object linking horizontal lines for use in spacing.
This engraver creates the following layout object(s):
Section 3.1.71 [NoteSpacing], page 311.

Section 2.2.89 [Rhythmic_column_engraver], page 235
Generate NoteColumn, an object that groups stems, note heads, and rests.
This engraver creates the following layout object(s):
Section 3.1.68 [NoteColumn], page 310.

Section 2.2.90 [Script_column_engraver], page 235
Find potentially colliding scripts and put them into a ScriptColumn object; that will fix the collisions.
This engraver creates the following layout object(s):
Section 3.1.87 [ScriptColumn], page 323.

Section 2.2.91 [Script_engraver], page 235
Handle note scripted articulations.
Music types accepted:
Section 1.2.6 [articulation-event], page 36
Properties (read)

    scriptDefinitions (list)
    The description of scripts. This is used by the Script_engraver for typesetting note-superscripts and subscripts. See ‘scm/script.scm’ for more information.

This engraver creates the following layout object(s):
Section 3.1.86 [Script], page 322.

Section 2.2.11 [Bend_engraver], page 210
Create fall spanners.
Music types accepted:
Section 1.2.10 [bend-after-event], page 36
This engraver creates the following layout object(s):
Section 3.1.20 [BendAfter], page 271.

Section 2.2.34 [Fingering_ engraver], page 217
Create fingering scripts.
Music types accepted:
Section 1.2.57 [stroke-finger-event], page 42 and Section 1.2.18
[fingering-event], page 37
This engraver creates the following layout object(s):
Section 3.1.37 [Fingering], page 285.

Section 2.2.27 [Dynamic_align_ engraver], page 215
Align hairpins and dynamic texts on a horizontal line
Properties (read)

  currentMusicalColumn (layout object)
  Grob that is X-parent to all non-breakable
  items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):
Section 3.1.34 [DynamicLineSpanner], page 282.

Section 2.2.63 [New_dynamic_ engraver], page 227
Create hairpins, dynamic texts, and their vertical alignments. The sym-
bols are collected onto a DynamicLineSpanner grob which takes care of
vertical positioning.
Music types accepted:
Section 1.2.53 [span-dynamic-event], page 41 and Section 1.2.2
[absolute-dynamic-event], page 35
Properties (read)

  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.’.

  currentMusicalColumn (layout object)
  Grob that is X-parent to all non-breakable
  items (note heads, lyrics, etc.).

  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 de-
crescendo, i.e., ‘dim.’.

This engraver creates the following layout object(s):
Section 3.1.35 [DynamicText], page 283, Section 3.1.36 [Dy-
namicTextSpanner], page 284 and Section 3.1.43 [Hairpin],
page 290.
Section 2.2.116 [Text_engraver], page 241
Create text scripts.
Music types accepted:
Section 1.2.59 [text-script-event], page 42
This engraver creates the following layout object(s):
Section 3.1.110 [TextScript], page 341.

Section 2.2.76 [Part_combine_engraver], page 231
Part combine engraver for orchestral scores: Print markings ‘a2’, ‘Solo’, ‘Solo II’, and ‘unisono’.
Music types accepted:
Section 1.2.38 [part-combine-event], page 39
Properties (read)

printPartCombineTexts (boolean)
Set ‘Solo’ and ‘A due’ texts in the part combiner?

soloText (markup)
The text for the start of a solo when part-combining.

soloIIText (markup)
The text for the start of a solo for voice ‘two’ when part-combining.

aDueText (markup)
Text to print at a unisono passage.

This engraver creates the following layout object(s):
Section 3.1.28 [CombineTextScript], page 277.

Section 2.2.95 [Slash_repeat_engraver], page 236
Make beat repeats.
Music types accepted:
Section 1.2.40 [percent-event], page 40
Properties (read)

measureLength (moment)
Length of one measure in the current time signature.

This engraver creates the following layout object(s):
Section 3.1.81 [RepeatSlash], page 320.

Section 2.2.77 [Percent_repeat_engraver], page 231
Make whole bar and double bar repeats.
Music types accepted:
Section 1.2.40 [percent-event], page 40
Properties (read)

countPercentRepeats (boolean)
If set, produce counters for percent repeats.
currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

measureLength (moment)
Length of one measure in the current time signature.

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.

Properties (write)

forbidBreak (boolean)
If set to #t, prevent a line break at this point.

This engraver creates the following layout object(s):
Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.76 [PercentRepeat], page 315 and Section 3.1.77 [PercentRepeatCounter], page 315.

Section 2.2.15 [Chord_tremolo_engraver], page 211
Generate beams for tremolo repeats.
Music types accepted:
Section 1.2.63 [tremolo-span-event], page 42
This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.64 [New_fingering_engraver], page 227
Create fingering scripts for notes in a new chord. This engraver is ill-named, since it also takes care of articulations and harmonic note heads.
Properties (read)

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.

harmonicDots (boolean)
If set, harmonic notes in dotted chords get dots.

strokeFingerOrientations (list)
See fingeringOrientations.

stringNumberOrientations (list)
See fingeringOrientations.

This engraver creates the following layout object(s):
Section 3.1.37 [Fingering], page 285, Section 3.1.86 [Script], page 322, Section 3.1.100 [StringNumber], page 332 and Section 3.1.101 [StrokeFinger], page 334.
Section 2.2.4 [Auto_beam_engraver], page 207
Generate beams based on measure characteristics and observed Stems. Uses beatLength, measureLength, and measurePosition to decide when to start and stop a beam. Overriding beaming is done through Section 2.2.107 [Stem_engraver], page 239 properties stemLeftBeamCount and stemRightBeamCount.

Music types accepted:
Section 1.2.9 [beam-forbid-event], page 36
Properties (read)

autoBeaming (boolean)
If set to true then beams are generated automatically.

autoBeamSettings (list)
Specifies when automatically generated beams should begin and end. See Section “Setting automatic beam behavior” in Notation Reference for more information.

beatLength (moment)
The length of one beat in this time signature.

subdivideBeams (boolean)
If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.39 [Grace_beam_engraver], page 219
Handle Beam events by engraving beams. If omitted, then notes are printed with flags instead of beams. Only engraves beams when we are at grace points in time.

Music types accepted:
Section 1.2.8 [beam-event], page 36
Properties (read)

beamMelismaBusy (boolean)
Signal if a beam is present.

beatLength (moment)
The length of one beat in this time signature.

subdivideBeams (boolean)
If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.9 [Beam_engraver], page 209
Handle Beam events by engraving beams. If omitted, then notes are printed with flags instead of beams.
Music types accepted:
Section 1.2.8 [beam-event], page 36

Properties (read)

beamMelismaBusy (boolean)
Signal if a beam is present.

beatLength (moment)
The length of one beat in this time signature.

subdivideBeams (boolean)
If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

Properties (write)

forbidBreak (boolean)
If set to $\#\#$, prevent a line break at this point.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.126 [Tweak_engraver], page 244
Read the tweaks property from the originating event, and set properties.

Section 2.2.87 [Rest_engraver], page 234
Engrave rests.

Music types accepted:
Section 1.2.44 [rest-event], page 40

Properties (read)

middleCPosition (number)
The place of the middle C, measured in half staff-spaces. Usually determined by looking at middleCClefPosition and middleCOffset.

This engraver creates the following layout object(s):
Section 3.1.84 [Rest], page 321.

Section 2.2.24 [Dots_engraver], page 214
Create Section 3.1.31 [Dots], page 279 objects for Section 3.2.80 [rhythmic-head-interface], page 397s.

This engraver creates the following layout object(s):
Section 3.1.31 [Dots], page 279.

Section 2.2.66 [Note_heads_engraver], page 228
Generate note heads.

Music types accepted:
Section 1.2.34 [note-event], page 39

Properties (read)

middleCPosition (number)
The place of the middle C, measured in half staff-spaces. Usually determined by looking at middleCClefPosition and middleCOffset.
staffLineLayoutFunction (procedure)
  Layout of staff lines, traditional, or semitone.

This engraver creates the following layout object(s):
Section 3.1.69 [NoteHead], page 310.

Section 2.2.13 [Breathing_sign_engraver], page 210
Create a breathing sign.
Music types accepted:
Section 1.2.12 [breathing-event], page 36
This engraver creates the following layout object(s):
Section 3.1.23 [BreathingSign], page 273.

Section 2.2.38 [Glissando_engraver], page 219
Engrave glissandi.
Music types accepted:
Section 1.2.19 [glissando-event], page 37
This engraver creates the following layout object(s):
Section 3.1.39 [Glissando], page 288.

Section 2.2.65 [Note_head_line_engraver], page 228
Engrave a line between two note heads, for example a glissando. If followVoice is set, staff switches also generate a line.
Properties (read)

  followVoice (boolean)
  If set, note heads are tracked across staff switches by a thin line.

This engraver creates the following layout object(s):
Section 3.1.39 [Glissando], page 288 and Section 3.1.126 [VoiceFollower], page 356.

Section 2.2.85 [Repeat_tie_engraver], page 234
Create repeat ties.
Music types accepted:
Section 1.2.43 [repeat-tie-event], page 40
This engraver creates the following layout object(s):
Section 3.1.82 [RepeatTie], page 320 and Section 3.1.83 [RepeatTieColumn], page 321.

Section 2.2.52 [Laissez_vibrer_engraver], page 223
Create laissez vibrer items.
Music types accepted:
Section 1.2.24 [laissez-vibrer-event], page 37
This engraver creates the following layout object(s):
Section 3.1.50 [LaissezVibrerTie], page 297 and Section 3.1.51 [LaissezVibrerTieColumn], page 297.

Section 2.2.36 [Forbid_line_break_engraver], page 218
Forbid line breaks when note heads are still playing at some point.
Properties (read)
busyGrobs (list)
   A queue of (end-moment . GROB) cons cells. This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Properties (write)

forbidBreak (boolean)
   If set to ##t, prevent a line break at this point.

Section 2.2.44 [Grob_pq_engraver], page 220
   Administrate when certain grobs (e.g., note heads) stop playing.

Properties (read)

busyGrobs (list)
   A queue of (end-moment . GROB) cons cells. This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Properties (write)

busyGrobs (list)
   A queue of (end-moment . GROB) cons cells. This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Section 2.2.124 [Trill_spanner_engraver], page 244
   Create trill spanner from an event.

Music types accepted:
Section 1.2.64 [trill-span-event], page 42

Properties (read)

currentCommandColumn (layout object)
   Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

currentMusicalColumn (layout object)
   Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):
Section 3.1.118 [TrillSpanner], page 349.

Section 2.2.117 [Text_spanner_engraver], page 242
   Create text spanner from an event.

Music types accepted:
Section 1.2.60 [text-span-event], page 42

This engraver creates the following layout object(s):
Section 3.1.111 [TextSpanner], page 343.

Section 2.2.62 [Multi_measure_rest_engraver], page 226
   Engrave multi-measure rests that are produced with ‘R’. It reads measurePosition and internalBarNumber to determine what number to print over the Section 3.1.63 [MultiMeasureRest], page 305. Reads
measureLength to determine whether it should use a whole rest or a breve rest to represent one measure.

Music types accepted:
Section 1.2.32 [multi-measure-text-event], page 38 and Section 1.2.31 [multi-measure-rest-event], page 38

Properties (read)

internalBarNumber (integer)
Contains the current bar number. This property is used for internal timekeeping, among others by the Accidental_engraver.

restNumberThreshold (number)
If a multimeasure rest has more measures than this, a number is printed.

breakableSeparationItem (layout object)
The breakable items in this time step, for this staff.

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

measurePosition (moment)
How much of the current measure have we had. This can be set manually to create incomplete measures.

measureLength (moment)
Length of one measure in the current time signature.

This engraver creates the following layout object(s):
Section 3.1.63 [MultiMeasureRest], page 305, Section 3.1.64 [MultiMeasureRestNumber], page 306 and Section 3.1.65 [MultiMeasureRestText], page 307.

Section 2.2.3 [Arpeggio_engraver], page 207
Generate an Arpeggio symbol.

Music types accepted:
Section 1.2.5 [arpeggio-event], page 35
This engraver creates the following layout object(s):
Section 3.1.9 [Arpeggio], page 263.

Section 2.2.72 [Output_property_engraver], page 229
Apply a procedure to any grob acknowledged.

Music types accepted:
Section 1.2.4 [apply-output-event], page 35

Section 2.2.83 [Pitched_trill_engraver], page 233
Print the bracketed note head after a note head with trill.

This engraver creates the following layout object(s):
Section 3.1.115 [TrillPitchAccidental], page 346, Section 3.1.116 [TrillPitchGroup], page 347 and Section 3.1.117 [TrillPitchHead], page 348.
Section 2.2.35 [Font_size_engraver], page 217
Put fontSize into font-size grob property.

Properties (read)

fontSize (number)
The relative size of all grobs in a context.

2.1.26 Voice

Corresponds to a voice on a staff. This context handles the conversion of dynamic signs, stems, beams, super- and subscripts, slurs, ties, and rests.

You have to instantiate this explicitly if you want to have multiple voices on the same staff.

This context creates the following layout object(s):

Section 3.1.9 [Arpeggio], page 263, Section 3.1.19 [Beam], page 270, Section 3.1.20 [BendAfter], page 271, Section 3.1.23 [BreathingSign], page 273, Section 3.1.27 [ClusterSpannerBeacon], page 276, Section 3.1.26 [ClusterSpanner], page 276, Section 3.1.28 [CombineTextScript], page 277, Section 3.1.31 [Dots], page 279, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.34 [DynamicLineSpanner], page 282, Section 3.1.36 [DynamicTextSpanner], page 284, Section 3.1.35 [DynamicText], page 283, Section 3.1.37 [Fingering], page 285, Section 3.1.39 [Glissando], page 288, Section 3.1.43 [Hairpin], page 290, Section 3.1.47 [InstrumentSwitch], page 294, Section 3.1.51 [LaissezVibrerTieColumn], page 297, Section 3.1.50 [LaissezVibrerTie], page 297, Section 3.1.54 [LigatureBracket], page 299, Section 3.1.64 [MultiMeasureRestNumber], page 306, Section 3.1.65 [MultiMeasureRestText], page 307, Section 3.1.63 [MultiMeasureRest], page 305, Section 3.1.68 [NoteColumn], page 310, Section 3.1.69 [NoteHead], page 310, Section 3.1.71 [NoteSpacing], page 311, Section 3.1.77 [PercentRepeatCounter], page 315, Section 3.1.76 [PercentRepeat], page 315, Section 3.1.78 [PhrasingSlur], page 317, Section 3.1.81 [RepeatSlash], page 320, Section 3.1.83 [RepeatTieColumn], page 321, Section 3.1.82 [RepeatTie], page 320, Section 3.1.84 [Rest], page 321, Section 3.1.87 [ScriptColumn], page 323, Section 3.1.86 [Script], page 322, Section 3.1.90 [Slur], page 324, Section 3.1.99 [StemTremolo], page 332, Section 3.1.98 [Stem], page 330, Section 3.1.100 [StringNumber], page 332, Section 3.1.101 [StrokeFinger], page 334, Section 3.1.110 [TextScript], page 341, Section 3.1.111 [TextSpanner], page 343, Section 3.1.113 [TieColumn], page 345, Section 3.1.112 [Tie], page 344, Section 3.1.115 [TrillPitchAccidental], page 346, Section 3.1.116 [TrillPitchGroup], page 347, Section 3.1.117 [TrillPitchHead], page 348, Section 3.1.118 [TrillSpanner], page 349, Section 3.1.119 [TupletBracket], page 350, Section 3.1.120 [TupletNumber], page 351 and Section 3.1.126 [VoiceFollower], page 356.

This context sets the following properties:

• Set translator property localKeySignature to ‘()’.

This context is a ‘bottom’ context; it cannot contain other contexts.

This context is built from the following engraver(s):

Section 2.2.94 [Skip_event_swallow_translator], page 236
Swallow \skip.

Section 2.2.49 [Instrument_switch_engraver], page 222
Create a cue text for taking instrument.

Properties (read)

instrumentCueName (markup)
The name to print if another instrument is to be taken.
This engraver creates the following layout object(s):
Section 3.1.47 [InstrumentSwitch], page 294.

**Section 2.2.40 [Grace_engraver], page 219**
Set font size and other properties for grace notes.
Properties (read)

```
graceSettings (list)
   Overrides for grace notes. This property should be manipulated through the add-grace-property function.
```

**Section 2.2.125 [Tuplet_engraver], page 244**
Catch tuplet events and generate appropriate bracket.
Music types accepted:
Section 1.2.65 [tuplet-span-event], page 42
Properties (read)

```
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.
```

This engraver creates the following layout object(s):
Section 3.1.119 [TupletBracket], page 350 and Section 3.1.120 [Tuplet-Number], page 351.

**Section 2.2.118 [Tie_engraver], page 242**
Generate ties between note heads of equal pitch.
Music types accepted:
Section 1.2.61 [tie-event], page 42
Properties (read)

```
tieWaitForNote (boolean)
   If true, tied notes do not have to follow each other directly. This can be used for writing out arpeggios.
```

Properties (write)

```
tieMelismaBusy (boolean)
   Signal whether a tie is present.
```

This engraver creates the following layout object(s):
Section 3.1.112 [Tie], page 344 and Section 3.1.113 [TieColumn], page 345.

**Section 2.2.96 [Slur_engraver], page 237**
Build slur grobs from slur events.
Music types accepted:
Section 1.2.48 [slur-event], page 40
Properties (read)
slurMelismaBusy (boolean)
    Signal if a slur is present.

doubleSlurs (boolean)
    If set, two slurs are created for every slurred note, one above and one below the chord.

This engraver creates the following layout object(s):
Section 3.1.90 [Slur], page 324.

Section 2.2.17 [Cluster_spanner_engraver], page 212
    Engrave a cluster using Spanner notation.
    Music types accepted:
    Section 1.2.13 [cluster-note-event], page 36
    This engraver creates the following layout object(s):
    Section 3.1.26 [ClusterSpanner], page 276 and Section 3.1.27 [ClusterSpannerBeacon], page 276.

Section 2.2.78 [Phrasing_slur_engraver], page 232
    Print phrasing slurs. Similar to Section 2.2.96 [Slur_engraver], page 237.
    Music types accepted:
    Section 1.2.42 [phrasing-slur-event], page 40
    This engraver creates the following layout object(s):
    Section 3.1.78 [PhrasingSlur], page 317.

Section 2.2.101 [Spanner_break_forbid_engraver], page 238
    Forbid breaks in certain spanners.

Section 2.2.69 [Note_spacing_engraver], page 229
    Generate NoteSpacing, an object linking horizontal lines for use in spacing.
    This engraver creates the following layout object(s):
    Section 3.1.71 [NoteSpacing], page 311.

Section 2.2.89 [Rhythmic_column_engraver], page 235
    Generate NoteColumn, an object that groups stems, note heads, and rests.
    This engraver creates the following layout object(s):
    Section 3.1.68 [NoteColumn], page 310.

Section 2.2.90 [Script_column_engraver], page 235
    Find potentially colliding scripts and put them into a ScriptColumn object; that will fix the collisions.
    This engraver creates the following layout object(s):
    Section 3.1.87 [ScriptColumn], page 323.

Section 2.2.91 [Script_engraver], page 235
    Handle note scripted articulations.
    Music types accepted:
    Section 1.2.6 [articulation-event], page 36
    Properties (read)
scriptDefinitions (list)
The description of scripts. This is used by the Script_engraver for typesetting note-superscripts and subscripts. See ‘scm/script.scm’ for more information.

This engraver creates the following layout object(s):
Section 3.1.86 [Script], page 322.

Section 2.2.11 [Bend_engraver], page 210
Create fall spanners.
Music types accepted:
Section 1.2.10 [bend-after-event], page 36
This engraver creates the following layout object(s):
Section 3.1.20 [BendAfter], page 271.

Section 2.2.34 [Fingering_engraver], page 217
Create fingering scripts.
Music types accepted:
Section 1.2.57 [stroke-finger-event], page 42 and Section 1.2.18 [fingering-event], page 37
This engraver creates the following layout object(s):
Section 3.1.37 [Fingering], page 285.

Section 2.2.27 [Dynamic_align_engraver], page 215
Align hairpins and dynamic texts on a horizontal line
Properties (read)

  currentMusicalColumn (layout object)
  Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):
Section 3.1.34 [DynamicLineSpanner], page 282.

Section 2.2.63 [New_dynamic_engraver], page 227
Create hairpins, dynamic texts, and their vertical alignments. The symbols are collected onto a DynamicLineSpanner grob which takes care of vertical positioning.
Music types accepted:
Section 1.2.53 [span-dynamic-event], page 41 and Section 1.2.2 [absolute-dynamic-event], page 35
Properties (read)

  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.’.

  currentMusicalColumn (layout object)
  Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).
**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.’.

This engraver creates the following layout object(s):
Section 3.1.35 [DynamicText], page 283, Section 3.1.36 [DynamicTextSpanner], page 284 and Section 3.1.43 [Hairpin], page 290.

**Section 2.2.116 [Text_engraver], page 241**
Create text scripts.
Music types accepted:
Section 1.2.59 [text-script-event], page 42
This engraver creates the following layout object(s):
Section 3.1.110 [TextScript], page 341.

**Section 2.2.76 [Part_combine_engraver], page 231**
Part combine engraver for orchestral scores: Print markings ‘a2’, ‘Solo’, ‘Solo II’, and ‘unisono’.
Music types accepted:
Section 1.2.38 [part-combine-event], page 39
Properties (read)

**printPartCombineTexts** (boolean)
Set ‘Solo’ and ‘A due’ texts in the part combiner?

**soloText** (markup)
The text for the start of a solo when part-combining.

**soloIIText** (markup)
The text for the start of a solo for voice ‘two’ when part-combining.

**aDueText** (markup)
Text to print at a unisono passage.

This engraver creates the following layout object(s):
Section 3.1.28 [CombineTextScript], page 277.

**Section 2.2.95 [Slash_repeat_engraver], page 236**
Make beat repeats.
Music types accepted:
Section 1.2.40 [percent-event], page 40
Properties (read)

**measureLength** (moment)
Length of one measure in the current time signature.
Chapter 2: Translation

This engraver creates the following layout object(s):
Section 3.1.81 [RepeatSlash], page 320.

Section 2.2.77 [Percent_repeat engraver], page 231
Make whole bar and double bar repeats.
Music types accepted:
Section 1.2.40 [percent-event], page 40
Properties (read)

  countPercentRepeats (boolean)
  If set, produce counters for percent repeats.

  currentCommandColumn (layout object)
  Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

  measureLength (moment)
  Length of one measure in the current time signature.

  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.

Properties (write)

  forbidBreak (boolean)
  If set to \##t, prevent a line break at this point.

This engraver creates the following layout object(s):
Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.76 [PercentRepeat], page 315 and Section 3.1.77 [PercentRepeatCounter], page 315.

Section 2.2.15 [Chord_tremolo_engraver], page 211
Generate beams for tremolo repeats.
Music types accepted:
Section 1.2.63 [tremolo-span-event], page 42
This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.64 [New_fingering_engraver], page 227
Create fingering scripts for notes in a new chord. This engraver is ill-named, since it also takes care of articulations and harmonic note heads.
Properties (read)

  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.

  harmonicDots (boolean)
  If set, harmonic notes in dotted chords get dots.
strokeFingerOrientations (list)
  See fingeringOrientations.

stringNumberOrientations (list)
  See fingeringOrientations.

This engraver creates the following layout object(s):
Section 3.1.37 [Fingering], page 285, Section 3.1.86 [Script], page 322,
Section 3.1.100 [StringNumber], page 332 and Section 3.1.101 [StrokeFinger], page 334.

Section 2.2.4 [Auto_beam_engraver], page 207
Generate beams based on measure characteristics and observed
Stems. Uses beatLength, measureLength, and measurePosition
to decide when to start and stop a beam. Overriding beaming is
done through Section 2.2.107 [Stem_engraver], page 239 properties
stemLeftBeamCount and stemRightBeamCount.
Music types accepted:
Section 1.2.9 [beam-forbid-event], page 36
Properties (read)
  autoBeaming (boolean)
    If set to true then beams are generated automatically.

  autoBeamSettings (list)
    Specifies when automatically generated beams
    should begin and end. See Section “Setting au-
    tomatic beam behavior” in Notation Reference
    for more information.

  beatLength (moment)
    The length of one beat in this time signature.

  subdivideBeams (boolean)
    If set, multiple beams will be subdivided at beat
    positions by only drawing one beam over the
    beat.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.39 [Grace_beam_engraver], page 219
Handle Beam events by engraving beams. If omitted, then notes are
printed with flags instead of beams. Only engraves beams when we are
at grace points in time.
Music types accepted:
Section 1.2.8 [beam-event], page 36
Properties (read)
  beamMelismaBusy (boolean)
    Signal if a beam is present.

  beatLength (moment)
    The length of one beat in this time signature.
subdivideBeams (boolean)
   If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.9 [Beam_engraver], page 209
Handle Beam events by engraving beams. If omitted, then notes are printed with flags instead of beams.
Music types accepted:
Section 1.2.8 [beam-event], page 36
Properties (read)
   beamMelismaBusy (boolean)
      Signal if a beam is present.

   beatLength (moment)
      The length of one beat in this time signature.

   subdivideBeams (boolean)
      If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

Properties (write)
   forbidBreak (boolean)
      If set to #\#, prevent a line break at this point.

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Section 2.2.107 [Stem_engraver], page 239
Create stems and single-stem tremolos. It also works together with the beam engraver for overriding beaming.
Music types accepted:
Section 1.2.62 [tremolo-event], page 42
Properties (read)
   tremoloFlags (integer)
      The number of tremolo flags to add if no number is specified.

   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.

This engraver creates the following layout object(s):
Section 3.1.98 [Stem], page 330 and Section 3.1.99 [StemTremolo], page 332.
Section 2.2.126 [Tweak_engraver], page 244
Read the \texttt{tweaks} property from the originating event, and set properties.

Section 2.2.87 [Rest_engraver], page 234
Engrave rests.
Music types accepted:
Section 1.2.44 [rest-event], page 40
Properties (read)
\begin{itemize}
\item \texttt{middleCPosition} (number)
  The place of the middle C, measured in half staff-spaces. Usually determined by looking at \texttt{middleCClefPosition} and \texttt{middleCOffset}.
\end{itemize}

This engraver creates the following layout object(s):
Section 3.1.84 [Rest], page 321.

Section 2.2.24 [Dots_engraver], page 214
Create Section 3.1.31 [Dots], page 279 objects for Section 3.2.80 [rhythmic-head-interface], page 397s.
This engraver creates the following layout object(s):
Section 3.1.31 [Dots], page 279.

Section 2.2.66 [Note_heads_engraver], page 228
Generate note heads.
Music types accepted:
Section 1.2.34 [note-event], page 39
Properties (read)
\begin{itemize}
\item \texttt{middleCPosition} (number)
  The place of the middle C, measured in half staff-spaces. Usually determined by looking at \texttt{middleCClefPosition} and \texttt{middleCOffset}.
\item \texttt{staffLineLayoutFunction} (procedure)
  Layout of staff lines, traditional, or semitone.
\end{itemize}

This engraver creates the following layout object(s):
Section 3.1.69 [NoteHead], page 310.

Section 2.2.13 [Breathing_sign_engraver], page 210
Create a breathing sign.
Music types accepted:
Section 1.2.12 [breathing-event], page 36
This engraver creates the following layout object(s):
Section 3.1.23 [BreathingSign], page 273.

Section 2.2.54 [Ligature_bracket_engraver], page 224
Handle \texttt{Ligature_events} by engraving \texttt{Ligature} brackets.
Music types accepted:
Section 1.2.26 [ligature-event], page 38
This engraver creates the following layout object(s):
Section 3.1.54 [LigatureBracket], page 299.
Section 2.2.38 [Glissando_engraver], page 219
Engrave glissandi.
Music types accepted:
Section 1.2.19 [glissando-event], page 37
This engraver creates the following layout object(s):
Section 3.1.39 [Glissando], page 288.

Section 2.2.65 [Note_head_line_engraver], page 228
Engrave a line between two note heads, for example a glissando. If followVoice is set, staff switches also generate a line.
Properties (read)

followVoice (boolean)
If set, note heads are tracked across staff switches by a thin line.

This engraver creates the following layout object(s):
Section 3.1.39 [Glissando], page 288 and Section 3.1.126 [VoiceFollower], page 356.

Section 2.2.85 [Repeat_tie_engraver], page 234
Create repeat ties.
Music types accepted:
Section 1.2.43 [repeat-tie-event], page 40
This engraver creates the following layout object(s):
Section 3.1.82 [RepeatTie], page 320 and Section 3.1.83 [RepeatTieColumn], page 321.

Section 2.2.52 [Laissez_vibrer_engraver], page 223
Create laissez vibrer items.
Music types accepted:
Section 1.2.24 [laissez-vibrer-event], page 37
This engraver creates the following layout object(s):
Section 3.1.50 [LaissezVibrerTie], page 297 and Section 3.1.51 [LaissezVibrerTieColumn], page 297.

Section 2.2.36 [Forbid_line_break_engraver], page 218
Forbid line breaks when note heads are still playing at some point.
Properties (read)

busyGrobs (list)
A queue of (end-moment . GROB) cons cells.
This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Properties (write)

forbidBreak (boolean)
If set to ##t, prevent a line break at this point.

Section 2.2.44 [Grob_pq_engraver], page 220
Administrate when certain grobs (e.g., note heads) stop playing.
Properties (read)
busynGrob (list)
A queue of (end-moment . GROB) cons cells. This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Properties (write)

busynGrob (list)
A queue of (end-moment . GROB) cons cells. This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Section 2.2.124 [Trill_spanner_ engraver], page 244
Create trill spanner from an event.
Music types accepted:
Section 1.2.64 [trill-span-event], page 42
Properties (read)

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

currentMusicalColumn (layout object)
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):
Section 3.1.118 [TrillSpanner], page 349.

Section 2.2.117 [Text spanner_ engraver], page 242
Create text spanner from an event.
Music types accepted:
Section 1.2.60 [text-span-event], page 42
This engraver creates the following layout object(s):
Section 3.1.111 [TextSpanner], page 343.

Section 2.2.62 [Multi_measure_rest_ engraver], page 226
Engrave multi-measure rests that are produced with ‘R’. It reads measurePosition and internalBarNumber to determine what number to print over the Section 3.1.63 [MultiMeasureRest], page 305. Reads measureLength to determine whether it should use a whole rest or a breve rest to represent one measure.
Music types accepted:
Section 1.2.32 [multi-measure-text-event], page 38 and Section 1.2.31 [multi-measure-rest-event], page 38
Properties (read)

internalBarNumber (integer)
Contains the current barnumber. This property is used for internal timekeeping, among others by the Accidental_ engraver.
restNumberThreshold (number)
  If a multimeasure rest has more measures than this, a number is printed.

breakableSeparationItem (layout object)
  The breakable items in this time step, for this staff.

currentCommandColumn (layout object)
  Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

measurePosition (moment)
  How much of the current measure have we had. This can be set manually to create incomplete measures.

measureLength (moment)
  Length of one measure in the current time signature.

This engraver creates the following layout object(s):
Section 3.1.63 [MultiMeasureRest], page 305, Section 3.1.64 [MultiMeasureRestNumber], page 306 and Section 3.1.65 [MultiMeasureRestText], page 307.

Section 2.2.3 [Arpeggio_engraver], page 207
  Generate an Arpeggio symbol.
  Music types accepted:
  Section 1.2.5 [arpeggio-event], page 35
  This engraver creates the following layout object(s):
  Section 3.1.9 [Arpeggio], page 263.

Section 2.2.72 [Output_property_engraver], page 229
  Apply a procedure to any grob acknowledged.
  Music types accepted:
  Section 1.2.4 [apply-output-event], page 35

Section 2.2.83 [Pitched_trill_engraver], page 233
  Print the bracketed note head after a note head with trill.
  This engraver creates the following layout object(s):
  Section 3.1.115 [TrillPitchAccidental], page 346, Section 3.1.116 [TrillPitchGroup], page 347 and Section 3.1.117 [TrillPitchHead], page 348.

Section 2.2.35 [Font_size_engraver], page 217
  Put fontSize into font-size grob property.
  Properties (read)

    fontSize (number)
      The relative size of all grobs in a context.

2.2 Engravers and Performers

See Section “Modifying context plug-ins” in Notation Reference.
2.2.1 Accidental_engraver

Make accidentals. Catch note heads, ties and notices key-change events. This engraver usually lives at Staff level, but reads the settings for Accidental at Voice level, so you can \override them at Voice.

Properties (read)

\begin{itemize}
\item \texttt{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.

\item \texttt{symbol} The symbol is the name of the context in which the following rules are to be applied. For example, if \texttt{context} is Section “Score” in Internals Reference then all staves share accidentals, and if \texttt{context} is Section “Staff” in Internals Reference then all voices in the same staff share accidentals, but staves do not.

\item \texttt{procedure} The procedure represents an accidental rule to be applied to the previously specified context.

\end{itemize}

\begin{itemize}
\item The procedure takes the following arguments:

\item \texttt{context} The current context to which the rule should be applied.

\item \texttt{pitch} The pitch of the note to be evaluated.

\item \texttt{barnum} The current bar number.

\item \texttt{measurepos} The current measure position.

\end{itemize}

\begin{itemize}
\item 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.

\item \texttt{autoCautionaries} (list)

\end{itemize}

\begin{itemize}
\item List similar to \texttt{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.

\item \texttt{internalBarNumber} (integer)

\end{itemize}

\begin{itemize}
\item Contains the current barnumber. This property is used for internal timekeeping, among others by the \texttt{Accidental_engraver}.

\item \texttt{extraNatural} (boolean)

\end{itemize}

\begin{itemize}
\item Whether to typeset an extra natural sign before accidentals changing from a non-natural to another non-natural.

\item \texttt{harmonicAccidentals} (boolean)

\end{itemize}

\begin{itemize}
\item If set, harmonic notes in chords get accidentals.

\item \texttt{keySignature} (list)

\end{itemize}

\begin{itemize}
\item The current key signature. This is an alist containing ($\text{step} . \text{alter}$) or ($\text{octave} . \text{step} . \text{alter}$), where \text{step} is a number in the range 0 to 6 and \text{alter} a fraction, denoting alteration. For alterations, use symbols, e.g. \texttt{keySignature = #\texttt{’}((6 . ,FLAT))}. 

\end{itemize}
localKeySignature (list)
The key signature at this point in the measure. The format is the same as for keySignature, but can also contain ((octave . name) . (alter barnumber . measureposition)) pairs.

Properties (write)

localKeySignature (list)
The key signature at this point in the measure. The format is the same as for keySignature, but can also contain ((octave . name) . (alter barnumber . measureposition)) pairs.

This engraver creates the following layout object(s):

Section 3.1.1 [Accidental], page 257, Section 3.1.2 [AccidentalCautionary], page 257, Section 3.1.3 [AccidentalPlacement], page 258 and Section 3.1.4 [AccidentalSuggestion], page 259.

Accidental_engraver is part of the following context(s): Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.20 [Staff], page 145 and Section 2.1.24 [VaticanaStaff], page 173.

2.2.2 Ambitus_engraver
This engraver creates the following layout object(s):

Section 3.1.3 [AccidentalPlacement], page 258, Section 3.1.5 [Ambitus], page 260, Section 3.1.6 [AmbitusAccidental], page 261, Section 3.1.7 [AmbitusLine], page 262 and Section 3.1.8 [AmbitusNoteHead], page 262.

Ambitus_engraver is not part of any context.

2.2.3 Arpeggio_engraver
Generate an Arpeggio symbol.

Music types accepted:

Section 1.2.5 [arpeggio-event], page 35

This engraver creates the following layout object(s):

Section 3.1.9 [Arpeggio], page 263.

Arpeggio_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.4 Auto_beam_engraver
Generate beams based on measure characteristics and observed Stems. Uses beatLength, measureLength, and measurePosition to decide when to start and stop a beam. Overriding beaming is done through Section 2.2.107 [Stem_engraver], page 239 properties stemLeftBeamCount and stemRightBeamCount.

Music types accepted:

Section 1.2.9 [beam-forbid-event], page 36

Properties (read)

autoBeaming (boolean)
If set to true then beams are generated automatically.
autoBeamSettings (list)
   Specifies when automatically generated beams should begin and end.
   See Section “Setting automatic beam behavior” in Notation Reference
   for more information.

beatLength (moment)
   The length of one beat in this time signature.

subdivideBeams (boolean)
   If set, multiple beams will be subdivided at beat positions by only draw-
   ing one beam over the beat.

This engraver creates the following layout object(s):
   Section 3.1.19 [Beam], page 270.

Auto_beam_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50,
   Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92,
   Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25
   [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.5 Axis_group_engraver
Group all objects created in this context in a VerticalAxisGroup spanner.

   Properties (read)
   
   currentCommandColumn (layout object)
   Grob that is X-parent to all current breakable (clef, key signature, etc.)
   items.

This engraver creates the following layout object(s):
   Section 3.1.125 [VerticalAxisGroup], page 355.

Axis_group_engraver is part of the following context(s): Section 2.1.5 [DrumStaff], page 62,
   Section 2.1.8 [FretBoards], page 80, Section 2.1.11 [GregorianTranscriptionStaff], page 83,
   Section 2.1.14 [MensuralStaff], page 106, Section 2.1.16 [NoteNames], page 127, Section 2.1.18
   [RhythmicStaff], page 130, Section 2.1.20 [Staff], page 145, Section 2.1.22 [TabStaff], page 155
   and Section 2.1.24 [VaticanaStaff], page 173.

2.2.6 Balloon_engraver
Create balloon texts.

   Music types accepted:
   Section 1.2.3 [annotate-output-event], page 35

This engraver creates the following layout object(s):
   Section 3.1.10 [BalloonTextItem], page 264.

Balloon_engraver is not part of any context.

2.2.7 Bar_engraver
Create barlines. This engraver is controlled through the whichBar property. If it has no bar
   line to create, it will forbid a linebreak at this point.

   Properties (read)
   
   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 Section “bar-line-interface” in Internals Reference.

Properties (write)

\textbf{forbidBreak} (boolean)  
If set to \#t, prevent a line break at this point.

This engraver creates the following layout object(s):

\textbf{Bar_engraver} is part of the following context(s): Section 2.1.5 [DrumStaff], page 62, Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.18 [RhythmicStaff], page 130, Section 2.1.20 [Staff], page 145, Section 2.1.22 [TabStaff], page 155 and Section 2.1.24 [VaticanaStaff], page 173.

\subsection{2.2.8 Bar_number_engraver}

A bar number is created whenever \texttt{measurePosition} is zero and when there is a bar line (i.e., when \texttt{whichBar} is set). It is put on top of all staves, and appears only at the left side of the staff. The staves are taken from \texttt{stavesFound}, which is maintained by Section 2.2.102 [Staff_collecting_engraver], page 238.

Properties (read)

\textbf{currentBarNumber} (integer)  
Contains the current bar number. This property is incremented at every bar line.

\textbf{whichBar} (string)  
This property is read to determine what type of bar line to create.  
Example:

\texttt{\set Staff.whichBar = "|:"}

This will create a start-repeat bar in this staff only. Valid values are described in Section “bar-line-interface” in Internals Reference.

\textbf{stavesFound} (list of grobs)  
A list of all staff-symbols found.

\textbf{barNumberVisibility} (procedure)  
A Procedure that takes an integer and returns whether the corresponding bar number should be printed.

This engraver creates the following layout object(s):

\textbf{Bar_number_engraver} is part of the following context(s): Section 2.1.19 [Score], page 133.

\subsection{2.2.9 Beam_engraver}

Handle Beam events by engraving beams. If omitted, then notes are printed with flags instead of beams.

Music types accepted:

\textbf{Section 1.2.8 [beam-event], page 36}

Properties (read)

\textbf{beamMelismaBusy} (boolean)  
Signal if a beam is present.
beatLength (moment)
   The length of one beat in this time signature.

subdivideBeams (boolean)
   If set, multiple beams will be subdivided at beat positions by only draw-
   ing one beam over the beat.

Properties (write)
   forbidBreak (boolean)
      If set to \#, prevent a line break at this point.

This engraver creates the following layout object(s):
   Section 3.1.19 [Beam], page 270.

   Beam_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50,
   Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92,
   Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25
   [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.10 Beam_performer

Music types accepted:
   Section 1.2.8 [beam-event], page 36

   Beam_performer is not part of any context.

2.2.11 Bend_engraver

Create fall spanners.

Music types accepted:
   Section 1.2.10 [bend-after-event], page 36

This engraver creates the following layout object(s):
   Section 3.1.20 [BendAfter], page 271.

   Bend_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50,
   Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92,
   Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25
   [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.12 Break_align_engraver

Align grobs with corresponding break-align-symbols into groups, and order the groups ac-
   cording to breakAlignOrder. The left edge of the alignment gets a separate group, with a
   symbol left-edge.

This engraver creates the following layout object(s):
   Section 3.1.21 [BreakAlignGroup], page 272, Section 3.1.22 [BreakAlignment], page 272 and
   Section 3.1.53 [LeftEdge], page 298.

   Break_align_engraver is part of the following context(s): Section 2.1.19 [Score], page 133.

2.2.13 Breathing_sign_engraver

Create a breathing sign.

Music types accepted:
   Section 1.2.12 [breathing-event], page 36

This engraver creates the following layout object(s):
   Section 3.1.23 [BreathingSign], page 273.
Breathing_sign_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.14 Chord_name_engraver

Catch note events and generate the appropriate chordname.

Music types accepted:
Section 1.2.34 [note-event], page 39

Properties (read)

chordChanges (boolean)
Only show changes in chords scheme?

chordNameExceptions (list)
An list of chord exceptions. Contains (chord . markup) entries.

chordNameFunction (procedure)
The function that converts lists of pitches to chord names.

chordNoteNamer (procedure)
A function that converts from a pitch object to a text markup. Used for single pitches.

chordRootNamer (procedure)
A function that converts from a pitch object to a text markup. Used for chords.

chordNameExceptions (list)
An list of chord exceptions. Contains (chord . markup) entries.

majorSevenSymbol (markup)
How should the major 7th be formatted in a chord name?

This engraver creates the following layout object(s):
Section 3.1.24 [ChordName], page 274.

Chord_name_engraver is part of the following context(s): Section 2.1.2 [ChordNames], page 48.

2.2.15 Chord_tremolo_engraver

Generate beams for tremolo repeats.

Music types accepted:
Section 1.2.63 [tremolo-span-event], page 42

This engraver creates the following layout object(s):
Section 3.1.19 [Beam], page 270.

Chord_tremolo_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.
2.2.16 Clef_engraver
Determine and set reference point for pitches.

Properties (read)

- 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.

- explicitClefVisibility (vector)
  ‘break-visibility’ function for clef changes.

- 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.

This engraver creates the following layout object(s):

Section 3.1.25 [Clef], page 275 and Section 3.1.72 [OctavateEight], page 311.

Clef_engraver is part of the following context(s): Section 2.1.5 [DrumStaff], page 62,
Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.14 [MensuralStaff], page 106,
Section 2.1.20 [Staff], page 145, Section 2.1.22 [TabStaff], page 155 and Section 2.1.24 [VaticanaStaff], page 173.

2.2.17 Cluster_spanner_engraver
Engrave a cluster using Spanner notation.

Music types accepted:

Section 1.2.13 [cluster-note-event], page 36

This engraver creates the following layout object(s):

Section 3.1.26 [ClusterSpanner], page 276 and Section 3.1.27 [ClusterSpannerBeacon],
page 276.

Cluster_spanner_engraver is part of the following context(s): Section 2.1.3 [CueVoice],
page 50, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice],
page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and
Section 2.1.26 [Voice], page 194.

2.2.18 Collision_engraver
Collect NoteColumns, and as soon as there are two or more, put them in a NoteCollision
object.

This engraver creates the following layout object(s):

Section 3.1.67 [NoteCollision], page 309.

Collision_engraver is part of the following context(s): Section 2.1.5 [DrumStaff], page 62,
Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.14 [MensuralStaff], page 106,
Section 2.1.20 [Staff], page 145, Section 2.1.22 [TabStaff], page 155 and Section 2.1.24 [VaticanaStaff], page 173.
2.2.19 Completion_heads_engraver

This engraver replaces \texttt{Note_heads_engraver}. It plays some trickery to break long notes and automatically tie them into the next measure.

Music types accepted:

\begin{itemize}
\item Section 1.2.61 \texttt{[tie-event]}, page 42 and Section 1.2.34 \texttt{[note-event]}, page 39
\end{itemize}

Properties (read)

\begin{itemize}
\item \texttt{middleCPosition} (number)
  The place of the middle C, measured in half staff-spaces. Usually determined by looking at \texttt{middleCClefPosition} and \texttt{middleCOffset}.

\item \texttt{measurePosition} (moment)
  How much of the current measure have we had. This can be set manually to create incomplete measures.

\item \texttt{measureLength} (moment)
  Length of one measure in the current time signature.
\end{itemize}

Properties (write)

\begin{itemize}
\item \texttt{completionBusy} (boolean)
  Whether a completion-note head is playing.
\end{itemize}

This engraver creates the following layout object(s):

\begin{itemize}
\item Section 3.1.31 \texttt{[Dots]}, page 279, Section 3.1.69 \texttt{[NoteHead]}, page 310 and Section 3.1.112 \texttt{[Tie]}, page 344.
\end{itemize}

\texttt{Completion_heads_engraver} is not part of any context.

2.2.20 Control_track_performer

\texttt{Control_track_performer} is not part of any context.

2.2.21 Custos_engraver

Engrave custodes.

This engraver creates the following layout object(s):

\begin{itemize}
\item Section 3.1.29 \texttt{[Custos]}, page 278.
\end{itemize}

\texttt{Custos_engraver} is part of the following context(s): Section 2.1.14 \texttt{[MensuralStaff]}, page 106 and Section 2.1.24 \texttt{[VaticanaStaff]}, page 173.

2.2.22 Default_bar_line_engraver

This engraver determines what kind of automatic bar lines should be produced, and sets \texttt{whichBar} accordingly. It should be at the same level as Section 2.2.122 \texttt{[Timing_translator]}, page 243.

Properties (read)

\begin{itemize}
\item \texttt{automaticBars} (boolean)
  If set to false then bar lines will not be printed automatically; they must be explicitly created with a \texttt{\bar} command. Unlike the \texttt{\cadenzaOn} keyword, measures are still counted. Bar line generation will resume according to that count if this property is unset.

\item \texttt{barAlways} (boolean)
  If set to true a bar line is drawn after each note.
\end{itemize}
defaultBarType (string)
Set the default type of bar line. See whichBar for information on available bar types.
This variable is read by Section “Timing translator” in Internals Reference at Section “Score” in Internals Reference level.

measureLength (moment)
Length of one measure in the current time signature.

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 Section “bar-line-interface” in Internals Reference.

measurePosition (moment)
How much of the current measure have we had. This can be set manually to create incomplete measures.

Properties (write)

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.

Default_bar_line_engraver is part of the following context(s): Section 2.1.19 [Score], page 133.

2.2.23 Dot_column_engraver
Engrave dots on dotted notes shifted to the right of the note. If omitted, then dots appear on top of the notes.

This engraver creates the following layout object(s):
Section 3.1.30 [DotColumn], page 279.

Dot_column_engraver is part of the following context(s): Section 2.1.5 [DrumStaff], page 62, Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.18 [RhythmicStaff], page 130, Section 2.1.20 [Staff], page 145, Section 2.1.22 [TabStaff], page 155 and Section 2.1.24 [VaticanaStaff], page 173.

2.2.24 Dots_engraver
Create Section 3.1.31 [Dots], page 279 objects for Section 3.2.80 [rhythmic-head-interface], page 397s.

This engraver creates the following layout object(s):
Section 3.1.31 [Dots], page 279.

Dots_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.
2.2.25 **Drum_note_performer**

Play drum notes.

Music types accepted:

Section 1.2.34 [note-event], page 39

*Drum_note_performer* is not part of any context.

2.2.26 **Drum_notes_engraver**

Generate drum note heads.

Music types accepted:

Section 1.2.34 [note-event], page 39

Properties (read)

- `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.

This engraver creates the following layout object(s):

Section 3.1.69 [NoteHead], page 310 and Section 3.1.86 [Script], page 322.

*Drum_notes_engraver* is part of the following context(s): Section 2.1.6 [DrumVoice], page 67.

2.2.27 **Dynamic_align_engraver**

Align hairpins and dynamic texts on a horizontal line

Properties (read)

- `currentMusicalColumn` (layout object)
  
  Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):

Section 3.1.34 [DynamicLineSpanner], page 282.

*Dynamic_align_engraver* is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.28 **Dynamic_engraver**

Create hairpins, dynamic texts, and their vertical alignments. The symbols are collected onto a DynamicLineSpanner grob which takes care of vertical positioning.

Music types accepted:

Section 1.2.53 [span-dynamic-event], page 41 and Section 1.2.2 [absolute-dynamic-event], page 35

This engraver creates the following layout object(s):

Section 3.1.34 [DynamicLineSpanner], page 282, Section 3.1.35 [DynamicText], page 283, Section 3.1.36 [DynamicTextSpanner], page 284 and Section 3.1.43 [Hairpin], page 290.

*Dynamic_engraver* is not part of any context.


2.2.29 Dynamic_performer
Music types accepted:
Section 1.2.2 [absolute-dynamic-event], page 35, Section 1.2.14 [crescendo-event], page 36 and Section 1.2.15 [decrescendo-event], page 36
Properties (read)

\texttt{dynamicAbsoluteVolumeFunction} (procedure)
\[
\text{[DOCUMENT-ME]}
\]

\texttt{instrumentEqualizer} (procedure)
A function taking a string (instrument name), and returning a \((\min, \max)\) pair of numbers for the loudness range of the instrument.

\texttt{midiMaximumVolume} (number)
Analogous to \texttt{midiMinimumVolume}.

\texttt{midiMinimumVolume} (number)
Set the minimum loudness for MIDI. Ranges from 0 to 1.

\texttt{midiInstrument} (string)
Name of the MIDI instrument to use.

\texttt{Dynamic_performer} is not part of any context.

2.2.30 Engraver
Base class for engravers. Does nothing, so it is not used.

\texttt{Engraver} is not part of any context.

2.2.31 Extender_engraver
Create lyric extenders.
Music types accepted:
Section 1.2.17 [extender-event], page 37
Properties (read)

\texttt{extendersOverRests} (boolean)
Whether to continue extenders as they cross a rest.

This engraver creates the following layout object(s):
Section 3.1.55 [LyricExtender], page 300.

\texttt{Extender_engraver} is part of the following context(s): Section 2.1.13 [Lyrics], page 104.

2.2.32 Figured_bass_engraver
Make figured bass numbers.
Music types accepted:
Section 1.2.7 [bass-figure-event], page 36 and Section 1.2.44 [rest-event], page 40
Properties (read)

\texttt{figuredBassAlterationDirection} (direction)
Where to put alterations relative to the main figure.

\texttt{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.

implicitBassFigures (list)
A list of bass figures that are not printed as numbers, but only as extender lines.

useBassFigureExtenders (boolean)
Whether to use extender lines for repeated bass figures.

ignoreFiguredBassRest (boolean)
Don’t swallow rest events.

This engraver creates the following layout object(s):
Section 3.1.13 [BassFigure], page 267, Section 3.1.14 [BassFigureAlignment], page 268, Section 3.1.16 [BassFigureBracket], page 269, Section 3.1.17 [BassFigureContinuation], page 269 and Section 3.1.18 [BassFigureLine], page 270.

Figured_bass_engraver is part of the following context(s): Section 2.1.5 [DrumStaff], page 62, Section 2.1.7 [FiguredBass], page 78, Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.20 [Staff], page 145, Section 2.1.22 [TabStaff], page 155 and Section 2.1.24 [VaticanaStaff], page 173.

2.2.33 Figured_bass_position_engraver
Position figured bass alignments over notes.

This engraver creates the following layout object(s):
Section 3.1.15 [BassFigureAlignmentPositioning], page 268.

Figured_bass_position_engraver is part of the following context(s): Section 2.1.5 [DrumStaff], page 62, Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.20 [Staff], page 145, Section 2.1.22 [TabStaff], page 155 and Section 2.1.24 [VaticanaStaff], page 173.

2.2.34 Fingering_engraver
Create fingering scripts.

Music types accepted:
Section 1.2.57 [stroke-finger-event], page 42 and Section 1.2.18 [fingering-event], page 37

This engraver creates the following layout object(s):
Section 3.1.37 [Fingering], page 285.

Fingering_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.35 Font_size_engraver
Put fontSize into font-size grob property.

Properties (read)

fontSize (number)
The relative size of all grobs in a context.

Font_size_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.5 [DrumStaff], page 62, Section 2.1.6 [DrumVoice], page 67, Section 2.1.8 [Fret-Boards], page 80, Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.13 [Lyrics], page 104, Section 2.1.14 [MensuralStaff],
2.2.36 **Forbid_line_break_engraver**
Forbid line breaks when note heads are still playing at some point.

Properties (read)

- `busyGrobs` (list)
  A queue of \texttt{(end-moment . GROB)} cons cells. This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Properties (write)

- `forbidBreak` (boolean)
  If set to \#t, prevent a line break at this point.

**Forbid_line_break_engraver** is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.37 **Fretboard_engraver**
Generate one or more tablature noteheads from event of type \texttt{NoteEvent}.

Music types accepted:

Section 1.2.34 [note-event], page 39 and Section 1.2.56 [string-number-event], page 41

Properties (read)

- `stringTunings` (list)
  The tablature strings tuning. It is a list of the pitch (in semitones) of each string (starting with the lower one).

- `minimumFret` (number)
  The tablature auto string-selecting mechanism selects the highest string with a fret at least \texttt{minimumFret}.

- `maximumFretStretch` (number)
  Don’t allocate frets further than this from specified frets.

- `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.

- `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.

- `predefinedDiagramTable` (hash table)
  The hash table of predefined fret diagrams to use in FretBoards.

This engraver creates the following layout object(s):

Section 3.1.38 [FretBoard], page 287.

**Fretboard_engraver** is part of the following context(s): Section 2.1.8 [FretBoards], page 80.
Chapter 2: Translation

2.2.38 Glissando\_engraver

Engrave glissandi.

Music types accepted:

Section 1.2.19 [glissando-event], page 37

This engraver creates the following layout object(s):

Section 3.1.39 [Glissando], page 288.

Glissando\_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.39 Grace\_beam\_engraver

Handle Beam events by engraving beams. If omitted, then notes are printed with flags instead of beams. Only engraves beams when we are at grace points in time.

Music types accepted:

Section 1.2.8 [beam-event], page 36

Properties (read)

\begin{itemize}
\item beam\_Melisma\_Busy (boolean)
\end{itemize}

Signal if a beam is present.

\begin{itemize}
\item \texttt{beatLength} (moment)
\end{itemize}

The length of one beat in this time signature.

\begin{itemize}
\item \texttt{subdivide\_Beams} (boolean)
\end{itemize}

If set, multiple beams will be subdivided at beat positions by only drawing one beam over the beat.

This engraver creates the following layout object(s):

Section 3.1.19 [Beam], page 270.

Grace\_beam\_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.40 Grace\_engraver

Set font size and other properties for grace notes.

Properties (read)

\begin{itemize}
\item grace\_Settings (list)
\end{itemize}

Overrides for grace notes. This property should be manipulated through the add\_grace\_property function.

Grace\_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.
2.2.41 Grace_spacing_ engraver

Bookkeeping of shortest starting and playing notes in grace note runs.

Properties (read)

\texttt{currentMusicalColumn} (layout object)
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):
Section 3.1.40 [GraceSpacing], page 289.
Grace_spacing_ engraver is part of the following context(s): Section 2.1.19 [Score], page 133.

2.2.42 Grid_line_span_ engraver

This engraver makes cross-staff lines: It catches all normal lines and draws a single span line across them.

This engraver creates the following layout object(s):
Section 3.1.41 [GridLine], page 289.
Grid_line_span_ engraver is not part of any context.

2.2.43 Grid_point_ engraver

Generate grid points.

Properties (read)

\texttt{gridInterval} (moment)
Interval for which to generate GridPoints.

This engraver creates the following layout object(s):
Section 3.1.42 [GridPoint], page 290.
Grid_point_ engraver is not part of any context.

2.2.44 Grob_pq_ engraver

Administrate when certain grobs (e.g., note heads) stop playing.

Properties (read)

\texttt{busyGrobs} (list)
A queue of (end-moment . GROB) cons cells. This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Properties (write)

\texttt{busyGrobs} (list)
A queue of (end-moment . GROB) cons cells. This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

Grob_pq_ engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.5 [DrumStaff], page 62, Section 2.1.6 [DrumVoice], page 67, Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.20 [Staff], page 145, Section 2.1.22 [TabStaff], page 155, Section 2.1.23 [TabVoice], page 161, Section 2.1.24 [VaticanaStaff], page 173, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.
2.2.45 Hara_kiri_engraver

Like Axis_group_engraver, but make a hara-kiri spanner, and add interesting items (i.e., note heads, lyric syllables, and normal rests).

Properties (read)

keepAliveInterfaces (list)
A list of symbols, signifying grob interfaces that are worth keeping a staff with remove-empty set around for.

This engraver creates the following layout object(s):
Section 3.1.125 [VerticalAxisGroup], page 355.

Hara_kiri_engraver is part of the following context(s): Section 2.1.2 [ChordNames], page 48, Section 2.1.7 [FiguredBass], page 78 and Section 2.1.13 [Lyrics], page 104.

2.2.46 Horizontal_bracket_engraver

Create horizontal brackets over notes for musical analysis purposes.

Music types accepted:
Section 1.2.35 [note-grouping-event], page 39

This engraver creates the following layout object(s):
Section 3.1.45 [HorizontalBracket], page 292.

Horizontal_bracket_engraver is not part of any context.

2.2.47 Hyphen_engraver

Create lyric hyphens and distance constraints between words.

Music types accepted:
Section 1.2.21 [hyphen-event], page 37

This engraver creates the following layout object(s):
Section 3.1.56 [LyricHyphen], page 301 and Section 3.1.57 [LyricSpace], page 302.

Hyphen_engraver is part of the following context(s): Section 2.1.13 [Lyrics], page 104.

2.2.48 Instrument_name_engraver

Create a system start text for instrument or vocal names.

Properties (read)

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

shortInstrumentName (markup)
See 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.

shortVocalName (markup)
Name of a vocal line, short version.

vocalName (markup)
Name of a vocal line.
This engraver creates the following layout object(s):
Section 3.1.46 [InstrumentName], page 293.

Instrument_name_engraver is part of the following context(s): Section 2.1.5 [DrumStaff], page 62, Section 2.1.8 [FretBoards], page 80, Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.13 [Lyrics], page 104, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.17 [PianoStaff], page 129, Section 2.1.18 [RhythmicStaff], page 130, Section 2.1.20 [Staff], page 145, Section 2.1.22 [TabStaff], page 155 and Section 2.1.24 [VaticanaStaff], page 173.

2.2.49 Instrument_switch_engraver
Create a cue text for taking instrument.

Properties (read)

instrumentCueName (markup)
The name to print if another instrument is to be taken.

This engraver creates the following layout object(s):
Section 3.1.47 [InstrumentSwitch], page 294.

Instrument_switch_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.50 Key_engraver
Engrave a key signature.

Music types accepted:
Section 1.2.22 [key-change-event], page 37

Properties (read)

createKeyOnClefChange (boolean)
Print a key signature whenever the clef is changed.

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.

extraNatural (boolean)
Whether to typeset an extra natural sign before accidentals changing from a non-natural to another non-natural.

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)).

lastKeySignature (list)
Last key signature before a key signature change.
printKeyCancellation (boolean)

Print restoration alterations before a key signature change.

Properties (write)

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)).

lastKeySignature (list)

Last key signature before a key signature change.

tonic (pitch)

The tonic of the current scale.

This engraver creates the following layout object(s):

Section 3.1.48 [KeyCancellation], page 295 and Section 3.1.49 [KeySignature], page 296.

Key_engraver is part of the following context(s): Section 2.1.11 [GregorianTranscription-Staff], page 83, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.20 [Staff], page 145 and Section 2.1.24 [VaticanaStaff], page 173.

2.2.51 Key_performer

Music types accepted:

Section 1.2.22 [key-change-event], page 37

Key_performer is not part of any context.

2.2.52 Laissez_vibrer_engraver

Create laissez vibrer items.

Music types accepted:

Section 1.2.24 [laissez-vibrer-event], page 37

This engraver creates the following layout object(s):

Section 3.1.50 [LaissezVibrerTie], page 297 and Section 3.1.51 [LaissezVibrerTieColumn], page 297.

Laissez_vibrer_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.53 Ledger_line_engraver

Create the spanner to draw ledger lines, and notices objects that need ledger lines.

This engraver creates the following layout object(s):

Section 3.1.52 [LedgerLineSpanner], page 298.

Ledger_line_engraver is part of the following context(s): Section 2.1.5 [DrumStaff], page 62, Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.18 [RhythmicStaff], page 130, Section 2.1.20 [Staff], page 145, Section 2.1.22 [TabStaff], page 155 and Section 2.1.24 [VaticanaStaff], page 173.
2.2.54 Ligature_bracket_engraver

Handle Ligature_events by engraving Ligature brackets.

Music types accepted:
- Section 1.2.26 [ligature-event], page 38

This engraver creates the following layout object(s):
- Section 3.1.54 [LigatureBracket], page 299.

Ligature_bracket_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.23 [TabVoice], page 161 and Section 2.1.26 [Voice], page 194.

2.2.55 Lyric_engraver

Engrave text for lyrics.

Music types accepted:
- Section 1.2.28 [lyric-event], page 38

Properties (read)
- ignoreMelismata (boolean)
  Ignore melismata for this Section “Lyrics” in Internals Reference line.

- lyricMelismaAlignment (direction)
  Alignment to use for a melisma syllable.

This engraver creates the following layout object(s):
- Section 3.1.58 [LyricText], page 302.

Lyric_engraver is part of the following context(s): Section 2.1.13 [Lyrics], page 104.

2.2.56 Lyric_performer

Music types accepted:
- Section 1.2.28 [lyric-event], page 38

Lyric_performer is not part of any context.

2.2.57 Mark_engraver

Create RehearsalMark objects. It puts them on top of all staves (which is taken from the property stavesFound). If moving this engraver to a different context, Section 2.2.102 [Staff_collecting_engraver], page 238 must move along, otherwise all marks end up on the same Y location.

Music types accepted:
- Section 1.2.29 [mark-event], page 38

Properties (read)
- markFormatter (procedure)
  A procedure taking as arguments the context and the rehearsal mark.
  It should return the formatted mark as a markup object.

- rehearsalMark (integer)
  The last rehearsal mark printed.

- stavesFound (list of grobs)
  A list of all staff-symbols found.

This engraver creates the following layout object(s):
- Section 3.1.80 [RehearsalMark], page 318.

Mark_engraver is part of the following context(s): Section 2.1.19 [Score], page 133.
2.2.58 Measure_grouping_engraver

Create MeasureGrouping to indicate beat subdivision.

Properties (read)

- **beatLength** (moment)
  The length of one beat in this time signature.

- **currentMusicalColumn** (layout object)
  Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

- **measurePosition** (moment)
  How much of the current measure have we had. This can be set manually to create incomplete measures.

- **beatGrouping** (list)
  A list of beatgroups, e.g., in 5/8 time ‘(2 3).

This engraver creates the following layout object(s):

Section 3.1.59 [MeasureGrouping], page 303.

Measure_grouping_engraver is not part of any context.

2.2.59 Melody_engraver

Create information for context dependent typesetting decisions.

This engraver creates the following layout object(s):

Section 3.1.60 [MelodyItem], page 304.

Melody_engraver is not part of any context.

2.2.60 Mensural_ligature_engraver

Handle Mensural_ligature_events by glueing special ligature heads together.

Music types accepted:

Section 1.2.26 [ligature-event], page 38

This engraver creates the following layout object(s):

Section 3.1.61 [MensuralLigature], page 304.

Mensural_ligature_engraver is part of the following context(s): Section 2.1.15 [MensuralVoice], page 116.

2.2.61 Metronome_mark_engraver

Engrave metronome marking. This delegates the formatting work to the function in the metronomeMarkFormatter property. The mark is put over all staves. The staves are taken from the stavesFound property, which is maintained by Section 2.2.102 [Staff_collecting_engraver], page 238.

Properties (read)

- **stavesFound** (list of grobs)
  A list of all staff-symbols found.

- **metronomeMarkFormatter** (procedure)
  How to produce a metronome markup. Called with four arguments: text, duration, count and context.

- **tempoUnitDuration** (duration)
  Unit for specifying tempo.
tempoUnitCount (number)
Count for specifying tempo.

tempoText (markup)
Text for tempo marks.

tempoHideNote (boolean)
Hide the note=count in tempo marks.

This engraver creates the following layout object(s):
Section 3.1.62 [MetronomeMark], page 304.

Metronome_mark_engraver is part of the following context(s): Section 2.1.19 [Score], page 133.

2.2.62 Multi_measure_rest_engraver
Engrave multi-measure rests that are produced with ‘R’. It reads measurePosition and internalBarNumber to determine what number to print over the Section 3.1.63 [MultiMeasureRest], page 305. Reads measureLength to determine whether it should use a whole rest or a breve rest to represent one measure.

Music types accepted:
Section 1.2.32 [multi-measure-text-event], page 38 and Section 1.2.31 [multi-measure-rest-event], page 38

Properties (read)

internalBarNumber (integer)
Contains the current bar number. This property is used for internal timekeeping, among others by the Accidental_engraver.

restNumberThreshold (number)
If a multi-measure rest has more measures than this, a number is printed.

breakableSeparationItem (layout object)
The breakable items in this time step, for this staff.

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

measurePosition (moment)
How much of the current measure have we had. This can be set manually to create incomplete measures.

measureLength (moment)
Length of one measure in the current time signature.

This engraver creates the following layout object(s):
Section 3.1.63 [MultiMeasureRest], page 305, Section 3.1.64 [MultiMeasureRestNumber], page 306 and Section 3.1.65 [MultiMeasureRestText], page 307.

Multi_measure_rest_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.
2.2.63 New\textunderscore dynamic\textunderscore engraver

Create hairpins, dynamic texts, and their vertical alignments. The symbols are collected onto a DynamicLineSpanner grob which takes care of vertical positioning.

Music types accepted:
Section 1.2.53 [span\textunderscore dynamic\textunderscore event], page 41 and Section 1.2.2 [absolute\textunderscore dynamic\textunderscore event], page 35

Properties (read)

\begin{itemize}
  \item \texttt{crescendoSpanner} (symbol)
    The type of spanner to be used for crescendi. Available values are \texttt{‘hairpin’} and \texttt{‘text’}. If unset, a hairpin crescendo is used.
  \item \texttt{crescendoText} (markup)
    The text to print at start of non-hairpin crescendo, i.e., \texttt{‘cresc.’}.
  \item \texttt{currentMusicalColumn} (layout object)
    Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).
  \item \texttt{decrescendoSpanner} (symbol)
    The type of spanner to be used for decrescendi. Available values are \texttt{‘hairpin’} and \texttt{‘text’}. If unset, a hairpin decrescendo is used.
  \item \texttt{decrescendoText} (markup)
    The text to print at start of non-hairpin decrescendo, i.e., \texttt{‘dim.’}.
\end{itemize}

This engraver creates the following layout object(s):
Section 3.1.35 [DynamicText], page 283, Section 3.1.36 [DynamicTextSpanner], page 284 and Section 3.1.43 [Hairpin], page 290.

\texttt{New\textunderscore dynamic\textunderscore engraver} is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.64 New\textunderscore fingering\textunderscore engraver

Create fingering scripts for notes in a new chord. This engraver is ill-named, since it also takes care of articulations and harmonic note heads.

Properties (read)

\begin{itemize}
  \item \texttt{fingeringOrientations} (list)
    A list of symbols, containing \texttt{‘left’}, \texttt{‘right’}, \texttt{‘up’} and/or \texttt{‘down’}. This list determines where fingerings are put relative to the chord being fingered.
  \item \texttt{harmonicDots} (boolean)
    If set, harmonic notes in dotted chords get dots.
  \item \texttt{strokeFingerOrientations} (list)
    See \texttt{fingeringOrientations}.
  \item \texttt{stringNumberOrientations} (list)
    See \texttt{fingeringOrientations}.
\end{itemize}

This engraver creates the following layout object(s):
Section 3.1.37 [Fingering], page 285, Section 3.1.86 [Script], page 322, Section 3.1.100 [StringNumber], page 332 and Section 3.1.101 [StrokeFinger], page 334.
New_fingering_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.65 Note_head_line_engraver

Engrave a line between two note heads, for example a glissando. If followVoice is set, staff switches also generate a line.

Properties (read)

followVoice (boolean)
   If set, note heads are tracked across staff switches by a thin line.

This engraver creates the following layout object(s):
   Section 3.1.39 [Glissando], page 288 and Section 3.1.126 [VoiceFollower], page 356.

Note_head_line_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.66 Note_heads_engraver

Generate note heads.

Music types accepted:
   Section 1.2.34 [note-event], page 39

Properties (read)

middleCPosition (number)
   The place of the middle C, measured in half staff-spaces. Usually determined by looking at middleCClefPosition and middleCCOffset.

staffLineLayoutFunction (procedure)
   Layout of staff lines, traditional, or semitone.

This engraver creates the following layout object(s):
   Section 3.1.69 [NoteHead], page 310.

Note_heads_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.67 Note_name_engraver

Print pitches as words.

Music types accepted:
   Section 1.2.34 [note-event], page 39

Properties (read)

printOctaveNames (boolean)
   Print octave marks for the NoteNames context.

This engraver creates the following layout object(s):
   Section 3.1.70 [NoteName], page 311.

Note_name_engraver is part of the following context(s): Section 2.1.16 [NoteNames], page 127.
2.2.68 **Note_performer**

Music types accepted:

Section 1.2.34 [note-event], page 39

*Note_performer* is not part of any context.

2.2.69 **Note_spacing_ engraver**

Generate *NoteSpacing*, an object linking horizontal lines for use in spacing.

This engraver creates the following layout object(s):

Section 3.1.71 [NoteSpacing], page 311.

*Note_spacing_ engraver* is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.70 **Note_swallow_ translator**

Swallow notes.

*Note_swallow_ translator* is part of the following context(s): Section 2.1.7 [FiguredBass], page 78.

2.2.71 **Ottava_spanner_ engraver**

Create a text spanner when the ottavation property changes.

**Properties** (read)

- **ottavation** (markup)
  If set, the text for an ottava spanner. Changing this creates a new text spanner.

- **originalMiddleCPosition** (integer)
  Used for temporary overriding middle C in octavation brackets.

- **currentMusicalColumn** (layout object)
  Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):

Section 3.1.73 [OttavaBracket], page 312.

*Ottava_spanner_ engraver* is part of the following context(s): Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.20 [Staff], page 145, Section 2.1.22 [TabStaff], page 155 and Section 2.1.24 [VaticanaStaff], page 173.

2.2.72 **Output_property_ engraver**

Apply a procedure to any grob acknowledged.

Music types accepted:

Section 1.2.4 [apply-output-event], page 35

*Output_property_ engraver* is part of the following context(s): Section 2.1.2 [Chord-Names], page 48, Section 2.1.3 [CueVoice], page 50, Section 2.1.5 [DrumStaff], page 62, Section 2.1.6 [DrumVoice], page 67, Section 2.1.8 [FretBoards], page 80, Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.18 [RhythmicStaff], page 130, Section 2.1.19 [Score], page 133, Section 2.1.20 [Staff], page 145,
2.2.73 Page_turn_engraver

Decide where page turns are allowed to go.

Music types accepted:
Section 1.2.11 [break-event], page 36

Properties (read)

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.

Page_turn_engraver is not part of any context.

2.2.74 Paper_column_engraver

Take care of generating columns.

This engraver decides whether a column is breakable. The default is that a column is always breakable. However, every Bar_engraver that does not have a barline at a certain point will set forbidBreaks in the score context to stop line breaks. In practice, this means that you can make a break point by creating a bar line (assuming that there are no beams or notes that prevent a break point).

Music types accepted:
Section 1.2.23 [label-event], page 37 and Section 1.2.11 [break-event], page 36

Properties (read)

forbidBreak (boolean)
If set to ##t, prevent a line break at this point.

Properties (write)

forbidBreak (boolean)
If set to ##t, prevent a line break at this point.

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

currentMusicalColumn (layout object)
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):
Section 3.1.66 [NonMusicalPaperColumn], page 308 and Section 3.1.74 [PaperColumn], page 314.

Paper_column_engraver is part of the following context(s): Section 2.1.19 [Score], page 133.

2.2.75 Parenthesis_engraver

Parenthesize objects whose music cause has the parenthesize property.

This engraver creates the following layout object(s):
Section 3.1.75 [ParenthesesItem], page 314.

Parenthesis_engraver is part of the following context(s): Section 2.1.19 [Score], page 133.
2.2.76 Part_combine_engraver

Part combine engraver for orchestral scores: Print markings ‘a2’, ‘Solo’, ‘Solo II’, and ‘unisono’.

Music types accepted:
Section 1.2.38 [part-combine-event], page 39

Properties (read)

`printPartCombineTexts` (boolean)
- Set ‘Solo’ and ‘A due’ texts in the part combiner?

`soloText` (markup)
- The text for the start of a solo when part-combining.

`soloIIText` (markup)
- The text for the start of a solo for voice ‘two’ when part-combining.

`aDueText` (markup)
- Text to print at a unisono passage.

This engraver creates the following layout object(s):
Section 3.1.28 [CombineTextScript], page 277.

*Part_combine_engraver* is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.77 Percent_repeat_engraver

Make whole bar and double bar repeats.

Music types accepted:
Section 1.2.40 [percent-event], page 40

Properties (read)

`countPercentRepeats` (boolean)
- If set, produce counters for percent repeats.

`currentCommandColumn` (layout object)
- Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

`measureLength` (moment)
- Length of one measure in the current time signature.

`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.

Properties (write)

`forbidBreak` (boolean)
- If set to `#t`, prevent a line break at this point.

This engraver creates the following layout object(s):
Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.33 [DoublePercentRepeat-Counter], page 281, Section 3.1.76 [PercentRepeat], page 315 and Section 3.1.77 [PercentRepeatCounter], page 315.


**Percent_repeat_engraver** is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.78 **Phrasing_slur_engraver**

Print phrasing slurs. Similar to Section 2.2.96 [Slur_engraver], page 237.

Music types accepted:

Section 1.2.42 [phrasing-slur-event], page 40

This engraver creates the following layout object(s):

Section 3.1.78 [PhrasingSlur], page 317.

**Phrasing_slur_engraver** is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.79 **Piano_pedal_align_engraver**

Align piano pedal symbols and brackets.

Properties (read)

- `currentCommandColumn` (layout object)
  - Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

This engraver creates the following layout object(s):

Section 3.1.92 [SostenutoPedalLineSpanner], page 326, Section 3.1.103 [SustainPedalLineSpanner], page 335 and Section 3.1.122 [UnaCordaPedalLineSpanner], page 353.

**Piano_pedal_align_engraver** is part of the following context(s): Section 2.1.5 [DrumStaff], page 62, Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.20 [Staff], page 145, Section 2.1.22 [TabStaff], page 155 and Section 2.1.24 [VaticanaStaff], page 173.

2.2.80 **Piano_pedal_engraver**

Engrave piano pedal symbols and brackets.

Music types accepted:

Section 1.2.66 [una-corda-event], page 43, Section 1.2.58 [sustain-event], page 42 and Section 1.2.51 [sostenuto-event], page 41

Properties (read)

- `currentCommandColumn` (layout object)
  - Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

- `pedalsSostenutoStrings` (list)
  - See `pedalSustainStrings`.

- `pedalsSostenutoStyle` (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.

This engraver creates the following layout object(s):
   Section 3.1.79 [PianoPedalBracket], page 318, Section 3.1.91 [SostenutoPedal], page 325,
   Section 3.1.102 [SustainPedal], page 335 and Section 3.1.121 [UnaCordaPedal], page 352.

Piano_pedal_engraver is part of the following context(s): Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.20 [Staff], page 145,
   Section 2.1.22 [TabStaff], page 155 and Section 2.1.24 [VaticanaStaff], page 173.

2.2.81 Piano_pedal_performer

Music types accepted:
   Section 1.2.66 [una-corda-event], page 43, Section 1.2.58 [sustain-event], page 42 and
   Section 1.2.51 [sostenuto-event], page 41

Piano_pedal_performer is not part of any context.

2.2.82 Pitch_squash_engraver

Set the vertical position of note heads to squashedPosition, if that property is set. This can be used to make a single-line staff demonstrating the rhythm of a melody.

Properties (read)

   squashedPosition (integer)
      Vertical position of squashing for Section “Pitch_squash_engraver” in Internals Reference.

Pitch_squash_engraver is part of the following context(s): Section 2.1.18 [RhythmicStaff], page 130.

2.2.83 Pitched_trill_engraver

Print the bracketed note head after a note head with trill.

This engraver creates the following layout object(s):
   Section 3.1.115 [TrillPitchAccidental], page 346, Section 3.1.116 [TrillPitchGroup], page 347 and
   Section 3.1.117 [TrillPitchHead], page 348.

Pitched_trill_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.84 Repeat_acknowledge_engraver

Acknowledge repeated music, and convert the contents of repeatCommands into an appropriate setting for whichBar.

Properties (read)

   doubleRepeatType (string)
      Set the default bar line for double repeats.
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.

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 Section “bar-line-interface” in Internals Reference.

Repeat_acknowledge_engraver is part of the following context(s): Section 2.1.19 [Score],
page 133.

2.2.85 Repeat_tie_engraver
Create repeat ties.

Music types accepted:
Section 1.2.43 [repeat-tie-event], page 40

This engraver creates the following layout object(s):
Section 3.1.82 [RepeatTie], page 320 and Section 3.1.83 [RepeatTieColumn], page 321.

Repeat_tie_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50,
Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92,
Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25
[VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.86 Rest_collision_engraver
Handle collisions of rests.

Properties (read)

busyGrobs (list)
A queue of (end-moment . GROB) cons cells. This is for internal (C++)
use only. This property contains the grobs which are still busy (e.g.
note heads, spanners, etc.).

This engraver creates the following layout object(s):
Section 3.1.85 [RestCollision], page 322.

Rest_collision_engraver is part of the following context(s): Section 2.1.5 [DrumStaff],
page 62, Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.14 [MensuralStaff],
page 106, Section 2.1.20 [Staff], page 145, Section 2.1.22 [TabStaff], page 155 and Section 2.1.24
[VaticanaStaff], page 173.

2.2.87 Rest_engraver
Engrave rests.

Music types accepted:
Section 1.2.44 [rest-event], page 40

Properties (read)

middleCPosition (number)
The place of the middle C, measured in half staff-spaces. Usually de-
termined by looking at middleCClefPosition and middleCOffset.
This engraver creates the following layout object(s):
Section 3.1.84 [Rest], page 321.

Rest_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.88 Rest_swallow_translator
Swallow rest.

Rest_swallow_translator is part of the following context(s): Section 2.1.2 [ChordNames], page 48 and Section 2.1.16 [NoteNames], page 127.

2.2.89 Rhythmic_column_engraver
Generate NoteColumn, an object that groups stems, note heads, and rests.

This engraver creates the following layout object(s):
Section 3.1.68 [NoteColumn], page 310.

Rhythmic_column_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.90 Script_column_engraver
Find potentially colliding scripts and put them into a ScriptColumn object; that will fix the collisions.

This engraver creates the following layout object(s):
Section 3.1.87 [ScriptColumn], page 323.

Script_column_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.91 Script_engraver
Handle note scripted articulations.

Music types accepted:
Section 1.2.6 [articulation-event], page 36

Properties (read)

scriptDefinitions (list)
The description of scripts. This is used by the Script_engraver for typesetting note-superscripts and subscripts. See 'scm/script.scm' for more information.

This engraver creates the following layout object(s):
Section 3.1.86 [Script], page 322.

Script_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.
2.2.92 Script_row_engraver

Determine order in horizontal side position elements.

This engraver creates the following layout object(s):

Section 3.1.88 [ScriptRow], page 323.

Script_row_engraver is part of the following context(s): Section 2.1.5 [DrumStaff], page 62, Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.20 [Staff], page 145, Section 2.1.22 [TabStaff], page 155 and Section 2.1.24 [VaticanaStaff], page 173.

2.2.93 Separating_line_group_engraver

Generate objects for computing spacing parameters.

Properties (read)

createSpacing (boolean)
Create StaffSpacing objects? Should be set for staves.

Properties (write)

hasStaffSpacing (boolean)
True if the current CommandColumn contains items that will affect spacing.

This engraver creates the following layout object(s):

Section 3.1.95 [StaffSpacing], page 329.

Separating_line_group_engraver is part of the following context(s): Section 2.1.2 [ChordNames], page 48, Section 2.1.5 [DrumStaff], page 62, Section 2.1.7 [FiguredBass], page 78, Section 2.1.8 [FretBoards], page 80, Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.16 [NoteNames], page 127, Section 2.1.18 [RhythmicStaff], page 130, Section 2.1.20 [Staff], page 145, Section 2.1.22 [TabStaff], page 155 and Section 2.1.24 [VaticanaStaff], page 173.

2.2.94 Skip_event_swallow_translator

Swallow \skip.

Skip_event_swallow_translator is part of the following context(s): Section 2.1.2 [ChordNames], page 48, Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.7 [FiguredBass], page 78, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.13 [Lyrics], page 104, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.16 [NoteNames], page 127, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.95 Slash_repeat_engraver

Make beat repeats.

Music types accepted:
Section 1.2.40 [percent-event], page 40

Properties (read)

measureLength (moment)
Length of one measure in the current time signature.

This engraver creates the following layout object(s):

Section 3.1.81 [RepeatSlash], page 320.
Slash_repeat_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.96 Slur_engraver

Build slur grobs from slur events.

Music types accepted:
Section 1.2.48 [slur-event], page 40

Properties (read)

slurMelismaBusy (boolean)
Signal if a slur is present.

doubleSlurs (boolean)
If set, two slurs are created for every slurred note, one above and one below the chord.

This engraver creates the following layout object(s):
Section 3.1.90 [Slur], page 324.

Slur_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.23 [TabVoice], page 161 and Section 2.1.26 [Voice], page 194.

2.2.97 Slur_performer

Music types accepted:
Section 1.2.48 [slur-event], page 40

Slur_performer is not part of any context.

2.2.98 Spacing_engraver

Make a SpacingSpanner and do bookkeeping of shortest starting and playing notes.

Music types accepted:
Section 1.2.52 [spacing-section-event], page 41

Properties (read)

currentMusicalColumn (layout object)
Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

currentCommandColumn (layout object)
Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

proportionalNotationDuration (moment)
Global override for shortest-playing duration. This is used for switching on proportional notation.

This engraver creates the following layout object(s):
Section 3.1.93 [SpacingSpanner], page 327.

Spacing_engraver is part of the following context(s): Section 2.1.19 [Score], page 133.
2.2.99 Span_arpeggio_engraver
Make arpeggios that span multiple staves.

Properties (read)

- `connectArpeggios` (boolean)
  - If set, connect arpeggios across piano staff.

This engraver creates the following layout object(s):

Section 3.1.9 [Arpeggio], page 263.

`Span_arpeggio_engraver` is part of the following context(s): Section 2.1.10 [GrandStaff], page 82, Section 2.1.17 [PianoStaff], page 129 and Section 2.1.21 [StaffGroup], page 154.

2.2.100 Span_bar_engraver
Make cross-staff bar lines: It catches all normal bar lines and draws a single span bar across them.

This engraver creates the following layout object(s):

Section 3.1.94 [SpanBar], page 328.

`Span_bar_engraver` is part of the following context(s): Section 2.1.10 [GrandStaff], page 82, Section 2.1.17 [PianoStaff], page 129 and Section 2.1.21 [StaffGroup], page 154.

2.2.101 Spanner_break_forbid_engraver
Forbid breaks in certain spanners.

`Spanner_break_forbid_engraver` is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.102 Staff_collecting_engraver
Maintain the `stavesFound` variable.

Properties (read)

- `stavesFound` (list of grobs)
  - A list of all staff-symbols found.

Properties (write)

- `stavesFound` (list of grobs)
  - A list of all staff-symbols found.

`Staff_collecting_engraver` is part of the following context(s): Section 2.1.5 [DrumStaff], page 62, Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.19 [Score], page 133, Section 2.1.20 [Staff], page 145, Section 2.1.22 [TabStaff], page 155 and Section 2.1.24 [VaticanaStaff], page 173.

2.2.103 Staff_performer
`Staff_performer` is not part of any context.

2.2.104 Staff_symbol_engraver
Create the constellation of five (default) staff lines.

Music types accepted:

- Section 1.2.55 [staff-span-event], page 41

This engraver creates the following layout object(s):
Section 3.1.96 [StaffSymbol], page 329.

*StaffSymbol* is part of the following context(s): Section 2.1.5 [DrumStaff], page 62, Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.18 [RhythmicStaff], page 130, Section 2.1.20 [Staff], page 145, Section 2.1.22 [TabStaff], page 155 and Section 2.1.24 [VaticanaStaff], page 173.

2.2.105 Stanza_number_align_engraver

This engraver ensures that stanza numbers are neatly aligned.

*Stanza_number_align_engraver* is part of the following context(s): Section 2.1.19 [Score], page 133.

2.2.106 Stanza_number_engraver

Engrave stanza numbers.

Properties (read)

- **stanza** (markup)
  - Stanza ‘number’ to print before the start of a verse. Use in **Lyrics** context.

This engraver creates the following layout object(s):

- Section 3.1.97 [StanzaNumber], page 330.

*Stanza_number_engraver* is part of the following context(s): Section 2.1.13 [Lyrics], page 104.

2.2.107 Stem_engraver

Create stems and single-stem tremolos. It also works together with the beam engraver for overriding beaming.

Music types accepted:

- Section 1.2.62 [tremolo-event], page 42

Properties (read)

- **tremoloFlags** (integer)
  - The number of tremolo flags to add if no number is specified.

- **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`.

This engraver creates the following layout object(s):

- Section 3.1.98 [Stem], page 330 and Section 3.1.99 [StemTremolo], page 332.

*Stem_engraver* is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161 and Section 2.1.26 [Voice], page 194.
2.2.108 String_number_engraver

Swallow string number events. The purpose of this engraver is to process tablatures for normal notation. To prevent warnings for unprocessed string number events to obscure real error messages, this engraver swallows them all.

String_number_engraver is part of the following context(s): Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.20 [Staff], page 145 and Section 2.1.24 [VaticanaStaff], page 173.

2.2.109 Swallow_engraver

This engraver swallows everything given to it silently. The purpose of this is to prevent spurious ‘event junked’ warnings.

Swallow_engraver is part of the following context(s): Section 2.1.4 [Devnull], page 62.

2.2.110 Swallow_performer

Swallow_performer is not part of any context.

2.2.111 System_start_delimiter_engraver

Create a system start delimiter (i.e., a SystemStartBar, SystemStartBrace, SystemStartBracket or SystemStartSquare spanner).

Properties (read)

* 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.

* currentCommandColumn (layout object)
  Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

This engraver creates the following layout object(s):

Section 3.1.105 [SystemStartBar], page 337, Section 3.1.106 [SystemStartBrace], page 338, Section 3.1.107 [SystemStartBracket], page 339 and Section 3.1.108 [SystemStartSquare], page 340.

System_start_delimiter_engraver is part of the following context(s): Section 2.1.1 [ChoirStaff], page 48, Section 2.1.10 [GrandStaff], page 82, Section 2.1.17 [PianoStaff], page 129, Section 2.1.19 [Score], page 133 and Section 2.1.21 [StaffGroup], page 154.

2.2.112 Tab_harmonic_engraver

In a tablature, parenthesize objects whose music cause has the parenthesize property.

Tab_harmonic_engraver is part of the following context(s): Section 2.1.23 [TabVoice], page 161.

2.2.113 Tab_note_heads_engraver

Generate one or more tablature noteheads from event of type NoteEvent.

Music types accepted:

Section 1.2.56 [string-number-event], page 41 and Section 1.2.34 [note-event], page 39

Properties (read)
middleCPosition (number)
   The place of the middle C, measured in half staff-spaces. Usually de-
termined by looking at middleCClefPosition and middleCOffset.

stringTunings (list)
   The tablature strings tuning. It is a list of the pitch (in semitones) of
each string (starting with the lower one).

minimumFret (number)
   The tablature auto string-selecting mechanism selects the highest string
with a fret at least minimumFret.

tablatureFormat (procedure)
   A function formatting a tablature note head. Called with three ar-
guments: string number, context and event. It returns the text as a
string.

highStringOne (boolean)
   Whether the first string is the string with highest pitch on the instru-
ment. This used by the automatic string selector for tablature notation.

stringOneTopmost (boolean)
   Whether the first string is printed on the top line of the tablature.

This engraver creates the following layout object(s):
Section 3.1.109 [TabNoteHead], page 340.
Tab_note_heads_engraver is part of the following context(s): Section 2.1.23 [TabVoice],
page 161.

2.2.114 Tab_staff_symbol_engraver
Create a tablature staff symbol, but look at stringTunings for the number of lines.

   Properties (read)
       stringTunings (list)
       The tablature strings tuning. It is a list of the pitch (in semitones) of
each string (starting with the lower one).

   This engraver creates the following layout object(s):
Section 3.1.96 [StaffSymbol], page 329.
Tab_staff_symbol_engraver is part of the following context(s): Section 2.1.22 [TabStaff],
page 155.

2.2.115 Tempo_performer

   Properties (read)
       tempoWholesPerMinute (moment)
       The tempo in whole notes per minute.

   Tempo_performer is not part of any context.

2.2.116 Text_engraver
Create text scripts.

   Music types accepted:
   Section 1.2.59 [text-script-event], page 42

   This engraver creates the following layout object(s):
Section 3.1.110 [TextScript], page 341.

**Text_engraver** is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

### 2.2.117 Text_spanner_engraver

Create text spanner from an event.

- **Music types accepted:**
  - Section 1.2.60 [text-span-event], page 42

- **This engraver creates the following layout object(s):**
  - Section 3.1.111 [TextSpanner], page 343.

**Text_spanner_engraver** is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

### 2.2.118 Tie_engraver

Generate ties between note heads of equal pitch.

- **Music types accepted:**
  - Section 1.2.61 [tie-event], page 42

  - **Properties (read)**
    - `tieWaitForNote` (boolean)
      - If true, tied notes do not have to follow each other directly. This can be used for writing out arpeggios.

  - **Properties (write)**
    - `tieMelismaBusy` (boolean)
      - Signal whether a tie is present.

  This engraver creates the following layout object(s):
  - Section 3.1.112 [Tie], page 344 and Section 3.1.113 [TieColumn], page 345.

**Tie_engraver** is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

### 2.2.119 Tie_performer

Generate ties between note heads of equal pitch.

- **Music types accepted:**
  - Section 1.2.61 [tie-event], page 42

  - **Properties (read)**
    - `tieMelismaBusy` (boolean)
      - Signal whether a tie is present.

**Tie_performer** is not part of any context.
2.2.120 Time_signature_engraver
Create a Section 3.1.114 [TimeSignature], page 345 whenever timeSignatureFraction changes.

Properties (read)

- implicitTimeSignatureVisibility (vector)
  break visibility for the default time signature.

- timeSignatureFraction (pair of numbers)
  A pair of numbers, signifying the time signature. For example, #'(4 . 4) is a 4/4 time signature.

This engraver creates the following layout object(s):
Section 3.1.114 [TimeSignature], page 345.

Time_signature_engraver is part of the following context(s): Section 2.1.5 [DrumStaff], page 62, Section 2.1.11 [GregorianTranscriptionStaff], page 83, Section 2.1.14 [MensuralStaff], page 106, Section 2.1.18 [RhythmicStaff], page 130, Section 2.1.20 [Staff], page 145 and Section 2.1.22 [TabStaff], page 155.

2.2.121 Time_signature_performer
Time_signature_performer is not part of any context.

2.2.122 Timing_translator
This engraver adds the alias Timing to its containing context. Responsible for synchronizing timing information from staves. Normally in Score. In order to create polyrhythmic music, this engraver should be removed from Score and placed in Staff.

Properties (read)

- internalBarNumber (integer)
  Contains the current bar number. This property is used for internal timekeeping, among others by the Accidental_engraver.

- currentBarNumber (integer)
  Contains the current bar number. This property is incremented at every bar line.

- 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.

Properties (write)

- internalBarNumber (integer)
  Contains the current bar number. This property is used for internal timekeeping, among others by the Accidental_engraver.

- currentBarNumber (integer)
  Contains the current bar number. This property is incremented at every bar line.

- measurePosition (moment)
  How much of the current measure have we had. This can be set manually to create incomplete measures.

Timing_translator is part of the following context(s): Section 2.1.19 [Score], page 133.
2.2.123 Translator
Base class. Not instantiated.

Translator is not part of any context.

2.2.124 Trill_spanner_engraver
Create trill spanner from an event.

Music types accepted:
Section 1.2.64 [trill-span-event], page 42

Properties (read)

- **currentCommandColumn**: layout object
  Grob that is X-parent to all current breakable (clef, key signature, etc.) items.
- **currentMusicalColumn**: layout object
  Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

This engraver creates the following layout object(s):
Section 3.1.118 [TrillSpanner], page 349.

Trill_spanner_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.125 Tuplet_engraver
Catch tuplet events and generate appropriate bracket.

Music types accepted:
Section 1.2.65 [tuplet-span-event], page 42

Properties (read)

- **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.

This engraver creates the following layout object(s):
Section 3.1.119 [TupletBracket], page 350 and Section 3.1.120 [TupletNumber], page 351.

Tuplet_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.

2.2.126 Tweak_engraver
Read the tweaks property from the originating event, and set properties.

Tweak_engraver is part of the following context(s): Section 2.1.3 [CueVoice], page 50, Section 2.1.6 [DrumVoice], page 67, Section 2.1.12 [GregorianTranscriptionVoice], page 92, Section 2.1.15 [MensuralVoice], page 116, Section 2.1.23 [TabVoice], page 161, Section 2.1.25 [VaticanaVoice], page 182 and Section 2.1.26 [Voice], page 194.
2.2.127 **Vaticana_ligature_engraver**

Handle ligatures by glueing special ligature heads together.

Music types accepted:
- Section 1.2.26 [ligature-event], page 38 and Section 1.2.41 [pes-or-flexa-event], page 40
- This engraver creates the following layout object(s):
  - Section 3.1.30 [DotColumn], page 279 and Section 3.1.123 [VaticanaLigature], page 354.
- **Vaticana_ligature_engraver** is part of the following context(s): Section 2.1.25 [VaticanaVoice], page 182.

2.2.128 **Vertical_align_engraver**

Catch groups (staves, lyrics lines, etc.) and stack them vertically.

Properties (read)
- **alignAboveContext** (string)
  Where to insert newly created context in vertical alignment.
- **alignBelowContext** (string)
  Where to insert newly created context in vertical alignment.

This engraver creates the following layout object(s):
- Section 3.1.124 [VerticalAlignment], page 354.
- **Vertical_align_engraver** is part of the following context(s): Section 2.1.19 [Score], page 133.

2.2.129 **Vertically_spaced_contexts_engraver**

Properties (read)
- **verticallySpacedContexts** (list)
  List of symbols, containing context names whose vertical axis groups should be taken into account for vertical spacing of systems.

Properties (write)
- **verticallySpacedContexts** (list)
  List of symbols, containing context names whose vertical axis groups should be taken into account for vertical spacing of systems.

**Vertically_spaced_contexts_engraver** is part of the following context(s): Section 2.1.19 [Score], page 133.

2.2.130 **Volta_engraver**

Make volta brackets.

Properties (read)
- **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.
- **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.
- **stavesFound** (list of grobs)
  A list of all staff-symbols found.
This engraver creates the following layout object(s):
Section 3.1.127 [VoltaBracket], page 356 and Section 3.1.128 [VoltaBracketSpanner], page 357.
Volta_engraver is part of the following context(s): Section 2.1.19 [Score], page 133.

2.3 Tunable context properties

aDueText (markup)
   Text to print at a unisono passage.

alignAboveContext (string)
   Where to insert newly created context in vertical alignment.

alignBassFigureAccidentals (boolean)
   If true, then the accidentals are aligned in bass figure context.

alignBelowContext (string)
   Where to insert newly created context in vertical 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 Section “Score” in Internals Reference then all staves share accidentals, and if context is Section “Staff” in Internals Reference 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 Section “Setting automatic beam behavior” in Notation Reference 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 num-
ber should be printed.

bassFigureFormatFunction (procedure)
A procedure that is called to produce the formatting for a BassFigure grob. It
takes a list of BassFigureEvents, 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 Section “Timing_translator” in Internals Reference at
   Section “Score” in Internals Reference 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.

figuringOrientations (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 GridPoints.

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 Section “Lyrics” in Internals Reference 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 "quotes.

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.

soloIIIText (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 Section “Pitch_squash_ engraver” in Internals Reference.

staffLineLayoutFunction (procedure)
   Layout of staff lines, traditional, or semitone.

stanzan (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 Section “bar-line-interface” in Internals Reference.

2.4 Internal context properties

associatedVoiceContext (context)
   The context object of the Voice that has the melody for this Lyrics.

barCheckLastFail (moment)
   Where in the measure did the last barcheck fail?

beamMelismaBusy (boolean)
   Signal if a beam is present.

breakableSeparationItem (layout object)
   The breakable items in this time step, for this staff.

busyGrobs (list)
   A queue of (end-moment . GROB) cons cells. This is for internal (C++) use only. This property contains the grobs which are still busy (e.g. note heads, spanners, etc.).

currentCommandColumn (layout object)
   Grob that is X-parent to all current breakable (clef, key signature, etc.) items.

currentMusicalColumn (layout object)
   Grob that is X-parent to all non-breakable items (note heads, lyrics, etc.).

dynamicAbsoluteVolumeFunction (procedure)
   [DOCUMENT-ME]
finalizations (list)
   A list of expressions to evaluate before proceeding to next time step. This is an internal variable.

graceSettings (list)
   Overrides for grace notes. This property should be manipulated through the add-grace-property function.

hasStaffSpacing (boolean)
   True if the current CommandColumn contains items that will affect spacing.

instrumentSupport (list of grobs)
   A list of grobs to attach the instrument name to.

lastKeySignature (list)
   Last key signature before a key signature change.

localKeySignature (list)
   The key signature at this point in the measure. The format is the same as for keySignature, but can also contain ((octave . name) . (alter barnumber . measureposition)) pairs.

melismaBusy (boolean)
   Signifies whether a melisma is active. This can be used to signal melismas on top of those automatically detected.

originalMiddleCPosition (integer)
   Used for temporary overriding middle C in octavation brackets.

quotedEventTypes (list)
   A list of symbols, representing the event types that should be duplicated for \quote commands.

rootSystem (layout object)
   The System object.

scriptDefinitions (list)
   The description of scripts. This is used by the Script_engraver for typesetting note-superscripts and subscripts. See 'scm/script.scm' for more information.

slurMelismaBusy (boolean)
   Signal if a slur is present.

stavesFound (list of grobs)
   A list of all staff-symbols found.

tieMelismaBusy (boolean)
   Signal whether a tie is present.
3 Backend

3.1 All layout objects

3.1.1 Accidental
Accidental objects are created by: Section 2.2.1 [Accidental_engraver], page 206.

Standard settings:

- **avoid-slur** (symbol):
  
  'inside
  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.

- **glyph-name-alist** (list):
  
  '((0 . accidentals.natural) (-1/2 . accidentals.flat) (1/2 . accidentals.sharp) (1 . accidentals.doublesharp) (-1 . accidentals.flatflat) (3/4 . accidentals.sharp.slashslash.stemstemstem) (1/4 . accidentals.sharp.slashslash.stem) (-1/4 . accidentals.mirroredflat) (-3/4 . accidentals.mirroredflat.flat))
  An alist of key-string pairs.

- **alteration** (number):
  accidental-interface::calc-alteration
  Alteration numbers for accidental.

- **stencil** (unknown):
  ly:accidental-interface::print
  The symbol to print.

- **Y-extent** (pair of numbers):
  ly:accidental-interface::height
  Hard coded extent in Y direction.

- **X-extent** (pair of numbers):
  ly:accidental-interface::width
  Hard coded extent in X direction.

This object supports the following interface(s): Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.1 [accidental-interface], page 358 and Section 3.2.37 [grob-interface], page 376.

3.1.2 AccidentalCautionary
AccidentalCautionary objects are created by: Section 2.2.1 [Accidental_engraver], page 206.

Standard settings:

- **avoid-slur** (symbol):
  
  'inside
  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.
parenthesized (boolean):
    #t
    Parenthesize this grob.

glyph-name-alist (list):
    '((0 . accidentals.natural) (-1/2 . accidentals.flat) (1/2
    . accidentals.sharp) (1 . accidentals.doublesharp) (-1.
    accidentals.flatflat) (3/4 . accidentals.sharp.slashslash.stemstemstem)
    (1/4 . accidentals.sharp.slashslash.stem)
    (-1/4 . accidentals.mirroredflat) (-3/4 .
    accidentals.mirroredflat.flat))
    An alist of key-string pairs.

alteration (number):
    accidental-interface::calc-alteration
    Alteration numbers for accidental.

Stencil (unknown):
    ly:accidental-interface::print
    The symbol to print.

Y-extent (pair of numbers):
    ly:accidental-interface::height
    Hard coded extent in Y direction.

This object supports the following interface(s): Section 3.2.42 [item-interface], page 382,
Section 3.2.31 [font-interface], page 371, Section 3.2.1 [accidental-interface], page 358 and
Section 3.2.37 [grob-interface], page 376.

3.1.3 AccidentalPlacement

AccidentalPlacement objects are created by: Section 2.2.1 [Accidental engraver], page 206 and
Section 2.2.2 [Ambitus engraver], page 207.

Standard settings:

left-padding (dimension, in staff space):
    0.2
    The amount of space that is put left to an object (e.g., a group of
    accidentals).

script-priority (number):
    -100
    A sorting key that determines in what order a script is within a stack
    of scripts.

direction (direction):
    -1
    If side-axis is 0 (or #X), then this property determines whether the ob-
    ject 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.

X-extent (pair of numbers):
    ly:axis-group-interface::width
    Hard coded extent in X direction.
right-padding (dimension, in staff space): 0.15
   Space to insert on the right side of an object (e.g., between note and its accidentals).

This object supports the following interface(s): Section 3.2.42 [item-interface], page 382, Section 3.2.2 [accidental-placement-interface], page 359 and Section 3.2.37 [grob-interface], page 376.

3.1.4 AccidentalSuggestion
AccidentalSuggestion objects are created by: Section 2.2.1 [Accidental engraver], page 206.

Standard settings:

   stencil (unknown):  
      ly:accidental-interface::print
      The symbol to print.

   X-extent (pair of numbers):  
      ly:accidental-interface::width
      Hard coded extent in X direction.

   Y-extent (pair of numbers):  
      ly:accidental-interface::height
      Hard coded extent in Y direction.

   X-offset (number):  
      #<simple-closure (#<primitive-generic +> #<simple-closure (#<primitive-procedure ly:self-alignment-interface::centered-on-x-parent>) #<simple-closure (#<primitive-procedure ly:self-alignment-interface::x-aligned-on-self>) >) >
      The horizontal amount that this object is moved relative to its X-parent.

   self-alignment-X (number):  
      0
      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.

   font-size (number):  
      -2
      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.

   glyph-name-alist (list):  
      '((0 . accidentals.natural) (-1/2 . accidentals.flat) (1/2 . accidentals.sharp) (1 . accidentals.doublesharp) (-1 . accidentals.flatflat) (3/4 . accidentals.sharp.slashslash.stemstemstem) (1/4 . accidentals.sharp.slashslash.stem) (-1/4 . accidentals.mirroredflat) (-3/4 . accidentals.mirroredflat.flat))
      An alist of key-string pairs.
alteration (number):
accidental-interface::calc-alteration
Alteration numbers for accidental.

Y-offset (number):
ly:side-position-interface::y-aligned-side
The vertical amount that this object is moved relative to its Y-parent.

direction (direction):
1
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.

staff-padding (dimension, in staff space):
0.25
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.

outside-staff-priority (number):
0
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.

script-priority (number):
0
A sorting key that determines in what order a script is within a stack of scripts.

side-axis (number):
1
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.

This object supports the following interface(s): Section 3.2.87 [side-position-interface], page 400, Section 3.2.83 [self-alignment-interface], page 398, Section 3.2.82 [script-interface], page 398, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.3 [accidental-suggestion-interface], page 360, Section 3.2.1 [accidental-interface], page 358 and Section 3.2.37 [grob-interface], page 376.

3.1.5 Ambitus

Ambitus objects are created by: Section 2.2.2 [Ambitus engraver], page 207.

Standard settings:

axes (list):
'(0 1)
List of axis numbers. In the case of alignment grobs, this should contain only one number.

X-extent (pair of numbers):
ly:axis-group-interface::width
Hard coded extent in X direction.
Y-extent (pair of numbers):
   ly:axis-group-interface::height
   Hard coded extent in Y direction.

space-alist (list):
   '((clef extra-space . 0.5) (key-signature extra-space . 0.0)
     (staff-bar extra-space . 0.0) (time-signature extra-space . 0.0)
     (first-note fixed-space . 0.0))
   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.

non-musical (boolean):
   #'t
   True if the grob belongs to a NonMusicalPaperColumn.

break-align-symbol (symbol):
   'ambitus
   This key is used for aligning and spacing breakable items.

break-visibility (vector):
   #(#f #f #t)
   A vector of 3 booleans, #(end-of-line unbroken begin-of-line).
   #t means visible, #f means killed.

This object supports the following interface(s): Section 3.2.42 [item-interface], page 382,
Section 3.2.15 [break-aligned-interface], page 366, Section 3.2.7 [axis-group-interface], page 361,
Section 3.2.5 [ambitus-interface], page 360 and Section 3.2.37 [grob-interface], page 376.

3.1.6 AmbitusAccidental
AmbitusAccidental objects are created by: Section 2.2.2 [Ambitus engraver], page 207.
Standard settings:
   font-family (symbol):
      'music
      The font family is the broadest category for selecting text fonts. Options
      include: sans, roman.

   padding (dimension, in staff space):
      0.5
      Add this much extra space between objects that are next to each other.

   X-offset (number):
      ly:side-position-interface::x-aligned-side
      The horizontal amount that this object is moved relative to its X-parent.

   direction (direction):
      -1
      If side-axis is 0 (or #X), then this property determines whether the ob-
      ject 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.
### 3.1.7 AmbitusLine

AmbitusLine objects are created by: Section 2.2.2 [Ambitus_engraver], page 207.

Standard settings:

- **stencil (unknown):**
  - ly:ambitus::print
    - The symbol to print.

- **thickness (number):**
  - 2
    - Line thickness, generally measured in line-thickness.

- **X-offset (number):**
  - ly:self-alignment-interface::centered-on-x-parent
    - The horizontal amount that this object is moved relative to its X-parent.

This object supports the following interface(s): Section 3.2.87 [side-position-interface], page 400, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.15 [break-aligned-interface], page 366, Section 3.2.1 [accidental-interface], page 358 and Section 3.2.37 [grob-interface], page 376.

### 3.1.8 AmbitusNoteHead

AmbitusNoteHead objects are created by: Section 2.2.2 [Ambitus_engraver], page 207.

Standard settings:

- **duration-log (integer):**
  - 2
    - The 2-log of the note head duration, i.e., 0 = whole note, 1 = half note, etc.
stencil (unknown):
  ly:note-head::print
  The symbol to print.

Y-offset (number):
  ly:staff-symbol-referencer::callback
  The vertical amount that this object is moved relative to its Y-parent.

This object supports the following interface(s): Section 3.2.97 [staff-symbol-referencer-interface], page 407, Section 3.2.80 [rhythmic-head-interface], page 397, Section 3.2.64 [note-head-interface], page 391, Section 3.2.46 [ledgered-interface], page 385, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.5 [ambitus-interface], page 360 and Section 3.2.37 [grob-interface], page 376.

3.1.9 Arpeggio

Arpeggio objects are created by: Section 2.2.3 [Arpeggio engraver], page 207 and Section 2.2.99 [Span_arpeggio_engraver], page 238.

Standard settings:

X-extent (pair of numbers):
  ly:arpeggio::width
  Hard coded extent in X direction.

stencil (unknown):
  ly:arpeggio::print
  The symbol to print.

Y-offset (number):
  ly:staff-symbol-referencer::callback
  The vertical amount that this object is moved relative to its Y-parent.

X-offset (number):
  ly:side-position-interface::x-aligned-side
  The horizontal amount that this object is moved relative to its X-parent.

direction (direction):
  -1
  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.

positions (pair of numbers):
  ly:arpeggio::calc-positions
  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.

padding (dimension, in staff space):
  0.5
  Add this much extra space between objects that are next to each other.
**script-priority** (number):

0

A sorting key that determines in what order a script is within a stack of scripts.

**side-axis** (number):

0

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.

**staff-position** (number):

0.0

Vertical position, measured in half staff spaces, counted from the middle line.

**Y-extent** (pair of numbers):

ly:arpeggio::height

Hard coded extent in Y direction.

This object supports the following interface(s): Section 3.2.97 [staff-symbol-referencer-interface], page 407, Section 3.2.87 [side-position-interface], page 400, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.6 [arpeggio-interface], page 361 and Section 3.2.37 [grob-interface], page 376.

### 3.1.10 BalloonTextItem

BalloonTextItem objects are created by: Section 2.2.6 [Balloon engraver], page 208.

Standard settings:

**stencil** (unknown):

ly:balloon-interface::print

The symbol to print.

**text** (markup):

#<procedure #f (grob)>

Text markup. See Section “Formatting text” in Notation Reference.

**X-offset** (number):

#<procedure #f (grob)>

The horizontal amount that this object is moved relative to its X-parent.

**Y-offset** (number):

#<procedure #f (grob)>

The vertical amount that this object is moved relative to its Y-parent.

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.8 [balloon-interface], page 362 and Section 3.2.37 [grob-interface], page 376.

### 3.1.11 BarLine

BarLine objects are created by: Section 2.2.7 [Bar engraver], page 208.

Standard settings:

**break-align-symbol** (symbol):

'staff-bar'

This key is used for aligning and spacing breakable items.
break-align-anchor (number):
   ly:bar-line::calc-anchor
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.

glyph (string):
   "|"
A string determining what ‘style’ of glyph is typeset. Valid choices
depend on the function that is reading this property.

gap (dimension, in staff space):
   0.4
Size of a gap in a variable symbol.

layer (integer):
   0
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.

break-visibility (vector):
   bar-line::calc-break-visibility
A vector of 3 booleans, #(end-of-line unbroken begin-of-line).
#t means visible, #f means killed.

non-musical (boolean):
   #t
True if the grob belongs to a NonMusicalPaperColumn.

stencil (unknown):
   ly:bar-line::print
The symbol to print.

bar-size (dimension, in staff space):
   ly:bar-line::calc-bar-size
The size of a bar line.

allow-span-bar (boolean):
   #t
If false, no inter-staff bar line will be created below this bar line.

space-alist (list):
   '((time-signature extra-space . 0.75) (custos minimum-space
   . 2.0) (clef minimum-space . 1.0) (key-signature extra-space
   . 1.0) (key-cancellation extra-space . 1.0) (first-note
   fixed-space . 1.3) (next-note semi-fixed-space . 0.9)
   (right-edge extra-space . 0.0))
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.

kern (dimension, in staff space):
   3.0
Amount of extra white space to add. For bar lines, this is the amount of space after a thick line.

**thin-kern** (number):

3.0

The space after a hair-line in a bar line.

**hair-thickness** (number):

1.9

Thickness of the thin line in a bar line.

**thick-thickness** (number):

6.0

Bar line thickness, measured in **line-thickness**.

This object supports the following interface(s): Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.15 [break-aligned-interface], page 366, Section 3.2.9 [bar-line-interface], page 362 and Section 3.2.37 [grob-interface], page 376.

### 3.1.12 BarNumber

BarNumber objects are created by: Section 2.2.8 [Bar number engraver], page 209.

Standard settings:

**stencil** (unknown):

ly:text-interface::print

The symbol to print.

**non-musical** (boolean):

#t

True if the grob belongs to a NonMusicalPaperColumn.

**break-visibility** (vector):

#(#f #f #t)

A vector of 3 booleans, #(@end-of-line unbroken begin-of-line). #t means visible, #f means killed.

**padding** (dimension, in staff space):

1.0

Add this much extra space between objects that are next to each other.

**direction** (direction):

1

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.

**font-family** (symbol):

'roman

The font family is the broadest category for selecting text fonts. Options include: sans, roman.

**font-size** (number):

-2
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.

**Y-offset** (number):
```
ly:side-position-interface::y-aligned-side
```
The vertical amount that this object is moved relative to its Y-parent.

**side-axis** (number):
```
1
```
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.

**outside-staff-priority** (number):
```
100
```
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.

**X-offset** (number):
```
```
The horizontal amount that this object is moved relative to its X-parent.

**self-alignment-X** (number):
```
1
```
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.

**break-align-symbols** (list):
```
'(left-edge staff-bar)
```
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).

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.87 [side-position-interface], page 400, Section 3.2.83 [self-alignment-interface], page 398, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.14 [break-alignable-interface], page 366 and Section 3.2.37 [grob-interface], page 376.

### 3.1.13 BassFigure

BassFigure objects are created by: Section 2.2.32 [Figured_bass_engraver], page 216.

Standard settings:

**stencil** (unknown):
```
ly:text-interface::print
```
The symbol to print.

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.79 [rhythmic-grob-interface], page 397, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.11 [bass-figure-interface], page 363 and Section 3.2.37 [grob-interface], page 376.
3.1.14 BassFigureAlignment

BassFigureAlignment objects are created by: Section 2.2.32 [Figured_bass_engraver], page 216.

Standard settings:

axes (list):

'(1)
List of axis numbers. In the case of alignment grobs, this should contain only one number.

threshold (pair of numbers):

'(2 . 1000)
(min . max), where min and max are dimensions in staff space.

Y-extent (pair of numbers):

ly:axis-group-interface::height
Hard coded extent in Y direction.

stacking-dir (direction):

-1
Stack objects in which direction?

padding (dimension, in staff space):

0.2
Add this much extra space between objects that are next to each other.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.10 [bass-figure-alignment-interface], page 363, Section 3.2.7 [axis-group-interface], page 361, Section 3.2.4 [align-interface], page 360 and Section 3.2.37 [grob-interface], page 376.

3.1.15 BassFigureAlignmentPositioning

BassFigureAlignmentPositioning objects are created by: Section 2.2.33 [Figured_bass_position_engraver], page 217.

Standard settings:

Y-offset (number):

ly:side-position-interface::y-aligned-side
The vertical amount that this object is moved relative to its Y-parent.

side-axis (number):

1
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.

direction (direction):

1
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.

Y-extent (pair of numbers):

ly:axis-group-interface::height
Hard coded extent in Y direction.
axes (list):

'(1)
List of axis numbers. In the case of alignment grobs, this should contain only one number.

staff-padding (dimension, in staff space):

1.0
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.

padding (dimension, in staff space):

0.5
Add this much extra space between objects that are next to each other.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.87 [side-position-interface], page 400, Section 3.2.7 [axis-group-interface], page 361 and Section 3.2.37 [grob-interface], page 376.

3.1.16 BassFigureBracket

BassFigureBracket objects are created by: Section 2.2.32 [Figured bass engraver], page 216.

Standard settings:

stencil (unknown):

ly:enclosing-bracket::print
The symbol to print.

X-extent (pair of numbers):

ly:enclosing-bracket::width
Hard coded extent in X direction.

dge-height (pair):

'(0.2 . 0.2)
A pair of numbers specifying the heights of the vertical edges: (left-height . right-height).

This object supports the following interface(s): Section 3.2.42 [item-interface], page 382, Section 3.2.28 [enclosing-bracket-interface], page 370 and Section 3.2.37 [grob-interface], page 376.

3.1.17 BassFigureContinuation

BassFigureContinuation objects are created by: Section 2.2.32 [Figured bass engraver], page 216.

Standard settings:

stencil (unknown):

ly:figured-bass-continuation::print
The symbol to print.

Y-offset (number):

ly:figured-bass-continuation::center-on-figures
The vertical amount that this object is moved relative to its Y-parent.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.29 [figured-bass-continuation-interface], page 371 and Section 3.2.37 [grob-interface], page 376.
3.1.18 BassFigureLine

BassFigureLine objects are created by: Section 2.2.32 [Figured_bass_engraver], page 216.

Standard settings:

axes (list):

' (1)

List of axis numbers. In the case of alignment grobs, this should contain only one number.

Y-extent (pair of numbers):

ly:axis-group-interface::height

Hard coded extent in Y direction.

vertical-skylines (unknown):

ly:axis-group-interface::calc-skylines

Two skylines, one above and one below this grob.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.7 [axis-group-interface], page 361 and Section 3.2.37 [grob-interface], page 376.

3.1.19 Beam

Beam objects are created by: Section 2.2.4 [Auto_beam_engraver], page 207, Section 2.2.9 [Beam_engraver], page 209, Section 2.2.15 [Chord_tremolo_engraver], page 211 and Section 2.2.39 [Grace_beam_engraver], page 219.

Standard settings:

gap (dimension, in staff space):

0.8

Size of a gap in a variable symbol.

positions (pair of numbers):

#<simple-closure #<simple-closure (#<procedure chain-grob-member-functions (grob value . funcs)> (#<primitive-procedure cons> 0 0) #<primitive-procedure ly:beam::calc-least-squares-positions> #<primitive-procedure ly:beam::slope-damping> #<primitive-procedure ly:beam::shift-region-to-valid> #<primitive-procedure ly:beam::quanting>) > >

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.

concaveness (number):

ly:beam::calc-concaveness

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.

direction (direction):

ly:beam::calc-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
Numerical values may also be used: \#UP=1, \#DOWN=-1, \#LEFT=-1, \#RIGHT=1, \#CENTER=0.

**beaming** (pair):

```
ly:beam::calc-beaming
```

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.

**stencil** (unknown):

```
ly:beam::print
```

The symbol to print.

**clip-edges** (boolean):

```
#t
```

Allow outward pointing beamlets at the edges of beams?

**thickness** (number):

```
0.48
```

Line thickness, generally measured in line-thickness.

**neutral-direction** (direction):

```
-1
```

Which direction to take in the center of the staff.

**beamed-stem-shorten** (list):

```
'(1.0 0.5 0.25)
```

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.

**damping** (number):

```
1
```

Amount of beam slope damping.

**auto-knee-gap** (dimension, in staff space):

```
5.5
```

If a gap is found between note heads where a horizontal beam fits that is larger than this number, make a kneed beam.

**font-family** (symbol):

```
'roman
```

The font family is the broadest category for selecting text fonts. Options include: sans, roman.

This object supports the following interface(s): 
Section 3.2.116 [unbreakable-spanner-interface], page 417, 
Section 3.2.97 [staff-symbol-referencer-interface], page 407, 
Section 3.2.94 [spanner-interface], page 405, 
Section 3.2.31 [font-interface], page 371, 
Section 3.2.12 [beam-interface], page 363 and 
Section 3.2.37 [grob-interface], page 376.

### 3.1.20 BendAfter

BendAfter objects are created by: Section 2.2.11 [Bend_engraver], page 210.

Standard settings:
stencil (unknown):
  bend::print
  The symbol to print.

minimum-length (dimension, in staff space):
  0.5
  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 be-
tween noteheads.

thickness (number):
  2.0
  Line thickness, generally measured in line-thickness.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405,
Section 3.2.13 [bend-after-interface], page 365 and Section 3.2.37 [grob-interface], page 376.

3.1.21 BreakAlignGroup

BreakAlignGroup objects are created by: Section 2.2.12 [Break_align_engraver], page 210.

Standard settings:

  axes (list):
    '(0)
    List of axis numbers. In the case of alignment grobs, this should contain
only one number.

  X-extent (pair of numbers):
    ly:axis-group-interface::width
    Hard coded extent in X direction.

  break-align-anchor (number):
    ly:break-aligned-interface::calc-average-anchor
    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-visibility (vector):
    ly:break-aligned-interface::calc-break-visibility
    A vector of 3 booleans, #(end-of-line unbroken begin-of-line).
    #t means visible, #f means killed.

This object supports the following interface(s): Section 3.2.42 [item-interface], page 382,
Section 3.2.15 [break-aligned-interface], page 366, Section 3.2.7 [axis-group-interface], page 361
and Section 3.2.37 [grob-interface], page 376.

3.1.22 BreakAlignment

BreakAlignment objects are created by: Section 2.2.12 [Break_align_engraver], page 210.

Standard settings:

  non-musical (boolean):
    #t
    True if the grob belongs to a NonMusicalPaperColumn.
stacking-dir (direction):
  1
  Stack objects in which direction?

X-extent (pair of numbers):
  ly:axis-group-interface::width
  Hard coded extent in X direction.

break-align-orders (vector):
  #((left-edge ambitus breathing-sign clef staff-bar key-
     cancellation key-signature time-signature custos) (left-edge
     ambitus breathing-sign clef staff-bar key-cancellation
     key-signature staff time-signature custos) (left-edge
     ambitus breathing-sign clef key-cancellation key-signature
     staff-bar time-signature custos))
  Defines the order in which prefatory matter (clefs, key signatures) ap-
  pears. 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))

axes (list):
  '(0)
  List of axis numbers. In the case of alignment grobs, this should contain
  only one number.

This object supports the following interface(s): Section 3.2.42 [item-interface], page 382,
Section 3.2.16 [break-alignment-interface], page 367, Section 3.2.7 [axis-group-interface],
page 361 and Section 3.2.37 [grob-interface], page 376.

3.1.23 BreathingSign
BreathingSign objects are created by: Section 2.2.13 [Breathing_sign_engraver], page 210.
Standard settings:

break-align-symbol (symbol):
  'breathing-sign
  This key is used for aligning and spacing breakable items.

non-musical (boolean):
  #t
  True if the grob belongs to a NonMusicalPaperColumn.

space-alist (list):
  '((ambitus extra-space . 2.0) (custos minimum-space .
     1.0) (key-signature minimum-space . 1.5) (time-signature
     minimum-space . 1.5) (staff-bar minimum-space . 1.5)
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.

### stencil (unknown):
- ly:text-interface::print
  The symbol to print.

### text (markup):
- '(#<procedure musicglyph-markup (layout props glyph-name)>
scripts.rcomma)
  Text markup. See Section “Formatting text” in Notation Reference.

### Y-offset (number):
- ly:breathing-sign::offset-callback
  The vertical amount that this object is moved relative to its Y-parent.

### break-visibility (vector):
- #(#t #t #f)
  A vector of 3 booleans, #(end-of-line unbroken begin-of-line).
  #t means visible, #f means killed.

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.17 [breathing-sign-interface], page 367, Section 3.2.15 [break-aligned-interface], page 366 and Section 3.2.37 [grob-interface], page 376.

### 3.1.24 ChordName

ChordName objects are created by: Section 2.2.14 [Chord_name_engraver], page 211.

#### Standard settings:

### stencil (unknown):
- ly:text-interface::print
  The symbol to print.

### after-line-breaking (boolean):
- ly:chord-name::after-line-breaking
  Dummy property, used to trigger callback for after-line-breaking.

### word-space (dimension, in staff space):
- 0.0
  Space to insert between words in texts.

### font-family (symbol):
- 'sans
  The font family is the broadest category for selecting text fonts. Options include: sans, roman.

### font-size (number):
- 1.5
  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.
This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.79 [rhythmic-grob-interface], page 397, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.18 [chord-name-interface], page 367 and Section 3.2.37 [grob-interface], page 376.

3.1.25 Clef

Clef objects are created by: Section 2.2.16 [Clef_engraver], page 212.

Standard settings:

- stencil (unknown):
  - ly:clef::print
    The symbol to print.

- non-musical (boolean):
  - #t
    True if the grob belongs to a NonMusicalPaperColumn.

- avoid-slur (symbol):
  - 'inside
    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.

- font-family (symbol):
  - 'music
    The font family is the broadest category for selecting text fonts. Options include: sans, roman.

- break-align-symbol (symbol):
  - 'clef
    This key is used for aligning and spacing breakable items.

- break-align-anchor (number):
  - ly:break-aligned-interface::calc-extent-aligned-anchor
    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-visibility (vector):
  - #(#{f #t #t})
    A vector of 3 booleans, #(end-of-line unbroken begin-of-line). #t means visible, #f means killed.

- space-alist (list):
  - '((ambitus extra-space . 2.0) (staff-bar extra-space . 0.7) (key-cancellation minimum-space . 3.5) (key-signature minimum-space . 3.5) (time-signature minimum-space . 4.2) (first-note minimum-fixed-space . 5.0) (next-note extra-space . 0.5) (right-edge extra-space . 0.5))
    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.
Y-offset (number):

\texttt{ly:staff-symbol-referencer::callback}

The vertical amount that this object is moved relative to its Y-parent.

This object supports the following interface(s): Section 3.2.97 [staff-symbol-referencer-interface], page 407, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.19 [clef-interface], page 368, Section 3.2.15 [break-aligned-interface], page 366 and Section 3.2.37 [grob-interface], page 376.

3.1.26 ClusterSpanner

ClusterSpanner objects are created by: Section 2.2.17 [Cluster_spanner_engraver], page 212.

Standard settings:

\texttt{springs-and-rods} (boolean):

\texttt{ly:spanner::set-spacing-rods}

Dummy variable for triggering spacing routines.

\texttt{stencil} (unknown):

\texttt{ly:cluster::print}

The symbol to print.

\texttt{minimum-length} (dimension, in staff space):

0.0

Try to make a spanner at least this long, normally in the horizontal direction. This requires an appropriate callback for the \texttt{springs-and-rods} property. If added to a \texttt{Tie}, this sets the minimum distance between noteheads.

\texttt{padding} (dimension, in staff space):

0.25

Add this much extra space between objects that are next to each other.

\texttt{style} (symbol):

'ramp

This setting determines in what style a grob is typeset. Valid choices depend on the \texttt{stencil} callback reading this property.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.21 [cluster-interface], page 368 and Section 3.2.37 [grob-interface], page 376.

3.1.27 ClusterSpannerBeacon

ClusterSpannerBeacon objects are created by: Section 2.2.17 [Cluster_spanner_engraver], page 212.

Standard settings:

\texttt{Y-extent} (pair of numbers):

\texttt{ly:cluster-beacon::height}

Hard coded extent in Y direction.

This object supports the following interface(s): Section 3.2.79 [rhythmic-grob-interface], page 397, Section 3.2.42 [item-interface], page 382, Section 3.2.20 [cluster-beacon-interface], page 368 and Section 3.2.37 [grob-interface], page 376.
3.1.28 CombineTextScript

CombineTextScript objects are created by: Section 2.2.76 [Part_combine engraver], page 231.

Standard settings:

- **stencil (unknown):**
  - `ly:text-interface::print`
  - The symbol to print.

- **extra-spacing-width (pair of numbers):**
  - `(+inf.0 . -inf.0)`
  - 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)`.

- **Y-offset (number):**
  - `ly:side-position-interface::y-aligned-side`
  - The vertical amount that this object is moved relative to its Y-parent.

- **X-offset (number):**
  - `ly:self-alignment-interface::x-aligned-on-self`
  - The horizontal amount that this object is moved relative to its X-parent.

- **direction (direction):**
  - 1
  - 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.

- **padding (dimension, in staff space):**
  - 0.5
  - Add this much extra space between objects that are next to each other.

- **staff-padding (dimension, in staff space):**
  - 0.5
  - 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.

- **script-priority (number):**
  - 200
  - A sorting key that determines in what order a script is within a stack of scripts.

- **baseline-skip (dimension, in staff space):**
  - 2
  - Distance between base lines of multiple lines of text.

- **side-axis (number):**
  - 1
  - 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.
avoid-slur (symbol):
   'outside
   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.

font-series (symbol):
   'bold
   Select the series of a font. Choices include medium, bold, bold-narrow, etc.

This object supports the following interface(s): Section 3.2.108 [text-script-interface], page 413, Section 3.2.107 [text-interface], page 412, Section 3.2.87 [side-position-interface], page 400, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.29 Custos

Custos objects are created by: Section 2.2.21 [Custos engraver], page 213.

Standard settings:

break-align-symbol (symbol):
   'custos
   This key is used for aligning and spacing breakable items.

non-musical (boolean):
   #t
   True if the grob belongs to a NonMusicalPaperColumn.

stencil (unknown):
   ly:custos::print
   The symbol to print.

break-visibility (vector):
   #( #t #f #f)
   A vector of 3 booleans, #(end-of-line unbroken begin-of-line).
   #t means visible, #f means killed.

style (symbol):
   'vaticana
   This setting determines in what style a grob is typeset. Valid choices depend on the stencil callback reading this property.

neutral-direction (direction):
   -1
   Which direction to take in the center of the staff.

Y-offset (number):
   ly:staff-symbol-referencer::callback
   The vertical amount that this object is moved relative to its Y-parent.

space-alist (list):
   '(((first-note minimum-fixed-space . 0.0) (right-edge extra-space . 0.1))
   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.
This object supports the following interface(s): Section 3.2.97 [staff-symbol-referencer-interface], page 407, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.22 [custos-interface], page 369, Section 3.2.15 [break-aligned-interface], page 366 and Section 3.2.37 [grob-interface], page 376.

### 3.1.30 DotColumn

DotColumn objects are created by: Section 2.2.23 [Dot_column_engraver], page 214 and Section 2.2.127 [Vaticana_ligature_engraver], page 245.

Standard settings:

- **axes** (list):
  - (0)
  - List of axis numbers. In the case of alignment grobs, this should contain only one number.

- **direction** (direction):
  - 1
  - 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.

- **X-extent** (pair of numbers):
  - ly:axis-group-interface::width
  - Hard coded extent in X direction.

This object supports the following interface(s): Section 3.2.42 [item-interface], page 382, Section 3.2.23 [dot-column-interface], page 369, Section 3.2.7 [axis-group-interface], page 361 and Section 3.2.37 [grob-interface], page 376.

### 3.1.31 Dots

Dots objects are created by: Section 2.2.19 [Completion_heads_engraver], page 213 and Section 2.2.24 [Dots_engraver], page 214.

Standard settings:

- **stencil** (unknown):
  - ly:dots::print
  - The symbol to print.

- **dot-count** (integer):
  - dots::calc-dot-count
  - The number of dots.

- **staff-position** (number):
  - dots::calc-staff-position
  - Vertical position, measured in half staff spaces, counted from the middle line.

This object supports the following interface(s): Section 3.2.97 [staff-symbol-referencer-interface], page 407, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.24 [dots-interface], page 369 and Section 3.2.37 [grob-interface], page 376.
## 3.1.32 DoublePercentRepeat

DoublePercentRepeat objects are created by: Section 2.2.77 [Percent_repeat_engraver], page 231.

Standard settings:

- **stencil** (unknown):
  
  ```lily
  ly:percent-repeat-item-interface::double-percent
  ```

  The symbol to print.

- **non-musical** (boolean):
  
  ```lily
  #t
  ```

  True if the grob belongs to a `NonMusicalPaperColumn`.

- **slope** (number):
  
  ```lily
  1.0
  ```

  The slope of this object.

- **dot-negative-kern** (number):
  
  ```lily
  0.75
  ```

  The space to remove between a dot and a slash in percent repeat glyphs. Larger values bring the two elements closer together.

- **slash-negative-kern** (number):
  
  ```lily
  1.6
  ```

  The space to remove between slashes in percent repeat glyphs. Larger values bring the two elements closer together.

- **font-encoding** (symbol):
  
  ```lily
  'fetaMusic
  ```

  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).

- **width** (dimension, in staff space):
  
  ```lily
  2.0
  ```

  The width of a grob measured in staff space.

- **thickness** (number):
  
  ```lily
  0.48
  ```

  Line thickness, generally measured in `line-thickness`.

- **break-align-symbol** (symbol):
  
  ```lily
  'staff-bar
  ```

  This key is used for aligning and spacing breakable items.

- **break-visibility** (vector):
  
  ```lily
  (#(t t f)
  ```

  A vector of 3 booleans, `#(end-of-line unbroken begin-of-line)`. 
  
  `t` means visible, `f` means killed.

This object supports the following interface(s): Section 3.2.72 [percent-repeat-item-interface], page 395, Section 3.2.71 [percent-repeat-interface], page 394, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.15 [break-aligned-interface], page 366 and Section 3.2.37 [grob-interface], page 376.
### 3.1.33 DoublePercentRepeatCounter

DoublePercentRepeatCounter objects are created by: Section 2.2.77 [Percent_repeat_engraver], page 231.

Standard settings:

- **Stencil (unknown):**
  ```
  ly:text-interface::print
  ``
  The symbol to print.

- **X-offset (number):**
  ```
  ``
  The horizontal amount that this object is moved relative to its X-parent.

- **Y-offset (number):**
  ```
  ly:side-position-interface::y-aligned-side
  ``
  The vertical amount that this object is moved relative to its Y-parent.

- **Font-encoding (symbol):**
  ```
  'fetaNumber
  ``
  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).

- **Self-alignment-X (number):**
  ```
  0
  ``
  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.

- **Font-size (number):**
  ```
  -2
  ``
  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.

- **Direction (direction):**
  ```
  1
  ``
  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.

- **Padding (dimension, in staff space):**
  ```
  0.2
  ``
  Add this much extra space between objects that are next to each other.

- **Staff-padding (dimension, in staff space):**
  ```
  0.25
  ```
Maintain this much space between reference points and the staff. Its effect is to align objects of differing sizes (like the dynamics \textit{p} and \textit{f}) on their baselines.

\texttt{side-axis} (number):

1

If the value is \texttt{#X} (or equivalently 0), the object is placed horizontally next to the other object. If the value is \texttt{#Y} or 1, it is placed vertically.

This object supports the following interface(s): Section 3.2.107 \texttt{[text-interface]}, page 412, Section 3.2.87 \texttt{[side-position-interface]}, page 400, Section 3.2.83 \texttt{[self-alignment-interface]}, page 398, Section 3.2.72 \texttt{[percent-repeat-item-interface]}, page 395, Section 3.2.71 \texttt{[percent-repeat-interface]}, page 394, Section 3.2.42 \texttt{[item-interface]}, page 382, Section 3.2.31 \texttt{[font-interface]}, page 371 and Section 3.2.37 \texttt{[grob-interface]}, page 376.

3.1.34 DynamicLineSpanner

DynamicLineSpanner objects are created by: Section 2.2.27 \texttt{[Dynamic_align_engraver]}, page 215 and Section 2.2.28 \texttt{[Dynamic_engraver]}, page 215.

Standard settings:

\texttt{axes} (list):

'(1)

List of axis numbers. In the case of alignment grobs, this should contain only one number.

\texttt{Y-offset} (number):

\texttt{ly:side-position-interface::y-aligned-side}

The vertical amount that this object is moved relative to its Y-parent.

\texttt{staff-padding} (dimension, in staff space):

0.1

Maintain this much space between reference points and the staff. Its effect is to align objects of differing sizes (like the dynamics \textit{p} and \textit{f}) on their baselines.

\texttt{padding} (dimension, in staff space):

0.6

Add this much extra space between objects that are next to each other.

\texttt{slur-padding} (number):

0.3

Extra distance between slur and script.

\texttt{minimum-space} (dimension, in staff space):

1.2

Minimum distance that the victim should move (after padding).

\texttt{direction} (direction):

-1

If \texttt{side-axis} is 0 (or \texttt{#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.
side-axis (number):
1
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.

outside-staff-priority (number):
250
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.

Y-extent (pair of numbers):
ly:axis-group-interface::height
Hard coded extent in Y direction.

X-extent (pair of numbers):
ly:axis-group-interface::width
Hard coded extent in X direction.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405,
Section 3.2.87 [side-position-interface], page 400, Section 3.2.26 [dynamic-line-spanner-interface],
page 370, Section 3.2.25 [dynamic-interface], page 370, Section 3.2.7 [axis-group-interface],
page 361 and Section 3.2.37 [grob-interface], page 376.

3.1.35 DynamicText
DynamicText objects are created by: Section 2.2.28 [Dynamic engraver], page 215 and
Section 2.2.63 [New dynamic engraver], page 227.

Standard settings:

stencil (unknown):
ly:text-interface::print
The symbol to print.

direction (direction):
ly:script-interface::calc-direction
If side-axis is 0 (or #X), then this property determines whether the ob-
ject 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.

X-offset (number):
ly:self-alignment-interface::x-aligned-on-self
The horizontal amount that this object is moved relative to its X-parent.

self-alignment-X (number):
0
Specify alignment of an object. The value -1 means left aligned, 0 cen-
tered, and 1 right-aligned in X direction. Other numerical values may
also be specified.

Y-offset (number):
ly:self-alignment-interface::y-aligned-on-self
The vertical amount that this object is moved relative to its Y-parent.
self-alignment-Y (number):
  0
Like self-alignment-X but for the Y axis.

font-series (symbol):
  'bold
Select the series of a font. Choices include medium, bold, bold-narrow, etc.

font-encoding (symbol):
  'fetaDynamic
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-shape (symbol):
  'italic
Select the shape of a font. Choices include upright, italic, caps.

extra-spacing-width (pair of numbers):
  '(+inf.0 . -inf.0)
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).

outside-staff-priority (number):
  250
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.

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.83 [self-alignment-interface], page 398, Section 3.2.82 [script-interface], page 398, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.25 [dynamic-interface], page 370 and Section 3.2.37 [grob-interface], page 376.

3.1.36 DynamicTextSpanner

DynamicTextSpanner objects are created by: Section 2.2.28 [Dynamic_engraver], page 215 and Section 2.2.63 [New_dynamic_engraver], page 227.

Standard settings:

font-shape (symbol):
  'italic
Select the shape of a font. Choices include upright, italic, caps.

style (symbol):
  'dashed-line
This setting determines in what style a grob is typeset. Valid choices depend on the stencil callback reading this property.

minimum-Y-extent (pair of numbers):
  '(-1 . 1)
Minimum size of an object in Y dimension, measured in \textit{staff-space} units.

\texttt{bound-details} (list):
  
  `((right (attach-dir . -1) (Y . 0) (padding . 0.75))
   (right-broken (attach-dir . 1) (padding . 0.0))
   (left (attach-dir . -1) (Y . 0) (stencil-offset 0 . -0.5) (padding . 0.5))
   (left-broken (attach-dir . 1)))`

An alist of properties for determining attachments of spanners to edges.

\texttt{stencil} (unknown):
  
  \texttt{ly:line-spanner::print}

The symbol to print.

\texttt{left-bound-info} (list):
  
  \texttt{ly:line-spanner::calc-left-bound-info-and-text}

An alist of properties for determining attachments of spanners to edges.

\texttt{right-bound-info} (list):
  
  \texttt{ly:line-spanner::calc-right-bound-info}

An alist of properties for determining attachments of spanners to edges.

\texttt{font-size} (number):
  
  1

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.

\texttt{dash-fraction} (number):
  
  0.2

Size of the dashes, relative to \texttt{dash-period}. Should be between 0.0 (no line) and 1.0 (continuous line).

\texttt{dash-period} (number):
  
  3.0

The length of one dash together with whitespace. If negative, no line is drawn at all.

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.94 [spanner-interface], page 405, Section 3.2.50 [line-spanner-interface], page 386, Section 3.2.49 [line-interface], page 385, Section 3.2.31 [font-interface], page 371, Section 3.2.27 [dynamic-text-spanner-interface], page 370, Section 3.2.25 [dynamic-interface], page 370 and Section 3.2.37 [grob-interface], page 376.

### 3.1.37 Fingering

Fingering objects are created by: Section 2.2.34 [Fingering_engraver], page 217 and Section 2.2.64 [New_fingering_engraver], page 227.

Standard settings:

\texttt{padding} (dimension, in staff space):
  
  0.5

Add this much extra space between objects that are next to each other.

\texttt{avoid-slur} (symbol):
  
  `around`
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.

**slur-padding** (number):

0.2

Extra distance between slur and script.

**staff-padding** (dimension, in staff space):

0.5

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.

**self-alignment-X** (number):

0

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):

0

Like **self-alignment-X** but for the Y axis.

**script-priority** (number):

100

A sorting key that determines in what order a script is within a stack of scripts.

**stencil** (unknown):

ly: text-interface::print

The symbol to print.

**direction** (direction):

ly: script-interface::calc-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.

**text** (markup):

fingering::calc-text

Text markup. See Section “Formatting text” in Notation Reference.

**font-encoding** (symbol):

'fetaNumber

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-size** (number):

-5
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.

This object supports the following interface(s): Section 3.2.108 [text-script-interface], page 413, Section 3.2.107 [text-interface], page 412, Section 3.2.87 [side-position-interface], page 400, Section 3.2.83 [self-alignment-interface], page 398, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.30 [finger-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

### 3.1.38 FretBoard

FretBoard objects are created by: Section 2.2.37 [Fretboard engraver], page 218.

Standard settings:

```
  stencil (unknown):
    fret-board::calc-stencil
    The symbol to print.

  fret-diagram-details (list):
    '((finger-code . below-string))
    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-position` – Location of dot in fret space. Default 0.6 for dots without labels, 0.95 `dot-radius` for dots with labels.
    - `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-label-font-mag` – The magnification of the font used to label the lowest fret number. Default 0.5.
    - `fret-label-vertical-offset` – The offset of the fret label from the center of the fret in direction parallel to strings. Default 0.
    - `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, landscape, and opposing-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 for normal orientation, 0.5 for landscape and opposing-landscape.
• string-thickness-factor – Factor for changing thickness of each string in the fret diagram. Thickness of string \( k \) is given by \( \text{thickness} \ast (1 + \text{string-thickness-factor}) ^ {(k-1)} \). Default 0.
• 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.

This object supports the following interface(s): Section 3.2.42 [item-interface], page 382, Section 3.2.32 [fret-diagram-interface], page 373, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.39 Glissando

Glissando objects are created by: Section 2.2.38 [Glissando engraver], page 219 and Section 2.2.65 [Note_head_line_engraver], page 228.

Standard settings:

style (symbol):
   'line
   This setting determines in what style a grob is typeset. Valid choices depend on the stencil callback reading this property.

gap (dimension, in staff space):
   0.5
   Size of a gap in a variable symbol.

zigzag-width (dimension, in staff space):
   0.75
   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.

X-extent (pair of numbers):
   #f
   Hard coded extent in X direction.

Y-extent (pair of numbers):
   #f
   Hard coded extent in Y direction.
bound-details (list):
   '((right (attach-dir . 0) (padding . 1.5)) (left (attach-dir . 0) (padding . 1.5)))
An alist of properties for determining attachments of spanners to edges.

stencil (unknown):
   ly:line-spanner::print
The symbol to print.

left-bound-info (list):
   ly:line-spanner::calc-left-bound-info
An alist of properties for determining attachments of spanners to edges.

right-bound-info (list):
   ly:line-spanner::calc-right-bound-info
An alist of properties for determining attachments of spanners to edges.

This object supports the following interface(s): Section 3.2.116 [unbreakable-spanner-interface], page 417, Section 3.2.94 [spanner-interface], page 405, Section 3.2.50 [line-spanner-interface], page 386, Section 3.2.49 [line-interface], page 385 and Section 3.2.37 [grob-interface], page 376.

3.1.40 GraceSpacing
GraceSpacing objects are created by: Section 2.2.41 [Grace_spacing_engraver], page 220.
Standard settings:

   common-shortest-duration (moment):
      grace-spacing::calc-shortest-duration
The most common shortest note length. This is used in spacing. Enlarging this sets the score tighter.

   spacing-increment (number):
      0.8
Add this much space for a doubled duration. Typically, the width of a note head. See also Section “spacing-spanner-interface” in Internals Reference.

   shortest-duration-space (dimension, in staff space):
      1.6
Start with this much space for the shortest duration. This is expressed in spacing-increment as unit. See also Section “spacing-spanner-interface” in Internals Reference.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.91 [spacing-options-interface], page 404, Section 3.2.33 [grace-spacing-interface], page 374 and Section 3.2.37 [grob-interface], page 376.

3.1.41 GridLine
GridLine objects are created by: Section 2.2.42 [Grid_line_span_engraver], page 220.
Standard settings:

   X-extent (pair of numbers):
      ly:grid-line-interface::width
Hard coded extent in X direction.
stencil (unknown):
    ly:grid-line-interface::print
    The symbol to print.

self-alignment-X (number):
    0
    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.

X-offset (number):
    #<simple-closure (#<primitive-generic +> #<simple-
closure (#<primitive-procedure ly:self-alignment-
interface::centered-on-x-parent>) #<simple-closure
(#<primitive-procedure ly:self-alignment-interface::x-
aligned-on-self>) >)>
    The horizontal amount that this object is moved relative to its X-parent.

layer (integer):
    0
    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.

This object supports the following interface(s): Section 3.2.83 [self-alignment-interface], page 398, Section 3.2.42 [item-interface], page 382, Section 3.2.35 [grid-line-interface], page 375 and Section 3.2.37 [grob-interface], page 376.

3.1.42 GridPoint

GridPoint objects are created by: Section 2.2.43 [Grid_point_engraver], page 220.

Standard settings:

    X-extent (pair of numbers):
        '(0 . 0)
        Hard coded extent in X direction.

    Y-extent (pair of numbers):
        '(0 . 0)
        Hard coded extent in Y direction.

This object supports the following interface(s): Section 3.2.42 [item-interface], page 382, Section 3.2.36 [grid-point-interface], page 376 and Section 3.2.37 [grob-interface], page 376.

3.1.43 Hairpin

Hairpin objects are created by: Section 2.2.28 [Dynamic_engraver], page 215 and Section 2.2.63 [New_dynamic_engraver], page 227.

Standard settings:

    stencil (unknown):
        ly:hairpin::print
        The symbol to print.

    springs-and-rods (boolean):
        ly:spanner::set-spacing-rods
        Dummy variable for triggering spacing routines.
after-line-breaking (boolean):
  ly:hairpin::after-line-breaking
  Dummy property, used to trigger callback for after-line-breaking.

grow-direction (direction):
  hairpin::calc-grow-direction
  Crescendo or decrescendo?

circled-tip (boolean):
  #f
  Put a circle at start/end of hairpins (al/del niente).

to-barline (boolean):
  #t
  If true, the spanner will stop at the bar line just before it would otherwise stop.

thickness (number):
  1.0
  Line thickness, generally measured in line-thickness.

height (dimension, in staff space):
  0.6666
  Height of an object in staff-space units.

minimum-length (dimension, in staff space):
  2.0
  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.

bound-padding (number):
  1.0
  The amount of padding to insert around spanner bounds.

self-alignment-Y (number):
  0
  Like self-alignment-X but for the Y axis.

Y-offset (number):
  ly:self-alignment-interface::y-aligned-on-self
  The vertical amount that this object is moved relative to its Y-parent.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.83 [self-alignment-interface], page 398, Section 3.2.49 [line-interface], page 385, Section 3.2.38 [hairpin-interface], page 379, Section 3.2.25 [dynamic-interface], page 370 and Section 3.2.37 [grob-interface], page 376.

3.1.44 HarmonicParenthesesItem
HarmonicParenthesesItem objects are created by: Section 2.2.112 [Tab_harmonic_engraver], page 240.

Standard settings:
stencil (unknown):
  parentheses-item::print
  The symbol to print.

padding (dimension, in staff space):
  0
  Add this much extra space between objects that are next to each other.

stencils (list):
  parentheses-item::calc-angled-bracket-stencils
  Multiple stencils, used as intermediate value.

This object supports the following interface(s): Section 3.2.70 [parentheses-interface], page 394, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.45 HorizontalBracket

HorizontalBracket objects are created by: Section 2.2.46 [Horizontal_bracket_engraver], page 221.

Standard settings:

  thickness (number):
    1.0
    Line thickness, generally measured in line-thickness.

stencil (unknown):
  ly:horizontal-bracket::print
  The symbol to print.

Y-offset (number):
  ly:side-position-interface::y-aligned-side
  The vertical amount that this object is moved relative to its Y-parent.

connect-to-neighbor (pair):
  ly:tuplet-bracket::calc-connect-to-neighbors
  Pair of booleans, indicating whether this grob looks as a continued break.

padding (dimension, in staff space):
  0.2
  Add this much extra space between objects that are next to each other.

staff-padding (dimension, in staff space):
  0.2
  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.

direction (direction):
  -1
  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.
side-axis (number):

1

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.

bracket-flare (pair of numbers):

'(0.5 . 0.5)

A pair of numbers specifying how much edges of brackets should slant outward. Value 0.0 means straight edges.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.87 [side-position-interface], page 400, Section 3.2.49 [line-interface], page 385, Section 3.2.40 [horizontal-bracket-interface], page 380 and Section 3.2.37 [grob-interface], page 376.

3.1.46 InstrumentName

InstrumentName objects are created by: Section 2.2.48 [Instrument_name_engraver], page 221.

Standard settings:

padding (dimension, in staff space):

0.3

Add this much extra space between objects that are next to each other.

stencil (unknown):

ly:system-start-text::print

The symbol to print.

X-offset (number):

ly:side-position-interface::x-aligned-side

The horizontal amount that this object is moved relative to its X-parent.

direction (direction):

-1

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.

self-alignment-Y (number):

0

Like self-alignment-X but for the Y axis.

self-alignment-X (number):

0

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.

This object supports the following interface(s): Section 3.2.105 [system-start-text-interface], page 412, Section 3.2.94 [spanner-interface], page 405, Section 3.2.87 [side-position-interface], page 400, Section 3.2.83 [self-alignment-interface], page 398, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.
### 3.1.47 InstrumentSwitch

InstrumentSwitch objects are created by: Section 2.2.49 [Instrument_switch_engraver], page 222.

Standard settings:

- **padding** (dimension, in staff space):
  - 0.5
  - Add this much extra space between objects that are next to each other.

- **stencil** (unknown):
  - ly: text-interface::print
  - The symbol to print.

- **Y-offset** (number):
  - ly: side-position-interface::y-aligned-side
  - The vertical amount that this object is moved relative to its Y-parent.

- **X-offset** (number):
  - ly: self-alignment-interface::x-aligned-on-self
  - The horizontal amount that this object is moved relative to its X-parent.

- **staff-padding** (dimension, in staff space):
  - 0.5
  - 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.

- **direction** (direction):
  - 1
  - 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.

- **side-axis** (number):
  - 1
  - 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.

- **self-alignment-X** (number):
  - -1
  - 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.

- **outside-staff-priority** (number):
  - 500
  - 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.

- **extra-spacing-width** (pair of numbers):
  - '(+inf.0 . -inf.0)
  - 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).

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.87 [side-position-interface], page 400, Section 3.2.83 [self-alignment-interface], page 398, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.48 KeyCancellation
KeyCancellation objects are created by: Section 2.2.50 [Key engraver], page 222.

Standard settings:

- **Stencil (unknown):**
  ly: key-signature-interface::print
  The symbol to print.

- **Glyph-name-alist (list):**
  '((0 . accidentals.natural))
  An alist of key-string pairs.

- **Space-alist (list):**
  '((time-signature extra-space . 1.25) (staff-bar extra-space . 0.6) (key-signature extra-space . 0.5) (right-edge extra-space . 0.5) (first-note fixed-space . 2.5))
  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.

- **Y-offset (number):**
  ly: staff-symbol-referencer::callback
  The vertical amount that this object is moved relative to its Y-parent.

- **Break-align-symbol (symbol):**
  'key-cancellation
  This key is used for aligning and spacing breakable items.

- **Break-visibility (vector):**
  #(t t f)
  A vector of 3 booleans, #(end-of-line unbroken begin-of-line).
  #t means visible, #f means killed.

- **Non-musical (boolean):**
  #t
  True if the grob belongs to a NonMusicalPaperColumn.

This object supports the following interface(s): Section 3.2.97 [staff-symbol-referencer-interface], page 407, Section 3.2.44 [key-signature-interface], page 384, Section 3.2.43 [key-cancellation-interface], page 384, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.15 [break-aligned-interface], page 366 and Section 3.2.37 [grob-interface], page 376.
3.1.49 KeySignature

KeySignature objects are created by: Section 2.2.50 [Key_engraver], page 222.

Standard settings:

stencil (unknown):
  ly:key-signature-interface::print
  The symbol to print.

avoid-slur (symbol):
  'inside
  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.

glyph-name-alist (list):
  '((0 . accidentals.natural) (-1/2 . accidentals.flat) (1/2 . accidentals.sharp) (3/4 . accidentals.sharp.slashslash.stemstemstem) (1/4 . accidentals.sharp.slashslash.stem) (-1/4 . accidentals.mirroredflat) (-3/4 . accidentals.mirroredflat.flat))
  An alist of key-string pairs.

space-alist (list):
  '((time-signature extra-space . 1.15) (staff-bar extra-space . 1.1) (right-edge extra-space . 0.5) (first-note fixed-space . 2.5))
  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.

Y-offset (number):
  ly:staff-symbol-referencer::callback
  The vertical amount that this object is moved relative to its Y-parent.

break-align-symbol (symbol):
  'key-signature
  This key is used for aligning and spacing breakable items.

break-align-anchor (number):
  ly:break-aligned-interface::calc-extent-aligned-anchor
  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-visibility (vector):
  #(#f #f #t)
  A vector of 3 booleans, #(end-of-line unbroken begin-of-line). #t means visible, #f means killed.

non-musical (boolean):
  #t
  True if the grob belongs to a NonMusicalPaperColumn.
This object supports the following interface(s): Section 3.2.97 [staff-symbol-reference-interface], page 407, Section 3.2.44 [key-signature-interface], page 384, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.15 [break-aligned-interface], page 366 and Section 3.2.37 [grob-interface], page 376.

3.1.50 LaissezVibrerTie

LaissezVibrerTie objects are created by: Section 2.2.52 [Laissez_vibrer_engraver], page 223.

Standard settings:

stencil (unknown):
   ly:tie::print
   The symbol to print.

control-points (list):
   ly:semi-tie::calc-control-points
   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.

direction (direction):
   ly:tie::calc-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.

head-direction (direction):
   -1
   Are the note heads left or right in a semitie?

thickness (number):
   1.0
   Line thickness, generally measured in line-thickness.

This object supports the following interface(s): Section 3.2.85 [semi-tie-interface], page 399, Section 3.2.42 [item-interface], page 382 and Section 3.2.37 [grob-interface], page 376.

3.1.51 LaissezVibrerTieColumn

LaissezVibrerTieColumn objects are created by: Section 2.2.52 [Laissez_vibrer_engraver], page 223.

Standard settings:

X-extent (pair of numbers):
   #f
   Hard coded extent in X direction.

Y-extent (pair of numbers):
   #f
   Hard coded extent in Y direction.

head-direction (direction):
   -1
   Are the note heads left or right in a semitie?

This object supports the following interface(s): Section 3.2.84 [semi-tie-column-interface], page 399, Section 3.2.42 [item-interface], page 382 and Section 3.2.37 [grob-interface], page 376.
3.1.52 LedgerLineSpanner

LedgerLineSpanner objects are created by: Section 2.2.53 [Ledger_line_engraver], page 223.

Standard settings:

- **springs-and-rods** (boolean):
  
  
  - `ly:ledger-line-spanner::set-spacing-rods`
  
  Dummy variable for triggering spacing routines.

- **stencil** (unknown):
  
  - `ly:ledger-line-spanner::print`
  
  The symbol to print.

- **X-extent** (pair of numbers):
  
  
  - `#f`
  
  Hard coded extent in X direction.

- **Y-extent** (pair of numbers):
  
  
  - `#f`
  
  Hard coded extent in Y direction.

- **minimum-length-fraction** (number):
  
  
  - `0.25`
  
  Minimum length of ledger line as fraction of note head size.

- **length-fraction** (number):
  
  
  - `0.25`
  
  Multiplier for lengths. Used for determining ledger lines and stem lengths.

- **layer** (integer):
  
  
  - `0`
  
  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.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.45 [ledger-line-spanner-interface], page 384 and Section 3.2.37 [grob-interface], page 376.

3.1.53 LeftEdge

LeftEdge objects are created by: Section 2.2.12 [Break_align_engraver], page 210.

Standard settings:

- **break-align-symbol** (symbol):
  
  
  - `left-edge`
  
  This key is used for aligning and spacing breakable items.

- **break-align-anchor** (number):
  
  
  - `ly:break-aligned-interface::calc-extent-aligned-anchor`
  
  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.

- **X-extent** (pair of numbers):
  
  
  - `'(0 . 0)`
  
  Hard coded extent in X direction.
non-musical (boolean):
    #t
    True if the grob belongs to a NonMusicalPaperColumn.

break-visibility (vector):
    #(#t #f #t)
    A vector of 3 booleans, #t means visible, #f means killed.

space-alist (list):
    '((custos extra-space . 0.0) (ambitus extra-space . 2.0)
     (time-signature extra-space . 1.0) (staff-bar extra-space . 0.0)
     (breathing-sign minimum-space . 0.0) (clef extra-space . 0.8)
     (first-note fixed-space . 2.0) (right-edge extra-space . 0.0)
     (key-signature extra-space . 0.0) (key-cancellation extra-space . 0.0))
    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-
spc or extra-space.

This object supports the following interface(s): Section 3.2.42 [item-interface], page 382,
Section 3.2.15 [break-aligned-interface], page 366 and Section 3.2.37 [grob-interface], page 376.

3.1.54 LigatureBracket

LigatureBracket objects are created by: Section 2.2.54 [Ligature_bracket_ engraver], page 224.

Standard settings:

padding (dimension, in staff space):
    2.0
    Add this much extra space between objects that are next to each other.

thickness (number):
    1.6
    Line thickness, generally measured in line-thickness.

edge-height (pair):
    '(0.7 . 0.7)
    A pair of numbers specifying the heights of the vertical edges: (left-
height . right-height).

shorten-pair (pair of numbers):
    '(-0.2 . -0.2)
    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.

direction (direction):
    1
    If side-axis is 0 (or #X), then this property determines whether the ob-
ject 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.
positions (pair of numbers):
   ly:tuplet-bracket::calc-positions
   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.

stencil (unknown):
   ly:tuplet-bracket::print
   The symbol to print.

staff-padding (dimension, in staff space):
   0.25
   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.

connect-to-neighbor (pair):
   ly:tuplet-bracket::calc-connect-to-neighbors
   Pair of booleans, indicating whether this grob looks as a continued break.

control-points (list):
   ly:tuplet-bracket::calc-control-points
   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.

This object supports the following interface(s): Section 3.2.14 [tuplet-bracket-interface], page 415, Section 3.2.94 [spanner-interface], page 405, Section 3.2.49 [line-interface], page 385 and Section 3.2.37 [grob-interface], page 376.

3.1.55 LyricExtender

LyricExtender objects are created by: Section 2.2.31 [Extender engraver], page 216.

Standard settings:

stencil (unknown):
   ly:lyric-extender::print
   The symbol to print.

thickness (number):
   0.8
   Line thickness, generally measured in line-thickness.

minimum-length (dimension, in staff space):
   1.5
   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.

Y-extent (pair of numbers):
   '(0 . 0)
   Hard coded extent in Y direction.
This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.53 [lyric-interface], page 388, Section 3.2.52 [lyric-hyphen-interface], page 386 and Section 3.2.37 [grob-interface], page 376.

3.1.56 LyricHyphen

LyricHyphen objects are created by: Section 2.2.47 [Hyphen engraved], page 221.

Standard settings:

- **thickness** (number):
  
  1.3

  Line thickness, generally measured in `line-thickness`.

- **height** (dimension, in staff space):
  
  0.42

  Height of an object in `staff-space` units.

- **dash-period** (number):
  
  10.0

  The length of one dash together with whitespace. If negative, no line is drawn at all.

- **length** (dimension, in staff space):
  
  0.66

  User override for the stem length of unbeamed stems.

- **minimum-length** (dimension, in staff space):
  
  0.3

  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-distance** (dimension, in staff space):
  
  0.1

  Minimum distance between rest and notes or beam.

- **padding** (dimension, in staff space):
  
  0.07

  Add this much extra space between objects that are next to each other.

- **springs-and-rods** (boolean):

  ly:lyric-hyphen::set-spacing-rods

  Dummy variable for triggering spacing routines.

- **stencil** (unknown):

  ly:lyric-hyphen::print

  The symbol to print.

- **Y-extent** (pair of numbers):

  ’(0 . 0)

  Hard coded extent in Y direction.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.53 [lyric-interface], page 388, Section 3.2.52 [lyric-hyphen-interface], page 387, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.
3.1.57 LyricSpace

LyricSpace objects are created by: Section 2.2.47 [Hyphen engraver], page 221.

Standard settings:

- **minimum-distance** (dimension, in staff space):
  
  0.45

  Minimum distance between rest and notes or beam.

- **springs-and-rods** (boolean):
  
  ly:lyric-hyphen::set-spacing-rods

  Dummy variable for triggering spacing routines.

- **padding** (dimension, in staff space):
  
  0.0

  Add this much extra space between objects that are next to each other.

- **Y-extent** (pair of numbers):
  
  #f

  Hard coded extent in Y direction.

- **X-extent** (pair of numbers):
  
  #f

  Hard coded extent in X direction.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.52 [lyric-hyphen-interface], page 387 and Section 3.2.37 [grob-interface], page 376.

3.1.58 LyricText

LyricText objects are created by: Section 2.2.55 [Lyric engraver], page 224.

Standard settings:

- **stencil** (unknown):
  
  lyric-text::print

  The symbol to print.

- **text** (markup):
  
  #<procedure #f (grob)>

  Text markup. See Section “Formatting text” in Notation Reference.

- **X-offset** (number):
  
  ly:self-alignment-interface::aligned-on-x-parent

  The horizontal amount that this object is moved relative to its X-parent.

- **self-alignment-X** (number):
  
  0

  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.

- **word-space** (dimension, in staff space):
  
  0.6

  Space to insert between words in texts.

- **font-series** (symbol):
  
  'bold-narrow

  Select the series of a font. Choices include medium, bold, bold-narrow, etc.
**font-size** (number):

1.0

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.

**extra-spacing-width** (pair of numbers):

'(0.0 . 0.0)

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).

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.83 [self-alignment-interface], page 398, Section 3.2.79 [rhythmic-grob-interface], page 397, Section 3.2.54 [lyric-syllable-interface], page 388, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

### 3.1.59 MeasureGrouping

MeasureGrouping objects are created by: Section 2.2.58 [Measure_grouping_engraver], page 225.

Standard settings:

**Y-offset** (number):

ly:side-position-interface::y-aligned-side

The vertical amount that this object is moved relative to its Y-parent.

**side-axis** (number):

1

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.

**stencil** (unknown):

ly:measure-grouping::print

The symbol to print.

**padding** (dimension, in staff space):

2

Add this much extra space between objects that are next to each other.

**direction** (direction):

1

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.

**thickness** (number):

1

Line thickness, generally measured in line-thickness.

**height** (dimension, in staff space):

2.0

Height of an object in staff-space units.
staff-padding (dimension, in staff space):
  3
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.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405,
Section 3.2.87 [side-position-interface], page 400, Section 3.2.56 [measure-grouping-interface],
page 388 and Section 3.2.37 [grob-interface], page 376.

3.1.60 MelodyItem
MelodyItem objects are created by: Section 2.2.59 [Melody_engraver], page 225.
Standard settings:
  neutral-direction (direction):
  -1
  Which direction to take in the center of the staff.

This object supports the following interface(s): Section 3.2.57 [melody-spanner-interface],
page 388, Section 3.2.42 [item-interface], page 382 and Section 3.2.37 [grob-interface], page 376.

3.1.61 MensuralLigature
MensuralLigature objects are created by: Section 2.2.60 [Mensural_ligature_engraver], page 225.
Standard settings:
  thickness (number):
  1.4
  Line thickness, generally measured in line-thickness.
  stencil (unknown):
  ly:mensural-ligature::print
  The symbol to print.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405,
Section 3.2.58 [mensural-ligature-interface], page 388, Section 3.2.31 [font-interface], page 371
and Section 3.2.37 [grob-interface], page 376.

3.1.62 MetronomeMark
MetronomeMark objects are created by: Section 2.2.61 [Metronome_mark_engraver], page 225.
Standard settings:
  stencil (unknown):
  ly:text-interface::print
  The symbol to print.
  Y-offset (number):
  ly:side-position-interface::y-aligned-side
  The vertical amount that this object is moved relative to its Y-parent.
  direction (direction):
  1
  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.
padding (dimension, in staff space):
  0.8
  Add this much extra space between objects that are next to each other.

side-axis (number):
  1
  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.

extra-spacing-width (pair of numbers):
  '(+inf.0 . -inf.0)
  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).

outside-staff-priority (number):
  1000
  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.

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412,
Section 3.2.87 [side-position-interface], page 400, Section 3.2.59 [metronome-mark-interface],
page 389, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371 and
Section 3.2.37 [grob-interface], page 376.

3.1.63 MultiMeasureRest

MultiMeasureRest objects are created by: Section 2.2.62 [Multi_measure_rest_engraver],
page 226.

Standard settings:

  stencil (unknown):
  ly:multi-measure-rest::print
  The symbol to print.

  springs-and-rods (boolean):
  ly:multi-measure-rest::set-spacing-rods
  Dummy variable for triggering spacing routines.

  Y-offset (number):
  ly:staff-symbol-referencer::callback
  The vertical amount that this object is moved relative to its Y-parent.

  staff-position (number):
  0
  Vertical position, measured in half staff spaces, counted from the middle
  line.

  expand-limit (integer):
  10
  Maximum number of measures expanded in church rests.

  thick-thickness (number):
  6.6
  Bar line thickness, measured in line-thickness.
**hair-thickness** (number):
2.0
Thickness of the thin line in a bar line.

**padding** (dimension, in staff space):
1
Add this much extra space between objects that are next to each other.

This object supports the following interface(s): Section 3.2.97 [staff-symbol-reference-interface], page 407, Section 3.2.94 [spanner-interface], page 405, Section 3.2.78 [rest-interface], page 397, Section 3.2.61 [multi-measure-rest-interface], page 389, Section 3.2.60 [multi-measure-interface], page 389, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

### 3.1.64 MultiMeasureRestNumber

MultiMeasureRestNumber objects are created by: Section 2.2.62 [Multi_measure_rest_engraver], page 226.

Standard settings:

**bound-padding** (number):
2.0
The amount of padding to insert around spanner bounds.

**springs-and-rods** (boolean):
ly:multi-measure-rest::set-text-rods
Dummy variable for triggering spacing routines.

**stencil** (unknown):
ly:text-interface::print
The symbol to print.

**X-offset** (number):
#<simple-closure (#<primitive-generic +> #<simple-closure (#<primitive-procedure ly:self-alignment-interface::x-aligned-on-self>>) #<simple-closure (#<primitive-procedure ly:self-alignment-interface::x-centered-on-y-parent>>)>
The horizontal amount that this object is moved relative to its X-parent.

**Y-offset** (number):
ly:side-position-interface::y-aligned-side
The vertical amount that this object is moved relative to its Y-parent.

**side-axis** (number):
1
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.

**self-alignment-X** (number):
0
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.

**direction** (direction):
1
If \texttt{side-axis} is 0 (or \texttt{#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.

\texttt{padding} (dimension, in staff space):
0.4
Add this much extra space between objects that are next to each other.

\texttt{staff-padding} (dimension, in staff space):
0.4
Maintain this much space between reference points and the staff. Its effect is to align objects of differing sizes (like the dynamics \texttt{p} and \texttt{f}) on their baselines.

\texttt{font-encoding} (symbol):
'fetaNumber
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 \texttt{fetaMusic} (Emmentaler), \texttt{fetaBraces} (Aybabtu), \texttt{fetaNumber} (Emmentaler), and \texttt{fetaDynamic} (Emmentaler).

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.94 [spanner-interface], page 405, Section 3.2.87 [side-position-interface], page 400, Section 3.2.83 [self-alignment-interface], page 398, Section 3.2.60 [multi-measure-interface], page 389, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

### 3.1.65 MultiMeasureRestText

MultiMeasureRestText objects are created by: Section 2.2.62 [Multi-measure-rest-engraver], page 226.

Standard settings:

\texttt{stencil} (unknown):
ly:text-interface::print
The symbol to print.

\texttt{X-offset} (number):

\begin{verbatim}
#<simple-closure (#<primitive-generic +> #<simple-closure (#<primitive-procedure ly:self-alignment-interface::x-centered-on-y-parent>) > #<simple-closure (#<primitive-procedure ly:self-alignment-interface::x-aligned-on-self>) >)
\end{verbatim}

The horizontal amount that this object is moved relative to its X-parent.

\texttt{Y-offset} (number):
ly:side-position-interface::y-aligned-side
The vertical amount that this object is moved relative to its Y-parent.

\texttt{self-alignment-X} (number):
0
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.
direction (direction):
    1
    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.

padding (dimension, in staff space):
    0.2
    Add this much extra space between objects that are next to each other.

staff-padding (dimension, in staff space):
    0.25
    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.

outside-staff-priority (number):
    450
    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.

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.94 [spanner-interface], page 405, Section 3.2.87 [side-position-interface], page 400, Section 3.2.83 [self-alignment-interface], page 398, Section 3.2.60 [multi-measure-interface], page 389, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.66 NonMusicalPaperColumn

NonMusicalPaperColumn objects are created by: Section 2.2.74 [Paper column engraver], page 230.

Standard settings:

allow-loose-spacing (boolean):
    #t
    If set, column can be detached from main spacing.

axes (list):
    '(0)
    List of axis numbers. In the case of alignment grobs, this should contain only one number.

before-line-breaking (boolean):
    ly:paper-column::before-line-breaking
    Dummy property, used to trigger a callback function.

X-extent (pair of numbers):
    ly:axis-group-interface::width
    Hard coded extent in X direction.

horizontal-skylines (unknown):
    ly:separation-item::calc-skylines
    Two skylines, one to the left and one to the right of this grob.
non-musical (boolean):
   #t
   True if the grob belongs to a NonMusicalPaperColumn.

line-break-permission (symbol):
   'allow
   Instructs the line breaker on whether to put a line break at this column. Can be force or allow.

page-break-permission (symbol):
   'allow
   Instructs the page breaker on whether to put a page break at this column. Can be force or allow.

full-measure-extra-space (number):
   1.0
   Extra space that is allocated at the beginning of a measure with only one note. This property is read from the NonMusicalPaperColumn that begins the measure.

This object supports the following interface(s): Section 3.2.89 [spaceable-grob-interface], page 403, Section 3.2.86 [separation-item-interface], page 400, Section 3.2.69 [paper-column-interface], page 393, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.7 [axis-group-interface], page 361 and Section 3.2.37 [grob-interface], page 376.

3.1.67 NoteCollision

NoteCollision objects are created by: Section 2.2.18 [Collision_engraver], page 212.

Standard settings:

   axes (list):
      '(0 1)
      List of axis numbers. In the case of alignment grobs, this should contain only one number.

   X-extent (pair of numbers):
      ly:axis-group-interface::width
      Hard coded extent in X direction.

   Y-extent (pair of numbers):
      ly:axis-group-interface::height
      Hard coded extent in Y direction.

   prefer-dotted-right (boolean):
      #t
      For note collisions, prefer to shift dotted up-note to the right, rather than shifting just the dot.

This object supports the following interface(s): Section 3.2.62 [note-collision-interface], page 390, Section 3.2.42 [item-interface], page 382, Section 3.2.7 [axis-group-interface], page 361 and Section 3.2.37 [grob-interface], page 376.
3.1.68 NoteColumn

NoteColumn objects are created by: Section 2.2.89 [Rhythmic_column_engraver], page 235.

Standard settings:

axes (list):
  '(0 1)
  List of axis numbers. In the case of alignment grobs, this should contain
  only one number.

X-extent (pair of numbers):
  ly:axis-group-interface::width
  Hard coded extent in X direction.

Y-extent (pair of numbers):
  ly:axis-group-interface::height
  Hard coded extent in Y direction.

horizontal-skylines (unknown):
  ly:separation-item::calc-skylines
  Two skylines, one to the left and one to the right of this grob.

This object supports the following interface(s): Section 3.2.86 [separation-item-interface],
page 400, Section 3.2.63 [note-column-interface], page 390, Section 3.2.42 [item-interface],
page 382, Section 3.2.7 [axis-group-interface], page 361 and Section 3.2.37 [grob-interface],
page 376.

3.1.69 NoteHead

NoteHead objects are created by: Section 2.2.19 [Completion_heads_engraver], page 213,
Section 2.2.26 [Drum_notes_engraver], page 215 and Section 2.2.66 [Note_heads_engraver],
page 228.

Standard settings:

stencil (unknown):
  ly:note-head::print
  The symbol to print.

duration-log (integer):
  note-head::calc-duration-log
  The 2-log of the note head duration, i.e., 0 = whole note, 1 = half note,
  etc.

stem-attachment (pair of numbers):
  ly:note-head::calc-stem-attachment
  An (x . y) pair where the stem attaches to the notehead.

Y-offset (number):
  ly:staff-symbol-referencer::callback
  The vertical amount that this object is moved relative to its Y-parent.

X-offset (number):
  ly:note-head::stem-x-shift
  The horizontal amount that this object is moved relative to its X-parent.

This object supports the following interface(s): Section 3.2.117 [vaticana-ligature-interface],
page 417, Section 3.2.97 [staff-symbol-referencer-interface], page 407, Section 3.2.80
3.1.70 NoteName

NoteName objects are created by: Section 2.2.67 [Note_name_engraver], page 228.

Standard settings:

\[\text{stencil (unknown):} \]
\[\text{ly: text-interface::print}\]

The symbol to print.

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.65 [note-name-interface], page 392, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.71 NoteSpacing

NoteSpacing objects are created by: Section 2.2.69 [Note_spacing_engraver], page 229.

Standard settings:

\[\text{stem-spacing-correction (number):} \]
\[0.5\]

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.

\[\text{same-direction-correction (number):} \]
\[0.25\]

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.

\[\text{space-to-barline (boolean):} \]
\[\text{#t}\]

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.

\[\text{knee-spacing-correction (number):} \]
\[1.0\]

Factor for the optical correction amount for kneed beams. Set between 0 for no correction and 1 for full correction.

This object supports the following interface(s): Section 3.2.90 [spacing-interface], page 404, Section 3.2.66 [note-spacing-interface], page 392, Section 3.2.42 [item-interface], page 382 and Section 3.2.37 [grob-interface], page 376.

3.1.72 OctavateEight

OctavateEight objects are created by: Section 2.2.16 [Clef_engraver], page 212.

Standard settings:
**self-alignment-X** (number):

0

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.

**break-visibility** (vector):

#(#f #f #t)

A vector of 3 booleans, #((end-of-line unbroken begin-of-line)). #t means visible, #f means killed.

**X-offset** (number):

#<simple-closure (#<primitive-generic +> #<simple-closure (#<primitive-procedure ly:self-alignment-interface::x-aligned-on-self>) #<simple-closure (#<primitive-procedure ly:self-alignment-interface::centered-on-x-parent>)>)

The horizontal amount that this object is moved relative to its X-parent.

**Y-offset** (number):

ly:side-position-interface::y-aligned-side

The vertical amount that this object is moved relative to its Y-parent.

**stencil** (unknown):

ly:text-interface::print

The symbol to print.

**font-shape** (symbol):

'italic

Select the shape of a font. Choices include upright, italic, caps.

**staff-padding** (dimension, in staff space):

0.2

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.

**font-size** (number):

-4

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.

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.87 [side-position-interface], page 400, Section 3.2.83 [self-alignment-interface], page 398, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

### 3.1.73 OttavaBracket

OttavaBracket objects are created by: Section 2.2.71 [Ottava_spanner_ engraver], page 229.

Standard settings:

**Y-offset** (number):

ly:side-position-interface::y-aligned-side

The vertical amount that this object is moved relative to its Y-parent.
stencil (unknown):
  ly:ottava-bracket::print
The symbol to print.

font-shape (symbol):
  'italic
  Select the shape of a font. Choices include upright, italic, caps.

shorten-pair (pair of numbers):
  '(0.0 . -0.6)
  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.

staff-padding (dimension, in staff space):
  1.0
  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.

padding (dimension, in staff space):
  0.5
  Add this much extra space between objects that are next to each other.

minimum-length (dimension, in staff space):
  1.0
  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.

style (symbol):
  'dashed-line
  This setting determines in what style a grob is typeset. Valid choices depend on the stencil callback reading this property.

dash-fraction (number):
  0.3
  Size of the dashes, relative to dash-period. Should be between 0.0 (no line) and 1.0 (continuous line).

direction (direction):
  1
  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.
outside-staff-priority (number):
   400
   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.

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412,
Section 3.2.94 [spanner-interface], page 405, Section 3.2.87 [side-position-interface], page 400,
Section 3.2.68 [ottava-bracket-interface], page 392, Section 3.2.49 [line-interface], page 385,
Section 3.2.40 [horizontal-bracket-interface], page 380, Section 3.2.31 [font-interface], page 371
and Section 3.2.37 [grob-interface], page 376.

3.1.74 PaperColumn

PaperColumn objects are created by: Section 2.2.74 [Paper_column_engraver], page 230.

Standard settings:

axes (list):
   '(0)
   List of axis numbers. In the case of alignment grobs, this should contain
   only one number.

allow-loose-spacing (boolean):
   #t
   If set, column can be detached from main spacing.

before-line-breaking (boolean):
   ly:paper-column::before-line-breaking
   Dummy property, used to trigger a callback function.

horizontal-skylines (unknown):
   ly:separation-item::calc-skylines
   Two skylines, one to the left and one to the right of this grob.

X-extent (pair of numbers):
   ly:axis-group-interface::width
   Hard coded extent in X direction.

This object supports the following interface(s): Section 3.2.89 [spaceable-grob-interface],
page 403, Section 3.2.86 [separation-item-interface], page 400, Section 3.2.69 [paper-column-interface],
page 393, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface],
page 371, Section 3.2.7 [axis-group-interface], page 361 and Section 3.2.37 [grob-interface],
page 376.

3.1.75 ParenthesesItem

ParenthesesItem objects are created by: Section 2.2.75 [Parenthesis_engraver], page 230.

Standard settings:

stencil (unknown):
   parentheses-item::print
   The symbol to print.

stencils (list):
   parentheses-item::calc-parenthesis-stencils
   Multiple stencils, used as intermediate value.
**font-size** (number):

-6

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.

**padding** (dimension, in staff space):

0.2

Add this much extra space between objects that are next to each other.

This object supports the following interface(s): Section 3.2.70 [parentheses-interface], page 394, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

### 3.1.76 PercentRepeat

PercentRepeat objects are created by: Section 2.2.77 [Percent_repeat_engraver], page 231.

**Standard settings:**

- **springs-and-rods** (boolean):
  - `ly:multi-measure-rest::set-spacing-rods`
  - Dummy variable for triggering spacing routines.

- **stencil** (unknown):
  - `ly:multi-measure-rest::percent`
  - The symbol to print.

- **slope** (number):
  - 1.0
  - The slope of this object.

- **thickness** (number):
  - 0.48
  - Line thickness, generally measured in `line-thickness`.

- **dot-negative-kern** (number):
  - 0.75
  - The space to remove between a dot and a slash in percent repeat glyphs. Larger values bring the two elements closer together.

- **font-encoding** (symbol):
  - `fetaMusic`
  - 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).

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.71 [percent-repeat-interface], page 394, Section 3.2.61 [multi-measure-rest-interface], page 389, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

### 3.1.77 PercentRepeatCounter

PercentRepeatCounter objects are created by: Section 2.2.77 [Percent_repeat_engraver], page 231.

**Standard settings:**
stencil (unknown):
    ly:text-interface::print
    The symbol to print.

X-offset (number):
    #<simple-closure (#<primitive-generic +> #<simple-closure
    (#<primitive-procedure ly:self-alignment-interface::x-
    centered-on-y-parent>) > #<simple-closure (#<primitive-
    procedure ly:self-alignment-interface::x-aligned-on-self>)
    >
    The horizontal amount that this object is moved relative to its X-parent.

Y-offset (number):
    ly:side-position-interface::y-aligned-side
    The vertical amount that this object is moved relative to its Y-parent.

self-alignment-X (number):
    0
    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.

direction (direction):
    1
    If side-axis is 0 (or #X), then this property determines whether the ob-
    ject 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.

padding (dimension, in staff space):
    0.2
    Add this much extra space between objects that are next to each other.

staff-padding (dimension, in staff space):
    0.25
    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.

font-size (number):
    -2
    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.

font-encoding (symbol):
    'fetaNumber
    The font encoding is the broadest category for selecting a font. Cur-
    rently, only lilypond’s system fonts (Emmentaler and Aybabtu) are
    using this property. Available values are fetaMusic (Emmentaler),
    fetaBraces (Aybabtu), fetaNumber (Emmentaler), and fetaDynamic
    (Emmentaler).
This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.94 [spanner-interface], page 405, Section 3.2.87 [side-position-interface], page 400, Section 3.2.83 [self-alignment-interface], page 398, Section 3.2.71 [percent-repeat-interface], page 394, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

### 3.1.78 PhrasingSlur

PhrasingSlur objects are created by: Section 2.2.78 [Phrasing_slur_engraver], page 232.

Standard settings:

- **control-points** (list):
  
  ```
  ly:slur::calc-control-points
  ```

  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.

- **direction** (direction):
  
  ```
  ly:slur::calc-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.

- **springs-and-rods** (boolean):
  
  ```
  ly:spanner::set-spacing-rods
  ```

  Dummy variable for triggering spacing routines.

- **Y-extent** (pair of numbers):
  
  ```
  ly:slur::height
  ```

  Hard coded extent in Y direction.

- **stencil** (unknown):
  
  ```
  ly:slur::print
  ```

  The symbol to print.

- **thickness** (number):
  
  ```
  1.1
  ```

  Line thickness, generally measured in line-thickness.

- **minimum-length** (dimension, in staff space):
  
  ```
  1.5
  ```

  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.

- **height-limit** (dimension, in staff space):
  
  ```
  2.0
  ```

  Maximum slur height: The longer the slur, the closer it is to this height.

- **ratio** (number):
  
  ```
  0.333
  ```

  Parameter for slur shape. The higher this number, the quicker the slur attains its height-limit.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.88 [slur-interface], page 402 and Section 3.2.37 [grob-interface], page 376.
3.1.79 PianoPedalBracket

PianoPedalBracket objects are created by: Section 2.2.80 [Piano_pedal_engraver], page 232.

Standard settings:

- **stencil** (unknown):
  
  ly:piano-pedal-bracket::print
  
  The symbol to print.

- **style** (symbol):
  
  'line
  
  This setting determines in what style a grob is typeset. Valid choices depend on the stencil callback reading this property.

- **bound-padding** (number):
  
  1.0
  
  The amount of padding to insert around spanner bounds.

- **direction** (direction):
  
  -1
  
  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.

- **bracket-flare** (pair of numbers):
  
  '(0.5 . 0.5)
  
  A pair of numbers specifying how much edges of brackets should slant outward. Value 0.0 means straight edges.

- **edge-height** (pair):
  
  '(1.0 . 1.0)
  
  A pair of numbers specifying the heights of the vertical edges: (left-height, right-height).

- **shorten-pair** (pair of numbers):
  
  '(0.0 . 0.0)
  
  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.

- **thickness** (number):
  
  1.0
  
  Line thickness, generally measured in line-thickness.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.74 [piano-pedal-interface], page 396, Section 3.2.73 [piano-pedal-bracket-interface], page 395, Section 3.2.49 [line-interface], page 385 and Section 3.2.37 [grob-interface], page 376.

3.1.80 RehearsalMark

RehearsalMark objects are created by: Section 2.2.57 [Mark_engraver], page 224.

Standard settings:

- **stencil** (unknown):
  
  ly:text-interface::print
  
  The symbol to print.
X-offset (number):
  #<simple-closure (#<primitive-generic +> #<simple-closure
  (#<primitive-procedure ly:break-alignable-interface::self-
  align-callback>) > #<simple-closure (#<primitive-procedure
  ly:self-alignment-interface::x-aligned-on-self>) >)) >
  The horizontal amount that this object is moved relative to its X-parent.

Y-offset (number):
  ly:side-position-interface::y-aligned-side
  The vertical amount that this object is moved relative to its Y-parent.

extra-spacing-width (pair of numbers):
  '(+inf.0 . -inf.0)
  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).

self-alignment-X (number):
  0
  Specify alignment of an object. The value -1 means left aligned, 0 cen-
  tered, and 1 right-aligned in X direction. Other numerical values may
  also be specified.

direction (direction):
  1
  If side-axis is 0 (or #X), then this property determines whether the ob-
  ject 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.

non-musical (boolean):
  #t
  True if the grob belongs to a NonMusicalPaperColumn.

font-size (number):
  2
  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.

baseline-skip (dimension, in staff space):
  2
  Distance between base lines of multiple lines of text.

break-visibility (vector):
  (#f #t #t)
  A vector of 3 booleans, (end-of-line unbroken begin-of-line).
  #t means visible, #f means killed.

break-align-symbols (list):
  '(staff-bar clef)
  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).
padding (dimension, in staff space):
0.8
Add this much extra space between objects that are next to each other.

outside-staff-priority (number):
1500
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.

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412,
Section 3.2.87 [side-position-interface], page 400, Section 3.2.83 [self-alignment-interface],
page 398, Section 3.2.55 [mark-interface], page 388, Section 3.2.42 [item-interface], page 382,
Section 3.2.31 [font-interface], page 371, Section 3.2.14 [break-alignable-interface], page 366 and
Section 3.2.37 [grob-interface], page 376.

3.1.81 RepeatSlash
RepeatSlash objects are created by: Section 2.2.95 [Slash_repeat_engraver], page 236.

Standard settings:

stencil (unknown):
ly:percent-repeat-item-interface::beat-slash
The symbol to print.

thickness (number):
0.48
Line thickness, generally measured in line-thickness.

slope (number):
1.7
The slope of this object.

This object supports the following interface(s): Section 3.2.79 [rhythmic-grob-interface],
page 397, Section 3.2.72 [percent-repeat-item-interface], page 395, Section 3.2.71 [percent-repeat-
interface], page 394, Section 3.2.42 [item-interface], page 382 and Section 3.2.37 [grob-interface],
page 376.

3.1.82 RepeatTie
RepeatTie objects are created by: Section 2.2.85 [Repeat_tie_engraver], page 234.

Standard settings:

stencil (unknown):
ly:tie::print
The symbol to print.

control-points (list):
ly:semi-tie::calc-control-points
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.

direction (direction):
ly:tie::calc-direction
If side-axis is 0 (or #X), then this property determines whether the ob-
ject 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.

**thickness** (number):

1.0

Line thickness, generally measured in line-thickness.

**head-direction** (direction):

1

Are the note heads left or right in a semitie?

This object supports the following interface(s): Section 3.2.85 [semi-tie-interface], page 399, Section 3.2.42 [item-interface], page 382 and Section 3.2.37 [grob-interface], page 376.

### 3.1.83 RepeatTieColumn

RepeatTieColumn objects are created by: Section 2.2.85 [Repeat_tie_engraver], page 234.

**Standard settings:**

**X-extent** (pair of numbers):

\#f

Hard coded extent in X direction.

**Y-extent** (pair of numbers):

\#f

Hard coded extent in Y direction.

**direction** (direction):

ly:tie::calc-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.

**head-direction** (direction):

ly:semi-tie-column::calc-head-direction

Are the note heads left or right in a semitie?

This object supports the following interface(s): Section 3.2.84 [semi-tie-column-interface], page 399, Section 3.2.42 [item-interface], page 382 and Section 3.2.37 [grob-interface], page 376.

### 3.1.84 Rest

Rest objects are created by: Section 2.2.87 [Rest_engraver], page 234.

**Standard settings:**

**stencil** (unknown):

ly:rest::print

The symbol to print.

**duration-log** (integer):

stem::calc-duration-log

The 2-log of the note head duration, i.e., 0 = whole note, 1 = half note, etc.
X-extent (pair of numbers):
  ly:rest::width
  Hard coded extent in X direction.

Y-extent (pair of numbers):
  ly:rest::height
  Hard coded extent in Y direction.

Y-offset (number):
  ly:rest::y-offset-callback
  The vertical amount that this object is moved relative to its Y-parent.

minimum-distance (dimension, in staff space):
  0.25
  Minimum distance between rest and notes or beam.

This object supports the following interface(s): Section 3.2.97 [staff-symbol-reference-interf
  ace], page 407, Section 3.2.80 [rhythmic-head-interface], page 397, Section 3.2.79 [rhythmic-
  grob-interface], page 397, Section 3.2.78 [rest-interface], page 397, Section 3.2.42 [item-interfa
  ce], page 382, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 3
  76.

3.1.85 RestCollision

RestCollision objects are created by: Section 2.2.86 [Rest_collision_engraver], page 234.

  Standard settings:

    minimum-distance (dimension, in staff space):
      0.75
      Minimum distance between rest and notes or beam.

This object supports the following interface(s): Section 3.2.77 [rest-collision-interface], page 3
  96, Section 3.2.42 [item-interface], page 382 and Section 3.2.37 [grob-interface], page 376.

3.1.86 Script

Script objects are created by: Section 2.2.26 [Drum_notes_engraver], page 215, Section 2.2.64 [New
  fingering_engraver], page 227 and Section 2.2.91 [Script_engraver], page 235.

  Standard settings:

    staff-padding (dimension, in staff space):
      0.25
      Maintain this much space between reference points and the staff. Its effect is to align objects of differ
ging sizes (like the dynamics p and f) on their baselines.

    X-offset (number):
      script-interface::calc-x-offset
      The horizontal amount that this object is moved relative to its X-parent.

    Y-offset (number):
      ly:side-position-interface::y-aligned-side
      The vertical amount that this object is moved relative to its Y-parent.

    side-axis (number):
      1
      If the value is #X (or equivalently 0), the object is placed horizontally next to the other object. If t
      he value is #Y or 1, it is placed vertically.
stencil (unknown):
    ly:script-interface::print
    The symbol to print.

direction (direction):
    ly:script-interface::calc-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.

font-encoding (symbol):
    'fetaMusic
    The font encoding is the broadest category for selecting a font. Cur-
    rently, only lilypond’s system fonts (Emmentaler and Aybabtu) are
    using this property. Available values are fetaMusic (Emmentaler),
    fetaBraces (Aybabtu), fetaNumber (Emmentaler), and fetaDynamic
    (Emmentaler).

This object supports the following interface(s): Section 3.2.87 [side-position-interface],
page 400, Section 3.2.82 [script-interface], page 398, Section 3.2.42 [item-interface], page 382,
Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.87 ScriptColumn

ScriptColumn objects are created by: Section 2.2.90 [Script_column_engraver], page 235.

Standard settings:

    before-line-breaking (boolean):
    ly:script-column::before-line-breaking
    Dummy property, used to trigger a callback function.

This object supports the following interface(s): Section 3.2.81 [script-column-interface],
page 397, Section 3.2.42 [item-interface], page 382 and Section 3.2.37 [grob-interface], page 376.

3.1.88 ScriptRow

ScriptRow objects are created by: Section 2.2.92 [Script_row_engraver], page 236.

Standard settings:

    before-line-breaking (boolean):
    ly:script-column::row-before-line-breaking
    Dummy property, used to trigger a callback function.

This object supports the following interface(s): Section 3.2.81 [script-column-interface],
page 397, Section 3.2.42 [item-interface], page 382 and Section 3.2.37 [grob-interface], page 376.

3.1.89 SeparationItem

SeparationItem objects are not created by any engraver.

Standard settings:

    avoid-slur (symbol):
    'inside
    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.
X-extent (pair of numbers):
  ly:axis-group-interface::width
  Hard coded extent in X direction.

Y-extent (pair of numbers):
  ly:axis-group-interface::height
  Hard coded extent in Y direction.

horizontal-skylines (unknown):
  ly:separation-item::calc-skylines
  Two skylines, one to the left and one to the right of this grob.

stencil (unknown):
  ly:separation-item::print
  The symbol to print.

This object supports the following interface(s): Section 3.2.86 [separation-item-interface], page 400, Section 3.2.42 [item-interface], page 382 and Section 3.2.37 [grob-interface], page 376.

3.1.90 Slur

Slur objects are created by: Section 2.2.96 [Slur engraver], page 237.

Standard settings:

  control-points (list):
    ly:slur::calc-control-points
    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.

  direction (direction):
    ly:slur::calc-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.

  springs-and-rods (boolean):
    ly:spanner::set-spacing-rods
    Dummy variable for triggering spacing routines.

Y-extent (pair of numbers):
  ly:slur::height
  Hard coded extent in Y direction.

stencil (unknown):
  ly:slur::print
  The symbol to print.

thickness (number):
  1.2
  Line thickness, generally measured in line-thickness.

line-thickness (number):
  0.8
  The thickness of the tie or slur contour.
minimum-length (dimension, in staff space):

1.5

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.

height-limit (dimension, in staff space):

2.0

Maximum slur height: The longer the slur, the closer it is to this height.

ratio (number):

0.25

Parameter for slur shape. The higher this number, the quicker the slur attains its height-limit.

avoid-slur (symbol):

'inside

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.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.88 [slur-interface], page 402 and Section 3.2.37 [grob-interface], page 376.

3.1.91 SostenutoPedal

SostenutoPedal objects are created by: Section 2.2.80 [Piano_pedal_engraver], page 232.

Standard settings:

stencil (unknown):

ly:text-interface::print

The symbol to print.

direction (direction):

1

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.

X-offset (number):

ly:self-alignment-interface::x-aligned-on-self

The horizontal amount that this object is moved relative to its X-parent.

extra-spacing-width (pair of numbers):

'(+inf.0 . -inf.0)

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).

padding (dimension, in staff space):

0.0

Add this much extra space between objects that are next to each other.
font-shape (symbol):
   'italic

Select the shape of a font. Choices include upright, italic, caps.

self-alignment-X (number):
   0

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.

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.83 [self-alignment-interface], page 398, Section 3.2.75 [piano-pedal-script-interface], page 396, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.92 SostenutoPedalLineSpanner

SostenutoPedalLineSpanner objects are created by: Section 2.2.79 [Piano_pedal_align_engraver], page 232.

Standard settings:

axes (list):
   '(1)

List of axis numbers. In the case of alignment grobs, this should contain only one number.

Y-extent (pair of numbers):
   ly:axis-group-interface::height

Hard coded extent in Y direction.

X-extent (pair of numbers):
   ly:axis-group-interface::width

Hard coded extent in X direction.

Y-offset (number):
   ly:side-position-interface::y-aligned-side

The vertical amount that this object is moved relative to its Y-parent.

outside-staff-priority (number):
   1000

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.

side-axis (number):
   1

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.

staff-padding (dimension, in staff space):
   1.0

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.
padding (dimension, in staff space):
  1.2
  Add this much extra space between objects that are next to each other.

minimum-space (dimension, in staff space):
  1.0
  Minimum distance that the victim should move (after padding).

direction (direction):
  -1
  If side-axis is 0 (or #X), then this property determines whether the ob-
  ject 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.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, 
Section 3.2.87 [side-position-interface], page 400, Section 3.2.74 [piano-pedal-interface], page 396, 
Section 3.2.7 [axis-group-interface], page 361 and Section 3.2.37 [grob-interface], page 376.

3.1.93 SpacingSpanner

SpacingSpanner objects are created by: Section 2.2.98 [Spacing_engraver], page 237.

Standard settings:

springs-and-rods (boolean):
  ly:spacing-spanner::set-springs
  Dummy variable for triggering spacing routines.

common-shortest-duration (moment):
  ly:spacing-spanner::calc-common-shortest-duration
  The most common shortest note length. This is used in spacing. En-
  larging this sets the score tighter.

average-spacing-wishes (boolean):
  #t
  If set, the spacing wishes are averaged over staves.

shortest-duration-space (dimension, in staff space):
  2.0
  Start with this much space for the shortest duration. This is expressed
  in spacing-increment as unit. See also Section "spacing-spanner-
  interface" in Internals Reference.

spacing-increment (number):
  1.2
  Add this much space for a doubled duration. Typically, the width of
  a note head. See also Section "spacing-spanner-interface" in Internals
  Reference.

base-shortest-duration (moment):
  #<Mom 3/16>
  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.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, 
Section 3.2.92 [spacing-spanner-interface], page 404, Section 3.2.91 [spacing-options-interface], 
page 404 and Section 3.2.37 [grob-interface], page 376.
3.1.94 SpanBar

SpanBar objects are created by: Section 2.2.100 [Span_bar_ engraver], page 238.

Standard settings:

`break-align-symbol` (symbol):
  `staff-bar`
  This key is used for aligning and spacing breakable items.

`Y-extent` (pair of numbers):
  `ly:axis-group-interface::height`
  Hard coded extent in Y direction.

`layer` (integer):
  0
  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.

`non-musical` (boolean):
  `#t`
  True if the grob belongs to a NonMusicalPaperColumn.

`stencil` (unknown):
  `ly:span-bar::print`
  The symbol to print.

`bar-size` (dimension, in staff space):
  `ly:span-bar::calc-bar-size`
  The size of a bar line.

`X-extent` (pair of numbers):
  `ly:span-bar::width`
  Hard coded extent in X direction.

`before-line-breaking` (boolean):
  `ly:span-bar::before-line-breaking`
  Dummy property, used to trigger a callback function.

`allow-span-bar` (boolean):
  `#t`
  If false, no inter-staff bar line will be created below this bar line.

`kern` (dimension, in staff space):
  3.0
  Amount of extra white space to add. For bar lines, this is the amount
  of space after a thick line.

`thin-kern` (number):
  3.0
  The space after a hair-line in a bar line.

`hair-thickness` (number):
  1.6
  Thickness of the thin line in a bar line.
thick-thickness (number):
   6.0
   Bar line thickness, measured in line-thickness.

This object supports the following interface(s): Section 3.2.93 [span-bar-interface], page 405, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.9 [bar-line-interface], page 362 and Section 3.2.37 [grob-interface], page 376.

3.1.95 StaffSpacing

StaffSpacing objects are created by: Section 2.2.93 [Separating_line_group_engraver], page 236.

Standard settings:

   non-musical (boolean):
      #t
      True if the grob belongs to a NonMusicalPaperColumn.

   stem-spacing-correction (number):
      0.4
      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.

This object supports the following interface(s): Section 3.2.95 [staff-spacing-interface], page 406, Section 3.2.90 [spacing-interface], page 404, Section 3.2.42 [item-interface], page 382 and Section 3.2.37 [grob-interface], page 376.

3.1.96 StaffSymbol

StaffSymbol objects are created by: Section 2.2.104 [Staff_symbol_engraver], page 238 and Section 2.2.114 [Tab_staff_symbol_engraver], page 241.

Standard settings:

   Y-extent (pair of numbers):
      ly:staff-symbol::height
      Hard coded extent in Y direction.

   stencil (unknown):
      ly:staff-symbol::print
      The symbol to print.

   line-count (integer):
      5
      The number of staff lines.

   ledger-line-thickness (pair of numbers):
      '(1.0 . 0.1)
      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.

   layer (integer):
      0
      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.
This object supports the following interface(s): Section 3.2.96 [staff-symbol-interface], page 406, Section 3.2.94 [spanner-interface], page 405 and Section 3.2.37 [grob-interface], page 376.

3.1.97 StanzaNumber

StanzaNumber objects are created by: Section 2.2.106 [Stanza_number_engraver], page 239.

Standard settings:

stencil (unknown):
   ly: text-interface::print
   The symbol to print.

font-series (symbol):
   'bold
   Select the series of a font. Choices include medium, bold, bold-narrow, etc.

padding (dimension, in staff space):
   1.0
   Add this much extra space between objects that are next to each other.

X-offset (number):
   ly: side-position-interface::x-aligned-side
   The horizontal amount that this object is moved relative to its X-parent.

side-axis (number):
   0
   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.

direction (direction):
   -1
   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.

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.98 [stanza-number-interface], page 407, Section 3.2.87 [side-position-interface], page 400, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.98 Stem

Stem objects are created by: Section 2.2.107 [Stem_engraver], page 239.

Standard settings:

direction (direction):
   ly: stem::calc-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.
duration-log (integer):
    stem::calc-duration-log
    The 2-log of the note head duration, i.e., 0 = whole note, 1 = half note, etc.

default-direction (direction):
    ly:stem::calc-default-direction
    Direction determined by note head positions.

stem-end-position (number):
    ly:stem::calc-stem-end-position
    Where does the stem end (the end is opposite to the support-head)?

neutral-direction (direction):
    -1
    Which direction to take in the center of the staff.

stencil (unknown):
    ly:stem::print
    The symbol to print.

X-extent (pair of numbers):
    ly:stem::width
    Hard coded extent in X direction.

Y-extent (pair of numbers):
    ly:stem::height
    Hard coded extent in Y direction.

length (dimension, in staff space):
    ly:stem::calc-length
    User override for the stem length of unbeamed stems.

thickness (number):
    1.3
    Line thickness, generally measured in line-thickness.

flag (unknown):
    ly:stem::calc-flag
    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.

beamlet-default-length (pair):
    '(1.1, 1.1)
    A pair of numbers. The first number specifies the default length of a beamlet that sticks out of the left hand side of this stem; the second number specifies the default length of the beamlet to the right. The actual length of a beamlet is determined by taking either the default length or the length specified by beamlet-max-length-proportion, whichever is smaller.

beamlet-max-length-proportion (pair):
    '(0.75, 0.75)
The maximum length of a beamlet, as a proportion of the distance between two adjacent stems.

**X-offset** (number):
- `ly:stem::offset-callback`
  The horizontal amount that this object is moved relative to its X-parent.

**Y-offset** (number):
- `ly:staff-symbol-referencer::callback`
  The vertical amount that this object is moved relative to its Y-parent.

This object supports the following interface(s): Section 3.2.99 [stem-interface], page 407, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

### 3.1.99 StemTremolo

StemTremolo objects are created by: Section 2.2.107 [Stem engraver], page 239.

**Standard settings:**

**Y-extent** (pair of numbers):
- `ly:stem-tremolo::height`
  Hard coded extent in Y direction.

**X-extent** (pair of numbers):
- `ly:stem-tremolo::width`
  Hard coded extent in X direction.

**stencil** (unknown):
- `ly:stem-tremolo::print`
  The symbol to print.

**slope** (number):
- `ly:stem-tremolo::calc-slope`
  The slope of this object.

**beam-width** (dimension, in staff space):
- `ly:stem-tremolo::calc-width`
  Width of the tremolo sign.

**style** (symbol):
- `ly:stem-tremolo::calc-style`
  This setting determines in what style a grob is typeset. Valid choices depend on the `stencil` callback reading this property.

**beam-thickness** (dimension, in staff space):
- 0.48
  Beam thickness, measured in staff-space units.

This object supports the following interface(s): Section 3.2.100 [stem-tremolo-interface], page 410, Section 3.2.42 [item-interface], page 382 and Section 3.2.37 [grob-interface], page 376.

### 3.1.100 StringNumber

StringNumber objects are created by: Section 2.2.64 [New_fingering_engraver], page 227.

**Standard settings:**
stencil (unknown):
  print-circled-text-callback
  The symbol to print.

text (markup):
  string-number::calc-text
  Text markup. See Section “Formatting text” in Notation Reference.

padding (dimension, in staff space):
  0.5
  Add this much extra space between objects that are next to each other.

staff-padding (dimension, in staff space):
  0.5
  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.

self-alignment-X (number):
  0
  Specify alignment of an object. The value -1 means left aligned, 0 cen-
tered, and 1 right-aligned in X direction. Other numerical values may
also be specified.

self-alignment-Y (number):
  0
  Like self-alignment-X but for the Y axis.

script-priority (number):
  100
  A sorting key that determines in what order a script is within a stack
of scripts.

avoid-slur (symbol):
  'around
  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.

font-encoding (symbol):
  'fetaNumber
  The font encoding is the broadest category for selecting a font. Cur-
cently, 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-size (number):
  -5
  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.

This object supports the following interface(s): Section 3.2.108 [text-script-interface],
page 413, Section 3.2.107 [text-interface], page 412, Section 3.2.101 [string-number-interface],
StrokeFinger objects are created by: Section 2.2.64 [New_fingering_ engraver], page 227.

Standard settings:

- **stencil (unknown):**
  
  `ly:text-interface::print`
  
  The symbol to print.

- **text (markup):**
  
  `stroke-finger::calc-text`
  
  Text markup. See Section “Formatting text” in Notation Reference.

- **digit-names (unknown):**
  
  `#(p i m a x)`
  
  Names for string finger digits.

- **padding (dimension, in staff space):**
  
  `0.5`
  
  Add this much extra space between objects that are next to each other.

- **staff-padding (dimension, in staff space):**
  
  `0.5`
  
  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.

- **self-alignment-X (number):**
  
  `0`
  
  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):**
  
  `0`
  
  Like self-alignment-X but for the Y axis.

- **script-priority (number):**
  
  `100`
  
  A sorting key that determines in what order a script is within a stack of scripts.

- **font-shape (symbol):**
  
  `'italic`
  
  Select the shape of a font. Choices include *upright*, *italic*, *caps*.

- **font-size (number):**
  
  `-4`
  
  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.
This object supports the following interface(s): Section 3.2.108 [text-script-interface], page 413, Section 3.2.107 [text-interface], page 412, Section 3.2.102 [stroke-finger-interface], page 410, Section 3.2.87 [side-position-interface], page 398, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

### 3.1.102 SustainPedal

SustainPedal objects are created by: Section 2.2.80 [Piano pedal engraver], page 232.

Standard settings:

- **extra-spacing-width** (pair of numbers):
  - `(+inf.0 . -inf.0)`
  
  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)`.

- **stencil** (unknown):
  - `ly:sustain-pedal::print`
  
  The symbol to print.

- **self-alignment-X** (number):
  - 0
  
  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.

- **direction** (direction):
  - 1
  
  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.

- **padding** (dimension, in staff space):
  - 0.0
  
  Add this much extra space between objects that are next to each other.

- **X-offset** (number):
  - `ly:self-alignment-interface::x-aligned-on-self`
  
  The horizontal amount that this object is moved relative to its X-parent.

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.83 [self-alignment-interface], page 398, Section 3.2.75 [piano-pedal-script-interface], page 396, Section 3.2.74 [piano-pedal-interface], page 396, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

### 3.1.103 SustainPedalLineSpanner

SustainPedalLineSpanner objects are created by: Section 2.2.79 [Piano pedal_align_engraver], page 232.

Standard settings:
axes (list):

(1)
List of axis numbers. In the case of alignment grobs, this should contain only one number.

Y-extent (pair of numbers):

ly:axis-group-interface::height
Hard coded extent in Y direction.

X-extent (pair of numbers):

ly:axis-group-interface::width
Hard coded extent in X direction.

Y-offset (number):

ly:side-position-interface::y-aligned-side
The vertical amount that this object is moved relative to its Y-parent.

outside-staff-priority (number):

1000
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.

side-axis (number):

1
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.

padding (dimension, in staff space):

1.2
Add this much extra space between objects that are next to each other.

staff-padding (dimension, in staff space):

1.2
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.

minimum-space (dimension, in staff space):

1.0
Minimum distance that the victim should move (after padding).

direction (direction):

-1
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.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.87 [side-position-interface], page 400, Section 3.2.74 [piano-pedal-interface], page 396, Section 3.2.7 [axis-group-interface], page 361 and Section 3.2.37 [grob-interface], page 376.
3.1.104 System

System objects are not created by any engraver.

Standard settings:

- **axes** (list):
  \[(0 1)\]
  List of axis numbers. In the case of alignment grobs, this should contain only one number.

- **X-extent** (pair of numbers):
  \[ly:axis-group-interface::width\]
  Hard coded extent in X direction.

- **Y-extent** (pair of numbers):
  \[ly:axis-group-interface::height\]
  Hard coded extent in Y direction.

- **vertical-skylines** (unknown):
  \[ly:axis-group-interface::calc-skylines\]
  Two skylines, one above and one below this grob.

- **max-stretch** (number):
  \[ly:axis-group-interface::calc-max-stretch\]
  The maximum amount that this VerticalAxisGroup can be vertically stretched (for example, in order to better fill a page).

This object supports the following interface(s): Section 3.2.103 [system-interface], page 410, Section 3.2.94 [spanner-interface], page 405, Section 3.2.7 [axis-group-interface], page 361 and Section 3.2.37 [grob-interface], page 376.

3.1.105 SystemStartBar

SystemStartBar objects are created by: Section 2.2.111 [System_start_delimiter_engraver], page 240.

Standard settings:

- **Y-extent** (pair of numbers):
  \[\#f\]
  Hard coded extent in Y direction.

- **padding** (dimension, in staff space):
  \[-0.1\]
  Add this much extra space between objects that are next to each other.

- **X-offset** (number):
  \[ly:side-position-interface::x-aligned-side\]
  The horizontal amount that this object is moved relative to its X-parent.

- **direction** (direction):
  \[-1\]
  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`. 
style (symbol):
    'bar-line
    This setting determines in what style a grob is typeset. Valid choices
depend on the stencil callback reading this property.

collapse-height (dimension, in staff space):
    5.0
    Minimum height of system start delimiter. If equal or smaller, the
bracket/brace/line is removed.

thickness (number):
    1.6
    Line thickness, generally measured in line-thickness.

stencil (unknown):
    ly:system-start-delimiter::print
    The symbol to print.

This object supports the following interface(s): Section 3.2.104 [system-start-delimiter-
interface], page 411, Section 3.2.94 [spanner-interface], page 405, Section 3.2.87 [side-position-
interface], page 400 and Section 3.2.37 [grob-interface], page 376.

3.1.106 SystemStartBrace
SystemStartBrace objects are created by: Section 2.2.111 [System_start_delimiter_engraver],
page 240.

Standard settings:
style (symbol):
    'brace
    This setting determines in what style a grob is typeset. Valid choices
depend on the stencil callback reading this property.

padding (dimension, in staff space):
    0.3
    Add this much extra space between objects that are next to each other.

stencil (unknown):
    ly:system-start-delimiter::print
    The symbol to print.

collapse-height (dimension, in staff space):
    5.0
    Minimum height of system start delimiter. If equal or smaller, the
bracket/brace/line is removed.

X-offset (number):
    ly:side-position-interface::x-aligned-side
    The horizontal amount that this object is moved relative to its X-parent.

direction (direction):
    -1
    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.
font-encoding (symbol):
`fetaBraces`

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).

Y-extent (pair of numbers):
```
#f
```

Hard coded extent in Y direction.

This object supports the following interface(s): Section 3.2.104 [system-start-delimiter-interface], page 411, Section 3.2.94 [spanner-interface], page 405, Section 3.2.87 [side-position-interface], page 400, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.107 SystemStartBracket

SystemStartBracket objects are created by: Section 2.2.111 [System_start_delimiter_engraver], page 240.

Standard settings:

Y-extent (pair of numbers):
```
#f
```

Hard coded extent in Y direction.

padding (dimension, in staff space):
```
0.8
```

Add this much extra space between objects that are next to each other.

X-offset (number):
```
ly:side-position-interface::x-aligned-side
```

The horizontal amount that this object is moved relative to its X-parent.

direction (direction):
```
-1
```

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`.

stencil (unknown):
```
ly:system-start-delimiter::print
```

The symbol to print.

style (symbol):
```
'bracket
```

This setting determines in what style a grob is typeset. Valid choices depend on the `stencil` callback reading this property.

collapse-height (dimension, in staff space):
```
5.0
```

Minimum height of system start delimiter. If equal or smaller, the bracket/brace/line is removed.
thickness (number):
0.45
Line thickness, generally measured in line-thickness.

This object supports the following interface(s): Section 3.2.104 [system-start-delimiter-interface], page 411, Section 3.2.94 [spanner-interface], page 405, Section 3.2.87 [side-position-interface], page 400, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.108 SystemStartSquare
SystemStartSquare objects are created by: Section 2.2.111 [System_start_delimiter_engraver], page 240.

Standard settings:

Y-extent (pair of numbers):
#f
Hard coded extent in Y direction.

X-offset (number):
ly:side-position-interface::x-aligned-side
The horizontal amount that this object is moved relative to its X-parent.

direction (direction):
-1
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.

stencil (unknown):
ly:system-start-delimiter::print
The symbol to print.

style (symbol):
'line-bracket
This setting determines in what style a grob is typeset. Valid choices depend on the stencil callback reading this property.

thickness (number):
1.0
Line thickness, generally measured in line-thickness.

This object supports the following interface(s): Section 3.2.104 [system-start-delimiter-interface], page 411, Section 3.2.94 [spanner-interface], page 405, Section 3.2.87 [side-position-interface], page 400, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.109 TabNoteHead
TabNoteHead objects are created by: Section 2.2.113 [Tab_note_heads_engraver], page 240.

Standard settings:

stencil (unknown):
ly:text-interface::print
The symbol to print.
duration-log (integer):
  note-head::calc-duration-log
  The 2-log of the note head duration, i.e., 0 = whole note, 1 = half note, etc.

Y-offset (number):
  ly:staff-symbol-referencer::callback
  The vertical amount that this object is moved relative to its Y-parent.

X-offset (number):
  ly:self-alignment-interface::x-aligned-on-self
  The horizontal amount that this object is moved relative to its X-parent.

direction (direction):
  0
  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.

font-size (number):
  -2
  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.

stem-attachment (pair of numbers):
  '(0.0 . 1.35)
  An (x . y) pair where the stem attaches to the notehead.

font-series (symbol):
  'bold
  Select the series of a font. Choices include medium, bold, bold-narrow, etc.

This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.97 [staff-symbol-referencer-interface], page 407, Section 3.2.80 [rhythmic-head-interface], page 397, Section 3.2.79 [rhythmic-grob-interface], page 397, Section 3.2.64 [note-head-interface], page 391, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.110 TextScript

TextScript objects are created by: Section 2.2.116 [Text engraver], page 241.

Standard settings:

extra-spacing-width (pair of numbers):
  '(+inf.0 . -inf.0)
  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).

X-offset (number):
  ly:self-alignment-interface::x-aligned-on-self
  The horizontal amount that this object is moved relative to its X-parent.
Y-offset (number):
  ly:side-position-interface::y-aligned-side
  The vertical amount that this object is moved relative to its Y-parent.

side-axis (number):
  1
  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.

direction (direction):
  -1
  If side-axis is 0 (or #X), then this property determines whether the ob-
  ject 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.

padding (dimension, in staff space):
  0.5
  Add this much extra space between objects that are next to each other.

staff-padding (dimension, in staff space):
  0.5
  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.

stencil (unknown):
  ly:text-interface::print
  The symbol to print.

direction (direction):
  -1
  If side-axis is 0 (or #X), then this property determines whether the ob-
  ject 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.

outside-staff-priority (number):
  450
  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.

avoid-slur (symbol):
  'around
  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.

slur-padding (number):
  0.5
  Extra distance between slur and script.
script-priority (number):  
200  
A sorting key that determines in what order a script is within a stack of scripts.

This object supports the following interface(s): Section 3.2.108 [text-script-interface], page 413, Section 3.2.107 [text-interface], page 412, Section 3.2.87 [side-position-interface], page 400, Section 3.2.83 [self-alignment-interface], page 398, Section 3.2.42 [item-interface], page 382, Section 3.2.41 [instrument-specific-markup-interface], page 380, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.111 TextSpanner

TextSpanner objects are created by: Section 2.2.117 [Text spanner engraver], page 242.

Standard settings:

Y-offset (number):
   ly:side-position-interface::y-aligned-side  
The vertical amount that this object is moved relative to its Y-parent.

font-shape (symbol):
   'italic  
Select the shape of a font. Choices include upright, italic, caps.

style (symbol):
   'dashed-line  
This setting determines in what style a grob is typeset. Valid choices depend on the stencil callback reading this property.

staff-padding (dimension, in staff space):
   0.8  
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.

dash-fraction (number):
   0.2  
Size of the dashes, relative to dash-period. Should be between 0.0 (no line) and 1.0 (continuous line).

dash-period (number):
   3.0  
The length of one dash together with whitespace. If negative, no line is drawn at all.

side-axis (number):
   1  
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.

direction (direction):
   1  
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.
outside-staff-priority (number): 350
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.

left-bound-info (list):
   ly:line-spanner::calc-left-bound-info
   An alist of properties for determining attachments of spanners to edges.

right-bound-info (list):
   ly:line-spanner::calc-right-bound-info
   An alist of properties for determining attachments of spanners to edges.

bound-details (list):
   '((left (Y . 0) (padding . 0.25) (attach-dir . -1)) (right (Y . 0) (padding . 0.25)))
   An alist of properties for determining attachments of spanners to edges.

stencil (unknown):
   ly:line-spanner::print
   The symbol to print.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.87 [side-position-interface], page 400, Section 3.2.50 [line-spanner-interface], page 386, Section 3.2.49 [line-interface], page 385, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.112 Tie

Tie objects are created by: Section 2.2.19 [Completion_heads_engraver], page 213 and Section 2.2.118 [Tie_engraver], page 242.

Standard settings:

control-points (list):
   ly:tie::calc-control-points
   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.

springs-and-rods (boolean):
   ly:spanner::set-spacing-rods
   Dummy variable for triggering spacing routines.

avoid-slur (symbol):
   'inside
   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.

direction (direction):
   ly:tie::calc-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.
neutral-direction (direction):
  1
  Which direction to take in the center of the staff.

stencil (unknown):
  ly:tie::print
  The symbol to print.

font-size (number):
  -6
  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.

thickness (number):
  1.2
  Line thickness, generally measured in line-thickness.

line-thickness (number):
  0.8
  The thickness of the tie or slur contour.

This object supports the following interface(s): Section 3.2.110 [tie-interface], page 414, Section 3.2.94 [spanner-interface], page 405 and Section 3.2.37 [grob-interface], page 376.

3.1.113 TieColumn

TieColumn objects are created by: Section 2.2.118 [Tie_engraver], page 242.

Standard settings:

before-line-breaking (boolean):
  ly:tie-column::before-line-breaking
  Dummy property, used to trigger a callback function.

X-extent (pair of numbers):
  #f
  Hard coded extent in X direction.

Y-extent (pair of numbers):
  #f
  Hard coded extent in Y direction.

This object supports the following interface(s): Section 3.2.109 [tie-column-interface], page 413, Section 3.2.94 [spanner-interface], page 405 and Section 3.2.37 [grob-interface], page 376.

3.1.114 TimeSignature

TimeSignature objects are created by: Section 2.2.120 [Time_signature_engraver], page 243.

Standard settings:

stencil (unknown):
  ly:time-signature::print
  The symbol to print.

break-align-symbol (symbol):
  'time-signature
  This key is used for aligning and spacing breakable items.
break-align-anchor (number):
   ly:break-aligned-interface::calc-extent-aligned-anchor
   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-visibility (vector):
   #(t t t)
   A vector of 3 booleans, #(end-of-line unbroken begin-of-line). #t means visible, #f means killed.

avoid-slur (symbol):
   'inside
   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.

extra-spacing-height (pair of numbers):
   '(-1.0 . 1.0)
   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).

space-alist (list):
   '((first-note fixed-space . 2.0) (right-edge extra-space . 0.5) (staff-bar minimum-space . 2.0))
   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.

non-musical (boolean):
   t
   True if the grob belongs to a NonMusicalPaperColumn.

style (symbol):
   'C
   This setting determines in what style a grob is typeset. Valid choices depend on the stencil callback reading this property.

This object supports the following interface(s): Section 3.2.111 [time-signature-interface], page 415, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.15 [break-aligned-interface], page 366 and Section 3.2.37 [grob-interface], page 376.

3.1.115 TrillPitchAccidental

TrillPitchAccidental objects are created by: Section 2.2.83 [Pitched_trill_engraver], page 233.

Standard settings:

X-offset (number):
   ly:side-position-interface::x-aligned-side
   The horizontal amount that this object is moved relative to its X-parent.
padding (dimension, in staff space):
0.2
Add this much extra space between objects that are next to each other.

direction (direction):
-1
If \texttt{side-axis} is 0 (or \texttt{#X}), then this property determines whether the object is placed \texttt{#LEFT}, \texttt{#CENTER} or \texttt{#RIGHT} with respect to the other object. Otherwise, it determines whether the object is placed \texttt{#UP}, \texttt{#CENTER} or \texttt{#DOWN}. Numerical values may also be used: \texttt{#UP}=1, \texttt{#DOWN}=-1, \texttt{#LEFT}=-1, \texttt{#RIGHT}=1, \texttt{#CENTER}=0.

font-size (number):
-4
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.

side-axis (number):
0
If the value is \texttt{#X} (or equivalently 0), the object is placed horizontally next to the other object. If the value is \texttt{#Y} or 1, it is placed vertically.

stencil (unknown):
\texttt{ly:accidental-interface::print}
The symbol to print.

Y-extent (pair of numbers):
\texttt{ly:accidental-interface::height}
Hard coded extent in Y direction.

glyph-name-alist (list):
'((0 . accidentals.natural) (-1/2 . accidentals.flat) (1/2 . accidentals.sharp) (1 . accidentals.doublesharp) (-1 . accidentals.flatflat) (3/4 . accidentals.sharp.slashslash.stemstemstem) (1/4 . accidentals.sharp.slashslash.stem) (-1/4 . accidentals.mirroredflat) (-3/4 . accidentals.mirroredflat.flat))
An alist of key-string pairs.

This object supports the following interface(s): Section 3.2.112 [trill-pitch-accidental-interface], page 415, Section 3.2.87 [side-position-interface], page 400, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.1 [accidental-interface], page 358 and Section 3.2.37 [grob-interface], page 376.

3.1.116 TrillPitchGroup
TrillPitchGroup objects are created by: Section 2.2.83 [Pitched_trill_engraver], page 233.

Standard settings:

X-offset (number):
\texttt{ly:side-position-interface::x-aligned-side}
The horizontal amount that this object is moved relative to its X-parent.

axes (list):
' (0)
List of axis numbers. In the case of alignment grobs, this should contain only one number.

**font-size** (number):

-4

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.

**stencil** (unknown):

- parenthesize-elements
  The symbol to print.

- stencils (list):
  - parentheses-item::calc-parenthesis-stencils
    Multiple stencils, used as intermediate value.

**direction** (direction):

- direction: 1
  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.

**side-axis** (number):

- side-axis: 0
  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.

**padding** (dimension, in staff space):

- padding: 0.3
  Add this much extra space between objects that are next to each other.

This object supports the following interface(s): Section 3.2.87 [side-position-interface], page 400, Section 3.2.70 [parentheses-interface], page 394, Section 3.2.64 [note-head-interface], page 391, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371, Section 3.2.7 [axis-group-interface], page 361 and Section 3.2.37 [grob-interface], page 376.

### 3.1.117 TrillPitchHead

TrillPitchHead objects are created by: Section 2.2.83 [Pitched_trill_engraver], page 233.

Standard settings:

- **stencil** (unknown):
  - ly:note-head::print
    The symbol to print.

- **duration-log** (integer):

  2
  The 2-log of the note head duration, i.e., 0 = whole note, 1 = half note, etc.

- **Y-offset** (number):

  ly:staff-symbol-referencer::callback
  The vertical amount that this object is moved relative to its Y-parent.
font-size (number):
-4
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.

This object supports the following interface(s): Section 3.2.97 [staff-symbol-refrencer-interface], page 407, Section 3.2.80 [rhythmic-head-interface], page 397, Section 3.2.76 [pitched-trill-interface], page 396, Section 3.2.46 [ledgered-interface], page 385, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.118 TrillSpanner
TrillSpanner objects are created by: Section 2.2.124 [Trill_spanner_ engraver], page 244.

Standard settings:

left-bound-info (list):
ly:line-spanner::calc-left-bound-info
An alist of properties for determining attachments of spanners to edges.

right-bound-info (list):
ly:line-spanner::calc-right-bound-info
An alist of properties for determining attachments of spanners to edges.

bound-details (list):
'((left (text #<procedure translate-scaled-markup (layout props offset arg)> (0.0 . -1.0) (#<procedure musicglyph-markup (layout props glyph-name)> scripts.trill)) (Y . 0) (stencil-offset -0.5 . 0) (padding . 1.5) (attach-dir . 0) (anchor-alignment . 0.15)) (left-broken (end-on-note . #t)) (right (Y . 0)))
An alist of properties for determining attachments of spanners to edges.

stencil (unknown):
ly:line-spanner::print
The symbol to print.

style (symbol):
'trill
This setting determines in what style a grob is typeset. Valid choices depend on the stencil callback reading this property.

staff-padding (dimension, in staff space):
1.0
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.

padding (dimension, in staff space):
0.5
Add this much extra space between objects that are next to each other.

direction (direction):
1
If $\text{side-axis}$ is 0 (or $\#X$), then this property determines whether the object is placed $\#\text{LEFT}$, $\#\text{CENTER}$ or $\#\text{RIGHT}$ with respect to the other object. Otherwise, it determines whether the object is placed $\#\text{UP}$, $\#\text{CENTER}$ or $\#\text{DOWN}$. Numerical values may also be used: $\#\text{UP}=1$, $\#\text{DOWN}=-1$, $\#\text{LEFT}=-1$, $\#\text{RIGHT}=1$, $\#\text{CENTER}=0$.

**Y-offset (number):**

$\text{ly:side-position-interface::y-aligned-side}$

The vertical amount that this object is moved relative to its Y-parent.

**side-axis (number):**

1

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.

**outside-staff-priority (number):**

50

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 $\text{outside-staff-priority}$ is closer to the staff.

This object supports the following interface(s): Section 3.2.113 [trill-spanner-interface], page 415, Section 3.2.94 [spanner-interface], page 405, Section 3.2.87 [side-position-interface], page 400, Section 3.2.50 [line-spanner-interface], page 386, Section 3.2.49 [line-interface], page 385, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

### 3.1.119 TupletBracket

TupletBracket objects are created by: Section 2.2.125 [Tuplet engraver], page 244.

**Standard settings:**

**padding (dimension, in staff space):**

1.1

Add this much extra space between objects that are next to each other.

**thickness (number):**

1.6

Line thickness, generally measured in $\text{line-thickness}$.

**edge-height (pair):**

'(0.7 . 0.7)

A pair of numbers specifying the heights of the vertical edges: ($\text{left-height}$, $\text{right-height}$).

**shorten-pair (pair of numbers):**

'(-0.2 . -0.2)

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.

**staff-padding (dimension, in staff space):**

0.25

Maintain this much space between reference points and the staff. Its effect is to align objects of differing sizes (like the dynamics $\text{p}$ and $\text{f}$) on their baselines.
full-length-to-extent (boolean):
    #t

    Run to the extent of the column for a full-length tuplet bracket.

direction (direction):
    ly:tuplet-bracket::calc-direction

    If side-axis is 0 (or #X), then this property determines whether the ob-
    ject 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.

positions (pair of numbers):
    ly:tuplet-bracket::calc-positions

    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.

connect-to-neighbor (pair):
    ly:tuplet-bracket::calc-connect-to-neighbors

    Pair of booleans, indicating whether this grob looks as a continued
    break.

control-points (list):
    ly:tuplet-bracket::calc-control-points

    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.

stencil (unknown):
    ly:tuplet-bracket::print

    The symbol to print.

This object supports the following interface(s): Section 3.2.114 [tuplet-bracket-interface],
page 415, Section 3.2.94 [spanner-interface], page 405, Section 3.2.49 [line-interface], page 385
and Section 3.2.37 [grob-interface], page 376.

3.1.120 TupletNumber

TupletNumber objects are created by: Section 2.2.125 [Tuplet_engraver], page 244.

Standard settings:

    stencil (unknown):
        ly:tuplet-number::print

        The symbol to print.

text (markup):
    tuplet-number::calc-denominator-text

    Text markup. See Section “Formatting text” in Notation Reference.

font-shape (symbol):
    'italic

    Select the shape of a font. Choices include upright, italic, caps.
font-size (number):
   ~2
   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.

avoid-slur (symbol):
   'inside
   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.

This object supports the following interface(s): Section 3.2.115 [tuplet-number-interface], page 417, Section 3.2.107 [text-interface], page 412, Section 3.2.94 [spanner-interface], page 405, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.121 UnaCordaPedal

UnaCordaPedal objects are created by: Section 2.2.80 [Piano_pedal_engraver], page 232.

Standard settings:

   stencil (unknown):
      ly:text-interface::print
      The symbol to print.

   font-shape (symbol):
      'italic
      Select the shape of a font. Choices include upright, italic, caps.

   extra-spacing-width (pair of numbers):
      '(+inf.0 . -inf.0)
      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).

   self-alignment-X (number):
      0
      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.

   direction (direction):
      1
      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.

   padding (dimension, in staff space):
      0.0
      Add this much extra space between objects that are next to each other.

   X-offset (number):
      ly:self-alignment-interface::x-aligned-on-self
      The horizontal amount that this object is moved relative to its X-parent.
This object supports the following interface(s): Section 3.2.107 [text-interface], page 412, Section 3.2.83 [self-alignment-interface], page 398, Section 3.2.75 [piano-pedal-script-interface], page 396, Section 3.2.42 [item-interface], page 382, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.122 UnaCordaPedalLineSpanner

UnaCordaPedalLineSpanner objects are created by: Section 2.2.79 [Piano pedal align engraver], page 232.

Standard settings:

- **axes (list):**
  ' (1)
  List of axis numbers. In the case of alignment grobs, this should contain only one number.

- **Y-extent (pair of numbers):**
  ly:axis-group-interface::height
  Hard coded extent in Y direction.

- **X-extent (pair of numbers):**
  ly:axis-group-interface::width
  Hard coded extent in X direction.

- **Y-offset (number):**
  ly:side-position-interface::y-aligned-side
  The vertical amount that this object is moved relative to its Y-parent.

- **outside-staff-priority (number):**
  1000
  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.

- **side-axis (number):**
  1
  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.

- **padding (dimension, in staff space):**
  1.2
  Add this much extra space between objects that are next to each other.

- **staff-padding (dimension, in staff space):**
  1.2
  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.

- **minimum-space (dimension, in staff space):**
  1.0
  Minimum distance that the victim should move (after padding).

- **direction (direction):**
  -1
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`.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.87 [side-position-interface], page 400, Section 3.2.74 [piano-pedal-interface], page 396, Section 3.2.7 [axis-group-interface], page 361 and Section 3.2.37 [grob-interface], page 376.

### 3.1.123 VaticanaLigature

VaticanaLigature objects are created by: Section 2.2.127 [Vaticana_ligature_ engraver], page 245.

Standard settings:

- `thickness` (number):
  - 0.6
  - Line thickness, generally measured in `line-thickness`.

- `stencil` (unknown):
  - `ly:vaticana-ligature::print`
  - The symbol to print.

This object supports the following interface(s): Section 3.2.117 [vaticana-ligature-interface], page 417, Section 3.2.94 [spanner-interface], page 405, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

### 3.1.124 VerticalAlignment

VerticalAlignment objects are created by: Section 2.2.128 [Vertical_align_ engraver], page 245.

Standard settings:

- `axes` (list):
  - `'(1)``
  - List of axis numbers. In the case of alignment grobs, this should contain only one number.

- `after-line-breaking` (boolean):
  - `ly:align-interface::stretch-after-break`
  - Dummy property, used to trigger callback for after-line-breaking.

- `Y-extent` (pair of numbers):
  - `ly:axis-group-interface::height`
  - Hard coded extent in Y direction.

- `X-extent` (pair of numbers):
  - `ly:axis-group-interface::width`
  - Hard coded extent in X direction.

- `stacking-dir` (direction):
  - `-1`
  - Stack objects in which direction?

- `padding` (dimension, in staff space):
  - 0.5
  - Add this much extra space between objects that are next to each other.
vertical-skylines (unknown):
  ly:axis-group-interface::combine-skylines
  Two skylines, one above and one below this grob.

max-stretch (number):
  0
  The maximum amount that this VerticalAxisGroup can be vertically stretched (for example, in order to better fill a page).

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.7 [axis-group-interface], page 361, Section 3.2.4 [align-interface], page 360 and Section 3.2.37 [grob-interface], page 376.

3.1.125 VerticalAxisGroup

VerticalAxisGroup objects are created by: Section 2.2.5 [Axis group engraver], page 208 and Section 2.2.45 [Hara kiri engraver], page 221.

Standard settings:

axes (list):
  '1
  List of axis numbers. In the case of alignment grobs, this should contain only one number.

Y-offset (number):
  ly:hara-kiri-group-spanner::force-hara-kiri-callback
  The vertical amount that this object is moved relative to its Y-parent.

Y-extent (pair of numbers):
  ly:hara-kiri-group-spanner::y-extent
  Hard coded extent in Y direction.

X-extent (pair of numbers):
  ly:axis-group-interface::width
  Hard coded extent in X direction.

vertical-skylines (unknown):
  ly:hara-kiri-group-spanner::calc-skylines
  Two skylines, one above and one below this grob.

max-stretch (number):
  ly:axis-group-interface::calc-max-stretch
  The maximum amount that this VerticalAxisGroup can be vertically stretched (for example, in order to better fill a page).

stencil (unknown):
  ly:axis-group-interface::print
  The symbol to print.

This object supports the following interface(s): Section 3.2.118 [vertically-spaceable-interface], page 418, Section 3.2.94 [spanner-interface], page 405, Section 3.2.39 [hara-kiri-group-spanner-interface], page 379, Section 3.2.7 [axis-group-interface], page 361 and Section 3.2.37 [grob-interface], page 376.
3.1.126 VoiceFollower

VoiceFollower objects are created by: Section 2.2.65 [Note_head_line_engraver], page 228.

Standard settings:

- **style** (symbol):
  - `'line`
  - This setting determines in what style a grob is typeset. Valid choices depend on the stencil callback reading this property.

- **gap** (dimension, in staff space):
  - `0.5`
  - Size of a gap in a variable symbol.

- **non-musical** (boolean):
  - `#t`
  - True if the grob belongs to a NonMusicalPaperColumn.

- **X-extent** (pair of numbers):
  - `#f`
  - Hard coded extent in X direction.

- **Y-extent** (pair of numbers):
  - `#f`
  - Hard coded extent in Y direction.

- **bound-details** (list):
  - `'(right (attach-dir . 0) (padding . 1.5)) (left (attach-dir . 0) (padding . 1.5)))`
  - An alist of properties for determining attachments of spanners to edges.

- **stencil** (unknown):
  - `ly:line-spanner::print`
  - The symbol to print.

- **left-bound-info** (list):
  - `ly:line-spanner::calc-left-bound-info`
  - An alist of properties for determining attachments of spanners to edges.

- **right-bound-info** (list):
  - `ly:line-spanner::calc-right-bound-info`
  - An alist of properties for determining attachments of spanners to edges.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.50 [line-spanner-interface], page 386, Section 3.2.49 [line-interface], page 385 and Section 3.2.37 [grob-interface], page 376.

3.1.127 VoltaBracket

VoltaBracket objects are created by: Section 2.2.130 [Volta_engraver], page 245.

Standard settings:

- **stencil** (unknown):
  - `ly:volta-bracket-interface::print`
  - The symbol to print.
font-encoding (symbol):
  'fetaNumber
  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).

thickness (number):
  1.6
  Line thickness, generally measured in line-thickness.

degree-height (pair):
  '(2.0 . 2.0)
  A pair of numbers specifying the heights of the vertical edges: (left-height . right-height).

font-size (number):
  -4
  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.

word-space (dimension, in staff space):
  0.6
  Space to insert between words in texts.

direction (direction):
  1
  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.

This object supports the following interface(s): Section 3.2.119 [volta-bracket-interface], page 418, Section 3.2.107 [text-interface], page 412, Section 3.2.94 [spanner-interface], page 405, Section 3.2.87 [side-position-interface], page 400, Section 3.2.49 [line-interface], page 385, Section 3.2.40 [horizontal-bracket-interface], page 380, Section 3.2.31 [font-interface], page 371 and Section 3.2.37 [grob-interface], page 376.

3.1.128 VoltaBracketSpanner
VoltaBracketSpanner objects are created by: Section 2.2.130 [Volta_engraver], page 245.

Standard settings:

axes (list):
  '1
  List of axis numbers. In the case of alignment grobs, this should contain only one number.

side-axis (number):
  1
  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.
direction (direction):
  1
  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.

padding (dimension, in staff space):
  1
  Add this much extra space between objects that are next to each other.

Y-offset (number):
  ly:side-position-interface::y-aligned-side
  The vertical amount that this object is moved relative to its Y-parent.

outside-staff-priority (number):
  100
  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.

Y-extent (pair of numbers):
  ly:axis-group-interface::height
  Hard coded extent in Y direction.

X-extent (pair of numbers):
  ly:axis-group-interface::width
  Hard coded extent in X direction.

no-alignment (boolean):
  #t
  If set, don’t place this grob in a VerticalAlignment; rather, place it using its own Y-offset callback.

This object supports the following interface(s): Section 3.2.94 [spanner-interface], page 405, Section 3.2.87 [side-position-interface], page 400, Section 3.2.7 [axis-group-interface], page 361 and Section 3.2.37 [grob-interface], page 376.

3.2 Graphical Object Interfaces

3.2.1 accidental-interface
A single accidental.

User settable properties:

  alteration (number)
  Alteration numbers for accidental.

  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.

  glyph-name-alist (list)
  An alist of key-string pairs.
parenthesized (boolean)
Parenthesize this grob.

restore-first (boolean)
Print a natural before the accidental.

Internal properties:

forced (boolean)
Manually forced accidental.

tie (layout object)
A pointer to a Tie object.

This grob interface is used in the following graphical object(s): Section 3.1.1 [Accidental], page 257, Section 3.1.2 [AccidentalCautionary], page 257, Section 3.1.4 [AccidentalSuggestion], page 259, Section 3.1.6 [AmbitusAccidental], page 261 and Section 3.1.115 [TrillPitchAccidental], page 346.

3.2.2 accidental-placement-interface
Resolve accidental collisions.

User settable properties:

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.

left-padding (dimension, in staff space)
The amount of space that is put left to an object (e.g., a group of accidentals).

padding (dimension, in staff space)
Add this much extra space between objects that are next to each other.

right-padding (dimension, in staff space)
Space to insert on the right side of an object (e.g., between note and its accidentals).

script-priority (number)
A sorting key that determines in what order a script is within a stack of scripts.

Internal properties:

accidental-grobs (list)
An alist with (notename . groblist) entries.

positioning-done (boolean)
Used to signal that a positioning element did its job. This ensures that a positioning is only done once.

This grob interface is used in the following graphical object(s): Section 3.1.3 [AccidentalPlacement], page 258.
3.2.3 accidental-suggestion-interface

An accidental, printed as a suggestion (typically: vertically over a note).

This grob interface is used in the following graphical object(s): Section 3.1.4 [AccidentalSuggestion], page 259.

3.2.4 align-interface

Order grobs from top to bottom, left to right, right to left or bottom to top. For vertical alignments of staves, the break-system-details of the left Section “NonMusicalPaperColumn” in Internals Reference may be set to tune vertical spacing. Set alignment-extra-space to add extra space for staves. Set fixed-alignment-extra-space to force staves in PianoStaffs further apart.

User settable properties:

align-dir (direction)
Which side to align? -1: left side, 0: around center of width, 1: right side.

axes (list)
List of axis numbers. In the case of alignment grobs, this should contain only one number.

padding (dimension, in staff space)
Add this much extra space between objects that are next to each other.

stacking-dir (direction)
Stack objects in which direction?

threshold (pair of numbers)
(min . max), where min and max are dimensions in staff space.

Internal properties:

elements (unknown)
A list of grobs; the type is depending on the grob where this is set in.

positioning-done (boolean)
Used to signal that a positioning element did its job. This ensures that a positioning is only done once.

This grob interface is used in the following graphical object(s): Section 3.1.14 [BassFigureAlignment], page 268 and Section 3.1.124 [VerticalAlignment], page 354.

3.2.5 ambitus-interface

The line between note heads for a pitch range.

User settable properties:

thickness (number)
Line thickness, generally measured in line-thickness.

Internal properties:

join-heads (boolean)
Whether to join the note heads of an ambitus grob with a vertical line.

note-heads (unknown)
A list of note head grobs.

This grob interface is used in the following graphical object(s): Section 3.1.5 [Ambitus], page 260, Section 3.1.7 [AmbitusLine], page 262 and Section 3.1.8 [AmbitusNoteHead], page 262.
3.2.6 arpeggio-interface
Functions and settings for drawing an arpeggio symbol (a wavy line left to noteheads.

User settable properties:

**arpeggio-direction** (direction)
If set, put an arrow on the arpeggio squiggly line.

**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.

**script-priority** (number)
A sorting key that determines in what order a script is within a stack of scripts.

Internal properties:

**stems** (unknown)
A list of stem objects, corresponding to the notes that the arpeggio has to be before.

This grob interface is used in the following graphical object(s): Section 3.1.9 [Arpeggio], page 263.

3.2.7 axis-group-interface
An object that groups other layout objects.

User settable properties:

**axes** (list)
List of axis numbers. In the case of alignment grobs, this should contain only one number.

**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.

**max-stretch** (number)
The maximum amount that this VerticalAxisGroup can be vertically stretched (for example, in order to better fill a page).

**no-alignment** (boolean)
If set, don’t place this grob in a VerticalAlignment; rather, place it using its own Y-offset callback.

**vertical-skylines** (unknown)
Two skylines, one above and one below this grob.

Internal properties:

**X-common** (layout object)
Common reference point for axis group.

**Y-common** (layout object)
See X-common.
adjacent-pure-heights (vector)
   Used by a VerticalAxisGroup to cache the Y-extents of different column ranges.

elements (unknown)
   A list of grobs; the type is depending on the grob where this is set in.

pure-Y-common (layout object)
   A cache of the common_refpoint_of_array of the elements grob set.

pure-relevant-items (unknown)
   A subset of elements that are relevant for finding the pure-Y-extent.

pure-relevant-spanners (unknown)
   A subset of elements that are relevant for finding the pure-Y-extent.

This grob interface is used in the following graphical object(s): Section 3.1.5 [Ambitus], page 260, Section 3.1.14 [BassFigureAlignment], page 268, Section 3.1.15 [BassFigureAlignment-Positioning], page 268, Section 3.1.18 [BassFigureLine], page 270, Section 3.1.21 [BreakAlignment], page 272, Section 3.1.22 [BreakAlignment], page 272, Section 3.1.30 [DotColumn], page 279, Section 3.1.34 [DynamicLineSpanner], page 282, Section 3.1.66 [NonMusicalPaper-Column], page 308, Section 3.1.67 [NoteCollision], page 309, Section 3.1.68 [NoteColumn], page 310, Section 3.1.74 [PaperColumn], page 314, Section 3.1.92 [SostenutoPedalLineSpanner], page 326, Section 3.1.103 [SustainPedalLineSpanner], page 335, Section 3.1.104 [System], page 337, Section 3.1.116 [TrillPitchGroup], page 347, Section 3.1.122 [UnaCordaPedalLineSpanner], page 353, Section 3.1.124 [VerticalAlignment], page 354, Section 3.1.125 [VerticalAxisGroup], page 355 and Section 3.1.128 [VoltaBracketSpanner], page 357.

3.2.8 balloon-interface
A collection of routines to put text balloons around an object.

User settable properties:

   padding (dimension, in staff space)
      Add this much extra space between objects that are next to each other.

   text (markup)
      Text markup. See Section “Formatting text” in Notation Reference.

This grob interface is used in the following graphical object(s): Section 3.1.10 [BalloonTextItem], page 264.

3.2.9 bar-line-interface
Bar line.

Print a special bar symbol. It replaces the regular bar symbol with a special symbol. The argument bartype is a string which specifies the kind of bar line to print. Options are :l, l:, :l:, :l.l:, :l.., ll, l., ,l., l.l, : and dashed.

These produce, respectively, a right repeat, a left repeat, a thick double repeat, a thin-thick double repeat, a thin-thick double repeat, a double bar, a start bar, an end bar, a thick double bar, a thin-thick-thin bar, a dotted bar and a dashed bar. In addition, there is an option ll: which is equivalent to l: except at line breaks, where it produces a double bar (ll) at the end of the line and a repeat sign (l:) at the beginning of the new line.

If bartype is set to empty then nothing is printed, but a line break is allowed at that spot.

   gap is used for the gaps in dashed bar lines.
User settable properties:

- **allow-span-bar** (boolean)
  - If false, no inter-staff bar line will be created below this bar line.

- **bar-size** (dimension, in staff space)
  - The size of a bar line.

- **gap** (dimension, in staff space)
  - Size of a gap in a variable symbol.

- **glyph** (string)
  - A string determining what ‘style’ of glyph is typeset. Valid choices depend on the function that is reading this property.

- **hair-thickness** (number)
  - Thickness of the thin line in a bar line.

- **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.

- **thick-thickness** (number)
  - Bar line thickness, measured in line-thickness.

- **thin-kern** (number)
  - The space after a hair-line in a bar line.

Internal properties:

- **bar-extent** (pair of numbers)
  - The Y-extent of the actual bar line. This may differ from Y-extent because it does not include the dots in a repeat bar line.

- **glyph-name** (string)
  - The glyph name within the font.

This grob interface is used in the following graphical object(s): Section 3.1.11 [BarLine], page 264 and Section 3.1.94 [SpanBar], page 328.

3.2.10 bass-figure-alignment-interface
Align a bass figure.

This grob interface is used in the following graphical object(s): Section 3.1.14 [BassFigure-Alignment], page 268.

3.2.11 bass-figure-interface
A bass figure text.

User settable properties:

- **implicit** (boolean)
  - Is this an implicit bass figure?

This grob interface is used in the following graphical object(s): Section 3.1.13 [BassFigure], page 267.

3.2.12 beam-interface
A beam.

The thickness property is the weight of beams, measured in staffspace. The direction property is not user-serviceable. Use the direction property of Stem instead.
User settable properties:

**annotation** (string)
Annotate a grob for debug purposes.

**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.

**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.

**break-overshoot** (pair of numbers)
How much does a broken spanner stick out of its bounds?

**clip-edges** (boolean)
Allow outward pointing beamlets at the edges of beams?

**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.

**damping** (number)
Amount of beam slope damping.

**direction** (direction)
If side-axis is 0 (or #X), then this property determines whether the ob-
ject 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.

**gap** (dimension, in staff space)
Size of a gap in a variable symbol.

**gap-count** (integer)
Number of gapped beams for tremolo.

**grow-direction** (direction)
Crescendo or decrescendo?

**inspect-quants** (pair of numbers)
If debugging is set, set beam and slur quants to this position, and print
the respective scores.

**knee** (boolean)
Is this beam kneed?

**length-fraction** (number)
Multiplier for lengths. Used for determining ledger lines and stem
lengths.
neutral-direction (direction)
Which direction to take in the center of the staff.

positions (pair of numbers)
Pair of staff coordinates \((\text{left} \cdot \text{right})\), where both \text{left} and \text{right} are in \text{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.

thickness (number)
Line thickness, generally measured in \text{line-thickness}.

Internal properties:

details (list)
A list of parameters for detailed grob behavior.
More information on the allowed parameters can be found by inspecting \text{lily/slur-scoring.cc}, \text{lily/beam-quanting.cc}, and \text{lily/tie-formatting-problem.cc}. Setting \text{debug-tie-scoring}, \text{debug-beam-scoring} or \text{debug-slur-scoring} also provides useful clues.

least-squares-dy (number)
The ideal beam slope, without damping.

normal-stems (unknown)
An array of visible stems.

quant-score (string)
The beam quanting score; stored for debugging.

quantized-positions (pair of numbers)
The beam positions after quanting.

shorten (dimension, in staff space)
The amount of space that a stem is shortened. Internally used to distribute beam shortening over stems.

stems (unknown)
A list of stem objects, corresponding to the notes that the arpeggio has to be before.

This grob interface is used in the following graphical object(s): Section 3.1.19 [Beam], page 270.

3.2.13 bend-after-interface
A doit or drop.

User settable properties:

thickness (number)
Line thickness, generally measured in \text{line-thickness}.

Internal properties:

delta-position (number)
The vertical position difference.

This grob interface is used in the following graphical object(s): Section 3.1.20 [BendAfter], page 271.
3.2.14 break-alignable-interface
Object that is aligned on a break alignment.

User settable properties:

- **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).

This grob interface is used in the following graphical object(s): Section 3.1.12 [BarNumber], page 266 and Section 3.1.80 [RehearsalMark], page 318.

3.2.15 break-aligned-interface
Items that are aligned in prefatory matter.

The spacing of these items is controlled by the **space-alist** property. It contains a list **break-align-symbols** with a specification of the associated space. The space specification can be

- **(minimum-space . spc)**
  Pad space until the distance is spc.

- **(fixed-space . spc)**
  Set a fixed space.

- **(semi-fixed-space . spc)**
  Set a space. Half of it is fixed and half is stretchable. (does not work at start of line. fixme)

- **(extra-space . spc)**
  Add spc amount of space.

Special keys for the alist are **first-note** and **next-note**, signifying the first note on a line, and the next note halfway a line.

Rules for this spacing are much more complicated than this. See [Wanske] page 126–134, [Ross] page 143–147.

User settable properties:

- **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-symbol** (symbol)
  This key is used for aligning and spacing breakable items.

- **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**.
This grob interface is used in the following graphical object(s): Section 3.1.5 [Ambitus], page 260, Section 3.1.6 [AmbitusAccidental], page 261, Section 3.1.11 [BarLine], page 264, Section 3.1.21 [BreakAlignGroup], page 272, Section 3.1.23 [BreathingSign], page 273, Section 3.1.25 [Clef], page 275, Section 3.1.29 [Custos], page 278, Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.48 [KeyCancellation], page 295, Section 3.1.49 [KeySignature], page 296, Section 3.1.53 [LeftEdge], page 298 and Section 3.1.114 [TimeSignature], page 345.

3.2.16 break-alignment-interface

The object that performs break alignment. See Section 3.2.15 [break-aligned-interface], page 366.

User settable properties:

- 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
  \(\overline{\text{\texttt{\textbackslash{\texttt{override Score.BreakAlignment \#'break-align-orders = \#(make-vector 3 '(span-bar breathing-sign staff-bar key clef time-signature)})}}}\}

Internal properties:

- positioning-done (boolean)
  Used to signal that a positioning element did its job. This ensures that a positioning is only done once.

This grob interface is used in the following graphical object(s): Section 3.1.22 [BreakAlignment], page 272.

3.2.17 breathing-sign-interface

A breathing sign.

User settable properties:

- direction (direction)
  If \texttt{side-axis} is \texttt{0} (or \texttt{#X}), then this property determines whether the object is placed \texttt{#LEFT}, \texttt{#CENTER} or \texttt{#RIGHT} with respect to the other object. Otherwise, it determines whether the object is placed \texttt{#UP}, \texttt{#CENTER} or \texttt{#DOWN}. Numerical values may also be used: \texttt{#UP}=1, \texttt{#DOWN}=-1, \texttt{#LEFT}=-1, \texttt{#RIGHT}=1, \texttt{#CENTER}=0.

This grob interface is used in the following graphical object(s): Section 3.1.23 [BreathingSign], page 273.

3.2.18 chord-name-interface

A chord name.
Internal properties:

begin-of-line-visible (boolean)
Used for marking ChordNames that should only show changes.

This grob interface is used in the following graphical object(s): Section 3.1.24 [ChordName], page 274.

3.2.19 clef-interface
A clef sign.

User settable properties:

full-size-change (boolean)
Don’t make a change clef smaller.

glyph (string)
A string determining what ‘style’ of glyph is typeset. Valid choices depend on the function that is reading this property.

non-default (boolean)
Set for manually specified clefs.

Internal properties:

glyph-name (string)
The glyph name within the font.

This grob interface is used in the following graphical object(s): Section 3.1.25 [Clef], page 275.

3.2.20 cluster-beacon-interface
A place holder for the cluster spanner to determine the vertical extents of a cluster spanner at this X position.

User settable properties:

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.

This grob interface is used in the following graphical object(s): Section 3.1.27 [ClusterSpannerBeacon], page 276.

3.2.21 cluster-interface
A graphically drawn musical cluster.

padding adds to the vertical extent of the shape (top and bottom).

The property style controls the shape of cluster segments. Valid values include leftsided-stairs, rightsided-stairs, centered-stairs, and ramp.

User settable properties:

padding (dimension, in staff space)
Add this much extra space between objects that are next to each other.

style (symbol)
This setting determines in what style a grob is typeset. Valid choices depend on the stencil callback reading this property.
Internal properties:

- **columns** *(unknown)*
  A list of grobs, typically containing `PaperColumn` or `NoteColumn` objects.

This grob interface is used in the following graphical object(s): Section 3.1.26 [ClusterSpanner], page 276.

### 3.2.22 custos-interface

A custos object. `style` can have four valid values: `mensural`, `vaticana`, `medicaea`, and `hufnagel`. `mensural` is the default style.

User settable properties:

- **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.

- **style** *(symbol)*
  This setting determines in what style a grob is typeset. Valid choices depend on the `stencil` callback reading this property.

This grob interface is used in the following graphical object(s): Section 3.1.29 [Custos], page 278.

### 3.2.23 dot-column-interface

Group dot objects so they form a column, and position dots so they do not clash with staff lines.

User settable properties:

- **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`.

Internal properties:

- **dots** *(unknown)*
  Multiple `Dots` objects.

- **positioning-done** *(boolean)*
  Used to signal that a positioning element did its job. This ensures that a positioning is only done once.

This grob interface is used in the following graphical object(s): Section 3.1.30 [DotColumn], page 279.

### 3.2.24 dots-interface

The dots to go with a notehead or rest. `direction` sets the preferred direction to move in case of staff line collisions. `style` defaults to undefined, which is normal 19th/20th century traditional style. Set `style` to `vaticana` for ancient type dots.
User settable properties:

**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.

**style (symbol)**
This setting determines in what style a grob is typeset. Valid choices depend on the `stencil` callback reading this property.

This grob interface is used in the following graphical object(s): Section 3.1.31 [Dots], page 279.

### 3.2.25 dynamic-interface
Any kind of loudness sign.

This grob interface is used in the following graphical object(s): Section 3.1.34 [DynamicLineSpanner], page 282, Section 3.1.35 [DynamicText], page 283, Section 3.1.36 [DynamicTextSpanner], page 284 and Section 3.1.43 [Hairpin], page 290.

### 3.2.26 dynamic-line-spanner-interface
Dynamic line spanner.

User settable properties:

**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.

This grob interface is used in the following graphical object(s): Section 3.1.34 [DynamicLineSpanner], page 282.

### 3.2.27 dynamic-text-spanner-interface
Dynamic text spanner.

User settable properties:

**text (markup)**
Text markup. See Section “Formatting text” in Notation Reference.

This grob interface is used in the following graphical object(s): Section 3.1.36 [DynamicTextSpanner], page 284.

### 3.2.28 enclosing-bracket-interface
Brackets alongside bass figures.

User settable properties:

**bracket-flare (pair of numbers)**
A pair of numbers specifying how much edges of brackets should slant outward. Value 0.0 means straight edges.
**edge-height** (pair)
A pair of numbers specifying the heights of the vertical edges: (left-height, right-height).

**padding** (dimension, in staff space)
Add this much extra space between objects that are next to each other.

**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.

**thickness** (number)
Line thickness, generally measured in line-thickness.

**Internal properties:**

**elements** (unknown)
A list of grobs; the type is depending on the grob where this is set in.

This grob interface is used in the following graphical object(s): Section 3.1.16 [BassFigure-Bracket], page 269.

3.2.29 **figured-bass-continuation-interface**
Simple extender line between bounds.

**User settable properties:**

**padding** (dimension, in staff space)
Add this much extra space between objects that are next to each other.

**thickness** (number)
Line thickness, generally measured in line-thickness.

**Internal properties:**

**figures** (unknown)
Figured bass objects for continuation line.

This grob interface is used in the following graphical object(s): Section 3.1.17 [BassFigure-Continuation], page 269.

3.2.30 **finger-interface**
A fingering instruction.

This grob interface is used in the following graphical object(s): Section 3.1.37 [Fingering], page 285.

3.2.31 **font-interface**
Any symbol that is typeset through fixed sets of glyphs, (i.e., fonts).

**User settable properties:**

**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.

### Internal properties:

**font** (font metric)
A cached font metric object.

This grob interface is used in the following graphical object(s): Section 3.1.1 [Accidental], page 257, Section 3.1.2 [AccidentalCautionary], page 257, Section 3.1.4 [AccidentalSuggestion], page 259, Section 3.1.6 [AmbitusAccidental], page 261, Section 3.1.7 [AmbitusLine], page 262, Section 3.1.8 [AmbitusNoteHead], page 262, Section 3.1.9 [Arpeggio], page 263, Section 3.1.10 [Balloontextitem], page 264, Section 3.1.11 [Barline], page 264, Section 3.1.12 [BarNumber], page 266, Section 3.1.13 [BassFigure], page 267, Section 3.1.19 [Beam], page 270, Section 3.1.23 [BreathingSign], page 273, Section 3.1.24 [ChordName], page 274, Section 3.1.25 [Clef], page 275, Section 3.1.28 [Combintextscript], page 277, Section 3.1.29 [Custos], page 278, Section 3.1.31 [Dots], page 279, Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.35 [DynamicText], page 283, Section 3.1.36 [DynamicTextSpanner], page 284, Section 3.1.37 [Fingering], page 285, Section 3.1.38 [Fretboard], page 287, Section 3.1.44 [HarmonicParenthesesItem], page 291, Section 3.1.46 [InstrumentName], page 293, Section 3.1.47 [InstrumentSwitch], page 294, Section 3.1.48 [KeyCancellation], page 295, Section 3.1.49 [KeySignature], page 296, Section 3.1.56 [LyricHyphen], page 301, Section 3.1.58 [LyricText], page 302, Section 3.1.61 [MensuralLigature], page 304, Section 3.1.62 [MetronomeMark], page 304, Section 3.1.63 [MultiMeasureRest], page 305, Section 3.1.64 [MultiMeasureRestNumber], page 306, Section 3.1.65 [MultiMeasureRestText], page 307, Section 3.1.66 [NonMusicalPaperColumn], page 308, Section 3.1.69 [NoteHead], page 310, Section 3.1.70 [NoteName], page 311, Section 3.1.72 [OctavateEight], page 311, Section 3.1.73 [Ottavabracket], page 312, Section 3.1.74 [PaperColumn], page 314, Section 3.1.75 [ParenthesesItem], page 314, Section 3.1.76 [PercentRepeat], page 315, Section 3.1.77 [PercentRepeatCounter], page 315, Section 3.1.80 [RehearsalMark], page 318, Section 3.1.84 [Rest], page 321, Section 3.1.86 [Script], page 322, Section 3.1.91 [SostenutoPedal], page 325, Section 3.1.94 [Spanbar], page 328, Section 3.1.97 [StanzaNumber], page 330, Section 3.1.98 [Stem], page 330, Section 3.1.100 [StringNumber], page 332, Section 3.1.101 [StrokeFinger], page 334, Section 3.1.102 [SustainPedal], page 335, Section 3.1.106 [SystemStartbrace], page 338, Section 3.1.107 [SystemStartBracket], page 339, Section 3.1.108 [SystemStartSquare], page 340, Section 3.1.109 [TabNoteHead], page 340, Section 3.1.110 [Textscript], page 341, Section 3.1.111 [TextSpanner], page 343, Section 3.1.114 [Timesignature], page 345, Section 3.1.115 [TrillPitchAccidental], page 346, Section 3.1.116 [Trillpitchgroup], page 347, Section 3.1.117 [Trill-
3.2.32 fret-diagram-interface

A fret diagram

User settable properties:

align-dir (direction)
Which side to align? -1: left side, 0: around center of width, 1: right side.

dot-placement-list (list)
List consisting of (description string-number fret-number finger-number) entries used to define fret diagrams.

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-position – Location of dot in fret space. Default 0.6 for dots without labels, 0.95-dot-radius for dots with labels.
- 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-label-font-mag – The magnification of the font used to label the lowest fret number. Default 0.5.
- fret-label-vertical-offset – The offset of the fret label from the center of the fret in direction parallel to strings. Default 0.
- 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, landscape, and opposing-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 for normal orientation, 0.5 for landscape and opposing-landscape.

• **string-thickness-factor** – Factor for changing thickness of each string in the fret diagram. Thickness of string \( k \) is given by \( \text{thickness} \times (1 + \text{string-thickness-factor})^{(k-1)} \). Default 0.

• **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.

```plaintext
size (number)
Size of object, relative to standard size.

thickness (number)
Line thickness, generally measured in line-thickness.
```

This grob interface is used in the following graphical object(s): Section 3.1.38 [FretBoard], page 287.

### 3.2.33 grace-spacing-interface

Keep track of durations in a run of grace notes.

**User settable properties:**

- **common-shortest-duration** (moment)
  The most common shortest note length. This is used in spacing. Enlarging this sets the score tighter.

**Internal properties:**

- **columns** (unknown)
  A list of grobs, typically containing `PaperColumn` or `NoteColumn` objects.

This grob interface is used in the following graphical object(s): Section 3.1.40 [GraceSpacing], page 289.

### 3.2.34 gregorian-ligature-interface

A gregorian ligature.

**Internal properties:**

- **ascendens** (boolean)
  Is this neume of ascending type?
**Chapter 3: Backend**

**auctum** (boolean)
Is this neume liquescentically augmented?

**cavum** (boolean)
Is this neume outlined?

**context-info** (integer)
Within a ligature, the final glyph or shape of a head may be affected by the left and/or right neighbour head. **context-info** holds for each head such information about the left and right neighbour, encoded as a bit mask.

**deminutum** (boolean)
Is this neume diminished?

**descendens** (boolean)
Is this neume of descendent type?

**inclinatum** (boolean)
Is this neume an inclinatum?

**linea** (boolean)
Attach vertical lines to this neume?

**oriscus** (boolean)
Is this neume an oriscus?

**pes-or-flexa** (boolean)
Shall this neume be joined with the previous head?

**prefix-set** (number)
A bit mask that holds all Gregorian head prefixes, such as \virga or \quilisma.

**quilisma** (boolean)
Is this neume a quilisma?

**stropha** (boolean)
Is this neume a stropha?

**virga** (boolean)
Is this neume a virga?

This grob interface is used in the following graphical object(s): Section 3.1.69 [NoteHead], page 310.

3.2.35 **grid-line-interface**

A line that is spanned between grid-points.

**User settable properties:**

**thickness** (number)
Line thickness, generally measured in line-thickness.

**Internal properties:**

**elements** (unknown)
A list of grobs; the type is depending on the grob where this is set in.

This grob interface is used in the following graphical object(s): Section 3.1.41 [GridLine], page 289.
3.2.36 grid-point-interface
A spanning point for grid lines.
This grob interface is used in the following graphical object(s): Section 3.1.42 [GridPoint], page 290.

3.2.37 grob-interface
A grob represents a piece of music notation.

All grobs have an X and Y position on the page. These X and Y positions are stored in a relative format, thus they can easily be combined by stacking them, hanging one grob to the side of another, or coupling them into grouping objects.

Each grob has a reference point (a.k.a. parent): The position of a grob is stored relative to that reference point. For example, the X reference point of a staccato dot usually is the note head that it applies to. When the note head is moved, the staccato dot moves along automatically.

A grob is often associated with a symbol, but some grobs do not print any symbols. They take care of grouping objects. For example, there is a separate grob that stacks staves vertically. The Section 3.1.67 [NoteCollision], page 309 object is also an abstract grob: It only moves around chords, but doesn’t print anything.

Grobs have properties (Scheme variables) that can be read and set. Two types of them exist: immutable and mutable. Immutable variables define the default style and behavior. They are shared between many objects. They can be changed using \override and \revert. Mutable properties are variables that are specific to one grob. Typically, lists of other objects, or results from computations are stored in mutable properties. In particular, every call to set-grob-property (or its C++ equivalent) sets a mutable property.

The properties after-line-breaking and before-line-breaking are dummies that are not user-serviceable.

User settable properties:

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.

after-line-breaking (boolean)
Dummy property, used to trigger callback for after-line-breaking.

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.

before-line-breaking (boolean)
Dummy property, used to trigger a callback function.

color (list)
The color of this grob.

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-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.

**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.

**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.

**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.

**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.

**springs-and-rods** (boolean)
Dummy variable for triggering spacing routines.

**stencil** (unknown)
The symbol to print.

**transparent** (boolean)
This makes the grob invisible.

**Internal properties:**

**axis-group-parent-X** (layout object)
Containing X axis group.

**axis-group-parent-Y** (layout object)
Containing Y axis group.

**cause** (any type)
Any kind of causation objects (i.e., music, or perhaps translator) that was the cause for this grob.
cross-staff (boolean)
For a beam or a stem, this is true if we depend on inter-staff spacing.

interfaces (list)
A list of symbols indicating the interfaces supported by this object. It is initialized from the meta field.

meta (list) Provide meta information. It is an alist with the entries name and interfaces.

pure-Y-offset-in-progress (boolean)
A debugging aid for catching cyclic dependencies.

staff-symbol (layout object)
The staff symbol grob that we are in.

This grob interface is used in the following graphical object(s): Section 3.1.1 [Accidental], page 257, Section 3.1.2 [AccidentalCautionary], page 257, Section 3.1.3 [AccidentalPlacement], page 258, Section 3.1.4 [AccidentalSuggestion], page 259, Section 3.1.5 [Ambitus], page 260, Section 3.1.6 [AmbitusAccidental], page 261, Section 3.1.7 [AmbitusLine], page 262, Section 3.1.8 [AmbitusNoteHead], page 262, Section 3.1.9 [Arpeggio], page 263, Section 3.1.10 [BalloonTextItem], page 264, Section 3.1.11 [BarLine], page 264, Section 3.1.12 [BarNumber], page 266, Section 3.1.13 [BassFigure], page 267, Section 3.1.14 [BassFigureAlignment], page 268, Section 3.1.15 [BassFigureAlignmentPositioning], page 268, Section 3.1.16 [BassFigureBracket], page 269, Section 3.1.17 [BassFigureContinuation], page 269, Section 3.1.18 [BassFigureLine], page 270, Section 3.1.19 [Beam], page 270, Section 3.1.20 [BendAfter], page 271, Section 3.1.21 [BreakAlignGroup], page 272, Section 3.1.22 [BreakAlignment], page 272, Section 3.1.23 [BreathingSign], page 273, Section 3.1.24 [ChordName], page 274, Section 3.1.25 [Clef], page 275, Section 3.1.26 [ClusterSpanner], page 276, Section 3.1.27 [ClusterSpannerBeacon], page 276, Section 3.1.28 [CombineTextScript], page 277, Section 3.1.29 [Custos], page 278, Section 3.1.30 [DotColumn], page 279, Section 3.1.31 [Dots], page 279, Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.34 [DynamicLineSpanner], page 282, Section 3.1.35 [DynamicText], page 283, Section 3.1.36 [DynamicTextSpanner], page 284, Section 3.1.37 [Fingering], page 285, Section 3.1.38 [FretBoard], page 287, Section 3.1.39 [Glissando], page 288, Section 3.1.40 [GraceSpacing], page 289, Section 3.1.41 [GridLine], page 289, Section 3.1.42 [GridPoint], page 290, Section 3.1.43 [Hairpin], page 290, Section 3.1.44 [HarmonicParenthesesItem], page 291, Section 3.1.45 [HorizontalBracket], page 292, Section 3.1.46 [InstrumentName], page 293, Section 3.1.47 [InstrumentSwitch], page 294, Section 3.1.48 [KeyCancellation], page 295, Section 3.1.49 [KeySignature], page 296, Section 3.1.50 [LaissezVibrerTie], page 297, Section 3.1.51 [LaissezVibrerTieColumn], page 297, Section 3.1.52 [LedgerLineSpanner], page 298, Section 3.1.53 [LeftEdge], page 298, Section 3.1.54 [LigatureBracket], page 299, Section 3.1.55 [LyricExtender], page 300, Section 3.1.56 [LyricHyphen], page 301, Section 3.1.57 [LyricSpace], page 302, Section 3.1.58 [LyricText], page 302, Section 3.1.59 [MeasureGrouping], page 303, Section 3.1.60 [MelodyItem], page 304, Section 3.1.61 [MensuralLigature], page 304, Section 3.1.62 [MetronomeMark], page 304, Section 3.1.63 [MultiMeasureRest], page 305, Section 3.1.64 [MultiMeasureRestNumber], page 306, Section 3.1.65 [MultiMeasureRestText], page 307, Section 3.1.66 [NonMusicalPaperColumn], page 308, Section 3.1.67 [NoteCollision], page 309, Section 3.1.68 [NoteColumn], page 310, Section 3.1.69 [NoteHead], page 310, Section 3.1.70 [NoteName], page 311, Section 3.1.71 [NoteSpacing], page 311, Section 3.1.72 [OctavateEight], page 311, Section 3.1.73 [OttavaBracket], page 312, Section 3.1.74 [PaperColumn], page 314, Section 3.1.75 [ParenthesesItem], page 314, Section 3.1.76 [PercentRepeat], page 315, Section 3.1.77 [PercentRepeatCounter], page 315, Section 3.1.78 [PhrasingSlur], page 317, Section 3.1.79 [PianoPedalBracket], page 318, Section 3.1.80 [RehearsalMark], page 318, Section 3.1.81 [RepeatSlash], page 320, Section 3.1.82 [RepeatTie], page 320, Section 3.1.83 [RepeatTieColumn], page 321, Section 3.1.84
3.2.38 hairpin-interface

A hairpin crescendo or decrescendo.

User settable properties:

- **bound-padding** (number)
  - The amount of padding to insert around spanner bounds.

- **circled-tip** (boolean)
  - Put a circle at start/end of hairpins (al/del niente).

- **grow-direction** (direction)
  - Crescendo or decrescendo?

- **height** (dimension, in staff space)
  - Height of an object in staff-space units.

Internal properties:

- **adjacent-hairpins** (unknown)
  - A list of directly neighboring hairpins.

This grob interface is used in the following graphical object(s): Section 3.1.43 [Hairpin], page 290.

3.2.39 hara-kiri-group-spanner-interface

A group spanner that keeps track of interesting items. If it doesn’t contain any after line breaking, it removes itself and all its children.

User settable properties:

- **remove-empty** (boolean)
  - If set, remove group if it contains no interesting items.
Chapter 3: Backend

remove-first (boolean)
Remove the first staff of an orchestral score?

Internal properties:

important-column-ranks (vector)
A cache of columns that contain items-worth-living data.

items-worth-living (unknown)
A list of interesting items. If empty in a particular staff, then that staff
is erased.

This grob interface is used in the following graphical object(s): Section 3.1.125 [VerticalAxisGroup], page 355.

3.2.40 horizontal-bracket-interface
A horizontal bracket encompassing notes.

User settable properties:

bracket-flare (pair of numbers)
A pair of numbers specifying how much edges of brackets should slant
outward. Value 0.0 means straight edges.

connect-to-neighbor (pair)
Pair of booleans, indicating whether this grob looks as a continued
break.

date-time (pair)
A pair of numbers specifying the heights of the vertical edges: (left-
height . right-height).

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.

Internal properties:

columns (unknown)
A list of grobs, typically containing PaperColumn or NoteColumn ob-
jects.

This grob interface is used in the following graphical object(s): Section 3.1.45 [Horizontal-
Bracket], page 292, Section 3.1.73 [OttavaBracket], page 312 and Section 3.1.127 [VoltaBracket],
page 356.

3.2.41 instrument-specific-markup-interface
Instrument-specific markup (like fret boards or harp pedal diagrams).

User settable properties:

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-position** – Location of dot in fret space. Default 0.6 for dots without labels, 0.95-dot-radius for dots with labels.

• **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 offset of the fret label from the center of the fret in direction parallel to strings. Default 0.

• **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, landscape, and opposing-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 for normal orientation, 0.5 for landscape and opposing-landscape.

• **string-thickness-factor** – Factor for changing thickness of each string in the fret diagram. Thickness of string $k$ is given by $\text{thickness} \times (1+\text{string-thickness-factor})^k$. Default 0.

• **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.
harp-pedal-details (list)

An alist of detailed grob properties for harp pedal diagrams. Each alist entry consists of a \texttt{(property value)} pair. The properties which can be included in harp-pedal-details include the following:

- \texttt{box-offset} – Vertical shift of the center of flat/sharp pedal boxes above/below the horizontal line. Default value 0.8.
- \texttt{box-width} – Width of each pedal box. Default value 0.4.
- \texttt{box-height} – Height of each pedal box. Default value 1.0.
- \texttt{space-before-divider} – Space between boxes before the first divider (so that the diagram can be made symmetric). Default value 0.8.
- \texttt{space-after-divider} – Space between boxes after the first divider. Default value 0.8.
- \texttt{circle-thickness} – Thickness (in unit of the line-thickness) of the ellipse around circled pedals. Default value 0.5.
- \texttt{circle-x-padding} – Padding in X direction of the ellipse around circled pedals. Default value 0.15.
- \texttt{circle-y-padding} – Padding in Y direction of the ellipse around circled pedals. Default value 0.2.

size (number)
Size of object, relative to standard size.

thickness (number)
Line thickness, generally measured in \texttt{line-thickness}.

This grob interface is used in the following graphical object(s): Section 3.1.110 [TextScript], page 341.

3.2.42 item-interface

Grobs can be distinguished in their role in the horizontal spacing. Many grobs define constraints on the spacing by their sizes, for example, note heads, clefs, stems, and all other symbols with a fixed shape. These grobs form a subtype called \texttt{Item}.

Some items need special treatment for line breaking. For example, a clef is normally only printed at the start of a line (i.e., after a line break). To model this, ‘breakable’ items (clef, key signature, bar lines, etc.) are copied twice. Then we have three versions of each breakable item: one version if there is no line break, one version that is printed before the line break (at the end of a system), and one version that is printed after the line break.

Whether these versions are visible and take up space is determined by the outcome of the \texttt{break-visibility} grob property, which is a function taking a direction (-1, 0 or 1) as an argument. It returns a cons of booleans, signifying whether this grob should be transparent and have no extent.

The following variables for \texttt{break-visibility} are predefined:

<table>
<thead>
<tr>
<th>grob will show:</th>
<th>before break</th>
<th>no break</th>
<th>after break</th>
</tr>
</thead>
<tbody>
<tr>
<td>all-invisible</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>begin-of-line-visible</td>
<td>no</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>end-of-line-visible</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>all-visible</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>begin-of-line-invisible</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>end-of-line-invisible</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>center-invisible</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
</tbody>
</table>
User settable properties:

**break-visibility** (vector)
A vector of 3 booleans, **(#(end-of-line unbroken begin-of-line)).**
#t means visible, #f means killed.

**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).

**non-musical** (boolean)
True if the grob belongs to a NonMusicalPaperColumn.

This grob interface is used in the following graphical object(s): Section 3.1.1 [Accidental], page 257, Section 3.1.2 [AccidentalCautionary], page 257, Section 3.1.3 [AccidentalPlacement], page 258, Section 3.1.4 [AccidentalSuggestion], page 259, Section 3.1.5 [Ambitus], page 260, Section 3.1.6 [AmbitusAccidental], page 261, Section 3.1.7 [AmbitusLine], page 262, Section 3.1.8 [AmbitusNoteHead], page 262, Section 3.1.9 [Arpeggio], page 263, Section 3.1.10 [BalloonTextItem], page 264, Section 3.1.11 [BarLine], page 264, Section 3.1.12 [BarNumber], page 266, Section 3.1.13 [BassFigure], page 267, Section 3.1.16 [BassFigureBracket], page 269, Section 3.1.21 [BreakAlignGroup], page 272, Section 3.1.22 [BreakAlignment], page 272, Section 3.1.23 [BreathingSign], page 273, Section 3.1.24 [ChordName], page 274, Section 3.1.25 [Clef], page 275, Section 3.1.27 [ClusterSpannerBeacon], page 276, Section 3.1.28 [CombineTextScript], page 277, Section 3.1.29 [Custos], page 278, Section 3.1.30 [DotColumn], page 279, Section 3.1.31 [Dots], page 279, Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.35 [DynamicText], page 283, Section 3.1.37 [Fingering], page 285, Section 3.1.38 [FretBoard], page 287, Section 3.1.41 [GridLine], page 289, Section 3.1.42 [GridPoint], page 290, Section 3.1.44 [HarmonicParenthesesItem], page 291, Section 3.1.47 [InstrumentSwitch], page 294, Section 3.1.48 [KeyCancellation], page 295, Section 3.1.49 [KeySignature], page 296, Section 3.1.50 [LaissezVibrerTie], page 297, Section 3.1.51 [LaissezVibrerTieColumn], page 297, Section 3.1.53 [LeftEdge], page 298, Section 3.1.58 [LyricText], page 302, Section 3.1.60 [MelodyItem], page 304, Section 3.1.62 [MetronomeMark], page 304, Section 3.1.66 [NonMusicalPaperColumn], page 308, Section 3.1.67 [NoteCollision], page 309, Section 3.1.68 [NoteColumn], page 310, Section 3.1.69 [NoteHead], page 310, Section 3.1.70 [NoteName], page 311, Section 3.1.71 [NoteSpacing], page 311, Section 3.1.72 [OctavateEight], page 311, Section 3.1.74 [PaperColumn], page 314, Section 3.1.75 [ParenthesesItem], page 314, Section 3.1.80 [RehearsalMark], page 318, Section 3.1.81 [RepeatSlash], page 320, Section 3.1.82 [RepeatTie], page 320, Section 3.1.83 [RepeatTieColumn], page 321, Section 3.1.84 [Rest], page 321, Section 3.1.85 [RestCollision], page 322, Section 3.1.86 [Script], page 322, Section 3.1.87 [ScriptColumn], page 323, Section 3.1.88 [ScriptRow], page 323, Section 3.1.89 [SeparationItem], page 323, Section 3.1.91 [SostenutoPedal], page 325, Section 3.1.94 [SpanBar], page 328, Section 3.1.95 [StaffSpacing], page 329, Section 3.1.97 [StanzaNumber], page 330, Section 3.1.98 [Stem], page 330, Section 3.1.99 [StemTremolo], page 332, Section 3.1.100 [StringNumber], page 332, Section 3.1.101 [StrokeFinger], page 334, Section 3.1.102 [SustainPedal], page 335, Section 3.1.109 [TabNoteHead], page 340, Section 3.1.110 [TextScript], page 341, Section 3.1.114...
3.2.43 key-cancellation-interface

A key cancellation.

This grob interface is used in the following graphical object(s): Section 3.1.48 [KeyCancellation], page 295.

3.2.44 key-signature-interface

A group of accidentals, to be printed as signature sign.

User settable properties:

- alteration-alist (list)
  List of (pitch . accidental) pairs for key signature.

- c0-position (integer)
  An integer indicating the position of middle C.

- glyph-name-alist (list)
  An alist of key-string pairs.

- 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.

This grob interface is used in the following graphical object(s): Section 3.1.48 [KeyCancellation], page 295 and Section 3.1.49 [KeySignature], page 296.

3.2.45 ledger-line-spanner-interface

This spanner draws the ledger lines of a staff. This is a separate grob because it has to process all potential collisions between all note heads.

User settable properties:

- gap (dimension, in staff space)
  Size of a gap in a variable symbol.

- length-fraction (number)
  Multiplier for lengths. Used for determining ledger lines and stem lengths.

- minimum-length-fraction (number)
  Minimum length of ledger line as fraction of note head size.

- thickness (number)
  Line thickness, generally measured in line-thickness.

Internal properties:

- note-heads (unknown)
  A list of note head grobs.

This grob interface is used in the following graphical object(s): Section 3.1.52 [LedgerLineSpanner], page 298.
3.2.46 ledgered-interface

Objects that need ledger lines, typically note heads. See also Section 3.2.45 [ledger-line-spanner-interface], page 384.

User settable properties:

  - no-ledgers (boolean)
    If set, don’t draw ledger lines on this object.

This grob interface is used in the following graphical object(s): Section 3.1.8 [AmbitusNoteHead], page 262, Section 3.1.69 [NoteHead], page 310 and Section 3.1.117 [TrillPitchHead], page 348.

3.2.47 ligature-bracket-interface

A bracket indicating a ligature in the original edition.

User settable properties:

  - height (dimension, in staff space)
    Height of an object in staff-space units.
  - thickness (number)
    Line thickness, generally measured in line-thickness.
  - width (dimension, in staff space)
    The width of a grob measured in staff space.

This grob interface is not used in any graphical object.

3.2.48 ligature-interface

A ligature.

This grob interface is not used in any graphical object.

3.2.49 line-interface

Generic line objects. Any object using lines supports this. The property style can be line, dashed-line, trill, dotted-line or zigzag.

For dashed-line, the length of the dashes is tuned with dash-fraction. If the latter is set to 0, a dotted line is produced. If dash-period is negative, the line is made transparent.

User settable properties:

  - arrow-length (number)
    Arrow length.
  - arrow-width (number)
    Arrow width.
  - 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.
  - style (symbol)
    This setting determines in what style a grob is typeset. Valid choices depend on the stencil callback reading this property.
**thickness** (number)
Line thickness, generally measured in `line-thickness`.

**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.

This grob interface is used in the following graphical object(s): Section 3.1.36 [DynamicTextSpanner], page 284, Section 3.1.39 [Glissando], page 288, Section 3.1.43 [Hairpin], page 290, Section 3.1.45 [HorizontalBracket], page 292, Section 3.1.54 [LigatureBracket], page 299, Section 3.1.73 [OttavaBracket], page 312, Section 3.1.79 [PianoPedalBracket], page 318, Section 3.1.111 [TextSpanner], page 343, Section 3.1.118 [TrillSpanner], page 349, Section 3.1.119 [TupletBracket], page 350, Section 3.1.126 [VoiceFollower], page 356 and Section 3.1.127 [VoltaBracket], page 356.

### 3.2.50 line-spanner-interface
Generic line drawn between two objects, e.g., for use with glissandi.

**User settable properties:**

- **bound-details** (list)
  An alist of properties for determining attachments of spanners to edges.

- **extra-dy** (number)
  Slope glissandi this much extra.

- **gap** (dimension, in staff space)
  Size of a gap in a variable symbol.

- **left-bound-info** (list)
  An alist of properties for determining attachments of spanners to edges.

- **right-bound-info** (list)
  An alist of properties for determining attachments of spanners to edges.

- **thickness** (number)
  Line thickness, generally measured in `line-thickness`.

- **to-barline** (boolean)
  If true, the spanner will stop at the bar line just before it would otherwise stop.

**Internal properties:**

- **note-columns** (pair)
  A list of `NoteColumn` grobs.

This grob interface is used in the following graphical object(s): Section 3.1.36 [DynamicTextSpanner], page 284, Section 3.1.39 [Glissando], page 288, Section 3.1.111 [TextSpanner], page 343, Section 3.1.118 [TrillSpanner], page 349 and Section 3.1.126 [VoiceFollower], page 356.

### 3.2.51 lyric-extender-interface
The extender is a simple line at the baseline of the lyric that helps show the length of a melisma (a tied or slurred note).
User settable properties:

**left-padding** (dimension, in staff space)
The amount of space that is put left to an object (e.g., a group of accidentals).

**next** (layout object)
Object that is next relation (e.g., the lyric syllable following an extender).

**right-padding** (dimension, in staff space)
Space to insert on the right side of an object (e.g., between note and its accidentals).

**thickness** (number)
Line thickness, generally measured in **line-thickness**.

Internal properties:

**heads** (unknown)
A list of note heads.

This grob interface is used in the following graphical object(s): Section 3.1.55 [LyricExtender], page 300.

### 3.2.52 lyric-hyphen-interface
A centered hyphen is simply a line between lyrics used to divide syllables.

User settable properties:

**dash-period** (number)
The length of one dash together with whitespace. If negative, no line is drawn at all.

**height** (dimension, in staff space)
Height of an object in **staff-space** units.

**length** (dimension, in staff space)
User override for the stem length of unbeamed stems.

**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.

**padding** (dimension, in staff space)
Add this much extra space between objects that are next to each other.

**thickness** (number)
Line thickness, generally measured in **line-thickness**.

This grob interface is used in the following graphical object(s): Section 3.1.56 [LyricHyphen], page 301 and Section 3.1.57 [LyricSpace], page 302.
3.2.53 lyric-interface
Any object that is related to lyrics.

This grob interface is used in the following graphical object(s): Section 3.1.55 [LyricExtender], page 300 and Section 3.1.56 [LyricHyphen], page 301.

3.2.54 lyric-syllable-interface
A single piece of lyrics.

This grob interface is used in the following graphical object(s): Section 3.1.58 [LyricText], page 302.

3.2.55 mark-interface
A rehearsal mark.

This grob interface is used in the following graphical object(s): Section 3.1.80 [RehearsalMark], page 318.

3.2.56 measure-grouping-interface
This object indicates groups of beats. Valid choices for style are bracket and triangle.

User settable properties:

- **height** (dimension, in staff space)
  Height of an object in staff-space units.

- **style** (symbol)
  This setting determines in what style a grob is typeset. Valid choices depend on the stencil callback reading this property.

- **thickness** (number)
  Line thickness, generally measured in line-thickness.

This grob interface is used in the following graphical object(s): Section 3.1.59 [Measure-Grouping], page 303.

3.2.57 melody-spanner-interface
Context dependent typesetting decisions.

User settable properties:

- **neutral-direction** (direction)
  Which direction to take in the center of the staff.

Internal properties:

- **stems** (unknown)
  A list of stem objects, corresponding to the notes that the arpeggio has to be before.

This grob interface is used in the following graphical object(s): Section 3.1.60 [MelodyItem], page 304.

3.2.58 mensural-ligature-interface
A mensural ligature.
User settable properties:

- **thickness** (number)
  Line thickness, generally measured in `line-thickness`.

**Internal properties:**

- **delta-position** (number)
  The vertical position difference.
- **flexa-width** (dimension, in staff space)
  The width of a flexa shape in a ligature grob in (in `staff-space` units).
- **head-width** (dimension, in staff space)
  The width of this ligature head.
- **join-right-amount** (number)
  DOCME
- **primitive** (integer)
  A pointer to a ligature primitive, i.e., an item similar to a note head that is part of a ligature.

This grob interface is used in the following graphical object(s): Section 3.1.61 [MensuralLigature], page 304 and Section 3.1.69 [NoteHead], page 310.

3.2.59 **metronome-mark-interface**

A metronome mark.

This grob interface is used in the following graphical object(s): Section 3.1.62 [MetronomeMark], page 304.

3.2.60 **multi-measure-interface**

Multi measure rest, and the text or number that is printed over it.

User settable properties:

- **bound-padding** (number)
  The amount of padding to insert around spanner bounds.

This grob interface is used in the following graphical object(s): Section 3.1.63 [MultiMeasureRest], page 305, Section 3.1.64 [MultiMeasureRestNumber], page 306 and Section 3.1.65 [MultiMeasureRestText], page 307.

3.2.61 **multi-measure-rest-interface**

A rest that spans a whole number of measures.

User settable properties:

- **bound-padding** (number)
  The amount of padding to insert around spanner bounds.
- **expand-limit** (integer)
  Maximum number of measures expanded in church rests.
- **hair-thickness** (number)
  Thickness of the thin line in a bar line.
- **measure-count** (integer)
  The number of measures for a multi-measure rest.
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.

thick-thickness (number)
Bar line thickness, measured in line-thickness.

Internal properties:

use-breve-rest (boolean)
Use breve rests for measures longer than a whole rest.

This grob interface is used in the following graphical object(s): Section 3.1.63 [MultiMeasureRest], page 305 and Section 3.1.76 [PercentRepeat], page 315.

3.2.62 note-collision-interface
An object that handles collisions between notes with different stem directions and horizontal shifts. Most of the interesting properties are to be set in Section 3.2.63 [note-column-interface], page 390: these are force-hshift and horizontal-shift.

User settable properties:

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 Section “note-collision-interface” in Internals Reference.
merge-differently-headed only applies to opposing stem directions (i.e., voice 1 & 2).

prefer-dotted-right (boolean)
For note collisions, prefer to shift dotted up-note to the right, rather than shifting just the dot.

Internal properties:

positioning-done (boolean)
Used to signal that a positioning element did its job. This ensures that a positioning is only done once.

This grob interface is used in the following graphical object(s): Section 3.1.67 [NoteCollision], page 309.

3.2.63 note-column-interface
Stem and noteheads combined.
User settable properties:

`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 Section “note-collision-interface” in Internals Reference.

`horizontal-shift` (integer)
An integer that identifies ranking of NoteColumns for horizontal shifting. This is used by Section “note-collision-interface” in Internals Reference.

`ignore-collision` (boolean)
If set, don’t do note collision resolution on this NoteColumn.

Internal properties:

`arpeggio` (layout object)
A pointer to an Arpeggio object.

`note-heads` (unknown)
A list of note head grobs.

`rest` (layout object)
A pointer to a Rest object.

`rest-collision` (layout object)
A rest collision that a rest is in.

`stem` (layout object)
A pointer to a Stem object.

This grob interface is used in the following graphical object(s): Section 3.1.68 [NoteColumn], page 310.

3.2.64 note-head-interface

A note head. There are many possible values for style. For a complete list, see Section “Note head styles” in Notation Reference.

User settable properties:

`note-names` (vector)
Vector of strings containing names for easy-notation note heads.

`stem-attachment` (pair of numbers)
An (x, y) pair where the stem attaches to the notehead.

`style` (symbol)
This setting determines in what style a grob is typeset. Valid choices depend on the stencil callback reading this property.

Internal properties:

`accidental-grob` (layout object)
The accidental for this note.

`glyph-name` (string)
The glyph name within the font.

This grob interface is used in the following graphical object(s): Section 3.1.8 [AmbitusNote-Head], page 262, Section 3.1.69 [NoteHead], page 310, Section 3.1.109 [TabNoteHead], page 340 and Section 3.1.116 [TrillPitchGroup], page 347.
3.2.65 note-name-interface

Note names.

This grob interface is used in the following graphical object(s): Section 3.1.70 [NoteName], page 311.

3.2.66 note-spacing-interface

This object calculates spacing wishes for individual voices.

User settable properties:

knee-spacing-correction (number)
Factor for the optical correction amount for kneed beams. Set between 0 for no correction and 1 for full correction.

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.

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.

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.

Internal properties:

left-items (unknown)
DOCME

right-items (unknown)
DOCME

This grob interface is used in the following graphical object(s): Section 3.1.71 [NoteSpacing], page 311.

3.2.67 only-prebreak-interface

Kill this grob after the line breaking process.

This grob interface is not used in any graphical object.

3.2.68 ottava-bracket-interface

An ottava bracket.

User settable properties:

bracket-flare (pair of numbers)
A pair of numbers specifying how much edges of brackets should slant outward. Value 0.0 means straight edges.
edge-height (pair)
A pair of numbers specifying the heights of the vertical edges: \((\text{left-height} \cdot \text{right-height})\).

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 \text{springs-and-rod}s property. If added to a Tie, this sets the minimum distance between noteheads.

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.

This grob interface is used in the following graphical object(s): Section 3.1.73 [OttavaBracket], page 312.

3.2.69 paper-column-interface
Paper_column objects form the top-most X parents for items. There are two types of columns: musical columns, where are attached to, and non-musical columns, where bar-lines, clefs, etc., are attached to. The spacing engine determines the X positions of these objects.

They are numbered, the first (leftmost) is column 0. Numbering happens before line breaking, and columns are not renumbered after line breaking. Since many columns go unused, you should only use the rank field to get ordering information. Two adjacent columns may have non-adjacent numbers.

User settable properties:

between-cols (pair)
Where to attach a loose column to.

full-measure-extra-space (number)
Extra space that is allocated at the beginning of a measure with only one note. This property is read from the NonMusicalPaperColumn that begins the measure.

labels (list)
List of labels (symbols) placed on a column

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.

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.

rhythmic-location (rhythmic location)
Where (bar number, measure position) in the score.

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.

used (boolean)
If set, this spacing column is kept in the spacing problem.

when (moment)
Global time step associated with this column happen?

Internal properties:

bounded-by-me (unknown)
A list of spanners that have this column as start/begin point. Only columns that have grobs or act as bounds are spaced.

grace-spacing (layout object)
A run of grace notes.

spacing (layout object)
The spacing spanner governing this section.

This grob interface is used in the following graphical object(s): Section 3.1.66 [NonMusical-PaperColumn], page 308 and Section 3.1.74 [PaperColumn], page 314.

3.2.70 parentheses-interface
Parentheses for other objects.

User settable properties:

padding (dimension, in staff space)
Add this much extra space between objects that are next to each other.

stencils (list)
Multiple stencils, used as intermediate value.

This grob interface is used in the following graphical object(s): Section 3.1.44 [HarmonicParenthesesItem], page 291, Section 3.1.75 [ParenthesesItem], page 314 and Section 3.1.116 [TrillPitchGroup], page 347.

3.2.71 percent-repeat-interface
Beat, Double and single measure repeats.
User settable properties:

**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.

**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.

**thickness** (number)
Line thickness, generally measured in line-thickness.

This grob interface is used in the following graphical object(s): Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.76 [PercentRepeat], page 315, Section 3.1.77 [PercentRepeatCounter], page 315 and Section 3.1.81 [RepeatSlash], page 320.

3.2.72 percent-repeat-item-interface
Repeats that look like percent signs.

User settable properties:

**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.

**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.

**thickness** (number)
Line thickness, generally measured in line-thickness.

This grob interface is used in the following graphical object(s): Section 3.1.32 [DoublePercentRepeat], page 280, Section 3.1.33 [DoublePercentRepeatCounter], page 281 and Section 3.1.81 [RepeatSlash], page 320.

3.2.73 piano-pedal-bracket-interface
The bracket of the piano pedal. It can be tuned through the regular bracket properties.

User settable properties:

**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.

**edge-height** (pair)
A pair of numbers specifying the heights of the vertical edges: (left-height . right-height).
**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.

**Internal properties:**

**pedal-text** (layout object)

A pointer to the text of a mixed-style piano pedal.

This grob interface is used in the following graphical object(s): Section 3.1.79 [PianoPedalBracket], page 318.

### 3.2.74 piano-pedal-interface

A piano pedal sign.

This grob interface is used in the following graphical object(s): Section 3.1.79 [PianoPedalBracket], page 318, Section 3.1.92 [SostenutoPedalLineSpanner], page 326, Section 3.1.102 [SustainPedal], page 335, Section 3.1.103 [SustainPedalLineSpanner], page 335 and Section 3.1.122 [UnaCordaPedalLineSpanner], page 353.

### 3.2.75 piano-pedal-script-interface

A piano pedal sign, fixed size.

This grob interface is used in the following graphical object(s): Section 3.1.91 [SostenutoPedal], page 325, Section 3.1.102 [SustainPedal], page 335 and Section 3.1.121 [UnaCordaPedal], page 352.

### 3.2.76 pitched-trill-interface

A note head to indicate trill pitches.

**Internal properties:**

**accidental-grob** (layout object)

The accidental for this note.

This grob interface is used in the following graphical object(s): Section 3.1.117 [TrillPitchHead], page 348.

### 3.2.77 rest-collision-interface

Move around ordinary rests (not multi-measure-rests) to avoid conflicts.

**User settable properties:**

**minimum-distance** (dimension, in staff space)

Minimum distance between rest and notes or beam.

**Internal properties:**

**elements** (unknown)

A list of grobs; the type is depending on the grob where this is set in.

**positioning-done** (boolean)

Used to signal that a positioning element did its job. This ensures that a positioning is only done once.

This grob interface is used in the following graphical object(s): Section 3.1.85 [RestCollision], page 322.
3.2.78 rest-interface

A rest symbol. The property style can be default, mensural, neomensural or classical.

**User settable properties:**

- **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.

- **minimum-distance (dimension, in staff space)**
  Minimum distance between rest and notes or beam.

- **style (symbol)**
  This setting determines in what style a grob is typeset. Valid choices depend on the stencil callback reading this property.

This grob interface is used in the following graphical object(s): Section 3.1.63 [MultiMeasureRest], page 305 and Section 3.1.84 [Rest], page 321.

3.2.79 rhythmic-grob-interface

Any object with a duration. Used to determine which grobs are interesting enough to maintain a hara-kiri staff.

This grob interface is used in the following graphical object(s): Section 3.1.13 [BassFigure], page 267, Section 3.1.24 [ChordName], page 274, Section 3.1.27 [ClusterSpannerBeacon], page 276, Section 3.1.58 [LyricText], page 302, Section 3.1.69 [NoteHead], page 310, Section 3.1.81 [RepeatSlash], page 320, Section 3.1.84 [Rest], page 321 and Section 3.1.109 [TabNoteHead], page 340.

3.2.80 rhythmic-head-interface

Note head or rest.

**User settable properties:**

- **duration-log (integer)**
  The 2-log of the note head duration, i.e., 0 = whole note, 1 = half note, etc.

**Internal properties:**

- **dot (layout object)**
  A reference to a Dots object.

- **stem (layout object)**
  A pointer to a Stem object.

This grob interface is used in the following graphical object(s): Section 3.1.8 [AmbitusNoteHead], page 262, Section 3.1.69 [NoteHead], page 310, Section 3.1.84 [Rest], page 321, Section 3.1.109 [TabNoteHead], page 340 and Section 3.1.117 [TrillPitchHead], page 348.

3.2.81 script-column-interface

An interface that sorts scripts according to their script-priority.

This grob interface is used in the following graphical object(s): Section 3.1.87 [ScriptColumn], page 323 and Section 3.1.88 [ScriptRow], page 323.
3.2.82 script-interface
An object that is put above or below a note.

User settable properties:

- **add-stem-support (boolean)**
  If set, the Stem object is included in this script’s support.

- **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.

- **script-priority (number)**
  A sorting key that determines in what order a script is within a stack of scripts.

- **slur-padding (number)**
  Extra distance between slur and script.

- **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.

Internal properties:

- **positioning-done (boolean)**
  Used to signal that a positioning element did its job. This ensures that a positioning is only done once.

- **script-stencil (pair)**
  A pair (type . arg) which acts as an index for looking up a Stencil object.

- **slur (layout object)**
  A pointer to a Slur object.

This grob interface is used in the following graphical object(s): Section 3.1.4 [AccidentalSuggestion], page 259, Section 3.1.35 [DynamicText], page 283 and Section 3.1.86 [Script], page 322.

3.2.83 self-alignment-interface
Position this object on itself and/or on its parent. To this end, the following functions are provided:

- **Self_alignment_interface::[xy]_aligned_on_self**
  Align self on reference point, using self-alignment-X and self-alignment-Y.

- **Self_alignment_interface::aligned_on_[xy]_parent**
  Shift the object so its own reference point is centered on the extent of the parent

User settable properties:

- **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.

This grob interface is used in the following graphical object(s): Section 3.1.4 [AccidentalSuggestion], page 259, Section 3.1.12 [BarNumber], page 266, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.35 [DynamicText], page 283, Section 3.1.37 [Fingering], page 285, Section 3.1.41 [GridLine], page 289, Section 3.1.43 [Hairpin], page 290, Section 3.1.46 [InstrumentName], page 293, Section 3.1.47 [InstrumentSwitch], page 294, Section 3.1.58 [LyricText], page 302, Section 3.1.64 [MultiMeasureRestNumber], page 306, Section 3.1.65 [MultiMeasureRestText], page 307, Section 3.1.72 [OctavateEight], page 311, Section 3.1.77 [PercentRepeatCounter], page 315, Section 3.1.80 [RehearsalMark], page 318, Section 3.1.91 [SostenutoPedal], page 325, Section 3.1.100 [StringNumber], page 332, Section 3.1.101 [StrokeFinger], page 334, Section 3.1.102 [SustainPedal], page 335, Section 3.1.110 [TextScript], page 341 and Section 3.1.121 [UnaCordaPedal], page 352.

### 3.2.84 semi-tie-column-interface

The interface for a column of l.v. (laissez vibrer) ties.

**User settable properties:**

- **head-direction** (direction)
  Are the note heads left or right in a semitie?

- **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.

**Internal properties:**

- **positioning-done** (boolean)
  Used to signal that a positioning element did its job. This ensures that a positioning is only done once.

This grob interface is used in the following graphical object(s): Section 3.1.51 [LaissezVibrerTieColumn], page 297 and Section 3.1.83 [RepeatTieColumn], page 321.

### 3.2.85 semi-tie-interface

A tie which is only on one side connected to a note head.

**User settable properties:**

- **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.

- **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.
**Chapter 3: Backend**

**3.2.86 separation-item-interface**

Item that computes widths to generate spacing rods.

**User settable properties:**

- **X-extent** (pair of numbers)
  Hard coded extent in X direction.

- **horizontal-skylines** (unknown)
  Two skylines, one to the left and one to the right of this grob.

- **padding** (dimension, in staff space)
  Add this much extra space between objects that are next to each other.

**Internal properties:**

- **conditional-elements** (unknown)
  Internal use only.

- **elements** (unknown)
  A list of grobs; the type is depending on the grob where this is set in.

This grob interface is used in the following graphical object(s): Section 3.1.66 [NonMusicalPaperColumn], page 308, Section 3.1.68 [NoteColumn], page 310, Section 3.1.74 [PaperColumn], page 314 and Section 3.1.89 [SeparationItem], page 323.

**3.2.87 side-position-interface**

Position a victim object (this one) next to other objects (the support). The property **direction** signifies where to put the victim object relative to the support (left or right, up or down?)

The routine also takes the size of the staff into account if **staff-padding** is set. If undefined, the staff symbol is ignored.

**head-direction** (direction)
Are the note heads left or right in a semitie?

**thickness** (number)
Line thickness, generally measured in **line-thickness**.

**Internal properties:**

- **details** (list)
  A list of parameters for detailed grob behavior.

  More information on the allowed parameters can be found by inspecting 'lily/slur-scoring.cc', 'lily/beam-quanting.cc', and 'lily/tie-formatting-problem.cc'. Setting **debug-tie-scoring**, **debug-beam-scoring** or **debug-slur-scoring** also provides useful clues.

- **note-head** (layout object)
  A single note head.

This grob interface is used in the following graphical object(s): Section 3.1.50 [LaissezVibrerTie], page 297 and Section 3.1.82 [RepeatTie], page 320.
User settable properties:

- **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`.

- **minimum-space** (dimension, in staff space)
  
  Minimum distance that the victim should move (after padding).

- **padding** (dimension, in staff space)
  
  Add this much extra space between objects that are next to each other.

- **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.

- **slur-padding** (number)
  
  Extra distance between slur and script.

- **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.

Internal properties:

- **direction-source** (layout object)
  
  In case `side-relative-direction` is set, which grob to get the direction from.

- **quantize-position** (boolean)
  
  If set, a vertical alignment is aligned to be within staff spaces.

- **side-support-elements** (unknown)
  
  The side support, a list of grobs.

This grob interface is used in the following graphical object(s): Section 3.1.4 [Accidental-Suggestion], page 259, Section 3.1.6 [AmbitusAccidental], page 261, Section 3.1.9 [Arpeggio], page 263, Section 3.1.12 [BarNumber], page 266, Section 3.1.15 [BassFigureAlignmentPositioning], page 268, Section 3.1.28 [CombineTextScript], page 277, Section 3.1.33 [DoublePercentRepeatCounter], page 281, Section 3.1.34 [DynamicLineSpanner], page 282, Section 3.1.37 [Fingering], page 285, Section 3.1.45 [HorizontalBracket], page 292, Section 3.1.46 [InstrumentName], page 293, Section 3.1.47 [InstrumentSwitch], page 294, Section 3.1.59 [MeasureGrouping], page 303, Section 3.1.62 [MetronomeMark], page 304, Section 3.1.64 [MultiMeasureRestNumber], page 306, Section 3.1.65 [MultiMeasureRestText], page 307, Section 3.1.72 [OctavateEight], page 311, Section 3.1.73 [OttavaBracket], page 312, Section 3.1.77 [PercentRepeatCounter], page 315, Section 3.1.80 [RehearsalMark], page 318, Section 3.1.86 [Script], page 322, Section 3.1.92 [SostenutoPedalLineSpanner], page 326, Section 3.1.97 [StanzaNumber], page 330, Section 3.1.100 [StringNumber], page 332, Section 3.1.101 [StrokeFinger], page 334, Section 3.1.103 [SustainPedalLineSpanner], page 335, Section 3.1.105 [SystemStartBar], page 337, Section 3.1.106 [SystemStartBrace], page 338, Section 3.1.107 [SystemStartBracket], page 339, Section 3.1.108 [SystemStartSquare], page 340, Section 3.1.110 [TextScript],...
3.2.88 slur-interface

A slur.

User settable properties:

- **annotation** (string)
  Annotate a grob for debug purposes.

- **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.

- **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.

- **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.

- **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.

- **eccentricity** (number)
  How asymmetrical to make a slur. Positive means move the center to the right.

- **height-limit** (dimension, in staff space)
  Maximum slur height: The longer the slur, the closer it is to this height.

- **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.

- **line-thickness** (number)
  The thickness of the tie or slur contour.

- **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.

**ratio** (number)
Parameter for slur shape. The higher this number, the quicker the slur attains its *height-limit*.

**thickness** (number)
Line thickness, generally measured in *line-thickness*.

### Internal properties:

**details** (list)
A list of parameters for detailed grob behavior.

More information on the allowed parameters can be found by inspecting ‘lily/slur-scoring.cc’, ‘lily/beam-quanting.cc’, and ‘lily/tie-formatting-problem.cc’. Setting debug-tie-scoring, debug-beam-scoring or debug-slur-scoring also provides useful clues.

**encompass-objects** (unknown)
Objects that a slur should avoid in addition to notes and stems.

**note-columns** (pair)
A list of `NoteColumn` grobs.

**quant-score** (string)
The beam quanting score; stored for debugging.

This grob interface is used in the following graphical object(s): Section 3.1.78 [PhrasingSlur], page 317 and Section 3.1.90 [Slur], page 324.

### 3.2.89 spaceable-grob-interface

A layout object that takes part in the spacing problem.

### User settable properties:

**allow-loose-spacing** (boolean)
If set, column can be detached from main spacing.

**keep-inside-line** (boolean)
If set, this column cannot have objects sticking into the margin.

**measure-length** (moment)
Length of a measure. Used in some spacing situations.

### Internal properties:

**ideal-distances** (list)

- `(obj . (dist . strength))` pairs.

**left-neighbors** (unknown)
A list of `spacing-wishes` grobs that are close to the current column. The closest `spacing-wishes` determine the actual distances between the columns.

**minimum-distances** (list)
A list of rods that have the format `(obj . dist)`. 
right-neighbors (unknown)
See left-neighbors.

spacing-wishes (unknown)
List of note spacing or staff spacing objects.

This grob interface is used in the following graphical object(s): Section 3.1.66 [NonMusical-PaperColumn], page 308 and Section 3.1.74 [PaperColumn], page 314.

3.2.90 spacing-interface
This object calculates the desired and minimum distances between two columns.

**Internal properties:**

left-items (unknown)
DOCME
	right-items (unknown)
DOCME

This grob interface is used in the following graphical object(s): Section 3.1.71 [NoteSpacing], page 311 and Section 3.1.95 [StaffSpacing], page 329.

3.2.91 spacing-options-interface
Supports setting of spacing variables.

**User settable properties:**

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 Section “spacing-spanner-interface” in Internals Reference.

spacing-increment (number)
Add this much space for a doubled duration. Typically, the width of a note head. See also Section “spacing-spanner-interface” in Internals Reference.

This grob interface is used in the following graphical object(s): Section 3.1.40 [GraceSpacing], page 289 and Section 3.1.93 [SpacingSpanner], page 327.

3.2.92 spacing-spanner-interface
The space taken by a note is dependent on its duration. Doubling a duration adds spacing-increment to the space. The most common shortest note gets shortest-duration-space. Notes that are even shorter are spaced proportionally to their duration.

Typically, the increment is the width of a black note head. In a piece with lots of 8th notes, and some 16th notes, the eighth note gets a 2 note heads width (i.e., the space following a note is a 1 note head width). A 16th note is followed by 0.5 note head width. The quarter note is followed by 3 NHW, the half by 4 NHW, etc.

**User settable properties:**

average-spacing-wishes (boolean)
If set, the spacing wishes are averaged over staves.

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.
common-shortest-duration (moment)
The most common shortest note length. This is used in spacing. Enlarging this sets the score tighter.

packed-spacing (boolean)
If set, the notes are spaced as tightly as possible.

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 Section “spacing-spanner-interface” in Internals Reference.

spacing-increment (number)
Add this much space for a doubled duration. Typically, the width of a note head. See also Section “spacing-spanner-interface” in Internals Reference.

strict-grace-spacing (boolean)
If set, main notes are spaced normally, then grace notes are put left of the musical columns for 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.

uniform-stretching (boolean)
If set, items stretch proportionally to their durations. This looks better in complex polyphonic patterns.

This grob interface is used in the following graphical object(s): Section 3.1.93 [SpacingSpanner], page 327.

3.2.93 span-bar-interface
A bar line that is spanned between other barlines. This interface is used for bar lines that connect different staves.

Internal properties:

  elements (unknown)
  A list of grobs; the type is depending on the grob where this is set in.

  glyph-name (string)
  The glyph name within the font.

This grob interface is used in the following graphical object(s): Section 3.1.94 [SpanBar], page 328.

3.2.94 spanner-interface
Some objects are horizontally spanned between objects. For example, slurs, beams, ties, etc. These grobs form a subtype called Spanner. All spanners have two span points (these must be Item objects), one on the left and one on the right. The left bound is also the X reference point of the spanner.

User settable properties:

  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.

**to-barline** (boolean)

If true, the spanner will stop at the bar line just before it would otherwise stop.

This grob interface is used in the following graphical object(s):
- Section 3.1.14 [BassFigureAlignment], page 268
- Section 3.1.15 [BassFigureAlignmentPositioning], page 268
- Section 3.1.17 [BassFigureContinuation], page 269
- Section 3.1.18 [BassFigureLine], page 270
- Section 3.1.19 [Beam], page 270
- Section 3.1.20 [BendAfter], page 271
- Section 3.1.26 [ClusterSpanner], page 276
- Section 3.1.34 [DynamicLineSpanner], page 282
- Section 3.1.36 [DynamicTextSpanner], page 284
- Section 3.1.39 [Glissando], page 288
- Section 3.1.40 [GraceSpacing], page 289
- Section 3.1.43 [Hairpin], page 290
- Section 3.1.45 [HorizontalBracket], page 292
- Section 3.1.46 [InstrumentName], page 293
- Section 3.1.52 [LedgerLineSpanner], page 298
- Section 3.1.54 [LigatureBracket], page 299
- Section 3.1.55 [LyricExtender], page 300
- Section 3.1.56 [LyricHyphen], page 301
- Section 3.1.57 [LyricSpace], page 302
- Section 3.1.59 [MeasureGrouping], page 303
- Section 3.1.61 [MensuralLigature], page 304
- Section 3.1.63 [MultiMeasureRest], page 305
- Section 3.1.64 [MultiMeasureRestNumber], page 306
- Section 3.1.65 [MultiMeasureRestText], page 307
- Section 3.1.73 [OttavaBracket], page 312
- Section 3.1.76 [PercentRepeat], page 315
- Section 3.1.77 [PercentRepeatCounter], page 315
- Section 3.1.78 [PhrasingSlur], page 317
- Section 3.1.79 [PianoPedalBracket], page 318
- Section 3.1.90 [Slur], page 324
- Section 3.1.92 [SostenutoPedalLineSpanner], page 326
- Section 3.1.93 [SpacingSpanner], page 327
- Section 3.1.96 [StaffSymbol], page 329
- Section 3.1.103 [SustainPedalLineSpanner], page 335
- Section 3.1.104 [System], page 337
- Section 3.1.105 [SystemStartBar], page 337
- Section 3.1.106 [SystemStartBrace], page 338
- Section 3.1.107 [SystemStartBracket], page 339
- Section 3.1.108 [SystemStartSquare], page 340
- Section 3.1.111 [TextSpanner], page 343
- Section 3.1.112 [Tie], page 344
- Section 3.1.113 [TieColumn], page 345
- Section 3.1.118 [TrillSpanner], page 349
- Section 3.1.119 [TupletBracket], page 350
- Section 3.1.120 [TupletNumber], page 351
- Section 3.1.122 [UnaCordaPedalLineSpanner], page 353
- Section 3.1.123 [VaticanaLigature], page 354
- Section 3.1.124 [VerticalAlignment], page 354
- Section 3.1.125 [VerticalAxisGroup], page 355
- Section 3.1.126 [VoiceFollower], page 356
- Section 3.1.127 [VoltaBracket], page 356
- Section 3.1.128 [VoltaBracketSpanner], page 357

### 3.2.95 staff-spacing-interface

This object calculates spacing details from a breakable symbol (left) to another object. For example, it takes care of optical spacing from a bar line to a note.

**User settable properties:**

- **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.

This grob interface is used in the following graphical object(s):
- Section 3.1.95 [StaffSpacing], page 329

### 3.2.96 staff-symbol-interface

This spanner draws the lines of a staff. A staff symbol defines a vertical unit, the **staff space**. Quantities that go by a half staff space are called **positions**. The center (i.e., middle line or space) is position 0. The length of the symbol may be set by hand through the **width** property.
User settable properties:

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.

line-count (integer)
The number of staff lines.

line-positions (list)
Vertical positions of staff lines.

staff-space (dimension, in staff space)
Amount of space between staff lines, expressed in global staff-space.

thickness (number)
Line thickness, generally measured in line-thickness.

width (dimension, in staff space)
The width of a grob measured in staff space.

This grob interface is used in the following graphical object(s): Section 3.1.96 [StaffSymbol], page 329.

3.2.97 staff-symbol-referencer-interface
An object whose Y position is meant relative to a staff symbol. These usually have Staff_symbol_referencer::callback in their Y-offset-callbacks.

User settable properties:

staff-position (number)
Vertical position, measured in half staff spaces, counted from the middle line.

This grob interface is used in the following graphical object(s): Section 3.1.7 [AmbitusLine], page 262, Section 3.1.8 [AmbitusNoteHead], page 262, Section 3.1.9 [Arpeggio], page 263, Section 3.1.19 [Beam], page 270, Section 3.1.25 [Clef], page 275, Section 3.1.29 [Custos], page 278, Section 3.1.31 [Dots], page 279, Section 3.1.48 [KeyCancellation], page 295, Section 3.1.49 [KeySignature], page 296, Section 3.1.63 [MultiMeasureRest], page 305, Section 3.1.69 [NoteHead], page 310, Section 3.1.84 [Rest], page 321, Section 3.1.109 [TabNoteHead], page 340 and Section 3.1.117 [TrillPitchHead], page 348.

3.2.98 stanza-number-interface
A stanza number, to be put in from of a lyrics line.

This grob interface is used in the following graphical object(s): Section 3.1.97 [StanzaNumber], page 330.

3.2.99 stem-interface
The stem represents the graphical stem. In addition, it internally connects note heads, beams, and tremolos. Rests and whole notes have invisible stems.

The following properties may be set in the details list.

beamed-lengths
List of stem lengths given beam multiplicity.

beamed-minimum-free-lengths
List of normal minimum free stem lengths (chord to beams) given beam multiplicity.
beamed-extreme-minimum-free-lengths

List of extreme minimum free stem lengths (chord to beams) given beam multiplicity.

lengths

Default stem lengths. The list gives a length for each flag count.

stem-shorten

How much a stem in a forced direction should be shortened. The list gives an amount depending on the number of flags and beams.

User settable properties:

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.

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.

beamlet-default-length (pair)

A pair of numbers. The first number specifies the default length of a beamlet that sticks out of the left hand side of this stem; the second number specifies the default length of the beamlet to the right. The actual length of a beamlet is determined by taking either the default length or the length specified by beamlet-max-length-proportion, whichever is smaller.

beamlet-max-length-proportion (pair)

The maximum length of a beamlet, as a proportion of the distance between two adjacent stems.

default-direction (direction)

Direction determined by note head positions.

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.

duration-log (integer)

The 2-log of the note head duration, i.e., 0 = whole note, 1 = half note, etc.

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-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.
french-beaming (boolean)

Use French beaming style for this stem. The stem stops at the innermost beams.

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.

max-beam-connect (integer)

Maximum number of beams to connect to beams from this stem. Further beams are typeset as beamlets.

neutral-direction (direction)

Which direction to take in the center of the staff.

no-stem-extend (boolean)

If set, notes with ledger lines do not get stems extending to the middle staff line.

stem-end-position (number)

Where does the stem end (the end is opposite to the support-head)?

stemlet-length (number)

How long should be a stem over a rest?

stroke-style (string)

Set to "grace" to turn stroke through flag on.

thickness (number)

Line thickness, generally measured in line-thickness.

**Internal properties:**

beam (layout object)

A pointer to the beam, if applicable.

details (list)

A list of parameters for detailed grob behavior.

More information on the allowed parameters can be found by inspecting ‘lily/slur-scoring.cc’, ‘lily/beam-quanting.cc’, and ‘lily/tie-formatting-problem.cc’. Setting debug-tie-scoring, debug-beam-scoring or debug-slur-scoring also provides useful clues.

note-heads (unknown)

A list of note head grobs.

positioning-done (boolean)

Used to signal that a positioning element did its job. This ensures that a positioning is only done once.

rests (unknown)

A list of rest objects.

stem-info (pair)

A cache of stem parameters.
tremolo-flag (layout object)
  The tremolo object on a stem.

This grob interface is used in the following graphical object(s): Section 3.1.98 [Stem], page 330.

3.2.100 stem-tremolo-interface
A beam slashing a stem to indicate a tremolo. The property style can be default or rectangle.

User settable properties:
  beam-thickness (dimension, in staff space)
    Beam thickness, measured in staff-space units.
  beam-width (dimension, in staff space)
    Width of the tremolo sign.
  flag-count (number)
    The number of tremolo beams.
  length-fraction (number)
    Multiplier for lengths. Used for determining ledger lines and stem lengths.
  slope (number)
    The slope of this object.
  style (symbol)
    This setting determines in what style a grob is typeset. Valid choices depend on the stencil callback reading this property.

Internal properties:
  stem (layout object)
    A pointer to a Stem object.

This grob interface is used in the following graphical object(s): Section 3.1.99 [StemTremolo], page 332.

3.2.101 string-number-interface
A string number instruction.

This grob interface is used in the following graphical object(s): Section 3.1.100 [StringNumber], page 332.

3.2.102 stroke-finger-interface
A right hand finger instruction.

User settable properties:
  digit-names (unknown)
    Names for string finger digits.

This grob interface is used in the following graphical object(s): Section 3.1.101 [StrokeFinger], page 334.

3.2.103 system-interface
This is the top-level object: Each object in a score ultimately has a System object as its X and Y parent.
User settable properties:

- **labels** (list)
  List of labels (symbols) placed on a column

- **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.

Internal properties:

- **all-elements** (unknown)
  A list of all grobs in this line. Its function is to protect objects from being garbage collected.

- **columns** (unknown)
  A list of grobs, typically containing PaperColumn or NoteColumn objects.

- **pure-Y-extent** (pair of numbers)
  The estimated height of a system.

- **skyline-distance** (number)
  The distance between this staff and the next one, as determined by a skyline algorithm.

- **spaceable-staves** (unknown)
  Objects to be spaced during page layout.

This grob interface is used in the following graphical object(s): Section 3.1.104 [System], page 337.

### 3.2.104 system-start-delimiter-interface

The brace, bracket or bar in front of the system. The following values for style are recognized:

- **bracket**
  A thick bracket, normally used to group similar instruments in a score. Default for StaffGroup. SystemStartBracket uses this style.

- **brace**
  A ‘piano style’ brace normally used for an instrument that uses two staves. The default style for GrandStaff. SystemStartBrace uses this style.

- **bar-line**
  A simple line between the staves in a score. Default for staves enclosed in << and >>. SystemStartBar uses this style.

- **line-bracket**
  A simple square, normally used for subgrouping instruments in a score. SystemStartSquare uses this style.

See also ‘input/regression/system-start-nesting.ly’.

User settable properties:

- **collapse-height** (dimension, in staff space)
  Minimum height of system start delimiter. If equal or smaller, the bracket/brace/line is removed.

- **style** (symbol)
  This setting determines in what style a grob is typeset. Valid choices depend on the stencil callback reading this property.
thickness (number)
Line thickness, generally measured in line-thickness.

This grob interface is used in the following graphical object(s): Section 3.1.105 [SystemStart-Bar], page 337, Section 3.1.106 [SystemStartBrace], page 338, Section 3.1.107 [SystemStartBracket], page 339 and Section 3.1.108 [SystemStartSquare], page 340.

3.2.105 system-start-text-interface
Text in front of the system.

User settable properties:

long-text (markup)
Text markup. See Section “Formatting text” in Notation Reference.

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.

text (markup)
Text markup. See Section “Formatting text” in Notation Reference.

This grob interface is used in the following graphical object(s): Section 3.1.46 [Instrument-Name], page 293.

3.2.106 tablature-interface
An interface for any notes set in a tablature staff.

This grob interface is not used in any graphical object.

3.2.107 text-interface
A Scheme markup text, see Section “Formatting text” in Notation Reference and Section “New markup command definition” in Notation Reference.

There are two important commands: ly:text-interface::print, which is a grob callback, and ly:text-interface::interpret-markup.

User settable properties:

baseline-skip (dimension, in staff space)
Distance between base lines of multiple lines of text.

text (markup)
Text markup. See Section “Formatting text” in Notation Reference.

text-direction (direction)
This controls the ordering of the words. The default RIGHT is for roman text. Arabic or Hebrew should use LEFT.

word-space (dimension, in staff space)
Space to insert between words in texts.

This grob interface is used in the following graphical object(s): Section 3.1.10 [Balloon-TextItem], page 264, Section 3.1.12 [BarNumber], page 266, Section 3.1.13 [BassFigure], page 267, Section 3.1.23 [BreathingSign], page 273, Section 3.1.24 [ChordName], page 274,
3.2.108 text-script-interface

An object that is put above or below a note.

**User settable properties:**

- **add-stem-support** (boolean)
  
  If set, the Stem object is included in this script’s support.

- **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.

- **script-priority** (number)
  
  A sorting key that determines in what order a script is within a stack of scripts.

**Internal properties:**

- **slur** (layout object)
  
  A pointer to a Slur object.

This grob interface is used in the following graphical object(s): Section 3.1.28 [CombineTextScript], page 277, Section 3.1.37 [Fingering], page 285, Section 3.1.100 [StringNumber], page 332, Section 3.1.101 [StrokeFinger], page 334, Section 3.1.102 [SustainPedal], page 335, Section 3.1.109 [TabNoteHead], page 340, Section 3.1.110 [TextScript], page 341, Section 3.1.120 [TupletNumber], page 351, Section 3.1.121 [UnaCordaPedal], page 352 and Section 3.1.127 [VoltaBracket], page 356.

3.2.109 tie-column-interface

Object that sets directions of multiple ties in a tied chord.

**User settable properties:**

- **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.

**Internal properties:**

- **positioning-done** (boolean)
  
  Used to signal that a positioning element did its job. This ensures that a positioning is only done once.
This grob interface is used in the following graphical object(s): Section 3.1.13 [TieColumn], page 345.

### 3.2.110 tie-interface

A horizontal curve connecting two noteheads.

#### User settable properties:

- **annotation** (string)
  Annotate a grob for debug purposes.

- **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.

- **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.

- **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.

- **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.

- **head-direction** (direction)
  Are the note heads left or right in a semitie?

- **line-thickness** (number)
  The thickness of the tie or slur contour.

- **neutral-direction** (direction)
  Which direction to take in the center of the staff.

- **staff-position** (number)
  Vertical position, measured in half staff spaces, counted from the middle line.

- **thickness** (number)
  Line thickness, generally measured in **line-thickness**.

#### Internal properties:

- **details** (list)
  A list of parameters for detailed grob behavior.

  More information on the allowed parameters can be found by inspecting ‘lily/slur-scoring.cc’, ‘lily/beam-quanting.cc’, and
lily/tie-formatting-problem.cc. Setting debug-tie-scoring, debug-beam-scoring or debug-slur-scoring also provides useful clues.

- **quant-score** (string)
  - The beam quanting score; stored for debugging.

- **separation-item** (layout object)
  - A separation item.

This grob interface is used in the following graphical object(s): Section 3.1.112 [Tie], page 344.

### 3.2.111 time-signature-interface

A time signature, in different styles. The following values for **style** are recognized:

- **C**
  - 4/4 and 2/2 are typeset as C and struck C, respectively. All other time signatures are written with two digits. The value **default** is equivalent to C.

- **neomensual**
  - 2/2, 3/2, 2/4, 3/4, 4/4, 6/4, 9/4, 4/8, 6/8, and 9/8 are typeset with neo-mensural style mensuration marks. All other time signatures are written with two digits.

- **mensural**
  - 2/2, 3/2, 2/4, 3/4, 4/4, 6/4, 9/4, 4/8, 6/8, and 9/8 are typeset with mensural style mensuration marks. All other time signatures are written with two digits.

- **single-digit**
  - All time signatures are typeset with a single digit, e.g., 3/2 is written as 3.

- **numbered**
  - All time signatures are typeset with two digits.

**User settable properties:**

- **fraction** (pair of numbers)
  - Numerator and denominator of a time signature object.

- **style** (symbol)
  - This setting determines in what style a grob is typeset. Valid choices depend on the **stencil** callback reading this property.

This grob interface is used in the following graphical object(s): Section 3.1.114 [TimeSignature], page 345.

### 3.2.112 trill-pitch-accidental-interface

An accidental for trill pitch.

This grob interface is used in the following graphical object(s): Section 3.1.115 [TrillPitchAccidental], page 346.

### 3.2.113 trill-spanner-interface

A trill spanner.

This grob interface is used in the following graphical object(s): Section 3.1.118 [TrillSpanner], page 349.

### 3.2.114 tuplet-bracket-interface

A bracket with a number in the middle, used for tuplets. When the bracket spans a line break, the value of **break-overshoot** determines how far it extends beyond the staff. At a line break, the markups in the **edge-text** are printed at the edges.
User settable properties:

- **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-overshoot** (pair of numbers)
  How much does a broken spanner stick out of its bounds?

- **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.

- **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.

- **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).

- **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.

- **gap** (dimension, in staff space)
  Size of a gap in a variable symbol.

- **padding** (dimension, in staff space)
  Add this much extra space between objects that are next to each other.

- **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.

- **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.
**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.

**thickness** (number)
Line thickness, generally measured in **line-thickness**.

**Internal properties:**

- **note-columns** (pair)
  A list of NoteColumn grobs.

- **tuplet-number** (layout object)
  The number for a bracket.

- **tuplets** (unknown)
  A list of smaller tuplet brackets.

This grob interface is used in the following graphical object(s): Section 3.1.54 [Ligature-Bracket], page 299 and Section 3.1.119 [TupletBracket], page 350.

### 3.2.115 tuplet-number-interface

The number for a bracket.

**User settable properties:**

- **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.

**Internal properties:**

- **bracket** (layout object)
  The bracket for a number.

This grob interface is used in the following graphical object(s): Section 3.1.120 [TupletNumber], page 351.

### 3.2.116 unbreakable-spanner-interface

A spanner that should not be broken across line breaks. Override with **breakable=#t**.

**User settable properties:**

- **breakable** (boolean)
  Allow breaks here.

This grob interface is used in the following graphical object(s): Section 3.1.19 [Beam], page 270 and Section 3.1.39 [Glissando], page 288.

### 3.2.117 vaticana-ligature-interface

A vaticana style Gregorian ligature.

**User settable properties:**

- **thickness** (number)
  Line thickness, generally measured in **line-thickness**.
Internal properties:

add-cauda (boolean)
Does this flexa require an additional cauda on the left side?

add-join (boolean)
Is this ligature head-joined with the next one by a vertical line?

add-stem (boolean)
Is this ligature head a virga and therefore needs an additional stem on the right side?

delta-position (number)
The vertical position difference.

flexa-height (dimension, in staff space)
The height of a flexa shape in a ligature grob (in staff-space units).

flexa-width (dimension, in staff space)
The width of a flexa shape in a ligature grob in (in staff-space units).

glyph-name (string)
The glyph name within the font.

x-offset (dimension, in staff space)
Extra horizontal offset for ligature heads.

This grob interface is used in the following graphical object(s): Section 3.1.69 [NoteHead], page 310 and Section 3.1.123 [VaticanaLigature], page 354.

3.2.118 vertically-spaceable-interface

Objects that should be kept at constant vertical distances. Typically: Section “VerticalAxis-Group” in Internals Reference objects of Section “Staff” in Internals Reference contexts.

This grob interface is used in the following graphical object(s): Section 3.1.125 [VerticalAxisGroup], page 355.

3.2.119 volta-bracket-interface

Volta bracket with number.

User settable properties:

height (dimension, in staff space)
Height of an object in staff-space units.

thickness (number)
Line thickness, generally measured in line-thickness.

Internal properties:

bars (unknown)
A list of bar line pointers.

This grob interface is used in the following graphical object(s): Section 3.1.127 [VoltaBracket], page 356.
3.3 User backend properties

**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.

beamlet-default-length (pair) A pair of numbers. The first number specifies the default length of a beamlet that sticks out of the left hand side of this stem; the second number specifies the default length of the beamlet to the right. The actual length of a beamlet is determined by taking either the default length or the length specified by beamlet-max-length-proportion, whichever is smaller.

beamlet-max-length-proportion (pair) The maximum length of a beamlet, as a proportion of the distance between two adjacent stems.

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.
conca
cveness (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.

dge-height (pair)
A pair of numbers specifying the heights of the vertical edges: (left-height .
right-height).

dge-text (pair)
A pair specifying the texts to be set at the edges: (left-text . right-text).

dexpnd-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 Section “note-collision-interface” in Internals Reference.

fraction (pair of numbers)
Numerator and denominator of a time signature object.

defrench-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-position – Location of dot in fret space. Default 0.6 for dots without labels, 0.95-dot-radius for dots with labels.
- 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-label-font-mag – The magnification of the font used to label the lowest fret number. Default 0.5.
- fret-label-vertical-offset – The offset of the fret label from the center of the fret in direction parallel to strings. Default 0.
- 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, landscape, and opposing-landscape. Default normal.
• **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 for normal orientation, 0.5 for landscape and opposing-landscape.

• **string-thickness-factor** – Factor for changing thickness of each string in the fret diagram. Thickness of string $k$ is given by $\text{thickness} \times (1 + \text{string-thickness-factor})^{(k-1)}$. Default 0.

• **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-measure-extra-space** (number)
Extra space that is allocated at the beginning of a measure with only one note. This property is read from the NonMusicalPaperColumn that begins the measure.

**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 NoteColumns for horizontal shifting. This is used by Section “note-collision-interface” in Internals Reference.

**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 Section “Formatting text” in Notation Reference.

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 Section “note-collision-interface” in Internals Reference.

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 Section “spacing-spanner-interface” in Internals Reference.

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.
slop (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 Section “spacing-spanner-interface” in Internals Reference.

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 be a stem over a rest?

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 for 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 Section “Formatting text” in Notation Reference.

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.

### 3.4 Internal backend properties

**X-common** (layout object)
Common reference point for axis group.

**Y-common** (layout object)
See X-common.

**accidental-grob** (layout object)
The accidental for this note.

**accidental-grobs** (list)
An alist with (notename . groblist) entries.

**add-cauda** (boolean)
Does this flexa require an additional cauda on the left side?

**add-join** (boolean)
Is this ligature head-joined with the next one by a vertical line?

**add-stem** (boolean)
Is this ligature head a virga and therefore needs an additional stem on the right side?

**adjacent-hairpins** (unknown)
A list of directly neighboring hairpins.

**adjacent-pure-heights** (vector)
Used by a VerticalAxisGroup to cache the Y-extents of different column ranges.

**all-elements** (unknown)
A list of all grobs in this line. Its function is to protect objects from being garbage collected.

**arpeggio** (layout object)
A pointer to an Arpeggio object.

**ascendens** (boolean)
Is this neume of ascending type?

**auctum** (boolean)
Is this neume liquescentically augmented?

**axis-group-parent-X** (layout object)
Containing X axis group.

**axis-group-parent-Y** (layout object)
Containing Y axis group.

**bar-extent** (pair of numbers)
The Y-extent of the actual bar line. This may differ from Y-extent because it does not include the dots in a repeat bar line.
bars (unknown)
A list of bar line pointers.

beam (layout object)
A pointer to the beam, if applicable.

begin-of-line-visible (boolean)
Used for marking ChordNames that should only show changes.

bounded-by-me (unknown)
A list of spanners that have this column as start/begin point. Only columns that have grobs or act as bounds are spaced.

bracket (layout object)
The bracket for a number.

cause (any type)
Any kind of causation objects (i.e., music, or perhaps translator) that was the cause for this grob.

cavum (boolean)
Is this neume outlined?

columns (unknown)
A list of grobs, typically containing PaperColumn or NoteColumn objects.

conditional-elements (unknown)
Internal use only.

context-info (integer)
Within a ligature, the final glyph or shape of a head may be affected by the left and/or right neighbour head. context-info holds for each head such information about the left and right neighbour, encoded as a bit mask.

cross-staff (boolean)
For a beam or a stem, this is true if we depend on inter-staff spacing.

delta-position (number)
The vertical position difference.

deminutum (boolean)
Is this neume diminished?

descendens (boolean)
Is this neume of descendent type?

details (list)
A list of parameters for detailed grob behavior.
More information on the allowed parameters can be found by inspecting ‘lily/slur-scoring.cc’, ‘lily/beam-quanting.cc’, and ‘lily/tie-formatting-problem.cc’. Setting debug-tie-scoring, debug-beam-scoring or debug-slur-scoring also provides useful clues.

direction-source (layout object)
In case side-relative-direction is set, which grob to get the direction from.

dot (layout object)
A reference to a Dots object.

dots (unknown)
Multiple Dots objects.
elements (unknown)
A list of grobs; the type is depending on the grob where this is set in.

encompass-objects (unknown)
Objects that a slur should avoid in addition to notes and stems.

figures (unknown)
Figured bass objects for continuation line.

flexa-height (dimension, in staff space)
The height of a flexa shape in a ligature grob (in \texttt{staff-space} units).

flexa-width (dimension, in staff space)
The width of a flexa shape in a ligature grob in (in \texttt{staff-space} units).

font (font metric)
A cached font metric object.

forced (boolean)
Manually forced accidental.

glyph-name (string)
The glyph name within the font.

grace-spacing (layout object)
A run of grace notes.

head-width (dimension, in staff space)
The width of this ligature head.

heads (unknown)
A list of note heads.

ideal-distances (list)
(obj . (dist . strength)) pairs.

important-column-ranks (vector)
A cache of columns that contain \texttt{items-worth-living} data.

inclinatum (boolean)
Is this neume an inclinatum?

interfaces (list)
A list of symbols indicating the interfaces supported by this object. It is initialized from the \texttt{meta} field.

items-worth-living (unknown)
A list of interesting items. If empty in a particular staff, then that staff is erased.

join-heads (boolean)
Whether to join the note heads of an ambitus grob with a vertical line.

join-right-amount (number)
DOCME

least-squares-dy (number)
The ideal beam slope, without damping.

left-items (unknown)
DOCME

left-neighbors (unknown)
A list of \texttt{spacing-wishes} grobs that are close to the current column.
The closest \texttt{spacing-wishes} determine the actual distances between the columns.
linea (boolean)
   Attach vertical lines to this neume?

meta (list)  Provide meta information. It is an alist with the entries name and interfaces.

minimum-distances (list)
   A list of rods that have the format (obj . dist).

normal-stems (unknown)
   An array of visible stems.

note-columns (pair)
   A list of NoteColumn grobs.

note-head (layout object)
   A single note head.

note-heads (unknown)
   A list of note head grobs.

oriscus (boolean)
   Is this neume an oriscus?

pedal-text (layout object)
   A pointer to the text of a mixed-style piano pedal.

pes-or-flexa (boolean)
   Shall this neume be joined with the previous head?

positioning-done (boolean)
   Used to signal that a positioning element did its job. This ensures that a positioning
   is only done once.

prefix-set (number)
   A bit mask that holds all Gregorian head prefixes, such as \virga or \quilisma.

primitive (integer)
   A pointer to a ligature primitive, i.e., an item similar to a note head that is part of
   a ligature.

pure-Y-common (layout object)
   A cache of the common_refpoint_of_array of the elements grob set.

pure-Y-extent (pair of numbers)
   The estimated height of a system.

pure-Y-offset-in-progress (boolean)
   A debugging aid for catching cyclic dependencies.

pure-relevant-items (unknown)
   A subset of elements that are relevant for finding the pure-Y-extent.

pure-relevant-spanners (unknown)
   A subset of elements that are relevant for finding the pure-Y-extent.

quant-score (string)
   The beam quanting score; stored for debugging.

quantize-position (boolean)
   If set, a vertical alignment is aligned to be within staff spaces.

quantized-positions (pair of numbers)
   The beam positions after quanting.
quilisma (boolean)
    Is this neume a quilisma?
rest (layout object)
    A pointer to a Rest object.
rest-collision (layout object)
    A rest collision that a rest is in.
rests (unknown)
    A list of rest objects.
right-items (unknown)
    DOCME
right-neighbors (unknown)
    See left-neighbors.
script-stencil (pair)
    A pair (type . arg) which acts as an index for looking up a Stencil object.
separation-item (layout object)
    A separation item.
shorten (dimension, in staff space)
    The amount of space that a stem is shortened. Internally used to distribute beam shortening over stems.
side-support-elements (unknown)
    The side support, a list of grobs.
skyline-distance (number)
    The distance between this staff and the next one, as determined by a skyline algorithm.
slur (layout object)
    A pointer to a Slur object.
spaceable-staves (unknown)
    Objects to be spaced during page layout.
spacing (layout object)
    The spacing spanner governing this section.
spacing-wishes (unknown)
    List of note spacing or staff spacing objects.
staff-symbol (layout object)
    The staff symbol grob that we are in.
stem (layout object)
    A pointer to a Stem object.
stem-info (pair)
    A cache of stem parameters.
stems (unknown)
    A list of stem objects, corresponding to the notes that the arpeggio has to be before.
stropha (boolean)
    Is this neume a stropha?
tie (layout object)
    A pointer to a Tie object.
tremolo-flag (layout object)
The tremolo object on a stem.

tuplet-number (layout object)
The number for a bracket.

tuplets (unknown)
A list of smaller tuplet brackets.

use-breve-rest (boolean)
Use breve rests for measures longer than a whole rest.

virga (boolean)
Is this neume a virga?

x-offset (dimension, in staff space)
Extra horizontal offset for ligature heads.
4 Scheme functions

dispatcher \( x \)  
Is \( x \) a Dispatcher object?  

listener \( x \)  
Is \( x \) a Listener object?  

ly:add-file-name-alist \( \text{alist} \)  
Add mappings for error messages from \( \text{alist} \).  

ly:add-interface \( a \ b \ c \)  
Add an interface description.  

ly:add-listener \( \text{list} \ \text{disp} \ \text{cl} \)  
Add the listener \( \text{list} \) to the dispatcher \( \text{disp} \). Whenever \( \text{disp} \) hears an event of class \( \text{cl} \), it is forwarded to \( \text{list} \).  

ly:add-option \( \text{sym} \ \text{val} \ \text{description} \)  
Add a program option \( \text{sym} \) with default \( \text{val} \).  

ly:all-grob-interfaces  
Get a hash table with all interface descriptions.  

ly:all-options  
Get all option settings in an alist.  

ly:all-stencil-expressions  
Return all symbols recognized as stencil expressions.  

ly:assoc-get \( \text{key} \ \text{alist} \ \text{default-value} \)  
Return value if \( \text{key} \) in \( \text{alist} \), else \text{default-value} \ (or \#f if not specified).  

ly:book-add-bookpart! \( \text{book-smob} \ \text{book-part} \)  
Add \text{book_part} to \text{book-smob} book part list.  

ly:book-add-score! \( \text{book-smob} \ \text{score} \)  
Add score to \text{book-smob} score list.  

ly:book-process \( \text{book-smob} \ \text{default-paper} \ \text{default-layout} \ \text{output} \)  
Print book. \text{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 \( \text{book-smob} \ \text{default-paper} \ \text{default-layout} \ \text{output} \)  
Print book. \text{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? \( x \)  
Is \( x \) a Box object?  

ly:bp \( \text{num} \)  
\( \text{num} \) bigpoints (1/72th inch).  

ly:bracket \( a \ iv \ t \ p \)  
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  
Send the stream event ev to the dispatcher disp.

**ly:camel-case->lisp-identifier**  
namen-sym  
Convert FooBar_Bla to foo-bar-bla style symbol.

**ly:chain-assoc-get**  
key achain dfault  
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**  
Plug a GUILE 1.6 and 1.7 memory leak by breaking a weak reference pointer cycle explicitly.

**ly:cm num**  
num cm.  
The Scheme code specified on command-line with ‘-e’.

**ly:connect-dispatchers**  
to from  
Make the dispatcher to listen to events from from.

**ly:context-event-source**  
context  
Return event-source of context context.

**ly:context-events-below**  
context  
Return a stream-distributor that distributes all events from context and all its subcontexts.

**ly:context-find**  
context name  
Find a parent of context that has name or alias name. Return #f if not found.

**ly:context-grob-definition**  
context name  
Return the definition of name (a symbol) within context as an alist.

**ly:context-id**  
context  
Return the ID string of context, i.e., for \context Voice = one ... return the string one.

**ly:context-name**  
context  
Return the name of context, i.e., for \context Voice = one ... return the symbol Voice.

**ly:context-now**  
context  
Return now-moment of context context.

**ly:context-parent**  
context  
Return the parent of context, #f if none.

**ly:context-property**  
c name  
Return the value of name from context c.

**ly:context-property-where-defined**  
context name  
Return the context above context where name is defined.
ly:context-pushpop-property  
\textit{context grob eltprop val}  
\begin{itemize}
\item Do a single \texttt{override} or \texttt{revert} operation in \textit{context}. The grob definition \textit{grob} is extended with \textit{eltprop} (if \textit{val} is specified) or reverted (if unspecified).
\end{itemize}

ly:context-set-property!  
\textit{context name val}  
\begin{itemize}
\item Set value of property \textit{name} in context \textit{context} to \textit{val}.
\end{itemize}

ly:context-unset-property  
\textit{context name}  
\begin{itemize}
\item Unset value of property \textit{name} in context \textit{context}.
\end{itemize}

ly:context?  
\textit{x}  
\begin{itemize}
\item Is \textit{x} a \texttt{Context} object?
\end{itemize}

ly:default-scale  
\begin{itemize}
\item Get the global default scale.
\end{itemize}

ly:dimension?  
\textit{d}  
\begin{itemize}
\item Return \textit{d} as a number. Used to distinguish length variables from normal numbers.
\end{itemize}

ly:dir?  
\textit{s}  
\begin{itemize}
\item A type predicate. The direction \textit{s} is \texttt{-1}, \texttt{0} or \texttt{1}, where \texttt{-1} represents left or down and \texttt{1} represents right or up.
\end{itemize}

ly:duration->string  
\textit{dur}  
\begin{itemize}
\item Convert \textit{dur} to a string.
\end{itemize}

ly:duration-dot-count  
\textit{dur}  
\begin{itemize}
\item Extract the dot count from \textit{dur}.
\end{itemize}

ly:duration-factor  
\textit{dur}  
\begin{itemize}
\item Extract the compression factor from \textit{dur}. Return it as a pair.
\end{itemize}

ly:duration-length  
\textit{dur}  
\begin{itemize}
\item The length of the duration as a \texttt{moment}.
\end{itemize}

ly:duration-log  
\textit{dur}  
\begin{itemize}
\item Extract the duration log from \textit{dur}.
\end{itemize}

ly:duration<?  
\textit{p1 p2}  
\begin{itemize}
\item Is \textit{p1} shorter than \textit{p2}?
\end{itemize}

ly:duration?  
\textit{x}  
\begin{itemize}
\item Is \textit{x} a \texttt{Duration} object?
\end{itemize}

ly:effective-prefix  
\begin{itemize}
\item Return effective prefix.
\end{itemize}

ly:error  
\textit{str rest}  
\begin{itemize}
\item A Scheme callable function to issue the error \textit{str}. The error is formatted with \texttt{format} and \textit{rest}.
\end{itemize}

ly:eval-simple-closure  
\textit{delayed closure scm-start scm-end}  
\begin{itemize}
\item Evaluate a simple \texttt{closure} with the given \texttt{delayed} argument. If \textit{scm-start} and \textit{scm-end} are defined, evaluate it purely with those start and end points.
\end{itemize}

ly:event-deep-copy  
\textit{m}  
\begin{itemize}
\item Copy \textit{m} and all sub expressions of \textit{m}.
\end{itemize}
**Chapter 4: Scheme functions**

--

**ly:event-property** `sev sym`
Get the property `sym` of stream `mus`. If `sym` is undefined, return `'()`.

**ly:event-set-property!** `ev sym val`
Set property `sym` in event `ev` to `val`.

**ly:expand-environment** `str`
Expand `$VAR` and `$(VAR)` in `str`.

**ly:export** `arg`
Export a Scheme object to the parser so it is treated as an identifier.

**ly:find-file** `name`
Return the absolute file name of `name`, or `#f` if not found.

**ly:font-config-add-directory** `dir`
Add directory `dir` to FontConfig.

**ly:font-config-add-font** `font`
Add font `font` to FontConfig.

**ly:font-config-display-fonts**
Dump a list of all fonts visible to FontConfig.

**ly:font-config-get-font-file** `name`
Get the file for font `name`.

**ly:font-design-size** `font`
Given the font metric `font`, return the design size, relative to the current output-scale.

**ly:font-file-name** `font`
Given the font metric `font`, return the corresponding file name.

**ly:font-get-glyph** `font name`
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`
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`
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`
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.
Chapter 4: Scheme functions

ly:font-magnification font
Given the font metric font, return the magnification, relative to the current output-scale.

ly:font-metric? x
Is x a Font_metric object?

ly:font-name font
Given the font metric font, return the corresponding name.

ly:font-sub-fonts font
Given the font metric font of an OpenType font, return the names of the subfonts within font.

ly:format str rest
LilyPond specific format, supporting ~a and ~[0-9]f.

ly:format-output context
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
Get a hash table with all LilyPond Scheme extension functions.

ly:get-all-translators
Return a list of all translator objects that may be instantiated.

ly:get-glyph font index
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
Return a list of all event classes that some translator listens to.

ly:get-option var
Get a global option setting.

ly:gettext original
A Scheme wrapper function for gettext.

ly:grobalist-chain grob global
Get an alist chain for grob grob, with global as the global default. If unspecified, font-defaults from the layout block is taken.

ly:grobalist-len grob-arr
Return the length of grob-arr.

ly:grobalist-ref grob-arr index
Retrieve the indexth element of grob-arr.

ly:grobalist? x
Is x a Grob_array object?

ly:grobalist-prop grob
Get the immutable properties of grob.
\textbf{ly:grob-common-refpoint} \textit{grob other axis} \hspace{1cm} \textbf{[Function]}
Find the common refpoint of \textit{grob} and \textit{other} for \textit{axis}.

\textbf{ly:grob-common-refpoint-of-array} \textit{grob others axis} \hspace{1cm} \textbf{[Function]}
Find the common refpoint of \textit{grob} and \textit{others} (a grob-array) for \textit{axis}.

\textbf{ly:grob-default-font} \textit{grob} \hspace{1cm} \textbf{[Function]}
Return the default font for grob \textit{grob}.

\textbf{ly:grob-extent} \textit{grob refp axis} \hspace{1cm} \textbf{[Function]}
Get the extent in \textit{axis} direction of \textit{grob} relative to the grob \textit{refp}.

\textbf{ly:grob-interfaces} \textit{grob} \hspace{1cm} \textbf{[Function]}
Return the interfaces list of grob \textit{grob}.

\textbf{ly:grob-layout} \textit{grob} \hspace{1cm} \textbf{[Function]}
Get \texttt{/layout} definition from grob \textit{grob}.

\textbf{ly:grob-object} \textit{grob sym} \hspace{1cm} \textbf{[Function]}
Return the value of a pointer in grob \textit{gob} of property \textit{sym}. It returns \texttt{'}()\texttt{'} (end-of-list) if \textit{sym} is undefined in \textit{gob}.

\textbf{ly:grob-original} \textit{grob} \hspace{1cm} \textbf{[Function]}
Return the unbroken original grob of \textit{grob}.

\textbf{ly:grob-parent} \textit{grob axis} \hspace{1cm} \textbf{[Function]}
Get the parent of \textit{grob}. \textit{axis} is 0 for the X-axis, 1 for the Y-axis.

\textbf{ly:grob-pq <?} \textit{a b} \hspace{1cm} \textbf{[Function]}
Compare two grob priority queue entries. This is an internal function.

\textbf{ly:grob-properties} \textit{grob} \hspace{1cm} \textbf{[Function]}
Get the mutable properties of \textit{grob}.

\textbf{ly:grob-property} \textit{grob sym deflt} \hspace{1cm} \textbf{[Function]}
Return the value of a value in grob \textit{gob} of property \textit{sym}. It returns \texttt{'}()\texttt{'} (end-of-list) or \textit{deft} (if specified) if \textit{sym} is undefined in \textit{gob}.

\textbf{ly:grob-property-data} \textit{grob sym} \hspace{1cm} \textbf{[Function]}
Retrieve \textit{sym} for \textit{grob} but don’t process callbacks.

\textbf{ly:grob-relative-coordinate} \textit{grob refp axis} \hspace{1cm} \textbf{[Function]}
Get the coordinate in \textit{axis} direction of \textit{grob} relative to the grob \textit{refp}.

\textbf{ly:grob-robust-relative-extent} \textit{grob refp axis} \hspace{1cm} \textbf{[Function]}
Get the extent in \textit{axis} direction of \textit{grob} relative to the grob \textit{refp}, or \texttt{(0,0)} if empty.

\textbf{ly:grob-script-priority-less} \textit{a b} \hspace{1cm} \textbf{[Function]}
Compare two grobs by script priority. For internal use.

\textbf{ly:grob-set-property!} \textit{grob sym val} \hspace{1cm} \textbf{[Function]}
Set \textit{sym} in grob \textit{grob} to value \textit{val}.

\textbf{ly:grob-staff-position} \textit{sg} \hspace{1cm} \textbf{[Function]}
Return the Y-position of \textit{sg} relative to the staff.

\textbf{ly:grob-suicide!} \textit{grob} \hspace{1cm} \textbf{[Function]}
Kill \textit{grob}.
**ly:grob-system grob**
Return the system grob of grob.

**ly:grob-translate-axis! grob d a**
Translate g on axis a over distance d.

**ly:grob? x**
Is x a Grob object?

**ly:gulp-file name size**
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**
Return a list of keys in tab.

**ly:inch num**
num inches.

**ly:input-both-locations sip**
Return input location in sip as (file-name first-line first-column last-line last-column).

**ly:input-file-line-char-column sip**
Return input location in sip as (file-name line char column).

**ly:input-location? x**
Is x an input-location?

**ly:input-message sip msg rest**
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**
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**
Parse expr, feed bits to func with first arg arg1 having offset offset.

**ly:intlog2 d**
The 2-logarithm of 1/d.

**ly:is-listened-event-class sym**
Is sym a listened event class?

**ly:item-break-dir it**
The break status direction of item it. -1 means end of line, 0 unbroken, and 1 beginning of line.

**ly:item? g**
Is g an Item object?

**ly:iterator? x**
Is x a Music_iterator object?

**ly:lexer-keywords lexer**
Return a list of (KEY . CODE) pairs, signifying the LilyPond reserved words list.
ly:ly-lily-lexer? x
Is x a Lily_lexer object?

ly:ly-lily-parser? x
Is x a Lily_parser object?

ly:make-book  paper  header  scores
Make a \book of paper and header (which may be #f as well) containing \scores.

ly:make-book-part scores
Make a \bookpart containing \scores.

ly:make-dispatcher
Return a newly created dispatcher.

ly:make-duration length  dotcount  num  den
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
Set up a global interpretation context, using the output block output-def. The context is returned.

ly:make-global-translator global
Create a translator group and connect it to the global context global. The translator group is returned.

ly:make-listener callback
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
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
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
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:ly-music? predicates or other type predicates.

ly:make-output-def
Make an output definition.

ly:make-page-label-marker label
Return page marker with label.
ly:make-page-permission-marker *symbol* permission
Return page marker with page breaking and turning permissions.

ly:make-pango-description-string *chain* size
Make a PangoFontDescription string for the property alist *chain* at size *size*.

ly:make-paper-outputter *port* format
Create an outputter that evaluates within output-format, writing to *port*.

ly:make-pitch octave note alter
*octave* is specified by an integer, zero for the octave containing middle C. *note* is a number indexing the global default scale, with 0 corresponding to pitch C and 6 usually corresponding to pitch B. *alter* is a rational number of 200-cent whole tones for alteration.

ly:make-prob type init rest
Create a Prob object.

ly:make-scale steps
Create a scale. The argument is a vector of rational numbers, each of which represents the number of 200 cent tones of a pitch above the tonic.

ly:make-score music
Return score with music encapsulated in score.

ly:make-simple-closure expr
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
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
Create a stream event of class *cl* with the given mutable property list.

ly:message str rest
A Scheme callable function to issue the message *str*. The message is formatted with format and *rest*.

ly:minimal-breaking pb
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
num mm.

ly:module->alist mod
Dump the contents of module *mod* as an alist.

ly:module-copy dest src
Copy all bindings from module *src* into *dest*. 
ly:modules-lookup \( modules \ sym \ def \)
Look up \( sym \) in the list \( modules \), returning the first occurrence. If not found, return \( \text{def} \) or \#f\) if \( \text{def} \) isn’t specified.

ly:moment-add \( a \ b \)
Add two moments.

ly:moment-div \( a \ b \)
Divide two moments.

ly:moment-grace-denominator \( \text{mom} \)
Extract denominator from grace timing.

ly:moment-grace-numerator \( \text{mom} \)
Extract numerator from grace timing.

ly:moment-main-denominator \( \text{mom} \)
Extract denominator from main timing.

ly:moment-main-numerator \( \text{mom} \)
Extract numerator from main timing.

ly:moment-mod \( a \ b \)
Modulo of two moments.

ly:moment-mul \( a \ b \)
Multiply two moments.

ly:moment-sub \( a \ b \)
Subtract two moments.

ly:moment<? \( a \ b \)
Compare two moments.

ly:moment? \( x \)
Is \( x \) a \text{Moment} object?

ly:music-compress \( m \ \text{factor} \)
Compress music object \( m \) by moment \( \text{factor} \).

ly:music-deep-copy \( m \)
Copy \( m \) and all sub expressions of \( m \).

ly:music-duration-compress \( \text{mus} \ \text{fact} \)
Compress \( \text{mus} \) by factor \( \text{fact} \), which is a \text{Moment}.

ly:music-duration-length \( \text{mus} \)
Extract the duration field from \( \text{mus} \) and return the length.

ly:music-function-extract \( x \)
Return the Scheme function inside \( x \).

ly:music-function? \( x \)
Is \( x \) a \text{music-function}?

ly:music-length \( \text{mus} \)
Get the length of music expression \( \text{mus} \) and return it as a \text{Moment} object.
ly: musica-list? lst
Type predicate: Return true if lst is a list of music objects.

ly: musica-mutable-properties mus
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: musica-output? x
Is x a Music_output object?

ly: musica-property mus sym dfault
Get the property sym of music expression mus. If sym is undefined, return '().

ly: musica-set-property! mus sym val
Set property sym in music expression mus to val.

ly: musica-transpose m p
Transpose m such that central C is mapped to p. Return m.

ly: musica? obj
Type predicate.

ly: note-head::stem-attachment font-metric glyph-name
Get attachment in font-metric for attaching a stem to notehead glyph-name.

ly: number->string s
Convert num to a string without generating many decimals.

ly: optimal-breaking pb
Optimally break (pages and lines) the Paper_book object pb to minimize badness in both vertical and horizontal spacing.

ly: option-usage
Print ly:set-option usage.

ly: otf->cff otf-file-name
Convert the contents of an OTF file to a CFF file, returning it as a string.

ly: otf-font-glyph-info font glyph
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
Extract a table tag from font. Return empty string for non-existent tag.

ly: otf-font? font
Is font an OpenType font?

ly: otf-glyph-list font
Return a list of glyph names for font.

ly: output-def-clone def
Clone output definition def.

ly: output-def-lookup pap sym def
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
Get the parent output definition of def.

ly:output-def-scope def
Get the variable scope inside def.

ly:output-def-set-variable! def sym val
Set an output definition def variable sym to val.

ly:output-def? def
Is def a layout definition?

ly:output-description output-def
Return the description of translators in output-def.

ly:output-formats
Formats passed to '--format' as a list of strings, used for the output.

ly:outputter-close outputter
Close port of outputter.

ly:outputter-dump-stencil outputter stencil
Dump stencil expr onto outputter.

ly:outputter-dump-string outputter str
Dump str onto outputter.

ly:outputter-output-scheme outputter expr
Eval expr in module of outputter.

ly:outputter-port outputter
Return output port for outputter.

ly:page-marker? x
Is x a Page_marker object?

ly:page-turn-breaking pb
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
Return alist of (ps-name file-name font-index) lists for Pango font f.

ly:pango-font? f
Is f a pango font?

ly:paper-book-pages pb
Return pages in book pb.

ly:paper-book-paper pb
Return pages in book pb.


ly:paper-book-systems  \textit{pb}
Return systems in book \textit{pb}.

ly:paper-book?  \textit{x}
Is \textit{x} a \texttt{Paper\_book} object?

ly:paper-fonts  \textit{bp}
Return fonts from the \texttt{paper} block \textit{bp}.

ly:paper-get-font  \texttt{paper-smob chain}
Return a font metric satisfying the font-qualifiers in the alist chain \textit{chain}. (An alist chain is a list of alists, containing grob properties.)

ly:paper-get-number  \texttt{layout-smob name}
Return the layout variable \textit{name}.

ly:paper-outputscale  \textit{bp}
Get output-scale for \textit{bp}.

ly:paper-score-paper-systems  \texttt{paper-score}
Return vector of \texttt{paper\_system} objects from \texttt{paper-score}.

ly:paper-system-minimum-distance  \texttt{sys1 sys2}
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?  \texttt{obj}
Type predicate.

ly:parse-file  \texttt{name}
Parse a single \texttt{.ly} file. Upon failure, throw \texttt{ly-file-failed} key.

ly:parser-clear-error  \texttt{parser}
Clear the error flag for the parser.

ly:parser-clone  \texttt{parser-smob}
Return a clone of \texttt{parser-smob}.

ly:parser-define!  \texttt{parser-smob symbol val}
Bind \texttt{symbol} to \texttt{val} in \texttt{parser-smob}'s module.

ly:parser-error  \texttt{parser msg input}
Display an error message and make the parser fail.

ly:parser-has-error?  \texttt{parser}
Does \texttt{parser} have an error flag?

ly:parser-lexer  \texttt{parser-smob}
Return the lexer for \texttt{parser-smob}.

ly:parser-lookup  \texttt{parser-smob symbol}
Look up \texttt{symbol} in \texttt{parser-smob}'s module. Return \texttt{'}()\texttt{'} if not defined.

ly:parser-output-name  \texttt{parser}
Return the base name of the output file.

ly:parser-parse-string  \texttt{parser-smob ly-code}
Parse the string \texttt{ly-code} with \texttt{parser-smob}. Upon failure, throw \texttt{ly-file-failed} key.
ly:parser-set-note-names parser names
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
Write performance to filename.

ly:pfb->pfa pfb-file-name
Convert the contents of a PFB file to PFA.

ly:pitch-alteration pp
Extract the alteration from pitch pp.

ly:pitch-diff pitch root
Return pitch delta such that pitch transposed by delta equals root.

ly:pitch-negate p
Negate p.

ly:pitch-notename pp
Extract the note name from pitch pp.

ly:pitch-octave pp
Extract the octave from pitch pp.

ly:pitch-quartertones pp
Calculate the number of quarter tones of pp from middle C.

ly:pitch-semitones pp
Calculate the number of semitones of pp from middle C.

ly:pitch-steps p
Number of steps counted from middle C of the pitch p.

ly:pitch-transpose p delta
Transpose p by the amount delta, where delta is relative to middle C.

ly:pitch<? p1 p2
Is p1 lexicographically smaller than p2?

ly:pitch? x
Is x a Pitch object?

ly:position-on-line? sg spos
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
Retrieve an alist of mutable properties.

ly:prob-mutable-properties prob
Retrieve an alist of mutable properties.

ly:prob-property obj sym default
Return the value for sym.

ly:prob-property? obj sym
Is boolean prop sym set?
ly:prob-set-property! obj sym value
   Set property sym of obj to value.

ly:prob-type? obj type
   Is obj the specified prob-type?

ly:prob? x
   Is x a Prob object?

ly:programming-error str rest
   A Scheme callable function to issue the internal warning str. The message is formatted with format and rest.

ly:progress str rest
   A Scheme callable function to print progress str. The message is formatted with format and rest.

ly:property-lookup-stats sym
   Return hash table with a property access corresponding to sym. Choices are prob, grob, and context.

ly:protects
   Return hash of protected objects.

ly:pt num
   num printer points.

ly:register-stencil-expression symbol
   Add symbol as head of a stencil expression.

ly:relative-group-extent elements common axis
   Determine the extent of elements relative to common in the axis direction.

ly:reset-all-fonts
   Forget all about previously loaded fonts.

ly:round-filled-box xext yext blot
   Make a Stencil object that prints a black box of dimensions xext, yext and roundness blot.

ly:round-filled-polygon points blot
   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
   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
   Add an output definition def to score.

ly:score-embedded-format score layout
   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
   Was there an error in the score?
ly:score-header score
Return score header.

ly:score-music score
Return score music.

ly:score-output-defs score
All output definitions in a score.

ly:score-set-header! score module
Set the score header.

ly:score? x
Is x a Score object?

ly:set-default-scale scale
Set the global default scale. This determines the tuning of pitches with no accidentals or key
signatures. The first pitch is C. Alterations are calculated relative to this scale. The number
of pitches in this scale determines the number of scale steps that make up an octave. Usually
the 7-note major scale.

ly:set-grob-modification-callback cb
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
Set the middleCPosition variable in context based on the variables middleCClefPosition and middleCOffset.

ly:set-option var val
Set a program option.

ly:set-property-cache-callback cb
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
Type predicate.

ly:skyline-pair? x
Is x a Skyline_pair object?

ly:skyline? x
Is x a Skyline object?

ly:smob-protects
Return LilyPond’s internal smob protection list.

ly:solve-spring-rod-problem springs rods length ragged
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 \((\text{ideal}, \text{inverse\_hook})\) and \text{rods} is of the form \((\text{idx1}, \text{idx2}, \text{distance})\).

\text{length} is a number, \text{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 \text{spring-count}+1 positions of the objects.

\text{ly:source-file? x} \quad \text{[Function]}
Is \(x\) a \text{Source\_file} object?

\text{ly:spanner-bound spanner dir} \quad \text{[Function]}
Get one of the bounds of \text{spanner}. \text{dir} is -1 for left, and 1 for right.

\text{ly:spanner-broken-into spanner} \quad \text{[Function]}
Return broken-into list for \text{spanner}.

\text{ly:spanner? g} \quad \text{[Function]}
Is \(g\) a spanner object?

\text{ly:staff-symbol-line-thickness grob} \quad \text{[Function]}
Returns the line-thickness of the staff associated with \text{grob}.

\text{ly:start-environment} \quad \text{[Function]}
Return the environment (a list of strings) that was in effect at program start.

\text{ly:stderr-redirect file-name mode} \quad \text{[Function]}
Redirect stderr to \text{file-name}, opened with \text{mode}.

\text{ly:stencil-add args} \quad \text{[Function]}
Combine stencils. Takes any number of arguments.

\text{ly:stencil-aligned-to stil axis dir} \quad \text{[Function]}
Align \text{stil} using its own extents. \text{dir} is a number. -1 and 1 are left and right, respectively. Other values are interpolated (so 0 means the center).

\text{ly:stencil-combine-at-edge first axis direction second padding minimum} \quad \text{[Function]}
Construct a stencil by putting \text{second} next to \text{first}. \text{axis} can be 0 (x-axis) or 1 (y-axis). \text{direction} can be -1 (left or down) or 1 (right or up). The stencils are juxtaposed with \text{padding} as extra space. If this puts the reference points closer than \text{minimum}, they are moved by the latter amount. \text{first} and \text{second} may also be '()' or \#f.

\text{ly:stencil-empty? stil} \quad \text{[Function]}
Return whether \text{stil} is empty.

\text{ly:stencil-expr stil} \quad \text{[Function]}
Return the expression of \text{stil}.

\text{ly:stencil-extent stil axis} \quad \text{[Function]}
Return a pair of numbers signifying the extent of \text{stil} in \text{axis} direction (0 or 1 for x and y axis, respectively).

\text{ly:stencil-fonts s} \quad \text{[Function]}
Analyze \text{s}, and return a list of fonts used in \text{s}.

\text{ly:stencil-in-color stc r g b} \quad \text{[Function]}
Put \text{stc} in a different color.
ly:stencil-rotate  stil angle x y  
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  
Return a stencil stil rotated angle degrees around point (x, y), given in absolute coordinates.

ly:stencil-translate  stil offset  
Return a stil, but translated by offset (a pair of numbers).

ly:stencil-translate-axis  stil amount axis  
Return a copy of stil but translated by amount in axis direction.

ly:stencil?  x  
Is x a Stencil object?

ly:stream-event?  x  
Is x a Stream_event object?

ly:string-substitute  a b s  
Replace string a by string b in string s.

ly:system-font-load  name  
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 Enmentaler 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  
Draw the system and return the prob containing its stencil.

ly:system-stretch  system amount-scm  
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  
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  
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-grob (ly:output-def-lookup layout 'text-font-defaults)). markup is the markup text to be processed.

ly:translator-description  me  
Return an alist of properties of translator me.

ly:translator-group?  x  
Is x a Translator_group object?

ly:translator-name  trans  
Return the type name of the translator object trans. The name is a symbol.
Chapter 4: Scheme functions

ly:translator? x
Is x a Translator object?

ly:transpose-key-alist l pit
Make a new key alist of l transposed by pitch pit.

ly:truncate-list! lst i
Take at most the first i of list lst.

ly:ttf->pfa ttf-file-name idx
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
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
Return the unit used for lengths as a string.

ly:usage
Print usage message.

ly:version
Return the current Lilypond version as a list, e.g., (1 3 127 uu1).

ly:warning str rest
A Scheme callable function to issue the warning str. The message is formatted with format and rest.

ly:wide-char->utf-8 wc
Encode the Unicode codepoint wc, an integer, as UTF-8.
Appendix A Indices

A.1 Concept index

(Index is nonexistent)

A.2 Function index

D
dispatcher ................................................. 439

L
listener ................................................ 439
ly:add-file-name-alist ............................. 439
ly:add-interface ................................ 439
ly:add-listener .................................... 439
ly:add-option ...................................... 439
ly:all-grob-interfaces ......................... 439
ly:all-options .................................... 439
ly:all-stencil-expressions ................. 439
ly:assoc-get .................................... 439
ly:book-add-bookpart! ....................... 439
ly:book-add-score! ............................ 439
ly:book-process .................................. 439
ly:book-process-to-systems ............ 439
ly:box? ........................................... 439
ly:bp ............................................. 439
ly:bracket ....................................... 439
ly:broadcast .................................... 440
ly:camel-case->lisp-identifier ........ 440
ly:chain-assoc-get ............................... 440
ly:clear-anonymous-modules ............. 440
ly:cm ............................................. 440
ly:command-line-code ......................... 440
ly:command-line-options ..................... 440
ly:command-line-verbose? .............. 440
ly:connect-dispatchers ..................... 440
ly:context-event-source ..................... 440
ly:context-events-below ..................... 440
ly:context-find ................................. 440
ly:context-grob-definition ................. 440
ly:context-id ................................ 440
ly:context-name ................................ 440
ly:context-now ................................ 440
ly:context-parent ................................ 440
ly:context-property-where-defined .... 440
ly:context-property ....................... 440
ly:context-pushpop-property ! .......... 441
ly:context-set-property! ................. 441
ly:context unset-property .................. 441
ly:context ........................................ 441
ly:default-scale ................................ 441
ly:dimension? .................................. 441
ly:dir? .......................................... 441
ly:duration->string ......................... 441
ly:duration-dot-count ...................... 441
ly:duration-factor ......................... 441
ly:duration-length ......................... 441
ly:duration-log ................................ 441
ly:duration-log-? ............................ 441
ly:duration? .................................... 441
ly:effective-prefix ......................... 441
ly:eval-simple-closure ..................... 441
ly:event-deep-copy ........................... 441
ly:event-property ................................ 442
ly:event-set-property! ................. 442
ly:expand-environment ..................... 442
ly:export ........................................ 442
ly:find-file ..................................... 442
ly:font-config-add-directory .......... 442
ly:font-config-add-font .................. 442
ly:font-config-display-fonts ........ 442
ly:font-config-get-font-file .......... 442
ly:font-design-size .......................... 442
ly:font-file-name ......................... 442
ly:font-get-glyph ......................... 442
ly:font-glyph-name-to-charcode .... 442
ly:font-glyph-name-to-index .......... 442
ly:font-index-to-charcode ............ 442
ly:font-magnification ....................... 443
ly:font-metric? ................................ 443
ly:font-name .................................. 443
ly:font-sub-fonts ................. 443
ly:format ....................................... 443
ly:format-output ............................ 443
ly:get-all-function-documentation .... 443
ly:get-all-translators ................. 443
ly:get-glyph .................................... 443
ly:get-listened-event-classes .......... 443
ly:get-option .................................. 443
ly:gettext ...................................... 443
ly:grob-alist-chain ......................... 443
ly:grob-array-length ....................... 443
ly:grob-array-ref ............................. 443
ly:grob-array? ................................ 443
ly:grob-basic-properties ................. 443
ly:grob-common-refpoint ............... 444
ly:grob-common-refpoint-of-array .... 444
ly:grob-default-font ...................... 444
ly:grob-extent ................................ 444
ly:grob-interfaces ......................... 444
ly:grob-layout ................................ 444
ly:grob-object ................................ 444
ly:grob-original .............................. 444
ly:grob-parent ................................ 444
ly:grob-pq? .................................. 444
ly:grob-properties ......................... 444
ly:grob-property ......................... 444
ly:grob-property-data .................... 444
ly:grob-relative-coordinate ............ 444