

LilyPond

The music typesetter

Snippets

The LilyPond development team

This document shows a selected set of LilyPond snippets from the [LilyPond Snippet Repository](#) (LSR). It is in the public domain.

We would like to address many thanks to Sebastiano Vigna for maintaining LSR web site and database, and the University of Milano for hosting LSR.

Please note that this document is not an exact subset of LSR: some snippets come from ‘`input/new`’ LilyPond sources directory, and snippets from LSR are converted through `convert-ly`, as LSR is based on a stable LilyPond version, and this document is for version 2.13.27.

Snippets are grouped by tags; tags listed in the table of contents match a section of LilyPond notation manual. Snippets may have several tags, and not all LSR tags may appear in this document.

In the HTML version of this document, you can click on the file name or figure for each example to see the corresponding input file.

For more information about how this manual fits with the other documentation, or to read this manual in other formats, see [Section “Manuals” in *General Information*](#).

If you are missing any manuals, the complete documentation can be found at <http://www.lilypond.org/>.

This document has been placed in the public domain.

For LilyPond version 2.13.27

Table of Contents

Pitches	1
Adding ambitus per voice	1
Altering the length of beamed stems	1
Ambitus with multiple voices	2
Ambitus	2
Applying note head styles depending on the step of the scale	3
Coloring notes depending on their pitch	4
Creating a sequence of notes on various pitches	5
Dodecaphonic-style accidentals for each note including naturals	6
Generating random notes	6
Hiding accidentals on tied notes at the beginning of a new system	7
Keep change clefs full sized	7
Makam example	7
Non-traditional key signatures	8
Numbers as easy note heads	8
Ottava text	10
Preventing extra naturals from being automatically added	10
Preventing natural signs from being printed when the key signature changes	10
Quoting another voice with transposition	11
Separating key cancellations from key signature changes	11
Transposing pitches with minimum accidentals ("Smart" transpose)	12
Tweaking clef properties	13
Rhythms	16
Adding beams slurs ties etc. when using tuplet and non-tuplet rhythms	16
Adding drum parts	16
Adjusting grace note spacing	17
Aligning bar numbers	17
Automatic beam subdivisions	18
Avoiding collisions with chord fingerings	19
Beam endings in Score context	19
Beam grouping in 7/8 time	20
Beams across line breaks	20
Changing beam knee gap	21
Changing form of multi-measure rests	21
Changing the number of augmentation dots per note	22
Changing the time signature without affecting the beaming	22
Changing the tuplet number	22
Changing time signatures inside a polymetric section using \scaleDurations	23
Chant or psalms notation	23
Compound time signatures	24
Conducting signs measure grouping signs	24
Controlling tuplet bracket visibility	25
Engraving ties manually	26
Engraving tremolos with floating beams	26
Entering several tuplets using only one \times command	27
Flat flags and beam nibs	27
Forcing rehearsal marks to start from a given letter or number	29

Grouping beats.....	29
Guitar strum rhythms.....	29
Heavily customized polymetric time signatures.....	31
Making an object invisible with the 'transparent' property.....	32
Making slurs with complex dash structure.....	32
Manually controlling beam positions.....	33
Merging multi-measure rests in a polyphonic part.....	33
Modifying tuplet bracket length.....	34
Moving dotted notes in polyphony.....	34
Multi-measure rest markup.....	35
Non-default tuplet numbers.....	35
Permitting line breaks within beamed tuplets.....	36
Positioning grace note beams at the height of normal note beams.....	37
Positioning grace notes with floating space.....	37
Positioning multi-measure rests.....	37
Printing bar numbers at regular intervals.....	38
Printing bar numbers inside boxes or circles.....	39
Printing metronome and rehearsal marks below the staff.....	39
Printing music with different time signatures.....	40
Printing the bar number for the first measure.....	43
Redefining grace note global defaults.....	43
Removing bar numbers from a score.....	44
Removing connecting bar lines on StaffGroup PianoStaff or GrandStaff.....	44
Rest styles.....	45
Reverting default beam endings.....	46
Rhythmic slashes.....	46
Skips in lyric mode (2).....	47
Skips in lyric mode.....	47
Stemlets.....	48
Sub-dividing beams.....	48
Three-sided box.....	49
Time signature printing only the numerator as a number (instead of the fraction).....	50
Tweaking grace layout within music.....	50
Using alternative flag styles.....	51
Using grace note slashes with normal heads.....	51
Using ties with arpeggios.....	52
Expressive marks.....	53
Adding beams slurs ties etc. when using tuplet and non-tuplet rhythms.....	53
Adding parentheses around an expressive mark or chordal note.....	54
Adjusting the shape of falls and doits.....	54
Asymmetric slurs.....	54
Breathing signs.....	55
Broken Crescendo Hairpin.....	55
Caesura ("railtracks") with fermata.....	56
Center text below hairpin dynamics.....	57
Changing \flageolet mark size.....	58
Changing text and spanner styles for text dynamics.....	58
Changing the appearance of a slur from solid to dotted or dashed.....	58
Changing the breath mark symbol.....	59
Changing the number of augmentation dots per note.....	59
Combining dynamics with markup texts.....	59
Contemporary glissando.....	60
Controlling spanner visibility after a line break.....	60

Controlling the vertical ordering of scripts	61
Creating a delayed turn.....	61
Creating arpeggios across notes in different voices	62
Creating cross-staff arpeggios in a piano staff	62
Creating cross-staff arpeggios in other contexts.....	63
Creating "real" parenthesized dynamics	64
Creating simultaneous rehearsal marks	64
Creating slurs across voices	65
Creating text spanners	65
Double glissando	66
Dynamics custom text spanner postfix	67
Dynamics text spanner postfix.....	67
Hairpins with different line styles	68
Hiding the extender line for text dynamics	69
Horizontally aligning custom dynamics (e.g. "sempre pp" "piu f" "subito p")	69
Inserting a caesura	73
Laissez vibrer ties	73
Line arrows	73
Making slurs with complex dash structure	74
Modifying default values for articulation shorthand notation	74
Moving slur positions vertically	75
Piano template with centered dynamics	75
Positioning arpeggios	77
Positioning text markups inside slurs	77
Printing hairpins using al niente notation	77
Printing metronome and rehearsal marks below the staff	78
Setting hairpin behavior at bar lines	78
Setting the minimum length of hairpins	78
Showing the same articulation above and below a note or chord	79
Snap-pizzicato ("Bartok pizzicato").....	79
Using double slurs for legato chords.....	80
Using the whiteout property	80
Vertical line as a baroque articulation mark	81
Vertically aligning dynamics across multiple notes.....	81
Repeats	82
Adding volta brackets to additional staves	82
Cross-staff tremolos	82
Engraving tremolos with floating beams.....	83
Isolated percent repeats	84
Measure counter	84
Percent repeat count visibility	84
Percent repeat counter.....	85
Positioning segno and coda (with line break)	85
Printing a repeat sign at the beginning of a piece	87
Shortening volta brackets	88
Volta below chords	88
Volta multi staff.....	89
Volta text markup using repeatCommands.....	90

Simultaneous notes	91
Additional voices to avoid collisions	91
Changing a single note's size in a chord	92
Changing partcombine texts	92
Clusters	92
Combining two parts on the same staff	93
Displaying complex chords	94
Double glissando	94
Forcing horizontal shift of notes	95
Making an object invisible with the 'transparent property	95
Moving dotted notes in polyphony	96
Suppressing warnings for clashing note columns	96
Staff notation	98
Adding ambitus per voice	98
Adding an extra staff at a line break	98
Adding an extra staff	99
Adding indicators to staves which get split after a break	100
Changing the number of lines in a staff	102
Changing the staff size	102
Changing the tempo without a metronome mark	103
Creating blank staves	103
Creating metronome marks in markup mode	105
Display bracket with only one staff in a system	105
Forcing measure width to adapt to MetronomeMark's width	106
Incipit	107
Inserting score fragments above a staff as markups	112
Letter tablature formatting	113
Making some staff lines thicker than the others	114
Measure counter	114
Mensurstriche layout (bar lines between the staves)	115
Modern TAB text clef	115
Nesting staves	115
Non-traditional key signatures	116
Printing metronome and rehearsal marks below the staff	117
Quoting another voice with transposition	117
Quoting another voice	118
Removing the first empty line	119
Tick bar lines	120
Time signature in parentheses	120
Tweaking clef properties	121
Use square bracket at the start of a staff group	122
Volta below chords	122
Volta multi staff	123

Editorial annotations	125
Adding fingerings to a score	125
Allowing fingerings to be printed inside the staff	125
Analysis brackets above the staff	125
Applying note head styles depending on the step of the scale	126
Avoiding collisions with chord fingerings	126
Blanking staff lines using the \whiteout command	127
Changing a single note's size in a chord	127
Changing the appearance of a slur from solid to dotted or dashed	128
Coloring notes depending on their pitch	128
Controlling the placement of chord fingerings	129
Creating a delayed turn	130
Creating blank staves	130
Default direction of stems on the center line of the staff	132
Drawing boxes around grobs	132
Drawing circles around various objects	132
Embedding native PostScript in a \markup block	133
Grid lines: changing their appearance	133
Grid lines: emphasizing rhythms and notes synchronization	134
Making some staff lines thicker than the others	136
Marking notes of spoken parts with a cross on the stem	136
Measure counter	137
Positioning fingering indications precisely	137
Positioning text markups inside slurs	138
Printing text from right to left	138
String number extender lines	138
Using PostScript to generate special note head shapes	139
Using the whiteout property	140
 Text	 141
Adding the current date to a score	141
Adjusting lyrics vertical spacing	141
Aligning and centering instrument names	142
Aligning marks with various notation objects	143
Aligning objects created with the \mark command	145
Blanking staff lines using the \whiteout command	145
Center text below hairpin dynamics	145
Centering markup on note heads automatically	146
Changing the default text font family	147
Combining dynamics with markup texts	148
Combining two parts on the same staff	148
Creating "real" parenthesized dynamics	149
Creating simultaneous rehearsal marks	150
Creating text spanners	150
Demonstrating all headers	151
Embedding native PostScript in a \markup block	153
Formatting lyrics syllables	153
How to put ties between syllables in lyrics	154
Lyrics alignment	154
Markup lines	154
Multi-measure rest markup	156
Ottava text	157
Outputting the version number	157

Piano template with centered lyrics	157
Printing marks at the end of a line	158
Printing marks on every staff	159
Printing text from right to left	159
Stand-alone two-column markup	160
String number extender lines	160
Three-sided box	161
UTF-8	162
Vocal ensemble template with lyrics aligned below and above the staves	163
Volta text markup using repeatCommands	165
Vocal music	166
Adding ambitus per voice	166
Adding indicators to staves which get split after a break	166
Adjusting lyrics vertical spacing	168
Ambitus with multiple voices	169
Ambitus	169
Changing stanza fonts	170
Chant or psalms notation	171
Forcing hyphens to be shown	171
Formatting lyrics syllables	172
How to put ties between syllables in lyrics	172
Lyrics alignment	172
Marking notes of spoken parts with a cross on the stem	173
Piano template with melody and lyrics	174
Single staff template with notes lyrics and chords	175
Single staff template with notes lyrics chords and frets	176
Single staff template with notes and lyrics	177
Skips in lyric mode (2)	177
Skips in lyric mode	178
Vertically aligning ossias and lyrics	178
Vertically centered common lyrics	179
Vocal ensemble template with automatic piano reduction	180
Vocal ensemble template with lyrics aligned below and above the staves	182
Vocal ensemble template with verse and refrain	183
Vocal ensemble template	185
Chords	188
Adding a figured bass above or below the notes	188
Adding bar lines to ChordNames context	188
Avoiding collisions with chord fingerings	189
Bar chords notation for Guitar (with Text Spanner)	189
Changing chord separator	191
Changing the chord names to German or semi-German notation	191
Changing the positions of figured bass alterations	192
Chord name exceptions	192
chord name major7	193
Clusters	193
Controlling the placement of chord fingerings	194
Displaying complex chords	194
Manually break figured bass extenders for only some numbers	195
Showing chords at changes	195
Simple lead sheet	196

Single staff template with notes lyrics and chords	196
Single staff template with notes lyrics chords and frets	197
Single staff template with notes and chords	198
Vertically centering paired figured bass extenders	198
Volta below chords	199
Keyboards	200
Accordion-discant symbols	200
Changing the text for sustain markings	203
Clusters	204
Controlling the placement of chord fingerings	204
Creating slurs across voices	205
Cross-staff tremolos	205
Fine-tuning pedal brackets	206
Indicating cross-staff chords with arpeggio bracket	206
Jazz combo template	207
Laissez vibrer ties	213
Piano template (simple)	214
Piano template with centered dynamics	214
Piano template with centered lyrics	215
Piano template with melody and lyrics	216
Vocal ensemble template with automatic piano reduction	217
Percussion	220
Adding drum parts	220
Heavily customized polymetric time signatures	220
Jazz combo template	222
Percussion beaters	227
Printing music with different time signatures	230
Fretted strings	235
Adding fingerings to a score	235
Adding fingerings to tablatures	235
Allowing fingerings to be printed inside the staff	236
Bar chords notation for Guitar (with Text Spanner)	236
chordChanges for FretBoards	237
Controlling the placement of chord fingerings	238
Customizing fretboard fret diagrams	239
Customizing markup fret diagrams	240
Defining predefined fretboards for other instruments	242
Faking a hammer in tablatures	244
Fingerings string indications and right-hand fingerings	244
Flamenco notation	244
Fret diagrams explained and developed	249
Guitar slides	256
Guitar strum rhythms	257
How to change fret diagram position	258
Jazz combo template	259
Laissez vibrer ties	265
Letter tablature formatting	266
Modern TAB text clef	266
Placement of right-hand fingerings	266
Polyphony in tablature	267

Stem and beam behavior in tablature	268
String number extender lines	268
Unfretted strings	270
Changing \flageolet mark size	270
Creating slurs across voices	270
Dotted harmonics	271
Snap-pizzicato ("Bartok pizzicato")	271
String quartet template (simple)	271
String quartet template with separate parts	273
Winds	276
Fingering symbols for wind instruments	276
Flute slap notation	276
Recorder fingering chart	277
Ancient notation	279
Adding a figured bass above or below the notes	279
Ancient fonts	279
Ancient notation template – modern transcription of gregorian music	284
Ancient notation template – modern transcription of mensural music	285
Ancient time signatures	290
Chant or psalms notation	291
Custodes	291
Incipit	292
Mensurstriche layout (bar lines between the staves)	298
Rest styles	298
Transcription of Ancient music with incipit	299
Vertical line as a baroque articulation mark	305
World music	306
Arabic improvisation	306
Makam example	306
Printing text from right to left	306
Contexts and engravers	308
Adding a figured bass above or below the notes	308
Adding an extra staff at a line break	308
Adding an extra staff	309
Changing MIDI output to one channel per voice	310
Changing time signatures inside a polymetric section using \scaleDurations	311
Chant or psalms notation	312
Creating blank staves	312
Defining an engraver in Scheme: ambitus engraver	314
Engravers one-by-one	320
Mensurstriche layout (bar lines between the staves)	326
Nesting staves	326
Removing bar numbers from a score	327
Use square bracket at the start of a staff group	327
Vocal ensemble template with lyrics aligned below and above the staves	328
Vocal ensemble template with verse and refrain	329

Tweaks and overrides	332
Adjusting grace note spacing	332
Altering the length of beamed stems	332
Analysis brackets above the staff	333
Asymmetric slurs	333
Avoiding collisions with chord fingerings	333
Caesura ("railtracks") with fermata	334
Centering markup on note heads automatically	335
Changing a single note's size in a chord	335
Changing form of multi-measure rests	336
Changing properties for individual grobs	336
Changing the default text font family	336
Changing the staff size	337
Changing the text for sustain markings	338
Controlling spanner visibility after a line break	338
Controlling the vertical ordering of scripts	339
Controlling tuplet bracket visibility	340
Creating a delayed turn	340
Creating simultaneous rehearsal marks	341
Creating text spanners	341
Custodes	342
Customizing fretboard fret diagrams	343
Customizing markup fret diagrams	345
Display bracket with only one staff in a system	346
Displaying grob ancestry	347
Dotted harmonics	349
Drawing boxes around grobs	349
Drawing circles around various objects	350
Dynamics custom text spanner postfix	350
Dynamics text spanner postfix	351
Fine-tuning pedal brackets	352
Forcing horizontal shift of notes	352
Fret diagrams explained and developed	353
Hairpins with different line styles	359
Horizontally aligning custom dynamics (e.g. "sempre pp" "piu f" "subito p")	360
How to change fret diagram position	363
Inserting a caesura	364
Keep change clefs full sized	365
Line arrows	365
Making an object invisible with the 'transparent property	366
Manually controlling beam positions	367
Mensurstriche layout (bar lines between the staves)	367
Moving dotted notes in polyphony	367
Moving slur positions vertically	368
Nesting staves	368
Percent repeat count visibility	369
Positioning arpeggios	370
Positioning multi-measure rests	370
Positioning text markups inside slurs	371
Printing a repeat sign at the beginning of a piece	371
Printing bar numbers inside boxes or circles	372
Printing metronome and rehearsal marks below the staff	372
Printing note names with and without an octave marker	373
Proportional strict notespacing	373

Removing connecting bar lines on StaffGroup PianoStaff or GrandStaff.....	374
Removing the first empty line	374
Rest styles	376
Rhythmic slashes.....	377
Separating key cancellations from key signature changes.....	377
Setting hairpin behavior at bar lines.....	378
Setting system separators.....	378
Showing the same articulation above and below a note or chord.....	381
String number extender lines	381
Suppressing warnings for clashing note columns.....	382
Time signature in parentheses	382
Time signature printing only the numerator as a number (instead of the fraction).....	383
Transcription of Ancient music with incipit	383
Tweaking clef properties.....	389
Tweaking grace layout within music	391
Using alternative flag styles.....	391
Using ly:grob-object to access grobs with \tweak.....	392
Using PostScript to generate special note head shapes.....	393
Using the \tweak command to tweak individual grobs.....	393
Vertically aligned dynamics and textscripts	394
Vertically aligning ossias and lyrics	394
Vertically centering paired figured bass extenders	395
Paper and layout.....	397
Aligning and centering instrument names	397
Book parts	398
Changing the staff size	402
Clip systems	403
Creating blank staves.....	406
Demonstrating all headers	407
Setting system separators.....	408
Table of contents.....	411
Titles	413
Adding the current date to a score.....	413
Aligning and centering instrument names	413
Demonstrating all headers.....	415
Outputting the version number	416
Spacing	417
Adjusting lyrics vertical spacing	417
Allowing fingerings to be printed inside the staff	417
Page label.....	418
Proportional strict notespacing.....	420
Vertically aligned dynamics and textscripts	421
Vertically aligning ossias and lyrics	421
MIDI.....	423
Changing MIDI output to one channel per voice	423
Changing the tempo without a metronome mark.....	424
Demo MidiInstruments	424

Templates	428
Ancient notation template – modern transcription of gregorian music.....	428
Ancient notation template – modern transcription of mensural music.....	429
Jazz combo template	434
Orchestra choir and piano template.....	440
Piano template (simple)	445
Piano template with centered dynamics.....	445
Piano template with centered lyrics.....	446
Piano template with melody and lyrics	447
Score for diatonic accordion	448
Single staff template with notes lyrics and chords	453
Single staff template with notes lyrics chords and frets	454
Single staff template with notes and chords	455
Single staff template with notes and lyrics	455
Single staff template with only notes.....	456
String quartet template (simple).....	456
String quartet template with separate parts.....	458
Vocal ensemble template with automatic piano reduction.....	460
Vocal ensemble template with lyrics aligned below and above the staves.....	462
Vocal ensemble template with verse and refrain	464
Vocal ensemble template	466

Pitches

Section “Pitches” in *Notation Reference*

Adding ambitus per voice

Ambitus can be added per voice. In this case, the ambitus must be moved manually to prevent collisions.

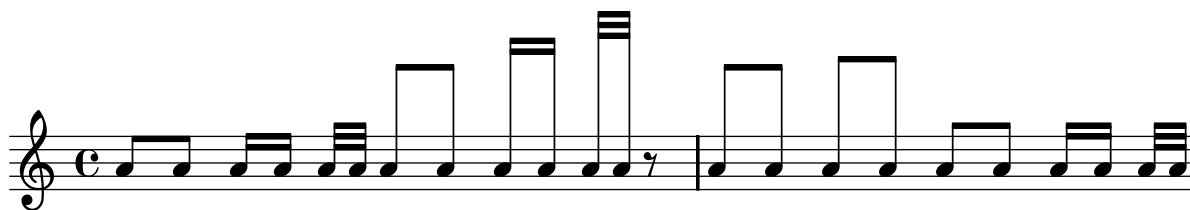
```
\new Staff <<
  \new Voice \with {
    \consists "Ambitus_engraver"
  } \relative c'' {
    \override Ambitus #'X-offset = #2.0
    \voiceOne
    c4 a d e
    f1
  }
  \new Voice \with {
    \consists "Ambitus_engraver"
  } \relative c' {
    \voiceTwo
    es4 f g as
    b1
  }
}>>
```



Altering the length of beamed stems

Stem lengths on beamed notes can be varied by overriding the `beamed-lengths` property of the `details` of the `Stem`. If a single value is used as an argument, the length applies to all stems. When multiple arguments are used, the first applies to eighth notes, the second to sixteenth notes and so on. The final argument also applies to all notes shorter than the note length of the final argument. Non-integer arguments may also be used.

```
\relative c'' {
  \override Stem #'details #'beamed-lengths = #'(2)
  a8[ a] a16[ a] a32[ a]
  \override Stem #'details #'beamed-lengths = #'(8 10 12)
  a8[ a] a16[ a] a32[ a] r8
  \override Stem #'details #'beamed-lengths = #'(8)
  a8[ a]
  \override Stem #'details #'beamed-lengths = #'(8.5)
  a8[ a]
  \revert Stem #'details
  a8[ a] a16[ a] a32[ a]
}
```



Ambitus with multiple voices

Adding the `Ambitus_engraver` to the `Staff` context creates a single ambitus per staff, even in the case of staves with multiple voices.

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



Ambitus

Ambitus indicate pitch ranges for voices.

Accidentals only show up if they are not part of the key signature. `AmbitusNoteHead` grobs also have ledger lines.

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

<<
  \new Staff {
    \relative c' {
      \time 2/4
      c4 f'
    }
  }
  \new Staff {
    \relative c' {
```

```

\time 2/4
\key d \major
cis4 as'
}
}
>>

```



Applying note head styles depending on the step of the scale

The `shapeNoteStyles` property can be used to define various note head styles for each step of the scale (as set by the key signature or the `tonic` property). This property requires a set of symbols, which can be purely arbitrary (geometrical expressions such as `triangle`, `cross`, and `xcircle` are allowed) or based on old American engraving tradition (some latin note names are also allowed).

That said, to imitate old American song books, there are several predefined note head styles available through shortcut commands such as `\aikenHeads` or `\sacredHarpHeads`.

This example shows different ways to obtain shape note heads, and demonstrates the ability to transpose a melody without losing the correspondence between harmonic functions and note head styles.

```

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

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

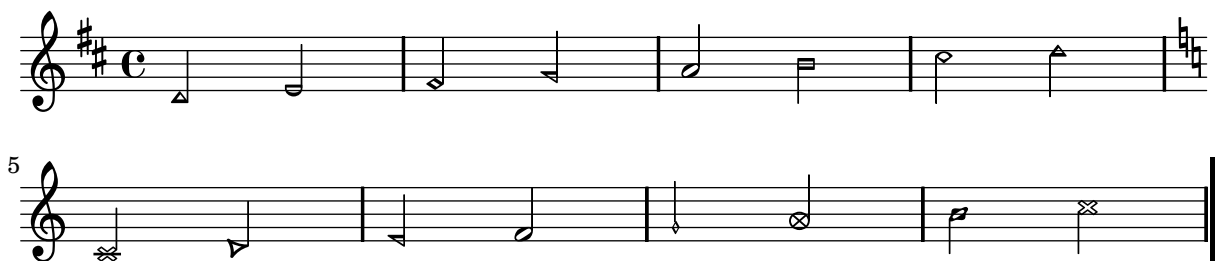
    \fragment
  }

  \break

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

    \fragment
  }
}

```



Coloring notes depending on their pitch

It is possible to color note heads depending on their pitch and/or their names: the function used in this example even makes it possible to distinguish enharmonics.

%Association list of pitches to colors.

```
#(define color-mapping
  (list
    (cons (ly:make-pitch 0 0 NATURAL) (x11-color 'red))
    (cons (ly:make-pitch 0 0 SHARP) (x11-color 'green))
    (cons (ly:make-pitch 0 1 FLAT) (x11-color 'green))
    (cons (ly:make-pitch 0 2 NATURAL) (x11-color 'red))
    (cons (ly:make-pitch 0 2 SHARP) (x11-color 'green))
    (cons (ly:make-pitch 0 3 FLAT) (x11-color 'red))
    (cons (ly:make-pitch 0 3 NATURAL) (x11-color 'green))
    (cons (ly:make-pitch 0 4 SHARP) (x11-color 'red))
    (cons (ly:make-pitch 0 5 NATURAL) (x11-color 'green))
    (cons (ly:make-pitch 0 5 FLAT) (x11-color 'red))
    (cons (ly:make-pitch 0 6 SHARP) (x11-color 'red))
    (cons (ly:make-pitch 0 1 NATURAL) (x11-color 'blue))
    (cons (ly:make-pitch 0 3 SHARP) (x11-color 'blue))
    (cons (ly:make-pitch 0 4 FLAT) (x11-color 'blue))
    (cons (ly:make-pitch 0 5 SHARP) (x11-color 'blue))
    (cons (ly:make-pitch 0 6 FLAT) (x11-color 'blue))))

%Compare pitch and alteration (not octave).
#(define (pitch-equals? p1 p2)
  (and
    (= (ly:pitch-alteration p1) (ly:pitch-alteration p2))
    (= (ly:pitch-notename p1) (ly:pitch-notename p2))))

#(define (pitch-to-color pitch)
  (let ((color (assoc pitch color-mapping pitch-equals?)))
    (if color
      (cdr color))))

#(define (color-notehead grob)
  (pitch-to-color
    (ly:event-property (event-cause grob) 'pitch)))

\score {
  \new Staff \relative c' {
    \override NoteHead #'color = #color-notehead
```

```

      c8 b d dis ees f g aes
    }
  }

```



Creating a sequence of notes on various pitches

In music that contains many occurrences of the same sequence of notes at different pitches, the following music function may prove useful. It takes a note, of which only the pitch is used. This example creates the rhythm used throughout Mars, from Gustav Holst's *The Planets*.

```

#(define (make-note-req p d)
  (make-music 'NoteEvent
    'duration d
    'pitch p))

#(define (make-note p d)
  (make-music 'EventChord
    'elements (list (make-note-req p d))))

#(define (make-triplet elt)
  (make-music 'TimeScaledMusic
    'denominator 3
    'numerator 2
    'element elt))

rhythm =
#(define-music-function (parser location note) (ly:music?)
  "Make the rhythm in Mars (the Planets) at the given note's pitch"
  (let ((p (ly:music-property
    (car (ly:music-property note 'elements))
    'pitch)))
    (make-sequential-music
      (list
        (make-triplet (make-sequential-music
          (list
            (make-note p (ly:make-duration 3 0 2 3))
            (make-note p (ly:make-duration 3 0 2 3))
            (make-note p (ly:make-duration 3 0 2 3))))))
        (make-note p (ly:make-duration 2 0))
        (make-note p (ly:make-duration 2 0))
        (make-note p (ly:make-duration 3 0))
        (make-note p (ly:make-duration 3 0))
        (make-note p (ly:make-duration 2 0))))))

\new Staff {
  \time 5/4
  \rhythm c'
  \rhythm c''

```

```
\rhythm g
}
```



Dodecaphonic-style accidentals for each note including naturals

In early 20th century works, starting with Schoenberg, Berg and Webern (the “Second” Viennese school), every pitch in the twelve-tone scale has to be regarded as equal, without any hierarchy such as the classical (tonal) degrees. Therefore, these composers print one accidental for each note, even at natural pitches, to emphasize their new approach to music theory and language.

This snippet shows how to achieve such notation rules.

```
\markup {
  This snippet is deprecated as of version 2.12 and
  will be removed from the documentation in 2.14.
}
```

This snippet is deprecated as of version 2.12 and will be removed from the documentation in 2.14. ■

Generating random notes

This Scheme-based snippet generates 24 random notes (or as many as required), based on the current time (or any randomish number specified instead, in order to obtain the same random notes each time): i.e., to get different random note patterns, just change this number.

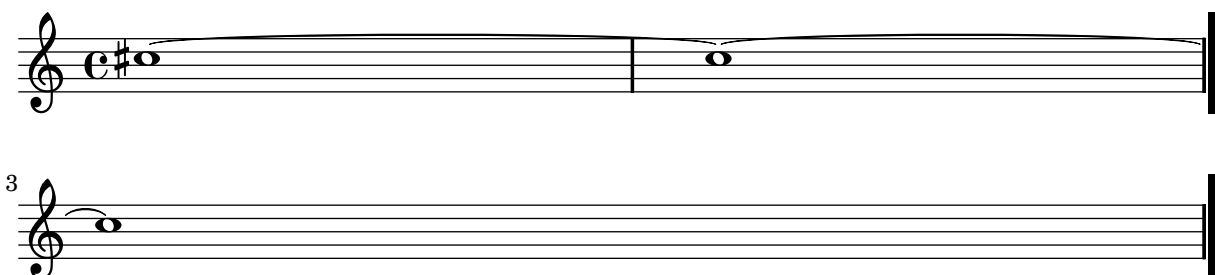
```
\score {
  {
    #(let ((random-state (seed->random-state (current-time))))
      (ly:export
        (make-sequential-music
          (map (lambda (x)
                (let ((idx (random 12 random-state)))
                  (make-event-chord
                    (list
                     (make-music 'NoteEvent
                               'duration (ly:make-duration 2 0 1 1)
                               'pitch (ly:make-pitch
                                       (quotient idx 7)
                                       (remainder idx 7)
                                       0)))))))
              (make-list 24))))))
  }
```



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

This shows how to hide accidentals on tied notes at the beginning of a new system.

```
\relative c' {
  \override Accidental #'hide-tied-accidental-after-break = ##t
  cis1~ cis~
  \break
  cis
}
```



Keep change clefs full sized

When a clef is changed, the clef sign displayed is smaller than the initial clef. This can be overridden with `full-size-change`.

```
\relative c' {
  \clef "treble"
  c1
  \clef "bass"
  c1
  \clef "treble"
  c1
  \override Staff.Clef #'full-size-change = ##t
  \clef "bass"
  c1
  \clef "treble"
  c1
  \revert Staff.Clef #'full-size-change
  \clef "bass"
  c1
  \clef "treble"
  c1
}
```



Makam example

Makam is a type of melody from Turkey using 1/9th-tone microtonal alterations. Consult the initialization file 'ly/makam.ly' for details of pitch names and alterations.

```
% Initialize makam settings
\include "makam.ly"

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



Non-traditional key signatures

The commonly used `\key` command sets the `keySignature` property, in the `Staff` context.

To create non-standard key signatures, set this property directly. The format of this command is a list:

```
\set Staff.keySignature = #`(((octave . step) . alter) ((octave . step) . alter)
...) where, for each element in the list, octave specifies the octave (0 being the octave
from middle C to the B above), step specifies the note within the octave (0 means C and
6 means B), and alter is ,SHARP ,FLAT ,DOUBLE-SHARP etc. (Note the leading comma.)
```

Alternatively, for each item in the list, using the more concise format `(step . alter)` specifies that the same alteration should hold in all octaves.

Here is an example of a possible key signature for generating a whole-tone scale:

```
\relative c' {
  \set Staff.keySignature = #`(((0 . 6) . ,FLAT)
                                ((0 . 5) . ,FLAT)
                                ((0 . 3) . ,SHARP))

  c4 d e fis
  aes4 bes c2
}
```



Numbers as easy note heads

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

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

```
#(define Ez_numbers_engraver
  (list
    (cons 'acknowledgers
      (list
        (cons 'note-head-interface
```

```

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

#(set-global-staff-size 26)

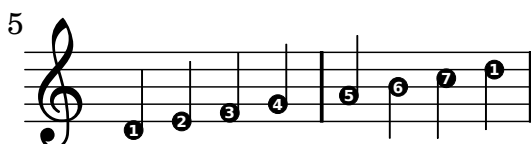
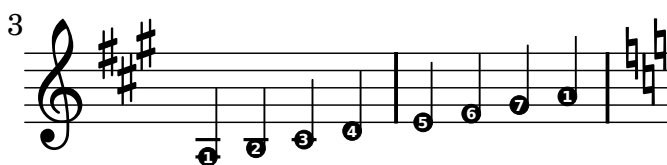
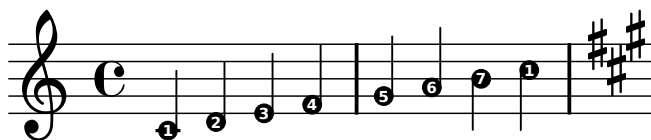
\layout {
  ragged-right = ##t
  \context {
    \Voice
    \consists \Ez_numbers_engraver
  }
}

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

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

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

```



Ottava text

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

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



Preventing extra naturals from being automatically added

In accordance with standard typesetting rules, a natural sign is printed before a sharp or flat if a previous accidental on the same note needs to be canceled. To change this behavior, set the `extraNatural` property to `f` in the `Staff` context.

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



Preventing natural signs from being printed when the key signature changes

When the key signature changes, natural signs are automatically printed to cancel any accidentals from previous key signatures. This may be prevented by setting to `f` the `printKeyCancellation` property in the `Staff` context.

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

}



Quoting another voice with transposition

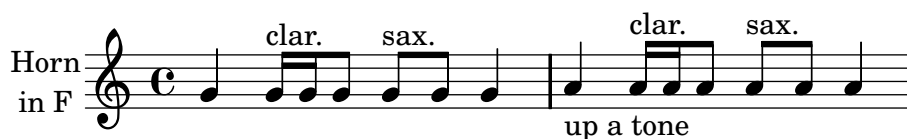
Quotations take into account the transposition of both source and target. In this example, all instruments play sounding middle C; the target is an instrument in F. The target part may be transposed using `\transpose`. In this case, all the pitches (including the quoted ones) are transposed.

```
\addQuote clarinet {
  \transposition bes
  \repeat unfold 8 { d'16 d' d'8 }
}

\addQuote sax {
  \transposition es'
  \repeat unfold 16 { a8 }
}

quoteTest = {
  % french horn
  \transposition f
  g'4
  << \quoteDuring #"clarinet" { \skip 4 } s4^"clar." >>
  << \quoteDuring #"sax" { \skip 4 } s4^"sax." >>
  g'4
}

{
  \set Staff.instrumentName =
    \markup {
      \center-column { Horn \line { in F } }
    }
  \quoteTest
  \transpose c' d' << \quoteTest s4_"up a tone" >>
}
```



Separating key cancellations from key signature changes

By default, the accidentals used for key cancellations are placed adjacent to those for key signature changes. This behavior can be changed by overriding the `'break-align-orders` property of the `BreakAlignment` grob.

The value of `'break-align-orders` is a vector of length 3, with quoted lists of breakable items as elements. This example only modifies the second list, moving `key-cancellation` before `staff-bar`; by modifying the second list, break alignment behavior only changes in the middle of a system, not at the beginning or the end.

```
\new Staff {
  \override Score.BreakAlignment #'break-align-orders =
    #'#((left-edge ambitus breathing-sign clef staff-bar
        key-cancellation key-signature time-signature custos)

        (left-edge ambitus breathing-sign clef key-cancellation
          staff-bar key-signature time-signature custos)

        (left-edge ambitus breathing-sign clef key-cancellation
          key-signature staff-bar time-signature custos))

  \key des \major
  c'1
  \bar "||"
  \key bes \major
  c'1
}
```



Transposing pitches with minimum accidentals ("Smart transpose")

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

Double accidentals should be removed

B sharp -> C

E sharp -> F

C flat -> B

F flat -> E

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

```
#(define (naturalize-pitch p)
  (let ((o (ly:pitch-octave p))
        (a (* 4 (ly:pitch-alteration p)))
        ;; alteration, a, in quarter tone steps,
        ;; for historical reasons
        (n (ly:pitch-notename p)))
    (cond
      ((and (> a 1) (or (eq? n 6) (eq? n 2)))
       (set! a (- a 2))
       (set! n (+ n 1)))
      ((and (< a -1) (or (eq? n 0) (eq? n 3)))
       (set! a (+ a 2))
       (set! n (- n 1)))))
```

```

(cond
  ((> a 2) (set! a (- a 4)) (set! n (+ n 1)))
  ((< a -2) (set! a (+ a 4)) (set! n (- n 1)))
  (if (< n 0) (begin (set! o (- o 1)) (set! n (+ n 7))))
  (if (> n 6) (begin (set! o (+ o 1)) (set! n (- n 7))))
  (ly:make-pitch o n (/ a 4)))

#(define (naturalize music)
  (let ((es (ly:music-property music 'elements))
        (e (ly:music-property music 'element))
        (p (ly:music-property music 'pitch)))
    (if (pair? es)
        (ly:music-set-property!
         music 'elements
         (map (lambda (x) (naturalize x)) es)))
        (if (ly:music? e)
            (ly:music-set-property!
             music 'element
             (naturalize e)))
            (if (ly:pitch? p)
                (begin
                 (set! p (naturalize-pitch p))
                 (ly:music-set-property! music 'pitch p)))
                music)))

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

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

\score {
  \new Staff {
    \transpose c ais { \music }
    \naturalizeMusic \transpose c ais { \music }
    \transpose c deses { \music }
    \naturalizeMusic \transpose c deses { \music }
  }
  \layout { }
}

```



Tweaking clef properties

The command `\clef "treble_8"` is equivalent to setting `clefGlyph`, `clefPosition` (which controls the vertical position of the clef), `middleCPosition` and `clefOctavation`. A clef is printed when any of the properties except `middleCPosition` are changed.

Note that changing the glyph, the position of the clef, or the octavation does not in itself change the position of subsequent notes on the staff: the position of middle C must also be specified to do this. The positional parameters are relative to the staff center line, positive numbers displacing upwards, counting one for each line and space. The `clefOctavation` value would normally be set to 7, -7, 15 or -15, but other values are valid.

When a clef change takes place at a line break the new clef symbol is printed at both the end of the previous line and the beginning of the new line by default. If the warning clef at the end of the previous line is not required it can be suppressed by setting the `Staff` property `explicitClefVisibility` to the value `end-of-line-invisible`. The default behavior can be recovered with `\unset Staff.explicitClefVisibility`.

The following examples show the possibilities when setting these properties manually. On the first line, the manual changes preserve the standard relative positioning of clefs and notes, whereas on the second line, they do not.

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

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

  % The following clef changes do not preserve
  % the normal relationship between notes and clefs:

  \set Staff.clefGlyph = #"clefs.F"
  \set Staff.clefPosition = #2
  c'1
  \set Staff.clefGlyph = #"clefs.G"
  c'1
  \set Staff.clefGlyph = #"clefs.C"
  c'1
  \set Staff.clefOctavation = #7
}
```

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

% Return to the normal clef:

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



Rhythms

Section “Rhythms” in *Notation Reference*

Adding beams slurs ties etc. when using tuplet and non-tuplet rhythms

LilyPond syntax can involve many unusual placements for parentheses, brackets etc., which might sometimes have to be interleaved. For example, when entering a manual beam, the left square bracket has to be placed after the starting note and its duration, not before. Similarly, the right square bracket should directly follow the note which is to be at the end of the requested beaming, even if this note happens to be inside a tuplet section. This snippet demonstrates how to combine manual beaming, manual slurs, ties and phrasing slurs with tuplet sections (enclosed within curly braces).

```
{
  r16[ g16 \times 2/3 { r16 e'8] }
  g16( a \times 2/3 { b d e' } )
  g8[( a \times 2/3 { b d' } e'] ~ }
  \time 2/4
  \times 4/5 { e'32\(( a b d' e' } a'4.\)
}
```



Adding drum parts

Using the powerful pre-configured tools such as the `\drummode` function and the `DrumStaff` context, inputting drum parts is quite easy: drums are placed at their own staff positions (with a special clef symbol) and have note heads according to the drum. Attaching an extra symbol to the drum or restricting the number of lines is possible.

```
drh = \drummode { cymc4.^"crash" hhc16^"h.h." hh hhc8 hho hhc8 hh16 hh hhc4 r4 r2 }
drl = \drummode { bd4 sn8 bd bd4 << bd ss >> bd8 tommh tommh bd tom1 tom1 bd tomfh16 tomfh
timb = \drummode { timh4 ssh timl8 ssh r timh r4 ssh8 timl r4 cb8 cb }
```

```
\score {
  <<
    \new DrumStaff \with {
      drumStyleTable = #timbales-style
      \override StaffSymbol #'line-count = #2
      \override BarLine #'bar-size = #2
    } <<
    \set Staff.instrumentName = #"timbales"
    \timb
  >>
  \new DrumStaff <<
    \set Staff.instrumentName = #"drums"
    \new DrumVoice { \stemUp \drh }
  >>
}
```

```

\new DrumVoice { \stemDown \drl }
>>
>>
\layout { }
\midi {
  \context {
    \Score
    tempoWholesPerMinute = #(ly:make-moment 120 4)
  }
}
}

```

Adjusting grace note spacing

The space given to grace notes can be adjusted using the `spacing-increment` property of `Score.GraceSpacing`.

```

graceNotes = {
  \grace { c4 c8 c16 c32 }
  c8
}

\relative c' {
  c8
  \graceNotes
  \override Score.GraceSpacing #'spacing-increment = #2.0
  \graceNotes
  \revert Score.GraceSpacing #'spacing-increment
  \graceNotes
}

```

Aligning bar numbers

Bar numbers by default are right-aligned to their parent object. This is usually the left edge of a line or, if numbers are printed within a line, the left hand side of a bar line. The numbers may also be positioned directly over the bar line or left-aligned to the bar line.

```

\relative c' {
  \set Score.currentBarNumber = #111
  \override Score.BarNumber #'break-visibility = #all-visible
}

```

```

% Increase the size of the bar number by 2
\override Score.BarNumber #'font-size = #2
% Print a bar number every second measure
\set Score.barNumberVisibility = #(every-nth-bar-number-visible 2)
c1 | c1
% Center-align bar numbers
\override Score.BarNumber #'self-alignment-X = #CENTER
c1 | c1
% Left-align bar numbers
\override Score.BarNumber #'self-alignment-X = #LEFT
c1 | c1
}

```



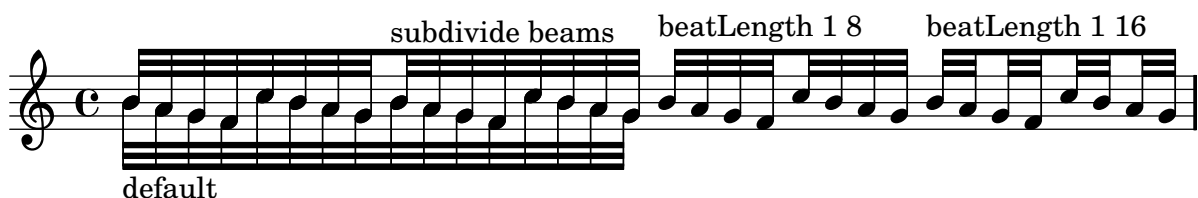
Automatic beam subdivisions

Beams can be subdivided automatically. By setting the property `subdivideBeams`, beams are subdivided at beat positions (as specified in `beatLength`).

```

\new Staff {
  \relative c' {
    <<
    {
      \voiceOne
      \set subdivideBeams = ##t
      b32[ a g f c' b a g
      b32^"subdivide beams" a g f c' b a g]
    }
    \new Voice {
      \voiceTwo
      b32_"default"[ a g f c' b a g
      b32 a g f c' b a g]
    }
  }
  >>
  \oneVoice
  \set beatLength = #(ly:make-moment 1 8)
  b32^"beatLength 1 8"[ a g f c' b a g]
  \set beatLength = #(ly:make-moment 1 16)
  b32^"beatLength 1 16"[ a g f c' b a g]
}

```



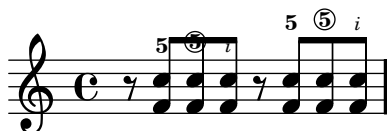
Avoiding collisions with chord fingerings

Fingerings and string numbers applied to individual notes will automatically avoid beams and stems, but this is not true by default for fingerings and string numbers applied to the individual notes of chords. The following example shows how this default behavior can be overridden.

```
\relative c' {
  \set fingeringOrientations = #'(up)
  \set stringNumberOrientations = #'(up)
  \set strokeFingerOrientations = #'(up)

  % Default behavior
  r8
  <f c'-5>8
  <f c'\5>8
  <f c'-\rightHandFinger #2 >8

  % Corrected to avoid collisions
  r8
  \override Fingering #'add-stem-support = ##t
  <f c'-5>8
  \override StringNumber #'add-stem-support = ##t
  <f c'\5>8
  \override StrokeFinger #'add-stem-support = ##t
  <f c'-\rightHandFinger #2 >8
}
```



Beam endings in Score context

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

```
\relative c'' {
  \time 5/4
  % Set default beaming for all staves
  \overrideBeamSettings #'Score #'(5 . 4) #'end
    #'(((1 . 8) . (3 4 3))
      ((1 . 16) . (6 8 6))
      ((1 . 32) . (12 16 12)))
  <<
  \new Staff {
    c8 c c c c c c c c c
  }
  \new Staff {
    % Modify beaming for just this staff
    \overrideBeamSettings #'Staff #'(5 . 4) #'end
      #'((* . (3 2)))
    c8 c c c c c c c c c
  }
}
```

```

\new Staff {
  % Inherit beaming from Score context
  <<
  {
    \voiceOne
    c8 c c c c c c c c c
  }
  % Modify beaming for this voice only
  \new Voice {
    \voiceTwo
    \overrideBeamSettings #'Voice #'(5 . 4) #'end
    #'((* . (3 2)))
    a8 a a a a a a a a a
  }
  >>
}
>>
}

```



Beam grouping in 7/8 time

There are no default automatic beam groupings specified for 7/8 time, so if automatic beams are required the grouping must be specified. For example, to group all beams 2-3-2 in 7/8 time, specify beam endings at 2/8 and 5/8:

```

\relative c'' {
  \time 7/8
  % rhythm 2-3-2
  a8 a a a a a a
  \overrideBeamSettings #'Score #'(7 . 8) #'end
  #'((* . (2 3 2)))
  a8 a a a a a a
}

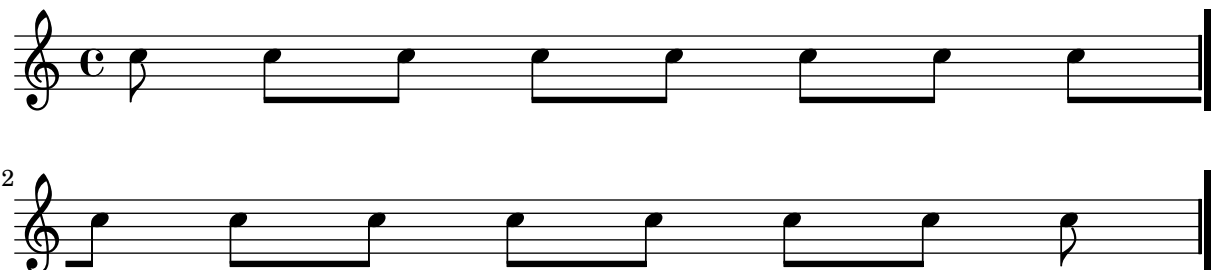
```



Beams across line breaks

Line breaks are normally forbidden when beams cross bar lines. This behavior can be changed as shown:

```
\relative c'' {
  \override Beam #'breakable = ##t
  c8 c[ c] c[ c] c[ c] c[ \break
  c8] c[ c] c[ c] c[ c] c
}
```



Changing beam knee gap

Kneed beams are inserted automatically when a large gap is detected between the note heads. This behavior can be tuned through the `auto-knee-gap` property. A kneed beam is drawn if the gap is larger than the value of `auto-knee-gap` plus the width of the beam object (which depends on the duration of the notes and the slope of the beam). By default `auto-knee-gap` is set to 5.5 staff spaces.

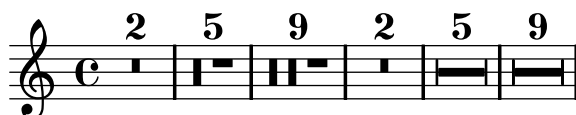
```
{
  f8 f''8 f8 f''8
  \override Beam #'auto-knee-gap = #6
  f8 f''8 f8 f''8
}
```



Changing form of multi-measure rests

If there are ten or fewer measures of rests, a series of longa and breve rests (called in German “Kirchenpausen” - church rests) is printed within the staff; otherwise a simple line is shown. This default number of ten may be changed by overriding the `expand-limit` property.

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



Changing the number of augmentation dots per note

This code demonstrates how to change the number of augmentation dots on a single note.

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



Changing the time signature without affecting the beaming

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

```
\markup {
  This snippet is deprecated as of 2.13.5 and will be removed in 2.14
}
```

This snippet is deprecated as of 2.13.5 and will be removed in 2.14

Changing the tuplet number

By default, only the numerator of the tuplet number is printed over the tuplet bracket, i.e., the denominator of the argument to the `\times` command. Alternatively, `num:den` of the tuplet number may be printed, or the tuplet number may be suppressed altogether.

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



Changing time signatures inside a polymetric section using `\scaleDurations`

The `measureLength` property, together with `measurePosition`, determines when a bar line is needed. However, when using `\scaleDurations`, the scaling of durations makes it difficult to change time signatures. In this case, `measureLength` should be set manually, using the `ly:make-moment` callback. The second argument must be the same as the second argument of `\scaleDurations`.

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

<<
\new Staff {
  \scaleDurations #'(8 . 5) {
    \time 6/8
    \set Timing.measureLength = #(ly:make-moment 6 5)
    b8 b b b b b
    \time 2/4
    \set Timing.measureLength = #(ly:make-moment 4 5)
    b4 b
  }
}
\new Staff {
  \clef bass
  \time 2/4
  c2 d e f
}
>>
```



Chant or psalms notation

This form of notation is used for the chant of the Psalms, where verses aren't always the same length.

```
stemOn = { \revert Staff.Stem #'transparent }
stemOff = { \override Staff.Stem #'transparent = ##t }
```

```
\score {
  \new Staff \with { \remove "Time_signature_engraver" }
  {
    \key g \minor
    \cadenzaOn
    \stemOff a'\breve bes'4 g'4
    \stemOn a'2 \bar "||"
    \stemOff a'\breve g'4 a'4
    \stemOn f'2 \bar "||"
    \stemOff a'\breve^{\markup { \italic flexe }}
    \stemOn g'2 \bar "||"
  }
}
```



Compound time signatures

Odd 20th century time signatures (such as "5/8") can often be played as compound time signatures (e.g. "3/8 + 2/8"), which combine two or more unequal metrics. LilyPond can make such music quite easy to read and play, by explicitly printing the compound time signatures and adapting the automatic beaming behavior. (Graphic measure grouping indications can also be added; see the appropriate snippet in this database.)

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

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

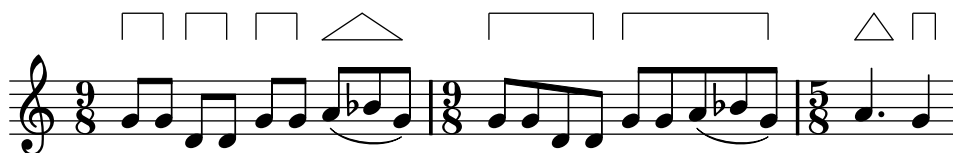


Conducting signs measure grouping signs

Beat grouping within a bar is controlled by the default grouping established in `beamSettings`. This grouping can be established by the use of `\overrideBeamSettings`. Alternatively, the

Scheme function `set-time-signature` can be used to both set the time signature and establish the default grouping rule. `set-time-signature`, takes three arguments: the number of beats, the beat length, and the internal grouping of beats in the measure. If the `Measure_grouping_engraver` is included in one of the display contexts, measure grouping signs will be created. Such signs ease reading rhythmically complex modern music. In the example, the 9/8 measure is grouped in two different patterns using the two different methods, while the 5/8 measure is grouped according to the default setting in `'scm/beam-settings.scm'`:

```
\score {
  \relative c'' {
    \time 9/8
    \overrideBeamSettings #'Score #'(9 . 8) #'end #'((* . (2 2 2 3)))
    g8 g d d g g a( bes g) |
    #(set-time-signature 9 8 '(4 5))
    g8 g d d g g a( bes g) |
    \time 5/8
    a4. g4 |
  }
  \layout {
    \context {
      \Staff
      \consists "Measure_grouping_engraver"
    }
  }
}
```



Controlling tuplet bracket visibility

The default behavior of tuplet-bracket visibility is to print a bracket unless there is a beam of the same length as the tuplet. To control the visibility of tuplet brackets, set the property `'bracket-visibility` to either `#t` (always print a bracket), `#f` (never print a bracket) or `#'if-no-beam` (only print a bracket if there is no beam).

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

\new Voice {
  \relative c' {
    << \music s4^"default" >>
    \override TupletBracket #'bracket-visibility = #'if-no-beam
    << \music s4^""if-no-beam" >>
    \override TupletBracket #'bracket-visibility = ##t
    << \music s4^"#t" >>
    \override TupletBracket #'bracket-visibility = ##f
    << \music s4^"#f" >>
  }
}
```

```
}
}
```



Engraving ties manually

Ties may be engraved manually by changing the `tie-configuration` property of the `TieColumn` object. The first number indicates the distance from the center of the staff in staff-spaces, and the second number indicates the direction (1 = up, -1 = down).

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



Engraving tremolos with floating beams

If a tremolo's total duration is less than a quarter-note, or exactly a half-note, or between a half-note and a whole-note, it is normally typeset with all beams touching the stems. Certain engraving styles typeset some of these beams as centered floating beams that do not touch the stems. The number of floating beams in this type of tremolo is controlled with the `'gap-count` property of the `Beam` object, and the size of the gaps between beams and stems is set with the `'gap` property.

```
\relative c' {
  \repeat tremolo 8 { a32 f }
  \override Beam #'gap-count = #1
  \repeat tremolo 8 { a32 f }
  \override Beam #'gap-count = #2
  \repeat tremolo 8 { a32 f }
  \override Beam #'gap-count = #3
  \repeat tremolo 8 { a32 f }

  \override Beam #'gap-count = #3
  \override Beam #'gap = #1.33
  \repeat tremolo 8 { a32 f }
}
```

```

\override Beam #'gap = #1
\repeat tremolo 8 { a32 f }
\override Beam #'gap = #0.67
\repeat tremolo 8 { a32 f }
\override Beam #'gap = #0.33
\repeat tremolo 8 { a32 f }
}

```



Entering several tuplets using only one `\times` command

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

In the example, two triplets are shown, while `\times` was entered only once.

Read the relevant sections of the Notation Reference for more information about `ly:make-moment`.

```

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

```



Flat flags and beam nibs

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

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

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

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

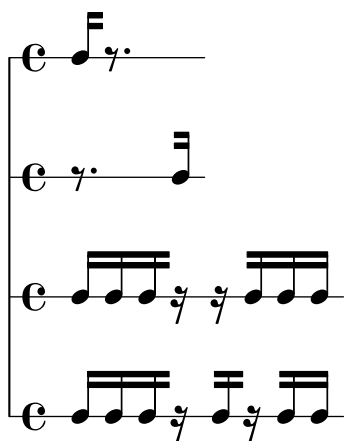
Sometimes it may make sense for a lone note surrounded by rests to carry both a left- and right-pointing flat flag. Do this with paired `[]` beam indicators alone (Example 4).

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

```

\score {
  <<
    % Example 1
    \new RhythmicStaff {
      \set stemLeftBeamCount = #0
      c16[]
      r8.
    }
    % Example 2
    \new RhythmicStaff {
      r8.
      \set stemRightBeamCount = #0
      c16[]
    }
    % Example 3
    \new RhythmicStaff {
      c16 c
      \set stemRightBeamCount = #2
      c16 r r
      \set stemLeftBeamCount = #2
      c16 c c
    }
    % Example 4
    \new RhythmicStaff {
      c16 c
      \set stemRightBeamCount = #2
      c16 r
      c16[]
      r16
      \set stemLeftBeamCount = #2
      c16 c
    }
  >>
}

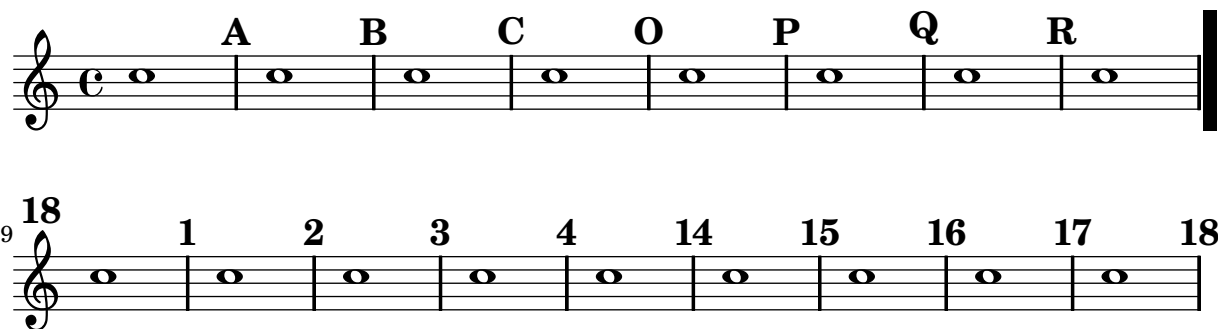
```



Forcing rehearsal marks to start from a given letter or number

This snippet demonstrates how to obtain automatic ordered rehearsal marks, but from the letter or number desired.

```
\relative c'' {
  c1 \mark \default
  c1 \mark \default
  c1 \mark \default
  c1 \mark #14
  c1 \mark \default
  c1 \mark \default
  c1 \mark \default
  c1 \mark \default
  \break
  \set Score.markFormatter = #format-mark-numbers
  c1 \mark #1
  c1 \mark \default
  c1 \mark \default
  c1 \mark \default
  c1 \mark #14
  c1 \mark \default
  c1 \mark \default
  c1 \mark \default
  c1 \mark \default
}
```



Grouping beats

Beaming patterns may be altered with the `beatGrouping` property:

```
\markup {
  This snippet is deprecated as of version 2.13.5 and will be removed
  in version 2.14.
}
```

This snippet is deprecated as of version 2.13.5 and will be removed in version 2.14.

Guitar strum rhythms

For guitar music, it is possible to show strum rhythms, along with melody notes, chord names and fret diagrams.

```

\include "predefined-guitar-fretboards.ly"
<<
  \new ChordNames {
    \chordmode {
      c1 | f | g | c
    }
  }
  \new FretBoards {
    \chordmode {
      c1 | f | g | c
    }
  }
  \new Voice \with {
    \consists "Pitch_squash_engraver"
  } {
    \relative c'' {
      \improvisationOn
      c4 c8 c c4 c8 c
      f4 f8 f f4 f8 f
      g4 g8 g g4 g8 g
      c4 c8 c c4 c8 c
    }
  }
  \new Voice = "melody" {
    \relative c'' {
      c2 e4 e4
      f2. r4
      g2. a4
      e4 c2.
    }
  }
  \new Lyrics {
    \lyricsto "melody" {
      This is my song.
      I like to sing.
    }
  }
>>

```

The image displays a musical score for a song. It features two staves: a guitar fretboard staff at the top and a vocal melody staff below it. The guitar staff shows four chords: C (fretboard diagram), F (fretboard diagram), G (fretboard diagram), and C (fretboard diagram). The vocal staff shows the melody for the lyrics "This is my song. I like to sing." The melody is written in a simple, accessible style, with notes corresponding to the lyrics. The lyrics are "This is my song. I like to sing." and are written below the vocal staff.

Heavily customized polymetric time signatures

Though the polymetric time signature shown was not the most essential item here, it has been included to show the beat of this piece (which is the template of a real Balkan song!).

```
#(define plus (markup #:vcenter "+"))
#(define ((custom-time-signature one two three four five six
                                seven eight nine ten eleven num) grob)
  (grob-interpret-markup grob
    (markup #:override '(baseline-skip . 0) #:number
      (#:line ((#:column (one num)) plus
                    (:column (two num)) plus
                    (:column (three num)) plus
                    (:column (four num)) plus
                    (:column (five num)) plus
                    (:column (six num)) plus
                    (:column (seven num)) plus
                    (:column (eight num)) plus
                    (:column (nine num)) plus
                    (:column (ten num)) plus
                    (:column (eleven num)))))))

melody = \relative c'' {
  \set Staff.instrumentName = #"Bb Sop."
  \key g \major
  #(set-time-signature 25 8 '(3 2 2 3 2 2 2 2 3 2 2))
  \override Staff.TimeSignature #'stencil =
    #(custom-time-signature "3" "2" "2" "3" "2" "2"
      "2" "2" "3" "2" "2" "8")
  c8 c c d4 c8 c b c b a4 g fis8 e d c b' c d e4-^ fis8 g \break
  c,4. d4 c4 d4. c4 d c2 d4. e4-^ d4
  c4. d4 c4 d4. c4 d c2 d4. e4-^ d4 \break
  c4. d4 c4 d4. c4 d c2 d4. e4-^ d4
  c4. d4 c4 d4. c4 d c2 d4. e4-^ d4 \break
}

drum = \new DrumStaff \drummode {
  \bar "|:" bd4.^ \markup { Drums } sn4 bd \bar ":" sn4.
  bd4 sn \bar ":" bd sn bd4. sn4 bd \bar ":@"
}

{
  \melody
  \drum
}
```



4

6

Drums

Making an object invisible with the 'transparent' property

Setting the `transparent` property will cause an object to be printed in “invisible ink”: the object is not printed, but all its other behavior is retained. The object still takes up space, it takes part in collisions, and slurs, ties and beams can be attached to it.

This snippet demonstrates how to connect different voices using ties. Normally, ties only connect two notes in the same voice. By introducing a tie in a different voice, and blanking the first up-stem in that voice, the tie appears to cross voices.

```
\relative c'' {
  \time 2/4
  <<
  {
    \once \override Stem #'transparent = ##t
    \once \override Stem #'length = #8
    b8 ~ b\noBeam
    \once \override Stem #'transparent = ##t
    \once \override Stem #'length = #8
    g8 ~ g\noBeam
  }
  \\
  {
    b8 g g e
  }
  >>
}
```

Making slurs with complex dash structure

Slurs can be made with complex dash patterns by defining the `dash-definition` property. `dash-definition` is a list of `dash-elements`. A `dash-element` is a list of parameters defining the dash behavior for a segment of the slur.

The slur is defined in terms of the bezier parameter `t` which ranges from 0 at the left end of the slur to 1 at the right end of the slur. `dash-element` is a list (`start-t stop-t dash-fraction dash-period`). The region of the slur from `start-t` to `stop-t` will have a fraction `dash-fraction` of each `dash-period` black. `dash-period` is defined in terms of staff spaces. `dash-fraction` is set to 1 for a solid slur.

```
\relative c' {
  \once \override
    Slur #'dash-definition = #'((0 0.3 0.1 0.75)
                                (0.3 0.6 1 1)
                                (0.65 1.0 0.4 0.75))
```

```

c4( d e f)
\once \override
  Slur #'dash-definition = #'((0 0.25 1 1)
                                (0.3 0.7 0.4 0.75)
                                (0.75 1.0 1 1))

c4( d e f)
}

```



Manually controlling beam positions

Beam positions may be controlled manually, by overriding the `positions` setting of the `Beam` grob.

```

\relative c' {
  \time 2/4
  % from upper staff-line (position 2) to center (position 0)
  \override Beam #'positions = #'(2 . 0)
  c8 c
  % from center to one above center (position 1)
  \override Beam #'positions = #'(0 . 1)
  c8 c
}

```



Merging multi-measure rests in a polyphonic part

When using multi-measure rests in a polyphonic staff, the rests will be placed differently depending on the voice they belong to. However they can be printed on the same staff line, using the following setting.

```
normalPos = \revert MultiMeasureRest #'staff-position
```

```

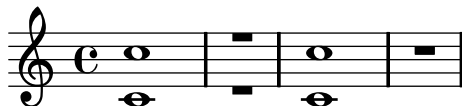
{
  <<
    {
      c''1
      R1
      c''1
      \normalPos
      R1
    }
    \\\
    {
      c'1
      R1
    }
  }

```

```

        c'1
        \normalPos
        R1
    }
    >>
}

```



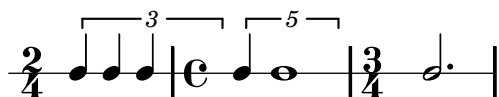
Modifying tuplet bracket length

Tuplet brackets can be made to run to prefatory matter or the next note. Default tuplet brackets end at the right edge of the final note of the tuplet; full-length tuplet brackets extend farther to the right, either to cover all the non-rhythmic notation up to the following note, or to cover only the whitespace before the next item of notation, be that a clef, time signature, key signature, or another note. The example shows how to switch tuplets to full length mode and how to modify what material they cover.

```

\new RhythmicStaff {
  % Set tuplets to be extendable...
  \set tupletFullLength = ##t
  % ...to cover all items up to the next note
  \set tupletFullLengthNote = ##t
  \time 2/4
  \times 2/3 { c4 c c }
  % ...or to cover just whitespace
  \set tupletFullLengthNote = ##f
  \time 4/4
  \times 4/5 { c4 c1 }
  \time 3/4
  c2.
}

```



Moving dotted notes in polyphony

When a dotted note in the upper voice is moved to avoid a collision with a note in another voice, the default is to move the upper note to the right. This behaviour can be over-ridden by using the `prefer-dotted-right` property of `NoteCollision`.

```

\new Staff \relative c' <<
{ f2. f4
  \override Staff.NoteCollision #'prefer-dotted-right = ##f
  f2. f4
  \override Staff.NoteCollision #'prefer-dotted-right = ##t
  f2. f4
}

```

```

\\
{ e4 e e e e e e e e e e }
>>

```



Multi-measure rest markup

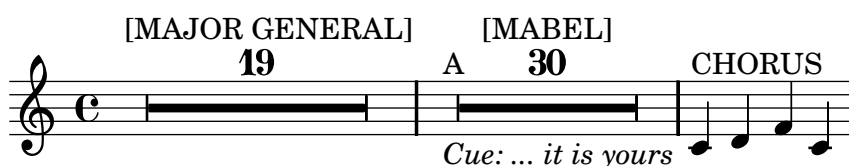
Markups attached to a multi-measure rest will be centered above or below it. Long markups attached to multi-measure rests do not cause the measure to expand. To expand a multi-measure rest to fit the markup, use a spacer rest with an attached markup before the multi-measure rest.

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

```

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

```



Non-default tuplet numbers

LilyPond also provides formatting functions to print tuplet numbers different than the actual fraction, as well as to append a note value to the tuplet number or tuplet fraction.

```

\relative c' {
  \once \override TupletNumber #'text =
    #(tuplet-number::non-default-tuplet-denominator-text 7)
  \times 2/3 { c4. c4. c4. c4. }
  \once \override TupletNumber #'text =
    #(tuplet-number::non-default-tuplet-fraction-text 12 7)
  \times 2/3 { c4. c4. c4. c4. }
  \once \override TupletNumber #'text =
    #(tuplet-number::append-note-wrapper
      (tuplet-number::non-default-tuplet-fraction-text 12 7) "8")
  \times 2/3 { c4. c4. c4. c4. }

  \once \override TupletNumber #'text =

```

```

    #(tuplet-number::append-note-wrapper
      tuplet-number::calc-denominator-text "4")
\time 2/3 { c8 c8 c8 c8 c8 c8 }
\once \override TupletNumber #'text =
    #(tuplet-number::append-note-wrapper
      tuplet-number::calc-fraction-text "4")
\time 2/3 { c8 c8 c8 c8 c8 c8 }

\once \override TupletNumber #'text =
    #(tuplet-number::fraction-with-notes "4." "8")
\time 2/3 { c4. c4. c4. c4. }
\once \override TupletNumber #'text =
    #(tuplet-number::non-default-fraction-with-notes 12 "8" 4 "4")
\time 2/3 { c4. c4. c4. c4. }
}

```



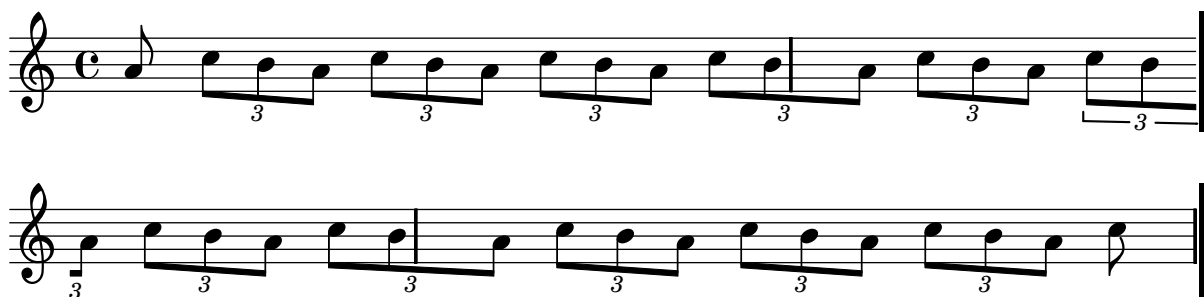
Permitting line breaks within beamed tuplets

This artificial example shows how both manual and automatic line breaks may be permitted to within a beamed tuplet. Note that such off-beat tuplets have to be beamed manually.

```

\layout {
  \context {
    \Voice
    % Permit line breaks within tuplets
    \remove "Forbid_line_break_engraver"
    % Allow beams to be broken at line breaks
    \override Beam #'breakable = ##t
  }
}
\relative c' {
  a8
  \repeat unfold 5 { \time 2/3 { c[ b a] } }
  % Insert a manual line break within a tuplet
  \time 2/3 { c[ b \bar "" \break a] }
  \repeat unfold 5 { \time 2/3 { c[ b a] } }
  c8
}

```



Positioning grace note beams at the height of normal note beams

When notes are placed on ledger lines, their beams are usually centred on the staff. Grace notes beams are shorter and grace notes on ledger lines may well have beams outside the staff. You can override this beaming for grace notes.

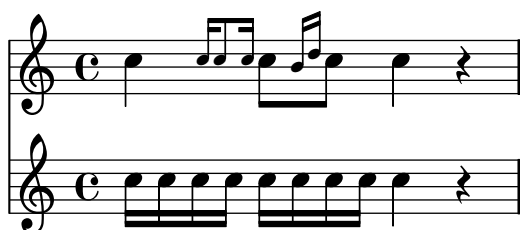
```
\relative c {
  f8[ e]
  \grace {
    f8[ e]
    \override Stem #'no-stem-extend = ##f
    f8[ e]
    \revert Stem #'no-stem-extend
  }
  f8[ e]
}
```



Positioning grace notes with floating space

Setting the property 'strict-grace-spacing makes the musical columns for grace notes 'floating', i.e., decoupled from the non-grace notes: first the normal notes are spaced, then the (musical columns of the) graces are put left of the musical columns for the main notes.

```
\relative c'' {
  <<
  \override Score.SpacingSpanner #'strict-grace-spacing = ##t
  \new Staff \new Voice {
    \afterGrace c4 { c16[ c8 c16] }
    c8[ \grace { b16[ d] } c8]
    c4 r
  }
  \new Staff {
    c16 c c c c c c c c4 r
  }
  >>
}
```



Positioning multi-measure rests

Unlike ordinary rests, there is no predefined command to change the staff position of a multi-measure rest symbol of either form by attaching it to a note. However, in polyphonic music

multi-measure rests in odd-numbered and even-numbered voices are vertically separated. The positioning of multi-measure rests can be controlled as follows:

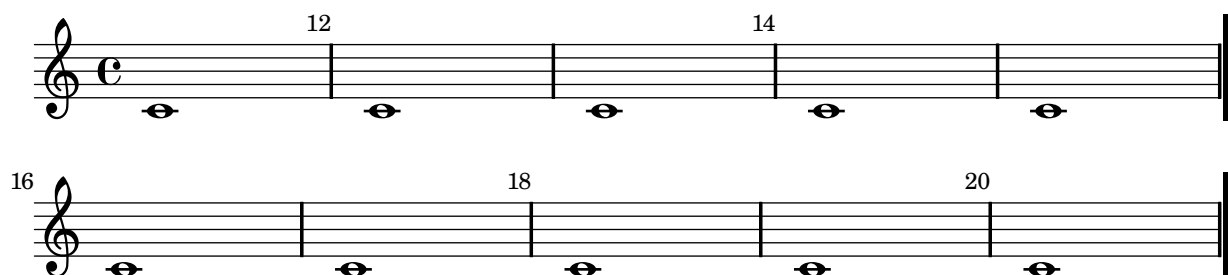
```
\relative c' {
  % MMR - Multi-Measure Rest
  % MMRs by default are set under the fourth line
  R1
  % They can be moved with an override
  \override MultiMeasureRest #'staff-position = #-2
  R1
  % A value of 0 is the default position;
  % the following trick moves the rest to the center line
  \override MultiMeasureRest #'staff-position = #-0.01
  R1
  % MMRs in odd-numbered voices are under the top line
  << { R1 } \\\ { a1 } >>
  % MMRs in even-numbered voices are under the bottom line
  << { c1 } \\\ { R1 } >>
  % They remain separated even in empty measures
  << { R1 } \\\ { R1 } >>
  % This brings them together even though there are two voices
  \compressFullBarRests
  <<
    \revert MultiMeasureRest #'staff-position
    { R1*3 }
    \\\
    \revert MultiMeasureRest #'staff-position
    { R1*3 }
  >>
}
```



Printing bar numbers at regular intervals

Bar numbers can be printed at regular intervals by setting the property `barNumberVisibility`. Here the bar numbers are printed every two measures except at the end of the line.

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



Printing bar numbers inside boxes or circles

Bar numbers can also be printed inside boxes or circles.

```
\relative c' {
  % Prevent bar numbers at the end of a line and permit them elsewhere
  \override Score.BarNumber #'break-visibility = #end-of-line-invisible
  \set Score.barNumberVisibility = #(every-nth-bar-number-visible 4)

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

  % Draw a box round the following bar number(s)
  \override Score.BarNumber #'stencil
    = #(make-stencil-boxer 0.1 0.25 ly:text-interface::print)
  \repeat unfold 5 { c1 }

  % Draw a circle round the following bar number(s)
  \override Score.BarNumber #'stencil
    = #(make-stencil-circler 0.1 0.25 ly:text-interface::print)
  \repeat unfold 4 { c1 } \bar "|."
}
```



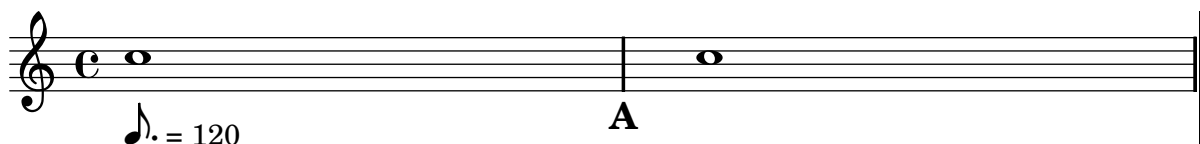
Printing metronome and rehearsal marks below the staff

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

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

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

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



Printing music with different time signatures

In the following snippet, two parts have a completely different time signature, yet remain synchronized. The bar lines can no longer be printed at the `Score` level; to allow independent bar lines in each part, the `Default_bar_line_engraver` and `Timing_translator` are moved from the `Score` context to the `Staff` context.

If bar numbers are required, the `Bar_number_engraver` should also be moved, since it relies on properties set by the `Timing_translator`; a `\with` block can be used to add bar numbers to the relevant staff.

```
\paper {
  indent = #0
  ragged-right = ##t
}

global = { \time 3/4 { s2.*3 } \bar "" \break { s2.*3 } }

\layout {
  \context {
    \Score
    \remove "Timing_translator"
    \remove "Default_bar_line_engraver"
    \remove "Bar_number_engraver"
    \override SpacingSpanner #'uniform-stretching = ##t
    \override SpacingSpanner #'strict-note-spacing = ##t
    proportionalNotationDuration = #(ly:make-moment 1 64)
  }
  \context {
    \Staff
    \consists "Timing_translator"
    \consists "Default_bar_line_engraver"
  }
  \context {
    \Voice
    \remove "Forbid_line_break_engraver"
    tupletFullLength = ##t
  }
}

Bassklarinette = \new Staff \with {
  \consists "Bar_number_engraver"
  barNumberVisibility = #(every-nth-bar-number-visible 2)
  \override BarNumber #'break-visibility = #end-of-line-invisible
} <<
\global {
  \bar "|"
  \clef treble
  \time 3/8
  d' '4.
```

```

\bar "|"
\time 3/4
r8 des''2( c''8)

\bar "|"
\time 7/8
r4. ees''2 ~

\bar "|"
\time 2/4
\tupletUp
\times 2/3 { ees''4 r4 d''4 ~ }

\bar "|"
\time 3/8
\tupletUp
\times 3/4 { d''4 r4 }

\bar "|"
\time 2/4
e''2

\bar "|"
\time 3/8
es''4.

\bar "|"
\time 3/4
r8 d''2 r8
\bar "|"
}
>>

```

```

Perkussion = \new StaffGroup <<
\new Staff <<
\global {
\bar "|"
\clef percussion
\time 3/4
r4 c'2 ~

\bar "|"
c'2.

\bar "|"
R2.

\bar "|"
r2 g'4 ~

\bar "|"
g'2. ~

```

```

        \bar "|"
        g'2.
    }
>>
\new Staff <<
  \global {
    \bar "|"
    \clef percussion
    \time 3/4
    R2.

    \bar "|"
    g'2. ~

    \bar "|"
    g'2.

    \bar "|"
    r4 g'2 ~

    \bar "|"
    g'2 r4

    \bar "|"
    g'2.
  }
>>
>>

\score {
  <<
    \Bassklarinette
    \Perkussion
  >>
}

```

The musical score is written for Bass Clarinet and Percussion. It consists of three measures. The first measure is for the Bass Clarinet, with a treble clef and a 3/4 time signature. It contains a half note G4, a quarter rest, and a quarter note G4. The second and third measures are for the Percussion, with a bass clef and a 3/4 time signature. They contain a half note G4, a quarter rest, and a quarter note G4. The score is written for Bass Clarinet and Percussion.

The image shows two systems of musical notation. The first system consists of a single staff with a treble clef and a key signature of one flat (B-flat). It contains several measures with complex rhythms, including triplets (marked '3'), groups of four (marked '4'), and groups of six (marked '6'). The second system consists of two staves, both with a treble clef and a key signature of one flat. The first staff begins with a measure number '8' and contains a series of notes and rests. The second staff contains a series of notes and rests, with a final measure marked by a double bar line.

Printing the bar number for the first measure

By default, the first bar number in a score is suppressed if it is less than or equal to '1'. By setting `barNumberVisibility` to `all-bar-numbers-visible`, any bar number can be printed for the first measure and all subsequent measures. Note that an empty bar line must be inserted before the first note for this to work.

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

The image shows two systems of musical notation. The first system consists of a single staff with a treble clef and a key signature of one flat (B-flat). It contains four measures, each with a single note. The first measure is marked with a bar number '1'. The second system consists of a single staff with a treble clef and a key signature of one flat. It contains four measures, each with a single note. The first measure is marked with a bar number '5'.

Redefining grace note global defaults

The global defaults for grace notes are stored in the identifiers `startGraceMusic`, `stopGraceMusic`, `startAcciaccaturaMusic`, `stopAcciaccaturaMusic`, `startAppoggiaturaMusic` and `stopAppoggiaturaMusic`, which are defined in the file `ly/grace-init.ly`. By redefining them other effects may be obtained.

```
startAcciaccaturaMusic = {
  s1*0(
    \override Stem #'stroke-style = #"grace"
    \slurDashed
  }
}
```

```
stopAcciaccaturaMusic = {
  \revert Stem #'stroke-style
  \slurSolid
  s1*0)
}

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

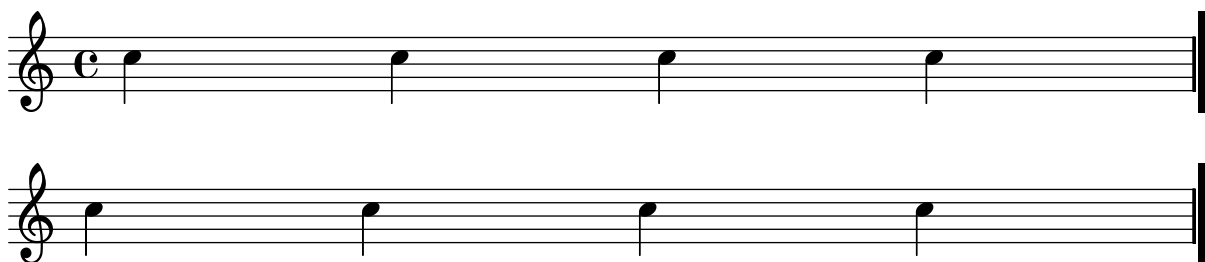


Removing bar numbers from a score

Bar numbers can be removed entirely by removing the `Bar_number_engraver` from the `Score` context.

```
\layout {
  \context {
    \Score
    \remove "Bar_number_engraver"
  }
}

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



Removing connecting bar lines on StaffGroup PianoStaff or GrandStaff

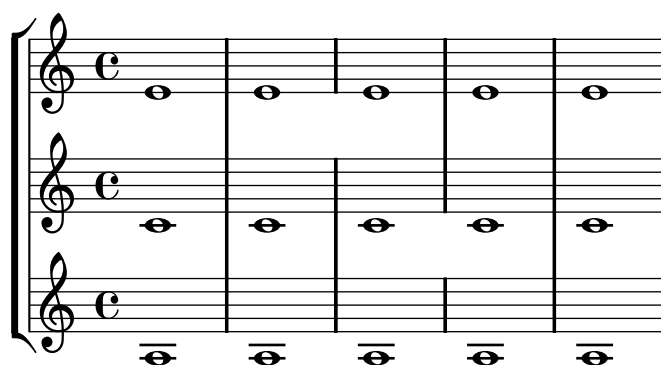
By default, bar lines in `StaffGroup`, `PianoStaff`, or `GrandStaff` groups are connected between the staves. This behaviour can be overridden on a staff-by-staff basis.

```
\relative c' {
  \new StaffGroup <<
    \new Staff {
      e1 | e
      \once \override Staff.BarLine #'allow-span-bar = ##f
      e1 | e | e
    }
  }
}
```

```

    }
    \new Staff {
      c1 | c | c
      \once \override Staff.BarLine #'allow-span-bar = ##f
      c1 | c
    }
    \new Staff {
      a1 | a | a | a | a
    }
  >>
}

```



Rest styles

Rests may be used in various styles.

```

\layout {
  indent = 0
  \context {
    \Staff
    \remove "Time_signature_engraver"
  }
}

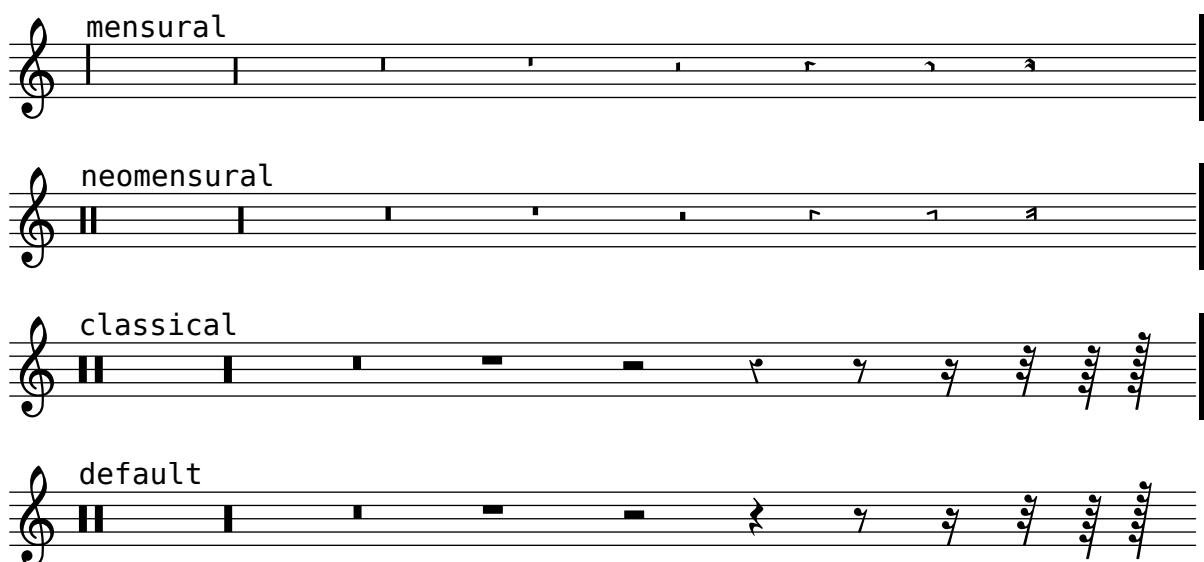
\new Staff \relative c {
  \cadenzaOn
  \override Staff.Rest #'style = #'mensural
  r\maxima^\markup \typewriter { mensural }
  r\longa r\breve r1 r2 r4 r8 r16 s32 s64 s128 s128
  \bar ""

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

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

```

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



Reverting default beam endings

To typeset beams grouped 3-4-3-2 in 12/8 it is no longer necessary first to override the default beam endings in 12/8, and then to set up the new beaming endings:

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

  % Default beaming
  a8 a a a a a a a a a a

  % Set new values for beam endings
  \overrideBeamSettings #'Score #'(12 . 8) #'end
  #'(( * . (3 4 3 2)))
  a8 a a a a a a a a a a
}
```



Rhythmic slashes

In “simple” lead-sheets, sometimes no actual notes are written, instead only “rhythmic patterns” and chords above the measures are notated giving the structure of a song. Such a feature is for example useful while creating/transcribing the structure of a song and also when sharing lead sheets with guitarists or jazz musicians. The standard support for this using `\repeat percent` is unsuitable here since the first beat has to be an ordinary note or rest. This example shows two solutions to this problem, by redefining ordinary rests to be printed as slashes. (If the duration of each beat is not a quarter note, replace the `r4` in the definitions with a rest of the appropriate duration).

```
% Macro to print single slash
rs = {
  \once \override Rest #'stencil = #ly:percent-repeat-item-interface::beat-slash
  \once \override Rest #'thickness = #0.48
  \once \override Rest #'slope = #1.7
  r4
}

% Function to print a specified number of slashes
comp = #(define-music-function (parser location count) (integer?)
  #{
    \override Rest #'stencil = #ly:percent-repeat-item-interface::beat-slash
    \override Rest #'thickness = #0.48
    \override Rest #'slope = #1.7
    \repeat unfold $count { r4 }
    \revert Rest #'stencil
  }
)

\score {
  \relative c' {
    c4 d e f |
    \rs \rs \rs \rs |
    \comp #4 |
  }
}
```



Skips in lyric mode (2)

Although `s` skips cannot be used in `\lyricmode` (it is taken to be a literal “s”, not a space), double quotes (“”) or underscores (`_`) are available. So for example:

```
<<
  \relative c' { a4 b c d }
  \new Lyrics \lyricmode { a4 " _ gap }
>>
```

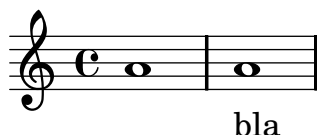


Skips in lyric mode

The `s` syntax for skips is only available in note mode and chord mode. In other situations, for example, when entering lyrics, using the `\skip` command is recommended.

```
<<
```

```
\relative c'' { a1 | a }
\new Lyrics \lyricmode { \skip 1 bla1 }
>>
```



Stemlets

In some notational conventions beams are allowed to extend over rests. Depending on preference, these beams may drop 'stemlets' to help the eye appreciate the rhythm better, and in some modern music the rest itself is omitted and only the stemlet remains.

This snippet shows a progression from traditional notation, to beams over the rest, to stemlets over the rest, to stemlets alone. Stemlets are generated by overriding the 'stemlet-length' property of `Stem`, while rests are hidden by setting 'transparent = ##t'.

Some `\markup` elements are included in the source to highlight the different notations.

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

{
  c'16^\markup { traditional } d' r f'
  g'16[^\markup { beams over rests } f' r d']

  % N.B. use Score.Stem to set for the whole score.
  \override Staff.Stem #'stemlet-length = #0.75

  c'16[^\markup { stemlets over rests } d' r f']
  g'16[^\markup { stemlets and no rests } f'
  \once \override Rest #'transparent = ##t
  r16 d']
}
```



Sub-dividing beams

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

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

```

c32[ c c c c c c c c]

% Set beam sub-group length to an eighth note
\set beatLength = #(ly:make-moment 1 8)
c32[ c c c c c c c c]

% Set beam sub-group length to a sixteenth note
\set beatLength = #(ly:make-moment 1 16)
c32[ c c c c c c c c]
}

```



Three-sided box

This example shows how to add a markup command to get a three sided box around some text (or other markup).

```

% New command to add a three sided box, with sides north, west and south
% Based on the box-stencil command defined in scm/stencil.scm
% Note that ";;" is used to comment a line in Scheme
#(define-public (NWS-box-stencil stencil thickness padding)
  "Add a box around STENCIL, producing a new stencil."
  (let* ((x-ext (interval-widen (ly:stencil-extent stencil X) padding))
        (y-ext (interval-widen (ly:stencil-extent stencil Y) padding))
        (y-rule (make-filled-box-stencil (cons 0 thickness) y-ext))
        (x-rule (make-filled-box-stencil
                  (interval-widen x-ext thickness) (cons 0 thickness))))
    ;; (set! stencil (ly:stencil-combine-at-edge stencil X 1 y-rule padding))
    (set! stencil (ly:stencil-combine-at-edge stencil X LEFT y-rule padding))
    (set! stencil (ly:stencil-combine-at-edge stencil Y UP x-rule 0.0))
    (set! stencil (ly:stencil-combine-at-edge stencil Y DOWN x-rule 0.0))
    stencil))

% The corresponding markup command, based on the \box command defined
% in scm/define-markup-commands.scm
#(define-markup-command (NWS-box layout props arg) (markup?)
  "Draw a box round @var{arg}. Looks at @code{thickness},
@code{box-padding} and @code{font-size} properties to determine line
thickness and padding around the markup."
  (let* ((th (chain-assoc-get 'thickness props 0.1))
        (size (chain-assoc-get 'font-size props 0))
        (pad (* (magstep size)
                 (chain-assoc-get 'box-padding props 0.2)))
        (m (interpret-markup layout props arg)))
    (NWS-box-stencil m th pad)))

% Test it:

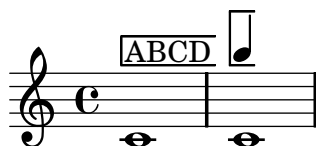
\relative c' {

```

```

c1^\markup { \NWS-box ABCD }
c1^\markup { \NWS-box \note #"4" #1.0 }
}

```



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

Sometimes, a time signature should not print the whole fraction (e.g. 7/4), but only the numerator (7 in this case). This can be easily done by using `\override Staff.TimeSignature #'style = #'single-digit` to change the style permanently. By using `\revert Staff.TimeSignature #'style`, this setting can be reversed. To apply the single-digit style to only one time signature, use the `\override` command and prefix it with a `\once`.

```

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

```



Tweaking grace layout within music

The layout of grace expressions can be changed throughout the music using the functions `add-grace-property` and `remove-grace-property`. The following example undefines the `Stem` direction for this grace, so that stems do not always point up, and changes the default note heads to crosses.

```

\relative c'' {
  \new Staff {

```

```

#(remove-grace-property 'Voice 'Stem 'direction)
#(add-grace-property 'Voice 'NoteHead 'style 'cross)
\new Voice {
  \acciaccatura { f16 } g4
  \grace { d16[ e] } f4
  \appoggiatura { f,32[ g a] } e2
}
}
}

```



Using alternative flag styles

Alternative styles of flag on eighth and shorter notes can be displayed by overriding the `flag` property of `Stem`. Valid values are `modern-straight-flag` and `old-straight-flag`.

```

testnotes = {
  \autoBeamOff
  c8 d16 c32 d64 \acciaccatura { c8 } d64 r4
}

\relative c' {
  \time 2/4
  \testnotes

  \override Stem #'flag = #modern-straight-flag
  \testnotes

  \override Stem #'flag = #old-straight-flag
  \testnotes

  \revert Stem #'flag
  \testnotes
}

```



Using grace note slashes with normal heads

The slash through the stem found in acciaccaturas can be applied in other situations.

```

\relative c' {
  \override Stem #'stroke-style = #"grace"

```

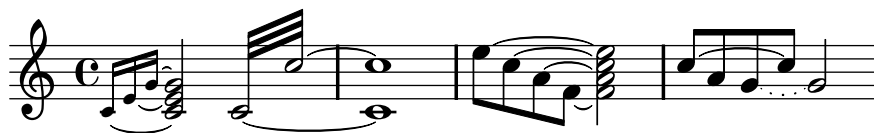
```
c8( d2) e8( f4)
}
```



Using ties with arpeggios

Ties are sometimes used to write out arpeggios. In this case, two tied notes need not be consecutive. This can be achieved by setting the `tieWaitForNote` property to `#t`. The same feature is also useful, for example, to tie a tremolo to a chord, but in principle, it can also be used for ordinary consecutive notes.

```
\relative c' {
  \set tieWaitForNote = ##t
  \grace { c16[ ~ e ~ g] ~ } <c, e g>2
  \repeat tremolo 8 { c32 ~ c' ~ } <c c,>1
  e8 ~ c ~ a ~ f ~ <e' c a f>2
  \tieUp
  c8 ~ a
  \tieDown
  \tieDotted
  g8 ~ c g2
}
```



Expressive marks

Section “Expressive marks” in *Notation Reference*

The presence of an accidental after a broken tie can be overridden.

```
\version "2.12.0"
```

```
\header {
  texidoc = "The presence of an accidental after a broken tie can be
  overridden."
}
\layout {
  ragged-right = ##t
}
```

```
mus = \relative c' {
  \override Accidental #'hide-tied-accidental-after-break = ##t
  f1~
  f2~f4 % ~ f8
  fis8 gis8 ~
  \break
  gis1
}
```

```
<<
  \new NoteNames \mus
  \new Voice { \key g \major \mus }
>>
```



Adding beams slurs ties etc. when using tuplet and non-tuplet rhythms

LilyPond syntax can involve many unusual placements for parentheses, brackets etc., which might sometimes have to be interleaved. For example, when entering a manual beam, the left square bracket has to be placed after the starting note and its duration, not before. Similarly, the right square bracket should directly follow the note which is to be at the end of the requested beaming, even if this note happens to be inside a tuplet section. This snippet demonstrates how to combine manual beaming, manual slurs, ties and phrasing slurs with tuplet sections (enclosed within curly braces).

```
{
  r16[ g16 \times 2/3 { r16 e'8] }
  g16( a \times 2/3 { b d e') }
  g8[( a \times 2/3 { b d') e'] ~ }
  \time 2/4
```

```
\times 4/5 { e'32\(( a b d' e' } a'4.\)
```



Adding parentheses around an expressive mark or chordal note

The `\parenthesize` function is a special tweak that encloses objects in parentheses. The associated grob is `ParenthesesItem`.

```
\relative c' {
  c2-\parenthesize ->
  \override ParenthesesItem #'padding = #0.1
  \override ParenthesesItem #'font-size = #-4
  <d \parenthesize f a>2
}
```



Adjusting the shape of falls and doits

The `shortest-duration-space` property may have to be tweaked to adjust the shape of falls and doits.

```
\relative c' {
  \override Score.SpacingSpanner #'shortest-duration-space = #4.0
  c2-\bendAfter #5
  c2-\bendAfter #-4.75
  c2-\bendAfter #8.5
  c2-\bendAfter #-6
}
```



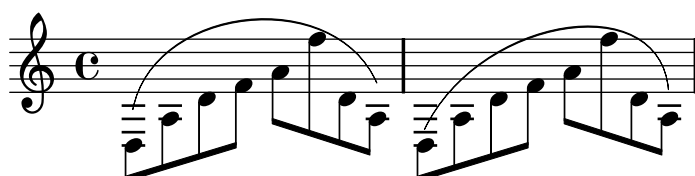
Asymmetric slurs

Slurs can be made asymmetric to match an asymmetric pattern of notes better.

```
slurNotes = { d,8( a' d f a f' d, a) }
```

```
\relative c' {
  \stemDown
  \slurUp
  \slurNotes
```

```
\once \override Slur #'eccentricity = #3.0
\slurNotes
}
```



Breathing signs

Breathing signs are available in different tastes: commas (default), ticks, vees and “railroad tracks” (caesura).

```
\new Staff \relative c'' {
  \key es \major
  \time 3/4
  % this bar contains no \breathe
  << { g4 as g } \ { es4 bes es } >> |
  % Modern notation:
  % by default, \breathe uses the rcomma, just as if saying:
  % \override BreathingSign #'text = \markup { \musicglyph #"scripts.rcomma" }
  << { g4 as g } \ { es4 \breathe bes es } >> |

  % rvarcomma and lvarcomma are variations of the default rcomma and lcomma
  % N.B.: must use Staff context here, since we start a Voice below
  \override Staff.BreathingSign #'text = \markup { \musicglyph #"scripts.rvarcomma" }
  << { g4 as g } \ { es4 \breathe bes es } >> |

  % vee
  \override BreathingSign #'text = \markup { \musicglyph #"scripts.upbow" }
  es8[ d es f g] \breathe f |

  % caesura
  \override BreathingSign #'text = \markup { \musicglyph #"scripts.caesura.curved" }
  es8[ d] \breathe es[ f g f] |
  es2 r4 \bar "||"
}
```



Broken Crescendo Hairpin

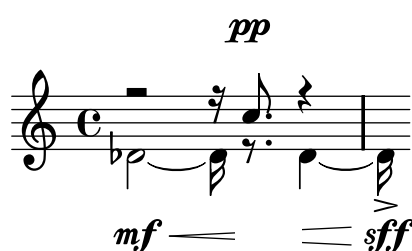
In order to make parts of a crescendo hairpin invisible, the following method is used: A white rectangle is drawn on top of the respective part of the crescendo hairpin, making it invisible. The rectangle is defined as postscript code within a text markup.

To fine-tune the position and size of the markup, the number preceding `setgray` in the postscript definition can be set to a value less than one, making it grey. The two numbers before

`scale` in the postscript code are responsible for the width and height of the rectangle, the two numbers before `translate` change the x- and y-origin of the rectangle.

Make sure to put the hairpin in a lower layer than the text markup to draw the rectangle over the hairpin.

```
\relative c' {
  <<
  {
    \dynamicUp
    \override DynamicLineSpanner #'staff-padding = #4
    r2 r16 c'8.\pp r4
  }
  \\\
  {
    \override DynamicLineSpanner #'layer = #0
    des,2\mf\< ~
    \override TextScript #'layer = #2
    des16_\markup {
      \postscript #"
        1.9 -8 translate
        5 4 scale
        1 setgray
        0 0 moveto
        0 1 lineto
        1 1 lineto
        1 0 lineto
        0 0 lineto
        fill"
    }
    r8. des4 ~ des16->\sff
  }
  >>
}
```



Caesura ("railtracks") with fermata

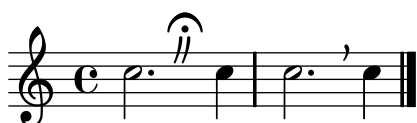
A caesura is sometimes denoted by a double “railtracks” breath mark with a fermata sign positioned above. This snippet shows an optically pleasing combination of railtracks and fermata.

```
\relative c' {
  c2.
  % construct the symbol
  \override BreathingSign #'text = \markup {
    \line {
      \musicglyph #"scripts.caesura.curved"
    }
  }
}
```

```

\translate #'(-1.75 . 1.6)
\musicglyph #"scripts.ufermata"
}
}
\breathe c4
% set the breathe mark back to normal
\revert BreathingSign #'text
c2. \breathe c4
\bar "|."
}

```



Center text below hairpin dynamics

This example provides a function to typeset a hairpin (de)crescendo with some additional text below it, such as “molto” or “poco”. The example also illustrates how to modify the way an object is normally printed, using some Scheme code.

```

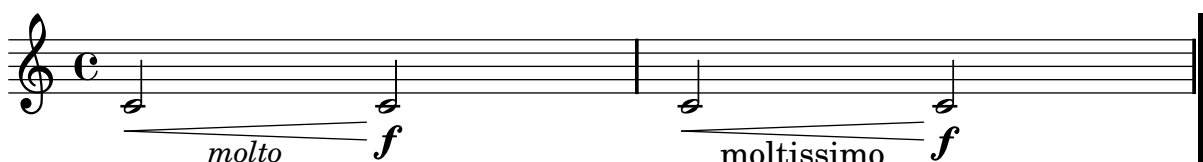
hairpinWithCenteredText =
#(define-music-function (parser location text) (markup?)
#{
  \override Voice.Hairpin #'stencil = #(lambda (grob)
    (ly:stencil-aligned-to
      (ly:stencil-combine-at-edge
        (ly:stencil-aligned-to (ly:hairpin::print grob) X CENTER)
        Y DOWN
        (ly:stencil-aligned-to (grob-interpret-markup grob $text) X CENTER))
      X LEFT))
  #})

hairpinMolto = \hairpinWithCenteredText \markup { \italic molto }
hairpinMore = \hairpinWithCenteredText \markup { \larger moltissimo }

\layout { ragged-right = ##f }

\relative c' {
  \hairpinMolto
  c2\< c\f
  \hairpinMore
  c2\< c\f
}

```



Changing \flageolet mark size

To make the \flageolet circle smaller use the following Scheme function.

```
smallFlageolet =
#(let ((m (make-articulation "flageolet")))
  (set! (ly:music-property m 'tweaks)
    (acons 'font-size -3
      (ly:music-property m 'tweaks)))
  m)

\layout { ragged-right = ##f }

\relative c' {
  d4~\flageolet_\markup { default size } d_\flageolet
  c4~\smallFlageolet_\markup { smaller } c_\smallFlageolet
}
```



Changing text and spanner styles for text dynamics

The text used for crescendos and decrescendos can be changed by modifying the context properties `crescendoText` and `decrescendoText`. The style of the spanner line can be changed by modifying the `'style` property of `DynamicTextSpanner`. The default value is `'hairpin`, and other possible values include `'line`, `'dashed-line` and `'dotted-line`.

```
\relative c' {
  \set crescendoText = \markup { \italic { cresc. poco } }
  \set crescendoSpanner = #'text
  \override DynamicTextSpanner #'style = #'dotted-line
  a2\< a
  a2 a
  a2 a
  a2 a\mf
}
```



Changing the appearance of a slur from solid to dotted or dashed

The appearance of slurs may be changed from solid to dotted or dashed.

```
\relative c' {
  c4( d e c)
  \slurDotted
  c4( d e c)
  \slurSolid
}
```

```

c4( d e c)
\slurDashed
c4( d e c)
\slurSolid
c4( d e c)
}

```



Changing the breath mark symbol

The glyph of the breath mark can be tuned by overriding the text property of the `BreathingSign` layout object with any markup text.

```

\relative c'' {
  c2
  \override BreathingSign #'text = \markup { \musicglyph #"scripts.rvarcomma" }
  \breathe
  d2
}

```



Changing the number of augmentation dots per note

This code demonstrates how to change the number of augmentation dots on a single note.

```

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

```



Combining dynamics with markup texts

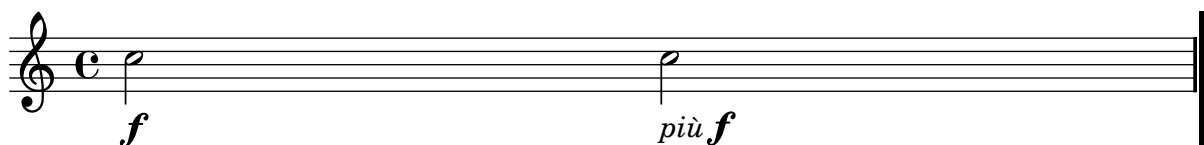
Some dynamics may involve text indications (such as “più forte” or “piano subito”). These can be produced using a `\markup` block.

```

piuF = \markup { \italic più \dynamic f }
\layout { ragged-right = ##f }

```

```
\relative c'' {
  c2\f c-\piuf
}
```



Contemporary glissando

A contemporary glissando without a final note can be typeset using a hidden note and cadenza timing.

```
\relative c'' {
  \time 3/4
  \override Glissando #'style = #'zigzag
  c4 c
  \cadenzaOn
  c4\glissando
  \hideNotes
  c,,4
  \unHideNotes
  \cadenzaOff
  \bar "|"
}
```



Controlling spanner visibility after a line break

The visibility of spanners which end on the first note following a line break is controlled by the `after-line-breaking` callback `ly:spanner::kill-zero-spanned-time`.

For objects such as glissandos and hairpins, the default behaviour is to hide the spanner after a break; disabling the callback will allow the left-broken span to be shown.

Conversely, spanners which are usually visible, such as text spans, can be hidden by enabling the callback.

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

\relative c'' {
  \override Hairpin #'to-barline = ##f
  \override Glissando #'breakable = ##t
  % show hairpin
  \override Hairpin #'after-line-breaking = ##t
  % hide text span
  \override TextSpanner #'after-line-breaking =
```

```

    #ly:spanner::kill-zero-spanned-time
e2\<\startTextSpan
% show glissando
\override Glissando #'after-line-breaking = ##t
f2\glissando
\break
f,1\!\stopTextSpan
}

```



Controlling the vertical ordering of scripts

The vertical ordering of scripts is controlled with the 'script-priority property. The lower this number, the closer it will be put to the note. In this example, the `TextScript` (the sharp symbol) first has the lowest priority, so it is put lowest in the first example. In the second, the prall trill (the `Script`) has the lowest, so it is on the inside. When two objects have the same priority, the order in which they are entered determines which one comes first.

```

\relative c''' {
  \once \override TextScript #'script-priority = #-100
  a2^\prall^\markup { \sharp }

  \once \override Script #'script-priority = #-100
  a2^\prall^\markup { \sharp }
}

```



Creating a delayed turn

Creating a delayed turn, where the lower note of the turn uses the accidental, requires several overrides. The `outside-staff-priority` property must be set to `#f`, as otherwise this would take precedence over the `avoid-slur` property. The value of `halign` is used to position the turn horizontally.

```

\relative c'' {
  \once \override TextScript #'avoid-slur = #'inside
  \once \override TextScript #'outside-staff-priority = ##f
  c2(^\markup \tiny \override #'(baseline-skip . 1) {
    \halign #-4

```

```

\center-column {
  \sharp
  \musicglyph #"scripts.turn"
}
}
d4.) c8
}

```



Creating arpeggios across notes in different voices

An arpeggio can be drawn across notes in different voices on the same staff if the `Span_arpeggio_engraver` is added to the `Staff` context:

```

\new Staff \with {
  \consists "Span_arpeggio_engraver"
}
\relative c' {
  \set Staff.connectArpeggios = ##t
  <<
    { <e' g>4\arpeggio <d f> <d f>2 }
    \\\
    { <d, f>2\arpeggio <g b>2 }
  >>
}

```



Creating cross-staff arpeggios in a piano staff

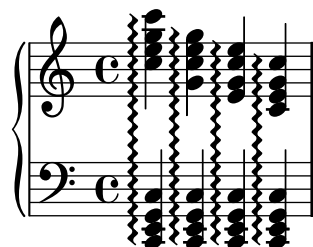
In a `PianoStaff`, it is possible to let an arpeggio cross between the staves by setting the property `PianoStaff.connectArpeggios`.

```

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

```

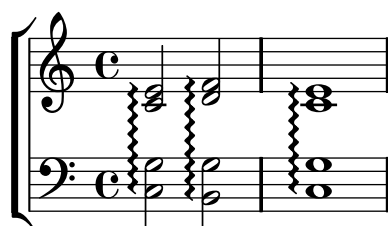
```
}
>>
```



Creating cross-staff arpeggios in other contexts

Cross-staff arpeggios can be created in contexts other than `GrandStaff`, `PianoStaff` and `StaffGroup` if the `Span_arpeggio_engraver` is included in the `Score` context.

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



Creating "real" parenthesized dynamics

Although the easiest way to add parentheses to a dynamic mark is to use a `\markup` block, this method has a downside: the created objects will behave like text markups, and not like dynamics.

However, it is possible to create a similar object using the equivalent Scheme code (as described in the Notation Reference), combined with the `make-dynamic-script` function. This way, the markup will be regarded as a dynamic, and therefore will remain compatible with commands such as `\dynamicUp` or `\dynamicDown`.

```
parenF = #(make-dynamic-script (markup #:line (#:normal-text #:italic
      #:fontsize 2 "(" #:hspace -0.8 #:dynamic "f" #:normal-text
      #:italic #:fontsize 2 ")"))))

\relative c'' {
  c4\parenF c c \dynamicUp c\parenF
}
```



Creating simultaneous rehearsal marks

Unlike text scripts, rehearsal marks cannot be stacked at a particular point in a score: only one `RehearsalMark` object is created. Using an invisible measure and bar line, an extra rehearsal mark can be added, giving the appearance of two marks in the same column. This method may also prove useful for placing rehearsal marks at both the end of one system and the start of the following system.

```
{
  \key a \major
  \set Score.markFormatter = #format-mark-box-letters
  \once \override Score.RehearsalMark #'outside-staff-priority = #5000
  \once \override Score.RehearsalMark #'self-alignment-X = #LEFT
  \once \override Score.RehearsalMark #'break-align-symbols = #'(key-signature)
  \mark \markup { \bold { Senza denti } }

  % the hidden measure and bar line
  % \cadenzaOn turns off automatic calculation of bar numbers
  \cadenzaOn
  \once \override Score.TimeSignature #'stencil = ##f
  \time 1/16
  s16 \bar ""
  \cadenzaOff

  \time 4/4
  \once \override Score.RehearsalMark #'self-alignment-X = #LEFT
  \mark \markup { \box \bold Intro }
  d'1
  \mark \default
}
```

$$\left. \begin{array}{l} d'1 \\ \end{array} \right\}$$


Creating slurs across voices

In some situations, it may be necessary to create slurs between notes from different voices.

The solution is to add invisible notes to one of the voices, using `\hideNotes`.

This example is measure 235 of the Ciaccona from Bach's 2nd Partita for solo violin, BWV 1004.

```
\relative c' {
  <<
  {
    d16( a') s a s a[ s a] s a[ s a]
  }
  \\\
  {
    \slurUp
    bes,16[ s e](
    \hideNotes a)
    \unHideNotes f[(
    \hideNotes a)
    \unHideNotes fis](
    \hideNotes a)
    \unHideNotes g[(
    \hideNotes a)
    \unHideNotes gis](
    \hideNotes a)
  }
  >>
}
```



Creating text spanners

The `\startTextSpan` and `\stopTextSpan` commands allow the creation of text spanners as easily as pedal indications or octavations. Override some properties of the `TextSpanner` object to modify its output.

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

```
\relative c' {
  \override TextSpanner #'(bound-details left text) = #"bla"
```

```

\override TextSpanner #'(bound-details right text) = #"blu"
a4 \startTextSpan
b4 c
a4 \stopTextSpan

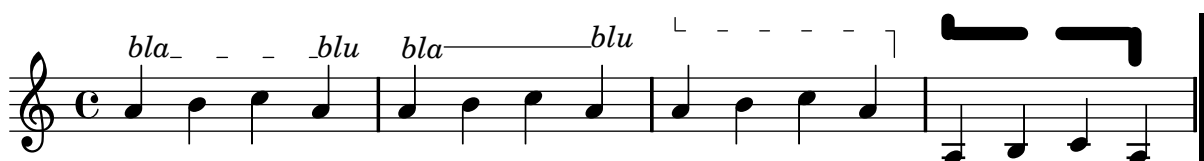
\override TextSpanner #'style = #'line
\once \override TextSpanner
  #'(bound-details left stencil-align-dir-y) = #CENTER
a4 \startTextSpan
b4 c
a4 \stopTextSpan

\override TextSpanner #'style = #'dashed-line
\override TextSpanner #'(bound-details left text) =
  \markup { \draw-line #'(0 . 1) }
\override TextSpanner #'(bound-details right text) =
  \markup { \draw-line #'(0 . -2) }
\once \override TextSpanner #'(bound-details right padding) = #-2

a4 \startTextSpan
b4 c
a4 \stopTextSpan

\set Staff.middleCPosition = #-13
\override TextSpanner #'dash-period = #10
\override TextSpanner #'dash-fraction = #0.5
\override TextSpanner #'thickness = #10
a4 \startTextSpan
b4 c
a4 \stopTextSpan
}

```



Double glissando

To connect chords with glissando lines, attach a second glissando to a hidden voice.

```

\relative c {
  \clef bass
  <<
  {
    % new voice ( = \voiceOne), hidden
    \hideNotes
    % attach glissando to note heads
    e2\glissando g
  }
  \\\
  {

```

```

    % original voice with chords rearranged so that
    % glissando is attached to a & c
    <e a,>2\glissando <g c,>
  }
>>
}

```



Dynamics custom text spanner postfix

Postfix functions for custom crescendo text spanners. The spanners should start on the first note of the measure. One has to use `-\mycresc`, otherwise the spanner start will rather be assigned to the next note.

% Two functions for (de)crescendo spanners where you can explicitly give the
% spanner text.

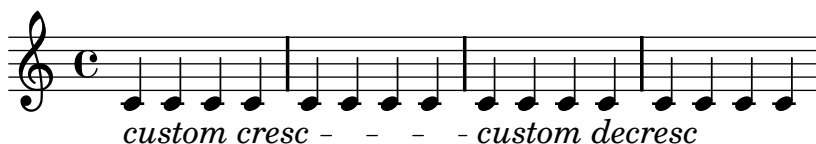
```

mycresc =
#(define-music-function (parser location mymarkup) (markup?)
  (make-music 'CrescendoEvent
    'span-direction START
    'span-type 'text
    'span-text mymarkup))

mydecresc =
#(define-music-function (parser location mymarkup) (markup?)
  (make-music 'DecrescendoEvent
    'span-direction START
    'span-type 'text
    'span-text mymarkup))

\relative c' {
  c4-\mycresc "custom cresc" c4 c4 c4 |
  c4 c4 c4 c4 |
  c4-\mydecresc "custom decresc" c4 c4 c4 |
  c4 c4\! c4 c4
}

```



Dynamics text spanner postfix

The `\cresc`, `\dim` and `\decresc` spanners can now be redefined as postfix operators and produce one text spanner. Defining custom spanners is also easy. Hairpin and text crescendi can be easily mixed. `\<` and `\>` produce hairpins by default, `\cresc` etc. produce text spanners by default.

```
% Some sample text dynamic spanners, to be used as postfix operators
crpoco =
#(make-music 'CrescendoEvent
      'span-direction START
      'span-type 'text
      'span-text "cresc. poco a poco")
% Redefine the existing \cresc, \dim and \decresc commands to use postfix syntax
cresc =
#(make-music 'CrescendoEvent
      'span-direction START
      'span-type 'text
      'span-text "cresc.")

dim =
#(make-music 'DecrescendoEvent
      'span-direction START
      'span-type 'text
      'span-text "dim.")

decresc =
#(make-music 'DecrescendoEvent
      'span-direction START
      'span-type 'text
      'span-text "decresc.")

\relative c' {
  c4\cresc d4 e4 f4 |
  g4 a4\! b4\crpoco c4 |
  c4 d4 e4 f4 |
  g4 a4\! b4\< c4 |
  g4\dim a4 b4\decresc c4\!
}
```

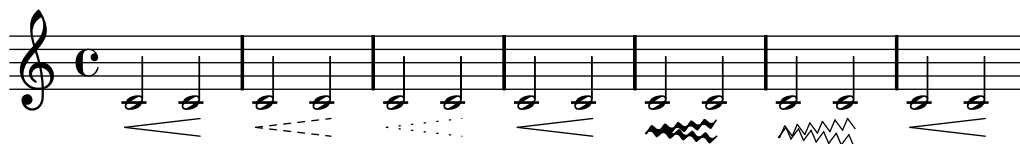


Hairpins with different line styles

Hairpins can take any style from `line-interface` - dashed-line, dotted-line, line, trill or zigzag.

```
\relative c' {
  c2\< c\!
  \override Hairpin #'style = #'dashed-line
  c2\< c\!
  \override Hairpin #'style = #'dotted-line
  c2\< c\!
  \override Hairpin #'style = #'line
  c2\< c\!
  \override Hairpin #'style = #'trill
  c2\< c\!
  \override Hairpin #'style = #'zigzag
  c2\< c\!
```

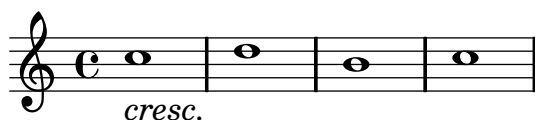
```
\revert Hairpin #'style
c2\< c\!
}
```



Hiding the extender line for text dynamics

Text style dynamic changes (such as *cresc.* and *dim.*) are printed with a dashed line showing their extent. This line can be suppressed in the following way:

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



Horizontally aligning custom dynamics (e.g. "sempre pp" "piu f" "subito p")

Some dynamic expressions involve additional text, like “sempre pp”. Since dynamics are usually centered under the note, the `\pp` would be displayed way after the note it applies to.

To correctly align the “sempre pp” horizontally, so that it is aligned as if it were only the `\pp`, there are several approaches:

- * Simply use `\once\override DynamicText #'X-offset = #-9.2` before the note with the dynamics to manually shift it to the correct position. Drawback: This has to be done manually each time you use that dynamic markup... * Add some padding (`\hspace 7.1`) into the definition of your custom dynamic mark, so that after LilyPond center-aligns it, it is already correctly aligned. Drawback: The padding really takes up that space and does not allow any other markup or dynamics to be shown in that position.

- * Shift the dynamic script `\once\override ... #'X-offset = ...` Drawback: `\once\override` is needed for every invocation!

- * Set the dimensions of the additional text to 0 (using `\with-dimensions '(0 . 0) '(0 . 0)`). Drawback: To LilyPond “sempre” has no extent, so it might put other stuff there and create collisions (which are not detected by the collision detection!). Also, there seems to be some spacing, so it’s not exactly the same alignment as without the additional text

- * Add an explicit shifting directly inside the scheme function for the dynamic-script.

- * Set an explicit alignment inside the dynamic-script. By default, this won’t have any effect, only if one sets `X-offset`! Drawback: One needs to set `DynamicText #'X-offset`, which will apply to all dynamic texts! Also, it is aligned at the right edge of the additional text, not at the center of `pp`.

```

\header { title = "Horizontally aligning custom dynamics" }

\paper { ragged-right = ##f }

% Solution 1: Using a simple markup with a particular halign value
% Drawback: It's a markup, not a dynamic command, so \dynamicDown
%           etc. will have no effect
semppMarkup = \markup { \halign #1.4 \italic "sempre" \dynamic "pp" }

% Solution 2: Using a dynamic script & shifting with
%           \once \override ... #'X-offset = ..
% Drawback: \once \override needed for every invocation
semppK =
#(make-dynamic-script
  (markup #:line
    (#:normal-text
      #:italic "sempre"
      #:dynamic "pp"))))

% Solution 3: Padding the dynamic script so the center-alignment
%           puts it at the correct position
% Drawback: the padding really reserves the space, nothing else can be there
semppT =
#(make-dynamic-script
  (markup #:line
    (#:normal-text
      #:italic "sempre"
      #:dynamic "pp"
      #:hspace 7.1))))

% Solution 4: Dynamic, setting the dimensions of the additional text to 0
% Drawback: To Lilypond "sempre" has no extent, so it might put
%           other stuff there => collisions
% Drawback: Also, there seems to be some spacing, so it's not exactly the
%           same alignment as without the additional text
semppM =
#(make-dynamic-script
  (markup #:line
    (#:with-dimensions '(0 . 0) '(0 . 0)
      #:right-align
      #:normal-text
      #:italic "sempre"
      #:dynamic "pp"))))

% Solution 5: Dynamic with explicit shifting inside the scheme function
semppG =
#(make-dynamic-script
  (markup #:hspace 0
    #:translate '(-18.85 . 0)
    #:line (#:normal-text
      #:italic "sempre"
      #:dynamic "pp"))))

```

```

% Solution 6: Dynamic with explicit alignment. This has only effect
%           if one sets X-offset!
% Drawback: One needs to set DynamicText #'X-offset!
% Drawback: Aligned at the right edge of the additional text,
%           not at the center of pp
semppMII =
#(make-dynamic-script
  (markup #:line (#:right-align
                  #:normal-text
                  #:italic "sempre"
                  #:dynamic "pp")))

\context StaffGroup <<
  \context Staff = "s" <<
    \set Staff.instrumentName = #"Normal"
    \relative c'' {
      \key es \major
      c4\pp c\p c c | c\ff c c\pp c
    }
  >>
  \context Staff = "sMarkup" <<
    \set Staff.instrumentName = \markup \column { Normal markup }
    \relative c'' {
      \key es \major
      c4-\semppMarkup c\p c c | c\ff c c-\semppMarkup c
    }
  >>
  \context Staff = "sK" <<
    \set Staff.instrumentName = \markup \column { Explicit shifting }
    \relative c'' {
      \key es \major
      \once \override DynamicText #'X-offset = #-9.2
      c4\semppK c\p c c
      c4\ff c
      \once \override DynamicText #'X-offset = #-9.2
      c4\semppK c
    }
  >>
  \context Staff = "sT" <<
    \set Staff.instrumentName = \markup \column { Right padding }
    \relative c'' {
      \key es \major
      c4\semppT c\p c c | c\ff c c\semppT c
    }
  >>
  \context Staff = "sM" <<
    \set Staff.instrumentName = \markup \column { Setting dimension "to zero" }
    \relative c'' {
      \key es \major
      c4\semppM c\p c c | c\ff c c\semppM c
    }
  >>

```

```

>>
\context Staff = "sG" <<
  \set Staff.instrumentName = \markup \column { Shifting inside dynamics }
  \relative c'' {
    \key es \major
    c4\semppG c\p c c | c\ff c c\semppG c
  }
>>
\context Staff = "sMII" <<
  \set Staff.instrumentName = \markup \column { Alignment inside dynamics }
  \relative c'' {
    \key es \major
    % Setting to ##f (false) gives the same result
    \override DynamicText #'X-offset = #0
    c4\semppMII c\p c c | c\ff c c\semppMII c
  }
>>
>>

```

Horizontally aligning custom dynamics

Normal	
Normal markup	
Explicit shifting	
Right padding	
Setting dimension to zero	
Shifting inside dynamics	
Alignment inside dynamics	

Inserting a caesura

Caesura marks can be created by overriding the 'text property of the `BreathingSign` object. A curved caesura mark is also available.

```
\relative c' {
  \override BreathingSign #'text = \markup {
    \musicglyph #"scripts.caesura.straight"
  }
  c8 e4. \breathe g8. e16 c4

  \override BreathingSign #'text = \markup {
    \musicglyph #"scripts.caesura.curved"
  }
  g8 e'4. \breathe g8. e16 c4
}
```



Laissez vibrer ties

Laissez vibrer ties have a fixed size. Their formatting can be tuned using 'tie-configuration.

```
\relative c' {
  <c e g>4\laissezVibrer r <c f g>\laissezVibrer r
  <c d f g>4\laissezVibrer r <c d f g>4.\laissezVibrer r8

  <c d e f>4\laissezVibrer r
  \override LaissezVibrerTieColumn #'tie-configuration
    = #`((-7 . ,DOWN)
      (-5 . ,DOWN)
      (-3 . ,UP)
      (-1 . ,UP))
  <c d e f>4\laissezVibrer r
}
```



Line arrows

Arrows can be applied to text-spanners and line-spanners (such as the Glissando).

```
\relative c' {
  \override TextSpanner #'bound-padding = #1.0
  \override TextSpanner #'style = #'line
  \override TextSpanner #'(bound-details right arrow) = ##t
  \override TextSpanner #'(bound-details left text) = #"fof"
  \override TextSpanner #'(bound-details right text) = #"gag"
  \override TextSpanner #'(bound-details right padding) = #0.6
```

```

\override TextSpanner #'(bound-details right stencil-align-dir-y) = #CENTER
\override TextSpanner #'(bound-details left stencil-align-dir-y) = #CENTER

\override Glissando #'(bound-details right arrow) = ##t
\override Glissando #'arrow-length = #0.5
\override Glissando #'arrow-width = #0.25

a8\startTextSpan gis a4 b\glissando b,
g'4 c\stopTextSpan c2
}

```



Making slurs with complex dash structure

Slurs can be made with complex dash patterns by defining the `dash-definition` property. `dash-definition` is a list of `dash-elements`. A `dash-element` is a list of parameters defining the dash behavior for a segment of the slur.

The slur is defined in terms of the bezier parameter `t` which ranges from 0 at the left end of the slur to 1 at the right end of the slur. `dash-element` is a list (`start-t stop-t dash-fraction dash-period`). The region of the slur from `start-t` to `stop-t` will have a fraction `dash-fraction` of each `dash-period` black. `dash-period` is defined in terms of staff spaces. `dash-fraction` is set to 1 for a solid slur.

```

\relative c' {
  \once \override
    Slur #'dash-definition = #'((0 0.3 0.1 0.75)
                                (0.3 0.6 1 1)
                                (0.65 1.0 0.4 0.75))

  c4( d e f)
  \once \override
    Slur #'dash-definition = #'((0 0.25 1 1)
                                (0.3 0.7 0.4 0.75)
                                (0.75 1.0 1 1))

  c4( d e f)
}

```



Modifying default values for articulation shorthand notation

The shorthands are defined in `'ly/script-init.ly'`, where the variables `dashHat`, `dashPlus`, `dashDash`, `dashBar`, `dashLarger`, `dashDot`, and `dashUnderscore` are assigned default values. The default values for the shorthands can be modified. For example, to associate the `-+` (`dashPlus`) shorthand with the trill symbol instead of the default `+` symbol, assign the value `trill` to the variable `dashPlus`:

```

\relative c' { c1-+ }

```

```
dashPlus = "trill"
```

```
\relative c' { c1-+ }
```



Moving slur positions vertically

The vertical position of a slur can be adjusted using the `positions` property of `Slur`. The property has 2 parameters, the first referring to the left end of the slur and the second to the right. The values of the parameters are not used by LilyPond to make an exact movement of the slur - instead it selects what placement of the slur looks best, taking into account the parameter values. Positive values move the slur up, and are appropriate for notes with stems down. Negative values move the slur down.

```
\relative c' {
  \stemDown
  e4( a)
  \override Slur #'positions = #'(1 . 1)
  e4( a)
  \override Slur #'positions = #'(2 . 2)
  e4( a)
  \override Slur #'positions = #'(3 . 3)
  e4( a)
  \override Slur #'positions = #'(4 . 4)
  e4( a)
  \override Slur #'positions = #'(5 . 5)
  e4( a)
  \override Slur #'positions = #'(0 . 5)
  e4( a)
  \override Slur #'positions = #'(5 . 0)
  e4( a)
}
```



Piano template with centered dynamics

Many piano scores have the dynamics centered between the two staves. The `Dynamics` context, if placed between staves, will automatically position dynamics correctly.

```

global = {
  \key c \major
  \time 4/4
}

upper = \relative c'' {
  \clef treble
  a4 b c d
}

lower = \relative c {
  \clef bass
  a2 c
}

dynamics = {
  s2\fff\> s4 s\!\pp
}

pedal = {
  s2\sustainOn s\sustainOff
}

\score {
  \new PianoStaff = "PianoStaff_pf" <<
    \new Staff = "Staff_pfUpper" << \global \upper >>
    \new Dynamics = "Dynamics_pf" \dynamics
    \new Staff = "Staff_pfLower" << \global \lower >>
    \new Dynamics = "pedal" \pedal
  >>
  \layout { }
}

\score {
  \new PianoStaff = "PianoStaff_pf" <<
    \new Staff = "Staff_pfUpper" << \global \upper \dynamics \pedal >>
    \new Staff = "Staff_pfLower" << \global \lower \dynamics \pedal >>
  >>
  \midi { }
}

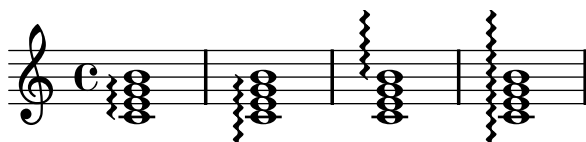
```



Positioning arpeggios

If you need to extend or shorten an arpeggio, you can modify the upper and lower start positions independently.

```
\relative c' {
  <c e g b>1\arpeggio
  \once \override Arpeggio #'positions = #'(-5 . 0)
  <c e g b>1\arpeggio
  \once \override Arpeggio #'positions = #'(0 . 5)
  <c e g b>1\arpeggio
  \once \override Arpeggio #'positions = #'(-5 . 5)
  <c e g b>1\arpeggio
}
```



Positioning text markups inside slurs

Text markups need to have the `outside-staff-priority` property set to false in order to be printed inside slurs.

```
\relative c' {
  \override TextScript #'avoid-slur = #'inside
  \override TextScript #'outside-staff-priority = ##f
  c2(^\markup { \halign #-10 \natural } d4.) c8
}
```



Printing hairpins using al niente notation

Hairpin dynamics may be printed with a circled tip (“al niente” notation) by setting the `circled-tip` property of the Hairpin object to `#t`.

```
\relative c' {
  \override Hairpin #'circled-tip = ##t
  c2\< c\!
  c4\> c\< c2\!
}
```



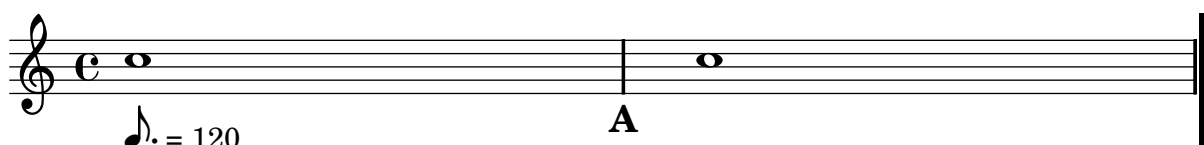
Printing metronome and rehearsal marks below the staff

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

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

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

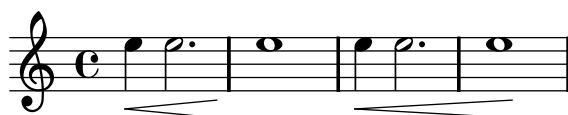
  % Rehearsal marks below the staff
  \override Score.RehearsalMark #'direction = #DOWN
  \mark \default
  c''1
}
```



Setting hairpin behavior at bar lines

If the note which ends a hairpin falls on a downbeat, the hairpin stops at the bar line immediately preceding. This behavior can be controlled by overriding the `'to-barline` property.

```
\relative c'' {
  e4\< e2.
  e1\!
  \override Hairpin #'to-barline = ##f
  e4\< e2.
  e1\!
}
```



Setting the minimum length of hairpins

If hairpins are too short, they can be lengthened by modifying the `minimum-length` property of the `Hairpin` object.

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



Showing the same articulation above and below a note or chord

By default, LilyPond does not allow the same articulation (e.g. an accent, a fermata, a flageolet, etc.) to be displayed above and below a note. For example, `c4_\fermata^\fermata` will only show a fermata below. The fermata above will simply be ignored. However, one can stick scripts (just like fingerings) inside a chord, which means it is possible to have as many articulations as desired. This approach has the advantage that it ignores the stem and positions the articulation relative to the note head. This can be seen in the case of the flageolets in the snippet. To mimic the behaviour of scripts outside a chord, `'add-stem-support` would be required. So, the solution is to write the note as a chord and add the articulations inside the `<...>`. The direction will always be above, but one can tweak this via a `\tweak: <c-\tweak #'direction #DOWN-\fermata^\fermata>`

```
% The same as \flageolet, just a little smaller
smallFlageolet =
#(let ((m (make-articulation "flageolet")))
  (set! (ly:music-property m 'tweaks)
    (acons 'font-size -2
      (ly:music-property m 'tweaks)))
  m)

\relative c' {
  s4^"wrong:"
  c_\fermata^\fermata % The second fermata is ignored!
  <e d'>^\smallFlageolet_\smallFlageolet

  % it works only if you wrap the note inside a chord. By default,
  % all articulations will be printed above, so you have to tweak
  % the direction.
  s4^"Works if written inside a chord:"
  <e-\tweak #'direction #DOWN -\smallFlageolet d'^\smallFlageolet>
  <e-\tweak #'direction #DOWN -\flageolet d'^\flageolet>
  <e-\tweak #'direction #DOWN -\smallFlageolet^\smallFlageolet>
  <e-\tweak #'direction #DOWN -\fermata^\fermata>
}
```

wrong: Works if written inside a chord:



Snap-pizzicato ("Bartok pizzicato")

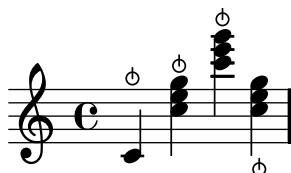
A snap-pizzicato (also known as "Bartok pizzicato") is a "strong pizzicato where the string is plucked vertically by snapping and rebounds off the fingerboard of the instrument" (Wikipedia). It is denoted by a circle with a vertical line going from the center upwards outside the circle.

```
\relative c' {
  c4\snappizzicato
  <c' e g>4\snappizzicato
}
```

```

<c' e g>4^\snappizzicato
<c, e g>4_\snappizzicato
}

```



Using double slurs for legato chords

Some composers write two slurs when they want legato chords. This can be achieved by setting `doubleSlurs`.

```

\relative c' {
  \set doubleSlurs = ##t
  <c e>4( <d f> <c e> <d f>)
}

```



Using the whiteout property

Any graphical object can be printed over a white background to mask parts of objects that lie beneath. This can be useful to improve the appearance of collisions in complex situations when repositioning objects is impractical. It is necessary to explicitly set the `layer` property to control which objects are masked by the white background.

In this example the collision of the tie with the time signature is improved by masking out the part of the tie that crosses the time signature by setting the `whiteout` property of `TimeSignature`. To do this `TimeSignature` is moved to a layer above `Tie`, which is left in the default layer of 1, and `StaffSymbol` is moved to a layer above `TimeSignature` so it is not masked.

```

{
  \override Score.StaffSymbol #'layer = #4
  \override Staff.TimeSignature #'layer = #3
  b'2 b'~
  \once \override Staff.TimeSignature #'whiteout = ##t
  \time 3/4
  b' r4
}

```



Vertical line as a baroque articulation mark

This short vertical line placed above the note is commonly used in baroque music. Its meaning can vary, but generally indicates notes that should be played with more “weight”. The following example demonstrates how to achieve such a notation.

```
upline =
#(let ((m (make-articulation "stopped")))
  (set! (ly:music-property m 'tweaks)
    (acons 'font-size 3
      (acons 'stencil (lambda (grob)
        (grob-interpret-markup
          grob
          (make-draw-line-markup '(0 . 1))))
        (ly:music-property m 'tweaks))))
  m)
```

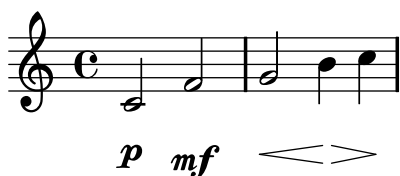
```
\relative c' {
  a'4^\upline a( c d')_\upline
}
```



Vertically aligning dynamics across multiple notes

Dynamics that occur at, begin on, or end on the same note will be vertically aligned. To ensure that dynamics are aligned when they do not occur on the same note, increase the **staff-padding** property of the **DynamicLineSpanner** object.

```
\relative c' {
  \override DynamicLineSpanner #'staff-padding = #4
  c2\p f\mf
  g2\< b4\> c\!
}
```



Repeats

Section “Repeats” in *Notation Reference*

Adding volta brackets to additional staves

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

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



Cross-staff tremolos

Since `\repeat tremolo` expects exactly two musical arguments for chord tremolos, the note or chord which changes staff within a cross-staff tremolo should be placed inside curly braces together with its `\change Staff` command.

```
\new PianoStaff <<
  \new Staff = "up" \relative c'' {
    \key a \major
    \time 3/8
    s4.
  }
  \new Staff = "down" \relative c'' {
    \key a \major
    \time 3/8
    \voiceOne
    \repeat tremolo 6 {
      <a e'>32
      {
        \change Staff = "up"

```

```

        \voiceTwo
        <cis a' dis>32
      }
    }
  }
>>

```



Engraving tremolos with floating beams

If a tremolo's total duration is less than a quarter-note, or exactly a half-note, or between a half-note and a whole-note, it is normally typeset with all beams touching the stems. Certain engraving styles typeset some of these beams as centered floating beams that do not touch the stems. The number of floating beams in this type of tremolo is controlled with the `'gap-count` property of the `Beam` object, and the size of the gaps between beams and stems is set with the `'gap` property.

```

\relative c'' {
  \repeat tremolo 8 { a32 f }
  \override Beam #'gap-count = #1
  \repeat tremolo 8 { a32 f }
  \override Beam #'gap-count = #2
  \repeat tremolo 8 { a32 f }
  \override Beam #'gap-count = #3
  \repeat tremolo 8 { a32 f }

  \override Beam #'gap-count = #3
  \override Beam #'gap = #1.33
  \repeat tremolo 8 { a32 f }
  \override Beam #'gap = #1
  \repeat tremolo 8 { a32 f }
  \override Beam #'gap = #0.67
  \repeat tremolo 8 { a32 f }
  \override Beam #'gap = #0.33
  \repeat tremolo 8 { a32 f }
}

```



Isolated percent repeats

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

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



Measure counter

This snippet provides a workaround for emitting measure counters using transparent percent repeats.

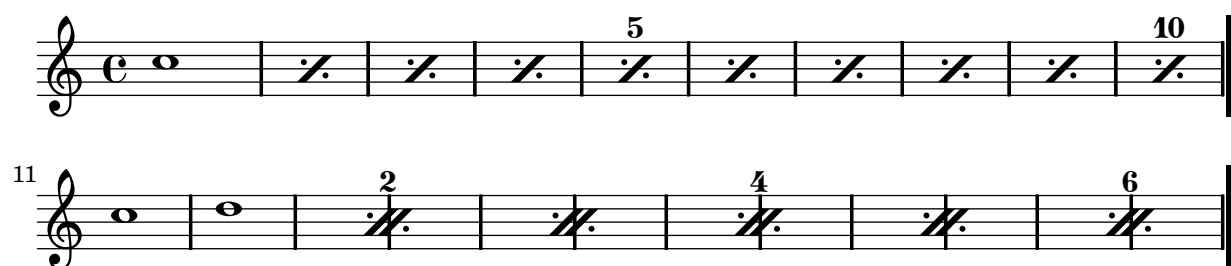
```
<<
  \context Voice = "foo" {
    \clef bass
    c4 r g r
    c4 r g r
    c4 r g r
    c4 r g r
  }
  \context Voice = "foo" {
    \set countPercentRepeats = ##t
    \override PercentRepeat #'transparent = ##t
    \override PercentRepeatCounter #'staff-padding = #1
    \repeat percent 4 { s1 }
  }
>>
```



Percent repeat count visibility

Percent repeat counters can be shown at regular intervals by setting the context property `repeatCountVisibility`.

```
\relative c'' {
  \set countPercentRepeats = ##t
  \set repeatCountVisibility = #(every-nth-repeat-count-visible 5)
  \repeat percent 10 { c1 } \break
  \set repeatCountVisibility = #(every-nth-repeat-count-visible 2)
  \repeat percent 6 { c1 d1 }
}
```



Percent repeat counter

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

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



Positioning segno and coda (with line break)

If you want to place an exiting segno sign and add text like “D.S. al Coda” next to it where usually the staff lines are you can use this snippet. The coda will resume in a new line. There is a variation documented in this snippet, where the coda will remain on the same line.

```
{
  \clef treble
  \key g \major
  \time 4/4
  \relative c'' {
    \repeat unfold 2 {
      | c4 c c c
    }

    % Set segno sign as rehearsal mark and adjust size if needed
    % \once \override Score.RehearsalMark #'font-size = #3
    \mark \markup { \musicglyph #"scripts.segno" }
    \repeat unfold 2 {
      | c4 c c c
    }

    % Set coda sign as rehearsal mark and adjust size if needed
    \once \override Score.RehearsalMark #'font-size = #4
    \mark \markup { \musicglyph #"scripts.coda" }
    \repeat unfold 2 {
      | c4 c c c
    }

    % Should Coda be on anew line?
    % Coda NOT on new line: use \nobreak
    % Coda on new line: DON'T use \nobreak
    % \noBreak
  }
}
```

```

\bar "||"

% Set segno sign as rehearsal mark and adjust size if needed
\once \override Score.RehearsalMark #'break-visibility = #begin-of-line-invisible
% \once \override Score.RehearsalMark #'font-size = #3
\mark \markup { \musicglyph #"scripts.segno" }

% Here begins the trickery!
% \cadenzaOn will suppress the bar count and \stopStaff removes the staff lines.
\cadenzaOn
  \stopStaff
    % Some examples of possible text-displays

    % text line-aligned
    % =====
    % Move text to the desired position
    % \once \override TextScript #'extra-offset = #'( 2 . -3.5 )
    % | s1*0^\markup { D.S. al Coda } }

    % text center-aligned
    % =====
    % Move text to the desired position
    % \once \override TextScript #'extra-offset = #'( 6 . -5.0 )
    % | s1*0^\markup { \center-column { D.S. "al Coda" } }

    % text and symbols center-aligned
    % =====
    % Move text to the desired position and tweak spacing for optimum text alignment
    % \once \override TextScript #'extra-offset = #'( 8 . -5.5 )
    \once \override TextScript #'word-space = #1.5
    \once \override TextScript #'X-offset = #8
    \once \override TextScript #'Y-offset = #1.5
    | s1*0^\markup { \center-column { "D.S. al Coda" \line { \musicglyph #"scripts.coda" } } }

    % Increasing the unfold counter will expand the staff-free space
    \repeat unfold 4 {
      s4 s4 s4 s4
      \bar ""
    }
    % Resume bar count and show staff lines again
  \startStaff
\cadenzaOff

% Should Coda be on new line?
% Coda NOT on new line: DON'T use \break
% Coda on new line: use \break
\break

% Show up, you clef and key!
\once \override Staff.KeySignature #'break-visibility = #end-of-line-invisible
\once \override Staff.Clef #'break-visibility = #end-of-line-invisible

```

```

% Set coda sign as rehearsal mark and adjust size and position

% Put the coda sign ontop of the (treble-)clef dependend on coda's line-position

% Coda NOT on new line, use this:
% \once \override Score.RehearsalMark #'extra-offset = #'( -2 . 1.75 )

% Coda on new line, use this:
\once \override Score.RehearsalMark #'extra-offset = #'( -8.42 . 1.75 )

\once \override Score.RehearsalMark #'font-size = #5
\mark \markup { \musicglyph #"scripts.coda" }

% The coda
\repeat unfold 5 {
  | c4 c c c
}
\bar"|"
}

```

Printing a repeat sign at the beginning of a piece

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

```

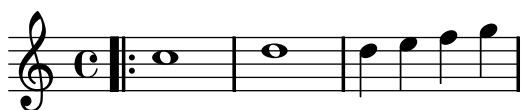
\relative c' {
  \once \override Score.BreakAlignment #'break-align-orders =
    #(make-vector 3 '(instrument-name
      left-edge
      ambitus
      span-bar
      breathing-sign
      clef
      key-signature
      time-signature
      staff-bar
      custos
      span-bar))
  \once \override Staff.TimeSignature #'space-alist =

```

```

#'((first-note . (fixed-space . 2.0))
   (right-edge . (extra-space . 0.5))
   ;; free up some space between time signature
   ;; and repeat bar line
   (staff-bar . (extra-space . 1)))
\bar "|:"
c1
d1
d4 e f g
}

```



Shortening volta brackets

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

```

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

```



Volta below chords

By adding the `Volta_engraver` to the relevant staff, voltes can be put under chords.

```

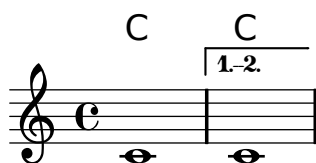
\score {
  <<
    \chords {
      c1
      c1
    }
    \new Staff \with {
      \consists "Volta_engraver"
    }
  }
}

```

```

    }
    {
        \repeat volta 2 { c'1 }
        \alternative { c' }
    }
>>
\layout {
    \context {
        \Score
        \remove "Volta_engraver"
    }
}
}

```



Volta multi staff

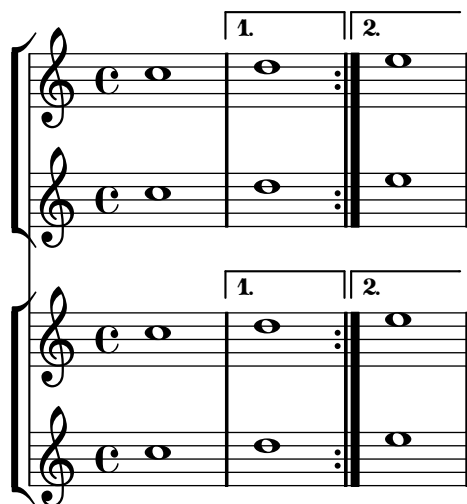
By adding the `Volta_engraver` to the relevant staff, volte can be put over staves other than the topmost one in a score.

```

voltaMusic = \relative c'' {
    \repeat volta 2 {
        c1
    }
    \alternative {
        d1
        e1
    }
}

<<
\new StaffGroup <<
    \new Staff \voltaMusic
    \new Staff \voltaMusic
>>
\new StaffGroup <<
    \new Staff \with { \consists "Volta_engraver" }
        \voltaMusic
    \new Staff \voltaMusic
>>
>>

```



Volta text markup using repeatCommands

Though voltes are best specified using `\repeat volta`, the context property `repeatCommands` must be used in cases where the volta text needs more advanced formatting with `\markup`.

Since `repeatCommands` takes a list, the simplest method of including markup is to use an identifier for the text and embed it in the command list using the Scheme syntax `#(list (list 'volta textIdentifier))`. Start- and end-repeat commands can be added as separate list elements:

```
voltaAdLib = \markup { 1. 2. 3... \text \italic { ad lib. } }
```

```
\relative c'' {
  c1
  \set Score.repeatCommands = #(list (list 'volta voltaAdLib) 'start-repeat)
  c4 b d e
  \set Score.repeatCommands = #'((volta #f) (volta "4.") end-repeat)
  f1
  \set Score.repeatCommands = #'((volta #f))
}
```



Simultaneous notes

Section “Simultaneous notes” in *Notation Reference*

Additional voices to avoid collisions

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

```
voiceFive = #(context-spec-music (make-voice-props-set 4) 'Voice)
```

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

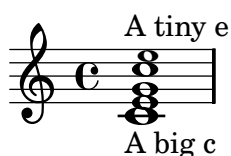


Changing a single note's size in a chord

Individual note heads in a chord can be modified with the `\tweak` command inside a chord, by altering the `font-size` property.

Inside the chord (within the brackets `< >`), before the note to be altered, place the `\tweak` command, followed by `#'font-size` and define the proper size like `#-2` (a tiny note head).

```
\relative c' {
  <\tweak #'font-size #+2 c e g c \tweak #'font-size #-2 e>1~\markup { A tiny e }_ \markup {
}
```



Changing partcombine texts

When using the automatic part combining feature, the printed text for the solo and unison sections may be changed:

```
\new Staff <<
  \set Staff.soloText = #"girl"
  \set Staff.soloIIText = #"boy"
  \set Staff.aDueText = #"together"
  \partcombine
  \relative c'' {
    g4 g r r
    a2 g
  }
  \relative c'' {
    r4 r a( b)
    a2 g
  }
}>>
```



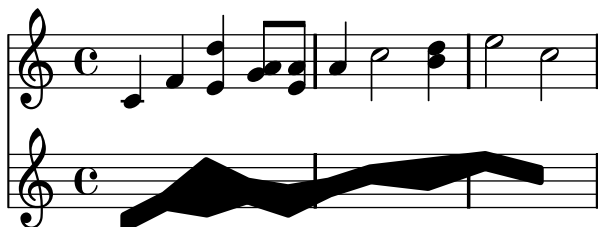
Clusters

Clusters are a device to denote that a complete range of notes is to be played.

```
fragment = \relative c' {
  c4 f <e d'>4
  <g a>8 <e a> a4 c2 <d b>4
  e2 c
}
```

```
<<
  \new Staff \fragment
```

```
\new Staff \makeClusters \fragment
>>
```



Combining two parts on the same staff

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

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

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

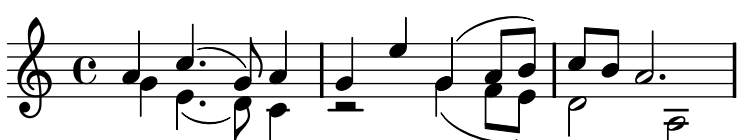

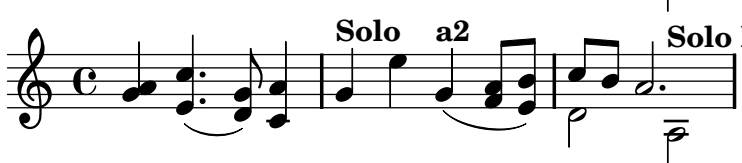
musicDown = \relative c'' {
  g4 e4.( d8) c4 |
  r2 g'4( f8 e) |
  d2 \stemDown a
}

\score {
  <<
    <<
      \new Staff {
        \set Staff.instrumentName = #"Standard polyphony"
        << \musicUp \\\musicDown >>
      }
      \new Staff \with { printPartCombineTexts = ##f } {
        \set Staff.instrumentName = #"PartCombine without texts"
        \partcombine \musicUp \musicDown
      }
      \new Staff {
        \set Staff.instrumentName = #"PartCombine with texts"
        \partcombine \musicUp \musicDown
      }
    >>
  >>
}
```

```

\layout {
  indent = 6.0\cm
  \context {
    \Score
    \override SystemStartBar #'collapse-height = #30
  }
}

```

Standard polyphony	
PartCombine without texts	
PartCombine with texts	

Displaying complex chords

Here is a way to display a chord where the same note is played twice with different accidentals.

```

fixA = {
  \once \override Stem #'length = #9
}
fixB = {
  \once \override NoteHead #'X-offset = #1.7
  \once \override Stem #'rotation = #'(45 0 0)
  \once \override Stem #'extra-offset = #'(-0.2 . -0.2)
  \once \override Stem #'flag-style = #'no-flag
  \once \override Accidental #'extra-offset = #'(4 . 0)
}

\relative c' {
  << { \fixA <b d!>8 } \ { \voiceThree \fixB dis } >> s
}

```



Double glissando

To connect chords with glissando lines, attach a second glissando to a hidden voice.

```

\relative c {
  \clef bass
  <<
  {

```

```

% new voice ( = \voiceOne), hidden
\hideNotes
% attach glissando to note heads
e2\glissando g
}
\\
{
% original voice with chords rearranged so that
% glissando is attached to a & c
<e a,>2\glissando <g c,>
}
>>
}

```



Forcing horizontal shift of notes

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

```

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

```



Making an object invisible with the 'transparent property

Setting the `transparent` property will cause an object to be printed in “invisible ink”: the object is not printed, but all its other behavior is retained. The object still takes up space, it takes part in collisions, and slurs, ties and beams can be attached to it.

This snippet demonstrates how to connect different voices using ties. Normally, ties only connect two notes in the same voice. By introducing a tie in a different voice, and blanking the first up-stem in that voice, the tie appears to cross voices.

```

\relative c'' {
  \time 2/4
  <<

```

```

{
  \once \override Stem #'transparent = ##t
  \once \override Stem #'length = #8
  b8 ~ b\noBeam
  \once \override Stem #'transparent = ##t
  \once \override Stem #'length = #8
  g8 ~ g\noBeam
}
\\
{
  b8 g g e
}
>>
}

```



Moving dotted notes in polyphony

When a dotted note in the upper voice is moved to avoid a collision with a note in another voice, the default is to move the upper note to the right. This behaviour can be over-ridden by using the `prefer-dotted-right` property of `NoteCollision`.

```

\new Staff \relative c' <<
{ f2. f4
  \override Staff.NoteCollision #'prefer-dotted-right = ##f
  f2. f4
  \override Staff.NoteCollision #'prefer-dotted-right = ##t
  f2. f4
}
\\
{ e4 e e e e e e e e e e }
>>

```



Suppressing warnings for clashing note columns

If notes from two voices with stems in the same direction are placed at the same position, and both voices have no shift or the same shift specified, the error message `'warning: ignoring too many clashing note columns'` will appear when compiling the LilyPond file. This message can be suppressed by setting the `'ignore-collision` property of the `NoteColumn` object to `##t`.

```

ignore = \override NoteColumn #'ignore-collision = ##t

\relative c' {
  <<
  \ignore

```

```
{ \stemDown f2 g }  
\\  
{ c2 c, }  
>>  
}
```



Staff notation

Section “Staff notation” in *Notation Reference*

Adding ambitus per voice

Ambitus can be added per voice. In this case, the ambitus must be moved manually to prevent collisions.

```
\new Staff <<
  \new Voice \with {
    \consists "Ambitus_engraver"
  } \relative c'' {
    \override Ambitus #'X-offset = #2.0
    \voiceOne
    c4 a d e
    f1
  }
  \new Voice \with {
    \consists "Ambitus_engraver"
  } \relative c' {
    \voiceTwo
    es4 f g as
    b1
  }
}>>
```



Adding an extra staff at a line break

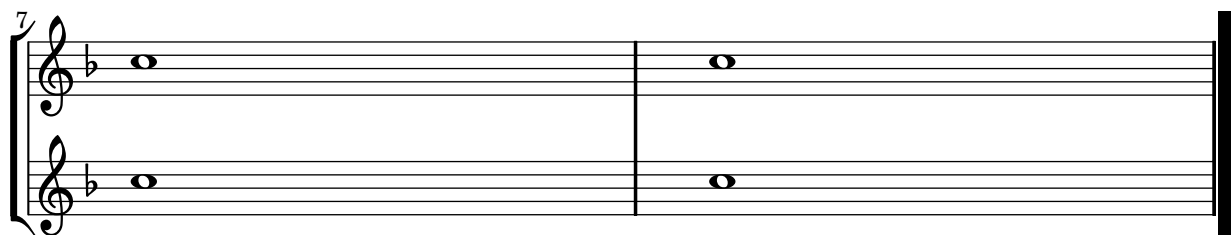
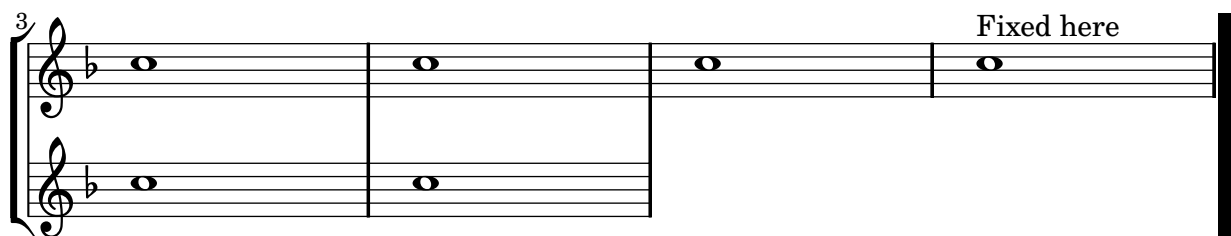
When adding a new staff at a line break, some extra space is unfortunately added at the end of the line before the break (to fit in a key signature change, which will never be printed anyway). The workaround is to add a setting of `Staff.explicitKeySignatureVisibility` as is shown in the example.

```
\score {
  \new StaffGroup \relative c'' {
    \new Staff
    \key f \major
    c1 c^"Unwanted extra space" \break
    << { c1 | c }
    \new Staff {
      \key f \major
      \once \override Staff.TimeSignature #'stencil = ##f
      c1 | c
    }
  }
  >>
  c1 | c^"Fixed here" \break
  << { c1 | c }
  \new Staff {
```

```

\once \set Staff.explicitKeySignatureVisibility = #end-of-line-invisible
\key f \major
\once \override Staff.TimeSignature #'stencil = ##f
c1 | c
}
>>
}
}

```



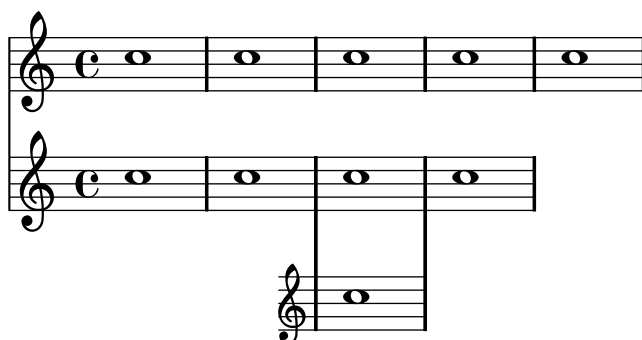
Adding an extra staff

An extra staff can be added (possibly temporarily) after the start of a piece.

```

\score {
  <<
    \new Staff \relative c'' { c1 | c | c | c | c }
    \new StaffGroup \relative c'' {
      \new Staff {
        c1 | c <<
          c1
          \new Staff {
            \once \override Staff.TimeSignature #'stencil = ##f
            c1
          }
        >>
        c1
      }
    }
  >>
}

```



Adding indicators to staves which get split after a break

This snippet defines the `\splitStaffBarLine` command, which adds arrows in north-east and south-east directions at a bar line, to denote that several voices sharing a staff will each continue on a staff of their own in the next system.

```
#(define-markup-command (arrow-at-angle layout props angle-deg length fill)
  (number? number? boolean?)
  (let* (
    ;; PI-OVER-180 and degrees->radians are taken from flag-styles.scm
    (PI-OVER-180 (/ (atan 1 1) 45))
    (degrees->radians (lambda (degrees) (* degrees PI-OVER-180)))
    (angle-rad (degrees->radians angle-deg))
    (target-x (* length (cos angle-rad)))
    (target-y (* length (sin angle-rad))))
    (interpret-markup layout props
      (markup
        #:translate (cons (/ target-x 2) (/ target-y 2))
        #:rotate angle-deg
        #:translate (cons (/ length -2) 0)
        #:concat (cons length 0)
        #:arrow-head X RIGHT fill))))))

splitStaffBarLineMarkup = \markup \with-dimensions #'(0 . 0) #'(0 . 0) {
  \combine
  \arrow-at-angle #45 #(sqrt 8) ##f
  \arrow-at-angle #-45 #(sqrt 8) ##f
}

splitStaffBarLine = {
  \once \override Staff.BarLine #'stencil =
    #(lambda (grob)
      (ly:stencil-combine-at-edge
        (ly:bar-line::print grob)
        X RIGHT
        (grob-interpret-markup grob splitStaffBarLineMarkup)
        0 0))
  \break
}

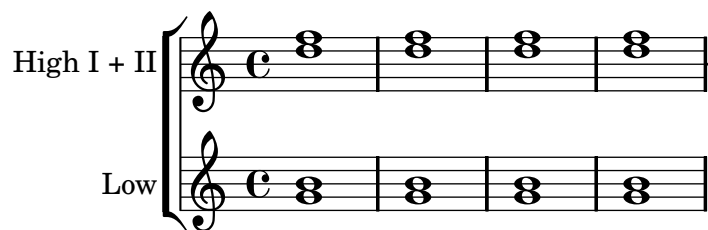
\paper {
  ragged-right = ##t
  short-indent = 5\mm
}
```

```

\score {
  <<
    \new ChoirStaff <<
      \new Staff \with { instrumentName = #"High I + II" } {
        <<
          \repeat unfold 4 f''1
          \\
          \repeat unfold 4 d''1
        >>
        \splitStaffBarLine
      }
      \new Staff \with { instrumentName = #"Low" } {
        <<
          \repeat unfold 4 b'1
          \\
          \repeat unfold 4 g'1
        >>
      }

      \new Staff \with { shortInstrumentName = #"H I" } {
        R1*4
        \repeat unfold 2 { r4 f''2 r4 } \repeat unfold 2 e''1
      }
      \new Staff \with { shortInstrumentName = #"H II" } {
        R1*4
        \repeat unfold 4 b'2 \repeat unfold 2 c''1
      }
      \new Staff \with { shortInstrumentName = #"L" } {
        R1*4
        <<
          \repeat unfold 4 g'1
          \\
          \repeat unfold 4 c'1
        >>
      }
    >>
  >>
  \layout {
    \context {
      \Staff \RemoveEmptyStaves
      \override VerticalAxisGroup #'remove-first = ##t
    }
  }
}

```



Changing the number of lines in a staff

The number of lines in a staff may be changed by overriding the `StaffSymbol` property `line-count`.

```
upper = \relative c'' {
  c4 d e f
}
```

```
lower = \relative c {
  \clef bass
  c4 b a g
}
```

```
\score {
  \context PianoStaff <<
    \new Staff {
      \upper
    }
    \new Staff {
      \override Staff.StaffSymbol #'line-count = #4
      \lower
    }
  >>
}
```



Changing the staff size

Though the simplest way to resize staves is to use `#{set-global-staff-size xx}`, an individual staff's size can be changed by scaling the properties `'staff-space` and `fontSize`.

```
<<
\new Staff {
  \relative c' {
    \dynamicDown
    c8\ff c c c c c c c
  }
}
\new Staff \with {
  fontSize = #-3
  \override StaffSymbol #'staff-space = #(magstep -3)
} {
  \clef bass
  c8 c c c c\ff c c c
}
>>
```



Changing the tempo without a metronome mark

To change the tempo in MIDI output without printing anything, make the metronome mark invisible.

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



Creating blank staves

To create blank staves, generate empty measures then remove the `Bar_number_engraver` from the `Score` context, and the `Time_signature_engraver`, `Clef_engraver` and `Bar_engraver` from the `Staff` context.

```

#(set-global-staff-size 20)

\score {
  {
    \repeat unfold 12 { s1 \break }
  }
  \layout {
    indent = 0\in
    \context {
      \Staff
      \remove "Time_signature_engraver"
      \remove "Clef_engraver"
      \remove "Bar_engraver"
    }
    \context {
      \Score
      \remove "Bar_number_engraver"
    }
  }
}

\paper {
  #(set-paper-size "letter")
  ragged-last-bottom = ##f
  line-width = 7.5\in
  left-margin = 0.5\in
  bottom-margin = 0.25\in
  top-margin = 0.25\in
}

```





Creating metronome marks in markup mode

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

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



Display bracket with only one staff in a system

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

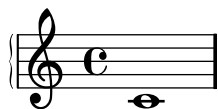
Note that in contexts such as **PianoStaff** and **GrandStaff** where the systems begin with a brace instead of a bracket, another property has to be set, as shown on the second system in the example.

```
\markup \left-column {
  \score {
    \new StaffGroup <<
      % Must be lower than the actual number of staff lines
      \override StaffGroup.SystemStartBracket #'collapse-height = #1
      \override Score.SystemStartBar #'collapse-height = #1
      \new Staff {
```

```

        c'1
      }
    >>
    \layout { }
  }
  \null
  \score {
    \new PianoStaff <<
      \override PianoStaff.SystemStartBrace #'collapse-height = #1
      \override Score.SystemStartBar #'collapse-height = #1
      \new Staff {
        c'1
      }
    >>
    \layout { }
  }
}

```



Forcing measure width to adapt to MetronomeMark's width

By default, metronome marks do not influence horizontal spacing. This has one downside: when using compressed rests, some metronome marks may be too close and therefore are printed vertically stacked, as demonstrated in the first part of this example. This can be solved through a simple override, as shown in the second half of the example.

```

example = {
  \tempo "Allegro"
  R1*6
  \tempo "Rall."
  R1*2
  \tempo "A tempo"
  R1*8
}

{
  \compressFullBarRests

  \example

  R1
  R1

  \override Score.MetronomeMark #'extra-spacing-width = #'(0 . 0)
  \example
}

```



Incipit

Incipits can be added using the instrument name grob, but keeping separate the instrument name definition and the incipit definition.

```
incipit =
#(define-music-function (parser location incipit-music) (ly:music?)
  #{
    \once \override Staff.InstrumentName #'self-alignment-X = #RIGHT
    \once \override Staff.InstrumentName #'self-alignment-Y = #UP
    \once \override Staff.InstrumentName #'Y-offset =
      #(lambda (grob)
        (+ 4 (system-start-text::calc-y-offset grob)))
    \once \override Staff.InstrumentName #'padding = #0.3
    \once \override Staff.InstrumentName #'stencil =
      #(lambda (grob)
        (let* ((instrument-name (ly:grob-property grob 'long-text))
              (layout (ly:output-def-clone (ly:grob-layout grob)))
              (music (make-sequential-music
                      (list (context-spec-music
                            (make-sequential-music
                              (list (make-property-set
                                    'instrumentName instrument-name)
                                    (make-grob-property-set
                                      'VerticalAxisGroup
                                      'Y-extent '(-4 . 4))))
                                'MensuralStaff)
                            $incipit-music)))
              (score (ly:make-score music))
              (mm (ly:output-def-lookup layout 'mm))
              (indent (ly:output-def-lookup layout 'indent))
              (width (ly:output-def-lookup layout 'incipit-width))
              (incipit-width (if (number? width)
                                (* width mm)
                                (* indent 0.5))))
          (ly:output-def-set-variable! layout 'indent (- indent
                                                            incipit-width))
          (ly:output-def-set-variable! layout 'line-width indent)
          (ly:output-def-set-variable! layout 'ragged-right #f)
          (ly:output-def-set-variable! layout 'ragged-last #f)
          (ly:output-def-set-variable! layout 'system-count 1)
          (ly:score-add-output-def! score layout)
          (ly:grob-set-property! grob 'long-text
                                (markup #:score score)))
      )
  )
)
```

```

        (system-start-text::print grob)))
#})

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

global = {
  \set Score.skipBars = ##t
  \key g \major
  \time 4/4

  % the actual music
  \skip 1*8

  % let finis bar go through all staves
  \override Staff.BarLine #'transparent = ##f

  % finis bar
  \bar "|."
}

discantusIncipit = <<
  \new MensuralVoice = "discantusIncipit" <<
    \repeat unfold 9 { s1 \noBreak }
    {
      \clef "neomensural-c1"
      \key f \major
      \time 2/2
      c'1.
    }
  >>
  \new Lyrics \lyricsto discantusIncipit { IV- }
>>

discantusNotes = {
  \transpose c' c'' {
    \clef "treble"
    d'2. d'4 |
    b e' d'2 |
    c'4 e'4.( d'8 c' b |
    a4) b a2 |
    b4.( c'8 d'4) c'4 |
    \once \override NoteHead #'transparent = ##t
    c'1 |
    b\breve |
  }
}

discantusLyrics = \lyricmode {
  Ju -- bi -- |
  la -- te De -- |
  o, om --
  nis ter -- |

```

```

    ra, __ om- |
    "... " |
    -us. |
}

altusIncipit = <<
  \new MensuralVoice = "altusIncipit" <<
    \repeat unfold 9 { s1 \noBreak }
    {
      \clef "neomensural-c3"
      \key f \major
      \time 2/2
      r1 f'1.
    }
  >>
  \new Lyrics \lyricsto altusIncipit { IV- }
>>

altusNotes = {
  \transpose c' c'' {
    \clef "treble"
    % two measures
    r2 g2. e4 fis g |
    a2 g4 e |
    fis g4.( fis16 e fis4) |
    g1 |
    \once \override NoteHead #'transparent = ##t
    g1 |
    g\breve |
  }
}

altusLyrics = \lyricmode {
  % two measures
  Ju -- bi -- la -- te |
  De -- o, om -- |
  nis ter -- ra, |
  "... " |
  -us. |
}

tenorIncipit = <<
  \new MensuralVoice = "tenorIncipit" <<
    \repeat unfold 9 { s1 \noBreak }
    {
      \clef "neomensural-c4"
      \key f \major
      \time 2/2
      r\longa
      r\breve
      r1 c'1.
    }
  >>

```

```

>>
\new Lyrics \lyricsto tenorIncipit { IV- }
>>

tenorNotes = {
  \transpose c' c' {
    \clef "treble_8"
    R1 |
    R1 |
    R1 |
    % two measures
    r2 d'2. d'4 b e' |
    \once \override NoteHead #'transparent = ##t
    e'1 |
    d'\breve |
  }
}

tenorLyrics = \lyricmode {
  % two measures
  Ju -- bi -- la -- te |
  "... " |
  -us.
}

bassusIncipit = <<
  \new MensuralVoice = "bassusIncipit" <<
    \repeat unfold 9 { s1 \noBreak }
    {
      \clef "bass"
      \key f \major
      \time 2/2
      %% incipit
      r\maxima
      f1.
    }
  >>
  \new Lyrics \lyricsto bassusIncipit { IV- }
>>

bassusNotes = {
  \transpose c' c' {
    \clef "bass"
    R1 |
    R1 |
    R1 |
    R1 |
    g2. e4 |
    \once \override NoteHead #'transparent = ##t
    e1 |
    g\breve |
  }
}

```

```

}

bassusLyrics = \lyricmode {
  Ju -- bi- |
  "... " |
  -us.
}

\score {
  <<
    \new StaffGroup = choirStaff <<
      \new Voice = "discantusNotes" <<
        \global
        \set Staff.instrumentName = #"Discantus"
        \incipit \discantusIncipit
        \discantusNotes
      >>
      \new Lyrics = "discantusLyrics" \lyricsto discantusNotes { \discantusLyrics }
      \new Voice = "altusNotes" <<
        \global
        \set Staff.instrumentName = #"Altus"
        \incipit \altusIncipit
        \altusNotes
      >>
      \new Lyrics = "altusLyrics" \lyricsto altusNotes { \altusLyrics }
      \new Voice = "tenorNotes" <<
        \global
        \set Staff.instrumentName = #"Tenor"
        \incipit \tenorIncipit
        \tenorNotes
      >>
      \new Lyrics = "tenorLyrics" \lyricsto tenorNotes { \tenorLyrics }
      \new Voice = "bassusNotes" <<
        \global
        \set Staff.instrumentName = #"Bassus"
        \incipit \bassusIncipit
        \bassusNotes
      >>
      \new Lyrics = "bassusLyrics" \lyricsto bassusNotes { \bassusLyrics }
    >>
  >>
  \layout {
    \context {
      \Score
      %% no bar lines in staves or lyrics
      \override BarLine #'transparent = ##t
    }
    %% the next two instructions keep the lyrics between the bar lines
    \context {
      \Lyrics
      \consists "Bar_engraver"
      \consists "Separating_line_group_engraver"
    }
  }
}

```

```

}
\context {
  \Voice
  %% no slurs
  \override Slur #'transparent = ##t
  %% Comment in the below "\remove" command to allow line
  %% breaking also at those bar lines where a note overlaps
  %% into the next measure. The command is commented out in this
  %% short example score, but especially for large scores, you
  %% will typically yield better line breaking and thus improve
  %% overall spacing if you comment in the following command.
  %%\remove "Forbid_line_break_engraver"
}
indent = 6\cm
incipit-width = 4\cm
}
}

```

Discantus

Altus

Tenor

Bassus

IV-

IV-

IV-

IV-

3

8

8

o, om - nis ter - ra, om- ... -us.

De - o, om - nis ter - ra, ... -us.

Ju - bi - la - te ... -us.

Ju - bi - la - te ... -us.

Inserting score fragments above a staff as markups

The `\markup` command is quite versatile. In this snippet, it contains a `\score` block instead of texts or marks.

```

tuning = \markup {
  \score {
    \new Staff \with { \remove "Time_signature_engraver" }
    {
      \clef bass
      <c, g, d g>1
    }
    \layout { ragged-right = ##t }
  }
}

\header {
  title = "Solo Cello Suites"
  subtitle = "Suite IV"
  subsubtitle = \markup { Originalstimmung: \general-align #Y #CENTER \tuning }
}

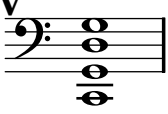
\layout { ragged-right = ##f }


\relative c'' {
  \time 4/8
  \times 2/3 { c8 d e } \times 2/3 { c d e }
  \times 2/3 { c8 d e } \times 2/3 { c d e }
  g8 a g a
  g8 a g a
}

```

Solo Cello Suites

Suite IV

Originalstimmung: 



Letter tablature formatting

Tablature can be formatted using letters instead of numbers.

```

music = \relative c {
  c4 d e f
  g4 a b c
  d4 e f g
}

```

```

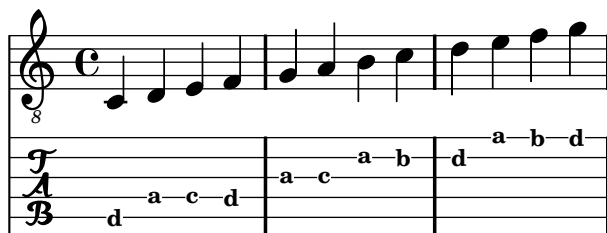
<<
\new Staff {
  \clef "G_8"
  \music
}

```

```

}
\new TabStaff \with {
  tablatureFormat = #fret-letter-tablature-format
}
{
  \music
}
>>

```



Making some staff lines thicker than the others

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

```

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

```



Measure counter

This snippet provides a workaround for emitting measure counters using transparent percent repeats.

```

<<
\context Voice = "foo" {
  \clef bass
  c4 r g r
  c4 r g r
  c4 r g r
  c4 r g r
}
\context Voice = "foo" {
  \set countPercentRepeats = ##t
  \override PercentRepeat #'transparent = ##t
  \override PercentRepeatCounter #'staff-padding = #1
  \repeat percent 4 { s1 }
}
>>

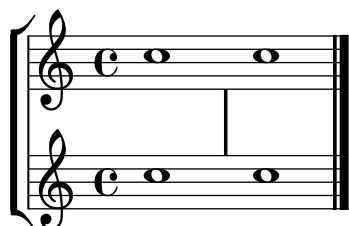
```



Mensurstriche layout (bar lines between the staves)

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

```
global = {
  \override Staff.BarLine #'transparent = ##t
  s1 s
  % the final bar line is not interrupted
  \revert Staff.BarLine #'transparent
  \bar " | ."
}
\new StaffGroup \relative c'' {
  <<
    \new Staff { << \global { c1 c } >> }
    \new Staff { << \global { c c } >> }
  >>
}
```



Modern TAB text clef

Use a markup text to replace the (TAB) clef glyph with a modern font.

```
\markup {
  This snippet is deprecated as of version 2.13.4 and will be removed
  in version 2.14.
}
```

This snippet is deprecated as of version 2.13.4 and will be removed in version 2.14.

Nesting staves

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

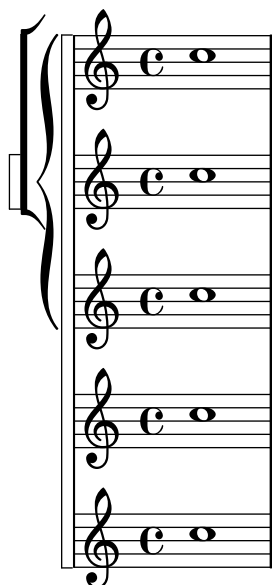
```
\new StaffGroup
\relative c'' <<
  \set StaffGroup.systemStartDelimiterHierarchy
    = #'(SystemStartSquare (SystemStartBrace (SystemStartBracket a
```

```

                                (SystemStartSquare b) ) c ) d)

\new Staff { c1 }
\new Staff { c1 }
\new Staff { c1 }
\new Staff { c1 }
\new Staff { c1 }
>>

```



Non-traditional key signatures

The commonly used `\key` command sets the `keySignature` property, in the `Staff` context.

To create non-standard key signatures, set this property directly. The format of this command is a list:

```
\set Staff.keySignature = #`(((octave . step) . alter) ((octave . step) . alter)
...)
```

where, for each element in the list, `octave` specifies the octave (0 being the octave from middle C to the B above), `step` specifies the note within the octave (0 means C and 6 means B), and `alter` is `,SHARP`, `,FLAT`, `,DOUBLE-SHARP` etc. (Note the leading comma.)

Alternatively, for each item in the list, using the more concise format `(step . alter)` specifies that the same alteration should hold in all octaves.

Here is an example of a possible key signature for generating a whole-tone scale:

```

\relative c' {
  \set Staff.keySignature = #`(((0 . 6) . ,FLAT)
                                ((0 . 5) . ,FLAT)
                                ((0 . 3) . ,SHARP))

  c4 d e fis
  aes4 bes c2
}

```



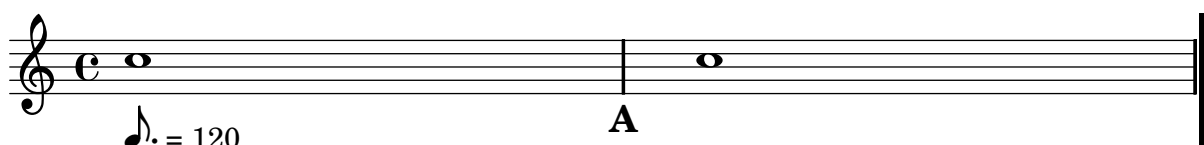
Printing metronome and rehearsal marks below the staff

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

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

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

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



Quoting another voice with transposition

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

```
\addQuote clarinet {
  \transposition bes
  \repeat unfold 8 { d'16 d' d'8 }
}

\addQuote sax {
  \transposition es'
  \repeat unfold 16 { a8 }
}

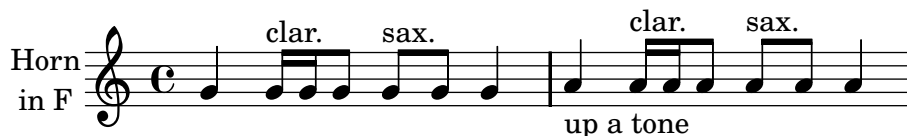
quoteTest = {
  % french horn
  \transposition f
  g'4
  << \quoteDuring #"clarinet" { \skip 4 } s4^"clar." >>
  << \quoteDuring #"sax" { \skip 4 } s4^"sax." >>
  g'4
}

{
  \set Staff.instrumentName =
  \markup {
```

```

\center-column { Horn \line { in F } }
}
\quoteTest
\transpose c' d' << \quoteTest s4_"up a tone" >>
}

```



Quoting another voice

The `quotedEventTypes` property determines the music event types which should be quoted. The default value is `(note-event rest-event tie-event beam-event tuplet-span-event)`, which means that only the notes, rests, ties, beams and tuplets of the quoted voice will appear in the `\quoteDuring` expression. In the following example, a 16th rest is not quoted since `rest-event` is not in `quotedEventTypes`.

For a list of event types, consult the “Music classes” section of the Internals Reference.

```

quoteMe = \relative c' {
  fis4 r16 a8.-> b4\ff c
}
\addQuote quoteMe \quoteMe

original = \relative c'' {
  c8 d s2
  \once \override NoteColumn #'ignore-collision = ##t
  es8 gis8
}

<<
\new Staff {
  \set Staff.instrumentName = #"quoteMe"
  \quoteMe
}
\new Staff {
  \set Staff.instrumentName = #"orig"
  \original
}
\new Staff \relative c'' <<
  \set Staff.instrumentName = #"orig+quote"
  \set Staff.quotedEventTypes =
    #'(note-event articulation-event)
  \original
  \new Voice {
    s4
    \set fontSize = #-4
    \override Stem #'length-fraction = #(magstep -4)
    \quoteDuring #"quoteMe" { \skip 2. }
  }
}
>>
>>

```

quoteMe

orig

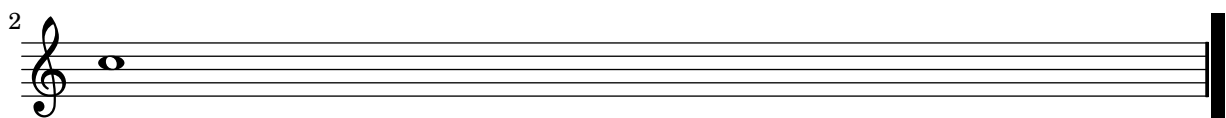
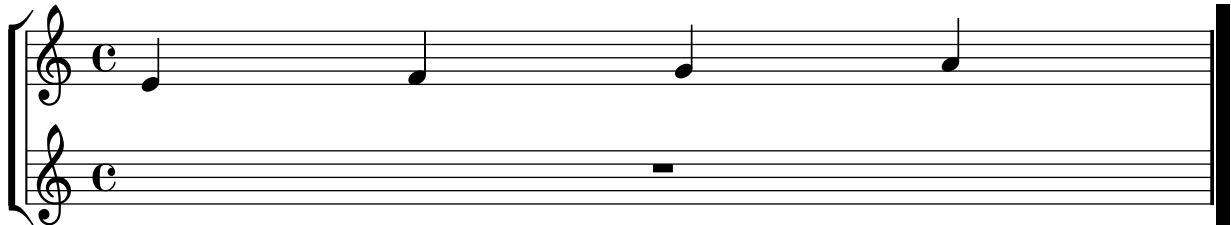
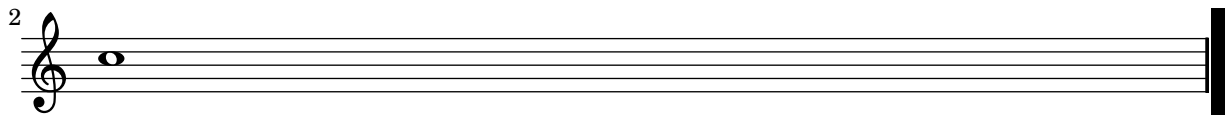
orig+quote

Removing the first empty line

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

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

```
\layout {
  \context {
    \Staff \RemoveEmptyStaves
    % To use the setting globally, uncomment the following line:
    % \override VerticalAxisGroup #'remove-first = ##t
  }
}
\new StaffGroup <<
  \new Staff \relative c' {
    e4 f g a \break
    c1
  }
  \new Staff {
    % To use the setting globally, comment this line,
    % uncomment the line in the \layout block above
    \override Staff.VerticalAxisGroup #'remove-first = ##t
    R1 \break
    R
  }
>>
\new StaffGroup <<
  \new Staff \relative c' {
    e4 f g a \break
    c1
  }
  \new Staff {
    R1 \break
    R
  }
>>
```



Tick bar lines

'Tick' bar lines are often used in music where the bar line is used only for coordination and is not meant to imply any rhythmic stress.

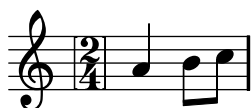
```
\relative c' {
  \set Score.defaultBarType = #"'"
  c4 d e f
  g4 f e d
  c4 d e f
  g4 f e d
  \bar "|."
}
```



Time signature in parentheses

The time signature can be enclosed within parentheses.

```
\relative c' {
  \override Staff.TimeSignature #'stencil = #(lambda (grob)
    (bracketify-stencil (ly:time-signature::print grob) Y 0.1 0.2 0.1))
  \time 2/4
  a4 b8 c
}
```



Tweaking clef properties

The command `\clef "treble_8"` is equivalent to setting `clefGlyph`, `clefPosition` (which controls the vertical position of the clef), `middleCPosition` and `clefOctavation`. A clef is printed when any of the properties except `middleCPosition` are changed.

Note that changing the glyph, the position of the clef, or the octavation does not in itself change the position of subsequent notes on the staff: the position of middle C must also be specified to do this. The positional parameters are relative to the staff center line, positive numbers displacing upwards, counting one for each line and space. The `clefOctavation` value would normally be set to 7, -7, 15 or -15, but other values are valid.

When a clef change takes place at a line break the new clef symbol is printed at both the end of the previous line and the beginning of the new line by default. If the warning clef at the end of the previous line is not required it can be suppressed by setting the `Staff` property `explicitClefVisibility` to the value `end-of-line-invisible`. The default behavior can be recovered with `\unset Staff.explicitClefVisibility`.

The following examples show the possibilities when setting these properties manually. On the first line, the manual changes preserve the standard relative positioning of clefs and notes, whereas on the second line, they do not.

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

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

  % The following clef changes do not preserve
  % the normal relationship between notes and clefs:

  \set Staff.clefGlyph = #"clefs.F"
  \set Staff.clefPosition = #2
  c'1
```

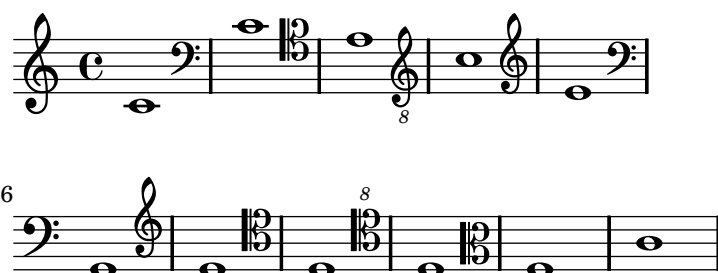
```

\set Staff.clefGlyph = #"clefs.G"
c'1
\set Staff.clefGlyph = #"clefs.C"
c'1
\set Staff.clefOctavation = #7
c'1
\set Staff.clefOctavation = #0
\set Staff.clefPosition = #0
c'1

% Return to the normal clef:

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

```



Use square bracket at the start of a staff group

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

```

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

```



Volta below chords

By adding the `Volta_engraver` to the relevant staff, volte can be put under chords.

```

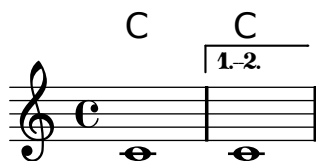
\score {
  <<
    \chords {

```

```

        c1
        c1
    }
    \new Staff \with {
        \consists "Volta_engraver"
    }
    {
        \repeat volta 2 { c'1 }
        \alternative { c' }
    }
>>
\layout {
    \context {
        \Score
        \remove "Volta_engraver"
    }
}
}

```



Volta multi staff

By adding the `Volta_engraver` to the relevant staff, volte can be put over staves other than the topmost one in a score.

```

voltaMusic = \relative c'' {
    \repeat volta 2 {
        c1
    }
    \alternative {
        d1
        e1
    }
}

<<
\new StaffGroup <<
    \new Staff \voltaMusic
    \new Staff \voltaMusic
>>
\new StaffGroup <<
    \new Staff \with { \consists "Volta_engraver" }
        \voltaMusic
    \new Staff \voltaMusic
>>
>>

```

The image displays two systems of musical notation, each consisting of two staves. The notation is in common time (C) and features a key signature of one flat (B-flat). The first system shows a first ending (1.) and a second ending (2.). The second system is identical to the first.

System 1:

- Staff 1: Treble clef, C-clef, common time. First measure: C4 (quarter), E4 (quarter). Second measure: B3 (half). Third measure: B3 (half), E4 (quarter), C4 (quarter).
- Staff 2: Treble clef, C-clef, common time. First measure: C4 (quarter), E4 (quarter). Second measure: B3 (half). Third measure: B3 (half), E4 (quarter), C4 (quarter).

System 2:

- Staff 1: Treble clef, C-clef, common time. First measure: C4 (quarter), E4 (quarter). Second measure: B3 (half). Third measure: B3 (half), E4 (quarter), C4 (quarter).
- Staff 2: Treble clef, C-clef, common time. First measure: C4 (quarter), E4 (quarter). Second measure: B3 (half). Third measure: B3 (half), E4 (quarter), C4 (quarter).

Editorial annotations

Section “Editorial annotations” in *Notation Reference*

Adding fingerings to a score

Fingering instructions can be entered using a simple syntax.

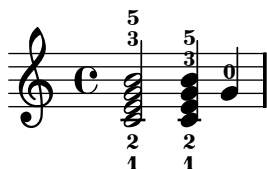
```
\relative c' ' {
  c4-1 d-2 f-4 e-3
}
```



Allowing fingerings to be printed inside the staff

By default, vertically oriented fingerings are positioned outside the staff. However, this behavior can be canceled. Note: you must use a chord construct <>, even if it is only a single note.

```
\relative c' {
  <c-1 e-2 g-3 b-5>2
  \override Fingering #'staff-padding = #'()
  <c-1 e-2 g-3 b-5>4 <g'-0>
}
```



Analysis brackets above the staff

Simple horizontal analysis brackets are added below the staff by default. The following example shows a way to place them above the staff instead.

```
\layout {
  \context {
    \Voice
    \consists "Horizontal_bracket_engraver"
  }
}
\relative c' ' {
  \once \override HorizontalBracket #'direction = #UP
  c2\startGroup
  d2\stopGroup
}
```



Applying note head styles depending on the step of the scale

The `shapeNoteStyles` property can be used to define various note head styles for each step of the scale (as set by the key signature or the `tonic` property). This property requires a set of symbols, which can be purely arbitrary (geometrical expressions such as `triangle`, `cross`, and `xcircle` are allowed) or based on old American engraving tradition (some latin note names are also allowed).

That said, to imitate old American song books, there are several predefined note head styles available through shortcut commands such as `\aikenHeads` or `\sacredHarpHeads`.

This example shows different ways to obtain shape note heads, and demonstrates the ability to transpose a melody without losing the correspondence between harmonic functions and note head styles.

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

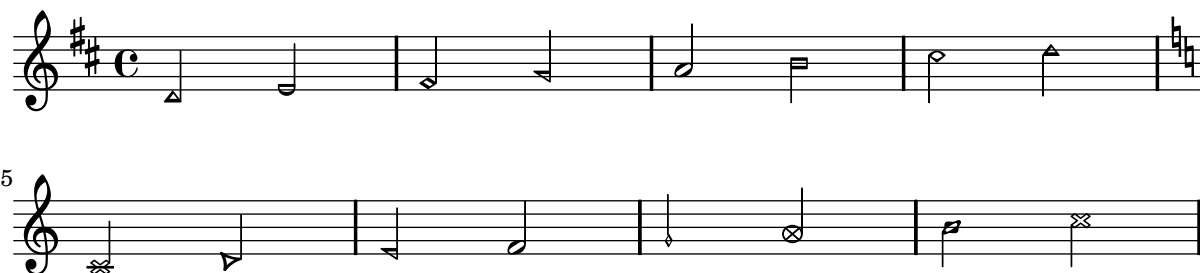
\new Staff {
  \transpose c d
  \relative c' {
    \set shapeNoteStyles = #'#(do re mi fa
                          #f la ti)

    \fragment
  }

  \break

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

    \fragment
  }
}
```



Avoiding collisions with chord fingerings

Fingerings and string numbers applied to individual notes will automatically avoid beams and stems, but this is not true by default for fingerings and string numbers applied to the individual notes of chords. The following example shows how this default behavior can be overridden.

```

\relative c' {
  \set fingeringOrientations = #'(up)
  \set stringNumberOrientations = #'(up)
  \set strokeFingerOrientations = #'(up)

  % Default behavior
  r8
  <f c'-5>8
  <f c'\5>8
  <f c'-\rightHandFinger #2 >8

  % Corrected to avoid collisions
  r8
  \override Fingering #'add-stem-support = ##t
  <f c'-5>8
  \override StringNumber #'add-stem-support = ##t
  <f c'\5>8
  \override StrokeFinger #'add-stem-support = ##t
  <f c'-\rightHandFinger #2 >8
}

```



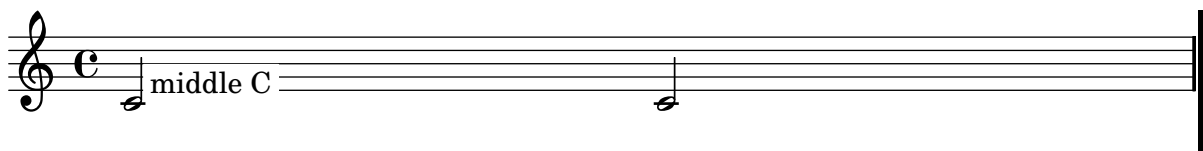
Blanking staff lines using the `\whiteout` command

The `\whiteout` command underlays a markup with a white box. Since staff lines are in a lower layer than most other grobs, this white box will not overlap any other grob.

```

\layout { ragged-right = ##f }
\relative c' {
  \override TextScript #'extra-offset = #'(2 . 4)
  c2-\markup { \whiteout \pad-markup #0.5 "middle C" } c
}

```



Changing a single note's size in a chord

Individual note heads in a chord can be modified with the `\tweak` command inside a chord, by altering the `font-size` property.

Inside the chord (within the brackets `< >`), before the note to be altered, place the `\tweak` command, followed by `font-size` and define the proper size like `#-2` (a tiny note head).

```

\relative c' {
  <\tweak #'font-size #-2 c e g c \tweak #'font-size #-2 e>1-\markup { A tiny e }-\markup {
}

```



Changing the appearance of a slur from solid to dotted or dashed

The appearance of slurs may be changed from solid to dotted or dashed.

```
\relative c' {
  c4( d e c)
  \slurDotted
  c4( d e c)
  \slurSolid
  c4( d e c)
  \slurDashed
  c4( d e c)
  \slurSolid
  c4( d e c)
}
```



Coloring notes depending on their pitch

It is possible to color note heads depending on their pitch and/or their names: the function used in this example even makes it possible to distinguish enharmonics.

```
%Association list of pitches to colors.
#(define color-mapping
  (list
    (cons (ly:make-pitch 0 0 NATURAL) (x11-color 'red))
    (cons (ly:make-pitch 0 0 SHARP) (x11-color 'green))
    (cons (ly:make-pitch 0 1 FLAT) (x11-color 'green))
    (cons (ly:make-pitch 0 2 NATURAL) (x11-color 'red))
    (cons (ly:make-pitch 0 2 SHARP) (x11-color 'green))
    (cons (ly:make-pitch 0 3 FLAT) (x11-color 'red))
    (cons (ly:make-pitch 0 3 NATURAL) (x11-color 'green))
    (cons (ly:make-pitch 0 4 SHARP) (x11-color 'red))
    (cons (ly:make-pitch 0 5 NATURAL) (x11-color 'green))
    (cons (ly:make-pitch 0 5 FLAT) (x11-color 'red))
    (cons (ly:make-pitch 0 6 SHARP) (x11-color 'red))
    (cons (ly:make-pitch 0 1 NATURAL) (x11-color 'blue))
    (cons (ly:make-pitch 0 3 SHARP) (x11-color 'blue))
    (cons (ly:make-pitch 0 4 FLAT) (x11-color 'blue))
    (cons (ly:make-pitch 0 5 SHARP) (x11-color 'blue))
  )
)
```

```

      (cons (ly:make-pitch 0 6 FLAT) (x11-color 'blue))))

%Compare pitch and alteration (not octave).
#(define (pitch-equals? p1 p2)
  (and
    (= (ly:pitch-alteration p1) (ly:pitch-alteration p2))
    (= (ly:pitch-notename p1) (ly:pitch-notename p2))))

#(define (pitch-to-color pitch)
  (let ((color (assoc pitch color-mapping pitch-equals?)))
    (if color
      (cdr color))))

#(define (color-notehead grob)
  (pitch-to-color
    (ly:event-property (event-cause grob) 'pitch)))

\score {
  \new Staff \relative c' {
    \override NoteHead #'color = #color-notehead
    c8 b d dis ees f g aes
  }
}

```



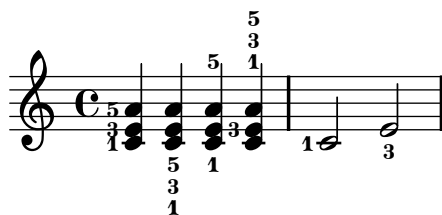
Controlling the placement of chord fingerings

The placement of fingering numbers can be controlled precisely. For fingering orientation to apply, you must use a chord construct <> even if it is a single note.

```

\relative c' {
  \set fingeringOrientations = #'(left)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(down)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(down right up)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(up)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(left)
  <c-1>2
  \set fingeringOrientations = #'(down)
  <e-3>2
}

```



Creating a delayed turn

Creating a delayed turn, where the lower note of the turn uses the accidental, requires several overrides. The `outside-staff-priority` property must be set to `#f`, as otherwise this would take precedence over the `avoid-slur` property. The value of `halign` is used to position the turn horizontally.

```
\relative c'' {
  \once \override TextScript #'avoid-slur = #'inside
  \once \override TextScript #'outside-staff-priority = ##f
  c2(~\markup \tiny \override #'(baseline-skip . 1) {
    \halign #-4
    \center-column {
      \sharp
      \musicglyph #"scripts.turn"
    }
  }
  d4.) c8
}
```



Creating blank staves

To create blank staves, generate empty measures then remove the `Bar_number_engraver` from the `Score` context, and the `Time_signature_engraver`, `Clef_engraver` and `Bar_engraver` from the `Staff` context.

```
 #(set-global-staff-size 20)

\score {
  {
    \repeat unfold 12 { s1 \break }
  }
  \layout {
    indent = 0\in
    \context {
      \Staff
      \remove "Time_signature_engraver"
      \remove "Clef_engraver"
      \remove "Bar_engraver"
    }
    \context {
      \Score
      \remove "Bar_number_engraver"
    }
  }
}
```

```
}  
  
\paper {  
  #(set-paper-size "letter")  
  ragged-last-bottom = ##f  
  line-width = 7.5\in  
  left-margin = 0.5\in  
  bottom-margin = 0.25\in  
  top-margin = 0.25\in  
}
```

Default direction of stems on the center line of the staff

The default direction of stems on the center line of the staff is set by the `Stem` property `neutral-direction`.

```
\relative c' {
  a4 b c b
  \override Stem #'neutral-direction = #up
  a4 b c b
  \override Stem #'neutral-direction = #down
  a4 b c b
}
```



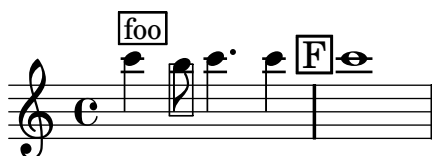
Drawing boxes around grobs

The `print`-function can be overridden to draw a box around an arbitrary grob.

```
\relative c' {
  \override TextScript #'stencil =
    #(make-stencil-boxer 0.1 0.3 ly:text-interface::print)
  c'4^"foo"

  \override Stem #'stencil =
    #(make-stencil-boxer 0.05 0.25 ly:stem::print)
  \override Score.RehearsalMark #'stencil =
    #(make-stencil-boxer 0.15 0.3 ly:text-interface::print)
  b8

  \revert Stem #'stencil
  c4. c4
  \mark "F"
  c1
}
```



Drawing circles around various objects

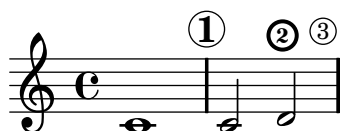
The `\circle` markup command draws circles around various objects, for example fingering indications. For other objects, specific tweaks may be required: this example demonstrates two strategies for rehearsal marks and measure numbers.

```
\relative c' {
  c1
```

```

\set Score.markFormatter =
  #(\lambda (mark context)
    (make-circle-markup (format-mark-numbers mark context)))
\mark \default
c2 d^\markup {
  \override #'(thickness . 3) {
    \circle \finger 2
  }
}
\override Score.BarNumber #'break-visibility = #all-visible
\override Score.BarNumber #'stencil =
  #(\make-stencil-circler 0.1 0.25 ly:text-interface::print)
}

```



Embedding native PostScript in a \markup block

PostScript code can be directly inserted inside a \markup block.

% PostScript is a registered trademark of Adobe Systems Inc.

```

\relative c'' {
  a4-\markup { \postscript #"3 4 moveto 5 3 rlineto stroke" }
  -\markup { \postscript #"[ 0 1 ] 0 setdash 3 5 moveto 5 -3 rlineto stroke " }

  b4-\markup { \postscript #"3 4 moveto 0 0 1 2 8 4 20 3.5 rcurveto stroke" }
  s2
  a'1
}

```



Grid lines: changing their appearance

The appearance of grid lines can be changed by overriding some of their properties.

```

\score {
  \new ChoirStaff <<
    \new Staff {
      \relative c'' {
        \stemUp
        c'4. d8 e8 f g4
      }
    }
  \new Staff {
    \relative c {
      % this moves them up one staff space from the default position
    }
  }
}

```

```

\override Score.GridLine #'extra-offset = #'(0.0 . 1.0)
\stemDown
\clef bass
\once \override Score.GridLine #'thickness = #5.0
c4
\once \override Score.GridLine #'thickness = #1.0
g'4
\once \override Score.GridLine #'thickness = #3.0
f4
\once \override Score.GridLine #'thickness = #5.0
e4
}
}
>>
\layout {
  \context {
    \Staff
    % set up grids
    \consists "Grid_point_engraver"
    % set the grid interval to one quarter note
    gridInterval = #(ly:make-moment 1 4)
  }
  \context {
    \Score
    \consists "Grid_line_span_engraver"
    % this moves them to the right half a staff space
    \override NoteColumn #'X-offset = #-0.5
  }
}
}

```



Grid lines: emphasizing rhythms and notes synchronization

Regular vertical lines can be drawn between staves to show note synchronization; however, in case of monophonic music, you may want to make the second staff invisible, and make the lines shorter like in this snippet.

```

\score {
  \new ChoirStaff {
    \relative c'' <<
    \new Staff {
      \time 12/8
      \stemUp

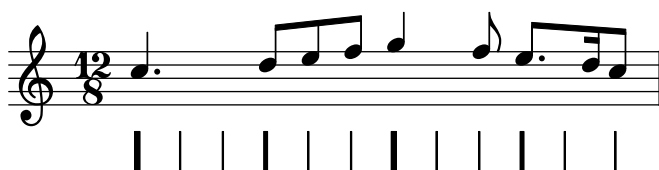
```

```

    c4. d8 e8 f g4 f8 e8. d16 c8
  }
  \new Staff {
    % hides staff and notes so that only the grid lines are visible
    \hideNotes
    \override Staff.BarLine #'transparent = ##t
    \override Staff.StaffSymbol #'line-count = #0
    \override Staff.TimeSignature #'transparent = ##t
    \override Staff.Clef #'transparent = ##t

    % dummy notes to force regular note spacing
    \once \override Score.GridLine #'thickness = #4.0
    c8 c c
    \once \override Score.GridLine #'thickness = #3.0
    c8 c c
    \once \override Score.GridLine #'thickness = #4.0
    c8 c c
    \once \override Score.GridLine #'thickness = #3.0
    c8 c c
  }
  >>
}
\layout {
  \context {
    \Score
    \consists "Grid_line_span_engraver"
    % center grid lines horizontally below note heads
    \override NoteColumn #'X-offset = #-0.5
  }
  \context {
    \Staff
    \consists "Grid_point_engraver"
    gridInterval = #(ly:make-moment 1 8)
    % set line length and positioning:
    % two staff spaces above center line on hidden staff
    % to four spaces below center line on visible staff
    \override GridPoint #'Y-extent = #'(2 . -4)
  }
  ragged-right = ##t
}
}

```



Making some staff lines thicker than the others

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

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



Marking notes of spoken parts with a cross on the stem

This example shows how to put crosses on stems. Mark the beginning of a spoken section with the `\speakOn` keyword, and end it with the `\speakOff` keyword.

```
speakOn = {
  \override Stem #'stencil = #(lambda (grob)
    (let* ((x-parent (ly:grob-parent grob X))
      (is-rest? (ly:grob? (ly:grob-object x-parent 'rest))))
    (if is-rest?
      empty-stencil
      (ly:stencil-combine-at-edge
        (ly:stem::print grob)
        Y
        (- (ly:grob-property grob 'direction))
        (grob-interpret-markup grob
          (markup #:hspace -1.025 #:fontsize -4
            #:musicglyph "noteheads.s2cross"))
        -2.3 0))))
}

speakOff = {
  \revert Stem #'stencil
}

\score {
  \new Staff {
    \relative c'' {
      a4 b a c
      \speakOn
      g4 f r g
      b4 r d e
      \speakOff
      c4 a g f
    }
  }
}
```



Measure counter

This snippet provides a workaround for emitting measure counters using transparent percent repeats.

```
<<
  \context Voice = "foo" {
    \clef bass
    c4 r g r
    c4 r g r
    c4 r g r
    c4 r g r
  }
  \context Voice = "foo" {
    \set countPercentRepeats = ##t
    \override PercentRepeat #'transparent = ##t
    \override PercentRepeatCounter #'staff-padding = #1
    \repeat percent 4 { s1 }
  }
>>
```



Positioning fingering indications precisely

Generally the options available for positioning the fingering of chords work well by default, but if one of the indications needs to be positioned more precisely the following tweak may be used. This is particularly useful for correcting the positioning when intervals of a second are involved.

```
\relative c' {
  \set fingeringOrientations = #'(left)
  <c-1 d-2 a'-5>4
  <c-1 d-\tweak #'extra-offset #'(0 . 0.7)-2 a'-5>4
  \set fingeringOrientations = #'(down)
  <c-1 d-2 a'-5>4
  <c-1 d-\tweak #'extra-offset #'(-1.2 . 0)-2 a'-5>4
  \set fingeringOrientations = #'(down right up)
  <c-1 d-2 a'-5>4
  <c-1 d-\tweak #'extra-offset #'(-1 . 1.2)-2 a'-5>4
  \set fingeringOrientations = #'(up)
  <c-1 d-2 a'-5>4
  <c-1 d-\tweak #'extra-offset #'(-1.2 . 0)-2 a'-5>4
}
```



Positioning text markups inside slurs

Text markups need to have the `outside-staff-priority` property set to `false` in order to be printed inside slurs.

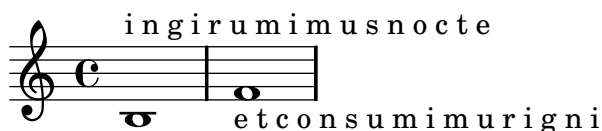
```
\relative c' {
  \override TextScript #'avoid-slur = #'inside
  \override TextScript #'outside-staff-priority = ##f
  c2(^\markup { \halign #-10 \natural } d4.) c8
}
```



Printing text from right to left

It is possible to print text from right to left in a markup object, as demonstrated here.

```
{
  b1^\markup {
    \line { i n g i r u m i m u s n o c t e }
  }
  f'_\markup {
    \override #'(text-direction . -1)
    \line { i n g i r u m i m u s n o c t e }
  }
}
```



String number extender lines

Make an extender line for string number indications, showing that a series of notes is supposed to be played all on the same string.

```
stringNumberSpanner =
#(define-music-function (parser location StringNumber) (string?)
  #{
    \override TextSpanner #'style = #'solid
    \override TextSpanner #'font-size = #-5
    \override TextSpanner #'(bound-details left stencil-align-dir-y) = #CENTER
    \override TextSpanner #'(bound-details left text) = \markup { \circle \number $StringNumber }
  })
```

```

\relative c {
  \clef "treble_8"
  \stringNumberSpanner "5"
  \textSpannerDown
  a8\startTextSpan
  b c d e f\stopTextSpan
  \stringNumberSpanner "4"
  g\startTextSpan a
  bes4 a g2\stopTextSpan
}

```



Using PostScript to generate special note head shapes

When a note head with a special shape cannot easily be generated with graphic markup, PostScript code can be used to generate the shape. This example shows how a parallelogram-shaped note head is generated.

```

parallelogram =
  #(ly:make-stencil (list 'embedded-ps
    "gsave
      currentpoint translate
      newpath
      0 0.25 moveto
      1.3125 0.75 lineto
      1.3125 -0.25 lineto
      0 -0.75 lineto
      closepath
      fill
      grestore" )
    (cons 0 1.3125)
    (cons 0 0))

myNoteHeads = \override NoteHead #'stencil = \parallelogram
normalNoteHeads = \revert NoteHead #'stencil

\relative c'' {
  \myNoteHeads
  g4 d'
  \normalNoteHeads
  <f, \tweak #'stencil \parallelogram b e>4 d
}

```



Using the whiteout property

Any graphical object can be printed over a white background to mask parts of objects that lie beneath. This can be useful to improve the appearance of collisions in complex situations when repositioning objects is impractical. It is necessary to explicitly set the **layer** property to control which objects are masked by the white background.

In this example the collision of the tie with the time signature is improved by masking out the part of the tie that crosses the time signature by setting the **whiteout** property of **TimeSignature**. To do this **TimeSignature** is moved to a layer above **Tie**, which is left in the default layer of 1, and **StaffSymbol** is moved to a layer above **TimeSignature** so it is not masked.

```
{
  \override Score.StaffSymbol #'layer = #4
  \override Staff.TimeSignature #'layer = #3
  b'2 b'~
  \once \override Staff.TimeSignature #'whiteout = ##t
  \time 3/4
  b' r4
}
```



Text

Section “Text” in *Notation Reference*

Adding the current date to a score

With a little Scheme code, the current date can easily be added to a score.

```
% first, define a variable to hold the formatted date:
date = #(strftime "%d-%m-%Y" (localtime (current-time)))
```

```
% use it in the title block:
\header {
  title = "Including the date!"
  subtitle = \date
}
```

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

```
% and use it in a \markup block:
\markup {
  \date
}
```

Including the date!

06-07-2010



06-07-2010

Adjusting lyrics vertical spacing

This snippet shows how to bring the lyrics line closer to the staff.

```
% Default layout:
```

```
<<
```

```
\new Staff \new Voice = melody \relative c' {
  c4 d e f
  g4 f e d
  c1
}
```

```
\new Lyrics \lyricsto melody { aa aa aa aa aa aa aa aa }
```

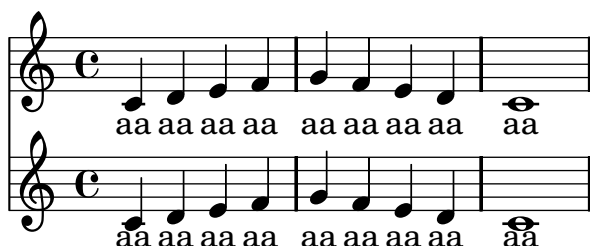
```
% Reducing the minimum space below the staff and above the lyrics:
```

```
\new Staff {
  \new Voice = melody \relative c' {
```

```

      c4 d e f
      g4 f e d
      c1
    }
  }
  \new Lyrics \with {
    \override VerticalAxisGroup #'inter-staff-spacing = #'((space . 1))
  }
  \lyricsto melody { aa aa aa aa aa aa aa aa aa }
>>

```



Aligning and centering instrument names

The horizontal alignment of instrument names is tweaked by changing the `Staff.InstrumentName #'self-alignment-X` property. The `\layout` variables `indent` and `short-indent` define the space in which the instrument names are aligned before the first and the following systems, respectively.

```

\paper {
  left-margin = 3\cm
}

\score {
  \new StaffGroup <<
    \new Staff {
      \override Staff.InstrumentName #'self-alignment-X = #LEFT
      \set Staff.instrumentName = \markup \left-column {
        "Left aligned"
        "instrument name"
      }
      \set Staff.shortInstrumentName = #"Left"
      c'1
      \break
      c'1
    }
    \new Staff {
      \override Staff.InstrumentName #'self-alignment-X = #CENTER
      \set Staff.instrumentName = \markup \center-column {
        Centered
        "instrument name"
      }
      \set Staff.shortInstrumentName = #"Centered"
      g'1
      g'1
    }
  }
}

```

```


\new Staff {
  \override Staff.InstrumentName #'self-alignment-X = #RIGHT
  \set Staff.instrumentName = \markup \right-column {
    "Right aligned"
    "instrument name"
  }
  \set Staff.shortInstrumentName = #"Right"
  e'1
  e'1
}
>>
\layout {
  ragged-right = ##t
  indent = 4\cm
  short-indent = 2\cm
}
}

```

Left aligned
instrument name

Centered
instrument name

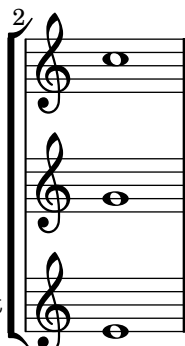
Right aligned
instrument name



Left

Centered

Right



Aligning marks with various notation objects

If specified, text marks may be aligned with notation objects other than bar lines. These objects include ambitus, breathing-sign, clef, custos, staff-bar, left-edge, key-cancellation, key-signature, and time-signature.

In such cases, text marks will be horizontally centered above the object. However this can be changed, as demonstrated on the second line of this example (in a score with multiple staves, this setting should be done for all the staves).

```

\relative c' {
  e1

```

```

% the RehearsalMark will be centered above the Clef

```

```

\override Score.RehearsalMark #'break-align-symbols = #'(clef)
\key a \major
\clef treble
\mark ""
e1

% the RehearsalMark will be centered above the TimeSignature
\override Score.RehearsalMark #'break-align-symbols = #'(time-signature)
\key a \major
\clef treble
\time 3/4
\mark \markup { \char ##x2193 }
e2.

% the RehearsalMark will be centered above the KeySignature
\override Score.RehearsalMark #'break-align-symbols = #'(key-signature)
\key a \major
\clef treble
\time 4/4
\mark \markup { \char ##x2193 }
e1

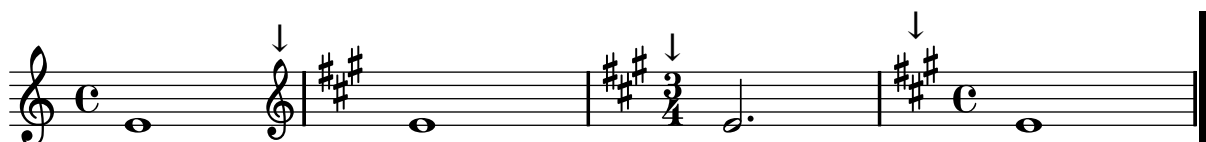
\break
e1

% the RehearsalMark will be aligned with the left edge of the KeySignature
\once \override Score.KeySignature #'break-align-anchor-alignment = #LEFT
\mark \markup { \char ##x2193 }
\key a \major
e1

% the RehearsalMark will be aligned with the right edge of the KeySignature
\once \override Score.KeySignature #'break-align-anchor-alignment = #RIGHT
\key a \major
\mark \markup { \char ##x2193 }
e1

% the RehearsalMark will be aligned with the left edge of the KeySignature
% and then shifted right by one unit.
\once \override Score.KeySignature #'break-align-anchor = #1
\key a \major
\mark \markup { \char ##x2193 }
e1
}

```





Aligning objects created with the `\mark` command

By default the `\mark` command centers objects over a bar line. This behavior can be modified to align at right or left.

```
\relative c' {
  c1 \mark "(Center)"
  c1
  \once \override Score.RehearsalMark #'self-alignment-X = #LEFT
  \mark "(Left)"
  c4 c c c
  c4 c c c
  \once \override Score.RehearsalMark #'self-alignment-X = #RIGHT
  \mark "(Right)"
  c1
}
```



Blanking staff lines using the `\whiteout` command

The `\whiteout` command underlays a markup with a white box. Since staff lines are in a lower layer than most other grobs, this white box will not overlap any other grob.

```
\layout { ragged-right = ##f }
\relative c' {
  \override TextScript #'extra-offset = #'(2 . 4)
  c2-\markup { \whiteout \pad-markup #0.5 "middle C" } c
}
```



Center text below hairpin dynamics

This example provides a function to typeset a hairpin (de)crescendo with some additional text below it, such as “molto” or “poco”. The example also illustrates how to modify the way an object is normally printed, using some Scheme code.

```
hairpinWithCenteredText =
#(define-music-function (parser location text) (markup?)
#{
  \override Voice.Hairpin #'stencil = #(lambda (grob)
    (ly:stencil-aligned-to
      (ly:stencil-combine-at-edge
```

```

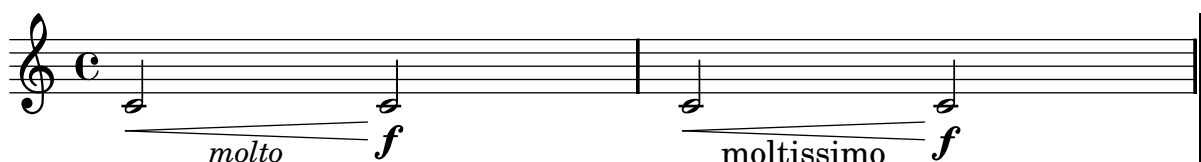
      (ly:stencil-aligned-to (ly:hairpin::print grob) X CENTER)
      Y DOWN
      (ly:stencil-aligned-to (grob-interpret-markup grob $text) X CENTER))
    X LEFT))
#})

hairpinMolto = \hairpinWithCenteredText \markup { \italic molto }
hairpinMore = \hairpinWithCenteredText \markup { \larger moltissimo }

\layout { ragged-right = ##f }

\relative c' {
  \hairpinMolto
  c2\< c\f
  \hairpinMore
  c2\< c\f
}

```



Centering markup on note heads automatically

For technical reasons, text scripts attached to note heads cannot easily be centered on a note head's width, unlike articulations.

Instead of using trial-and-error offset tweaks, this snippet accesses a note head (or rest) from the `TextScript` object's horizontal parent (a paper column), using its extent to correct the positioning.

```

textScriptCenterOnNote = \override TextScript #'X-offset =
#(lambda (grob)
  (let* ((paper-col (ly:grob-parent grob X))
        (elts (ly:grob-object paper-col 'elements))
        (rhythmic-head grob))

    (for-each
      (lambda (idx)
        (let ((elt (ly:grob-array-ref elts idx)))
          (if (grob::has-interface elt
                                   'rhythmic-grob-interface)
              (set! rhythmic-head elt))))
      (reverse (iota (ly:grob-array-length elts))))

    (+
      (ly:self-alignment-interface::x-aligned-on-self grob)
      (interval-center
        (ly:grob-robust-relative-extent rhythmic-head rhythmic-head X))))))

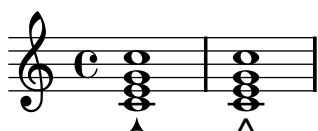
\relative c' {
  \override TextScript #'self-alignment-X = #CENTER

```

```

\textScriptCenterOnNote
<c e g c>1-\markup \arrow-head #Y #UP ##t
<c e g c>1-\markup \huge "^"
}

```



Changing the default text font family

The default font families for text can be overridden with `make-pango-font-tree`.

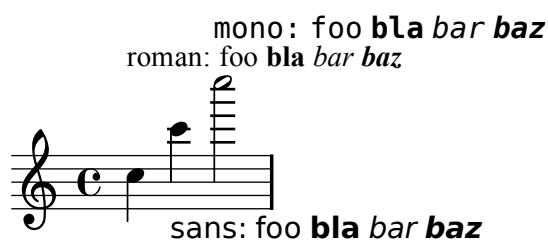
```

\paper {
  % change for other default global staff size.
  myStaffSize = #20
  %{
    run
    lilypond -dshow-available-fonts blabla
    to show all fonts available in the process log.
  %}

  #(define fonts
    (make-pango-font-tree "Times New Roman"
                          "Nimbus Sans"
                          "Luxi Mono"
                          "Helvetica"
                          "Courier"
      (/ myStaffSize 20)))
}

\relative c' {
  c4^\markup {
    roman: foo \bold bla \italic bar \italic \bold baz
  }
  c'4_\markup {
    \override #'(font-family . sans)
    {
      sans: foo \bold bla \italic bar \italic \bold baz
    }
  }
  c'2^\markup {
    \override #'(font-family . typewriter)
    {
      mono: foo \bold bla \italic bar \italic \bold baz
    }
  }
}

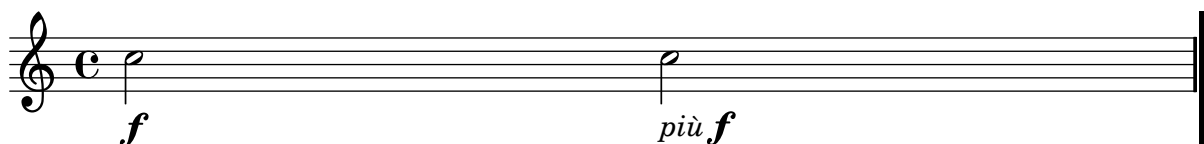
```



Combining dynamics with markup texts

Some dynamics may involve text indications (such as “più forte” or “piano subito”). These can be produced using a `\markup` block.

```
piuF = \markup { \italic più \dynamic f }
\layout { ragged-right = ##f }
\relative c'' {
  c2\f c-\piuF
}
```



Combining two parts on the same staff

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

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

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



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

\score {
  <<
  <<
  \new Staff {
    \set Staff.instrumentName = #"Standard polyphony"
    << \musicUp \\\musicDown >>
  }
}
```

```

\new Staff \with { printPartCombineTexts = ##f } {
  \set Staff.instrumentName = #"PartCombine without texts"
  \partcombine \musicUp \musicDown
}
\new Staff {
  \set Staff.instrumentName = #"PartCombine with texts"
  \partcombine \musicUp \musicDown
}
>>
>>
\layout {
  indent = 6.0\cm
  \context {
    \Score
    \override SystemStartBar #'collapse-height = #30
  }
}
}

```

Standard polyphony	
PartCombine without texts	
PartCombine with texts	

Creating "real" parenthesized dynamics

Although the easiest way to add parentheses to a dynamic mark is to use a `\markup` block, this method has a downside: the created objects will behave like text markups, and not like dynamics.

However, it is possible to create a similar object using the equivalent Scheme code (as described in the Notation Reference), combined with the `make-dynamic-script` function. This way, the markup will be regarded as a dynamic, and therefore will remain compatible with commands such as `\dynamicUp` or `\dynamicDown`.

```

parenF = #(make-dynamic-script (markup #:line (#:normal-text #:italic
  #:fontsize 2 "(" #:hspace -0.8 #:dynamic "f" #:normal-text
  #:italic #:fontsize 2 ")"))))

\relative c' {
  c4\parenF c c \dynamicUp c\parenF
}

```



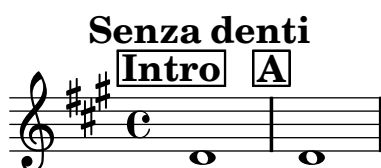
Creating simultaneous rehearsal marks

Unlike text scripts, rehearsal marks cannot be stacked at a particular point in a score: only one `RehearsalMark` object is created. Using an invisible measure and bar line, an extra rehearsal mark can be added, giving the appearance of two marks in the same column. This method may also prove useful for placing rehearsal marks at both the end of one system and the start of the following system.

```
{
  \key a \major
  \set Score.markFormatter = #format-mark-box-letters
  \once \override Score.RehearsalMark #'outside-staff-priority = #5000
  \once \override Score.RehearsalMark #'self-alignment-X = #LEFT
  \once \override Score.RehearsalMark #'break-align-symbols = #'(key-signature)
  \mark \markup { \bold { Senza denti } }

  % the hidden measure and bar line
  % \cadenzaOn turns off automatic calculation of bar numbers
  \cadenzaOn
  \once \override Score.TimeSignature #'stencil = ##f
  \time 1/16
  s16 \bar ""
  \cadenzaOff

  \time 4/4
  \once \override Score.RehearsalMark #'self-alignment-X = #LEFT
  \mark \markup { \box \bold Intro }
  d'1
  \mark \default
  d'1
}
```



Creating text spanners

The `\startTextSpan` and `\stopTextSpan` commands allow the creation of text spanners as easily as pedal indications or octavations. Override some properties of the `TextSpanner` object to modify its output.

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

\relative c'' {
  \override TextSpanner #'(bound-details left text) = #"bla"
  \override TextSpanner #'(bound-details right text) = #"blu"
  a4 \startTextSpan
  b4 c
}
```

```

a4 \stopTextSpan

\override TextSpanner #'style = #'line
\once \override TextSpanner
  #'(bound-details left stencil-align-dir-y) = #CENTER
a4 \startTextSpan
b4 c
a4 \stopTextSpan

\override TextSpanner #'style = #'dashed-line
\override TextSpanner #'(bound-details left text) =
  \markup { \draw-line #'(0 . 1) }
\override TextSpanner #'(bound-details right text) =
  \markup { \draw-line #'(0 . -2) }
\once \override TextSpanner #'(bound-details right padding) = #-2

a4 \startTextSpan
b4 c
a4 \stopTextSpan

\set Staff.middleCPosition = #-13
\override TextSpanner #'dash-period = #10
\override TextSpanner #'dash-fraction = #0.5
\override TextSpanner #'thickness = #10
a4 \startTextSpan
b4 c
a4 \stopTextSpan
}

```



Demonstrating all headers

All header fields with special meanings.

```

\header {
  copyright = "copyright"
  title = "title"
  subtitle = "subtitle"
  composer = "composer"
  arranger = "arranger"
  instrument = "instrument"
  metre = "metre"
  opus = "opus"
  piece = "piece"
  poet = "poet"
  texidoc = "All header fields with special meanings."
  copyright = "public domain"
  enteredby = "jcn"
}

```

```

    source = "urtext"
}

\layout {
  ragged-right = ##f
}

\score {
  \relative c'' { c1 | c | c | c }
}

\score {
  \relative c'' { c1 | c | c | c }
  \header {
    title = "localtitle"
    subtitle = "localsubtitle"
    composer = "localcomposer"
    arranger = "localarranger"
    instrument = "localinstrument"
    metre = "localmetre"
    opus = "localopus"
    piece = "localpiece"
    poet = "localpoet"
    copyright = "localcopyright"
  }
}

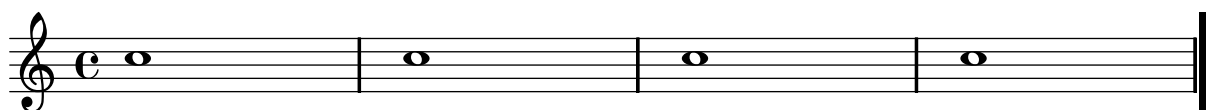
```

	title	
	subtitle	
poet	instrument	composer
		arranger

piece		opus
-------	--	------



localpiece		localopus
------------	--	-----------



Embedding native PostScript in a \markup block

PostScript code can be directly inserted inside a \markup block.

% PostScript is a registered trademark of Adobe Systems Inc.

```
\relative c'' {
  a4-\markup { \postscript #"3 4 moveto 5 3 rlineto stroke" }
  -\markup { \postscript #"[ 0 1 ] 0 setdash 3 5 moveto 5 -3 rlineto stroke " }

  b4-\markup { \postscript #"3 4 moveto 0 0 1 2 8 4 20 3.5 rcurveto stroke" }
  s2
  a'1
}
```



Formatting lyrics syllables

To format individual syllables in lyrics, use \markup { } on these lyrics.

% Tip taken from <http://lists.gnu.org/archive/html/lilypond-user/2007-12/msg00215.html>

```
\header {
  title = "Markup can be used inside lyrics!"
}

mel = \relative c'' { c4 c c c }
lyr = \lyricmode {
  Lyrics \markup { \italic "can" } \markup { \with-color #red "contain" }
  \markup { \fontsize #8 \bold "Markup!" }
}

<<
  \context Voice = melody \mel
  \context Lyrics \lyricsto melody \lyr
>>
```

Markup can be used inside lyrics!



Lyrics *can* **contain** **Markup!**

How to put ties between syllables in lyrics

This can be achieved by separating those syllables by tildes.

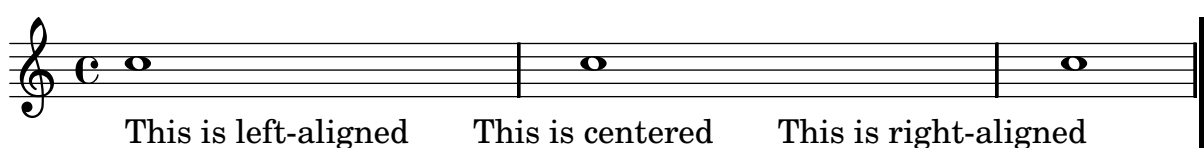
```
\lyrics {
  wa~o~a
}
```

wa~o~a

Lyrics alignment

Horizontal alignment for lyrics can be set by overriding the `self-alignment-X` property of the `LyricText` object. #-1 is left, #0 is center and #1 is right; however, you can use `#LEFT`, `#CENTER` and `#RIGHT` as well.

```
\layout { ragged-right = ##f }
\relative c'' {
  c1
  c1
  c1
}
\addlyrics {
  \once \override LyricText #'self-alignment-X = #LEFT
  "This is left-aligned"
  \once \override LyricText #'self-alignment-X = #CENTER
  "This is centered"
  \once \override LyricText #'self-alignment-X = #1
  "This is right-aligned"
}
```



Markup lines

Text that can spread over pages is entered with the `\markuplines` command.

```

#(set-default-paper-size "a6")

#(define-markup-list-command (paragraph layout props args) (markup-list?)
  (interpret-markup-list layout props
    (make-justified-lines-markup-list (cons (make-hspace-markup 2) args))))

% Candide, Voltaire
\markuplines {
  \override-lines #'(baseline-skip . 2.5) {
    \paragraph {
      Il y avait en Westphalie, dans le château de M. le baron de
      Thunder-ten-tronckh, un jeune garçon à qui la nature avait donné
      les mœurs les plus douces. Sa physionomie annonçait son âme.
    }
  }
}
```

Il avait le jugement assez droit, avec l'esprit le plus simple ; c'est, je crois, pour cette raison qu'on le nommait Candide. Les anciens domestiques de la maison soupçonnaient qu'il était fils de la sœur de monsieur le baron et d'un bon et honnête gentilhomme du voisinage, que cette demoiselle ne voulut jamais épouser parce qu'il n'avait pu prouver que soixante et onze quartiers, et que le reste de son arbre généalogique avait été perdu par l'injure du temps.

}

\paragraph {

Monsieur le baron était un des plus puissants seigneurs de la Westphalie, car son château avait une porte et des fenêtres. Sa grande salle même était ornée d'une tapisserie. Tous les chiens de ses basses-cours composaient une meute dans le besoin ; ses palefreniers étaient ses piqueurs; le vicaire du village était son grand-aumônier. Ils l'appelaient tous monseigneur, et ils riaient quand il faisait des contes.

}

}

}

Il y avait en Westphalie, dans le château de M. le baron de Thunder-ten-tronckh, un jeune garçon à qui la nature avait donné les mœurs les plus douces. Sa physionomie annonçait son âme. Il avait le jugement assez droit, avec l'esprit le plus simple ; c'est, je crois, pour cette raison qu'on le nommait Candide. Les anciens domestiques de la maison soupçonnaient qu'il était fils de la sœur de monsieur le baron et d'un bon et honnête gentilhomme du voisinage, que cette demoiselle ne voulut jamais épouser parce qu'il n'avait pu prouver que soixante et onze quartiers, et que le reste de son arbre généalogique avait été perdu par l'injure du temps.

Monsieur le baron était un des plus puissants seigneurs de la Westphalie, car son château avait une porte et des fenêtres. Sa grande salle même était ornée d'une tapisserie. Tous les chiens de ses basses-cours composaient une meute dans le besoin ; ses palefreniers étaient ses piqueurs; le vicaire du village était son grand-aumônier. Ils l'appelaient tous monseigneur, et ils riaient quand il faisait des contes.

Multi-measure rest markup

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

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

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

The musical notation shows three measures on a single staff. The first measure is a multi-measure rest of 19 measures, with the label '[MAJOR GENERAL]' centered above it and '19' centered below it. The second measure is a multi-measure rest of 30 measures, with the label '[MABEL]' centered above it and '30' centered below it. The third measure contains the notes C4, D4, F4, and C5, with the label 'CHORUS' centered above it. A cue line 'Cue: ... it is yours' is written below the second measure, aligned with the start of the 30-measure rest.

Ottava text

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

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



Outputting the version number

By putting the output of `lilypond-version` into a lyric, it is possible to print the version number of LilyPond in a score, or in a document generated with `lilypond-book`. Another possibility is to append the version number to the doc-string, in this manner:

```
\score {
  \new Lyrics {
    \override Score.RehearsalMark #'self-alignment-X = #LEFT
    \mark #(string-append "Processed with LilyPond version " (lilypond-version))
    s2
  }
}
```

Processed with LilyPond version 2.13.27

Piano template with centered lyrics

Instead of having a full staff for the melody and lyrics, lyrics can be centered between the staves of a piano staff.

```
upper = \relative c'' {
  \clef treble
  \key c \major
  \time 4/4

  a4 b c d
}
```

```
lower = \relative c {
  \clef bass
```

```

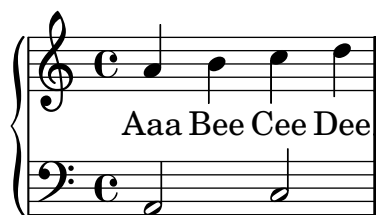
\key c \major
\time 4/4

a2 c
}

text = \lyricmode {
  Aaa Bee Cee Dee
}

\score {
  \new GrandStaff <<
    \new Staff = upper { \new Voice = "singer" \upper }
    \new Lyrics \lyricsto "singer" \text
    \new Staff = lower { \lower }
  >>
  \layout {
    \context {
      \GrandStaff
      \accepts "Lyrics"
    }
    \context {
      \Lyrics
      \consists "Bar_engraver"
    }
  }
  \midi { }
}

```



Printing marks at the end of a line

Marks can be printed at the end of the current line, instead of the beginning of the following line. In such cases, it might be preferable to align the right end of the mark with the bar line.

```

\relative c' ' {
  g2 c
  d,2 a'
  \once \override Score.RehearsalMark #'break-visibility = #end-of-line-visible
  \once \override Score.RehearsalMark #'self-alignment-X = #RIGHT
  \mark "D.C. al Fine"
  \break
  g2 b,
  c1 \bar "||"
}

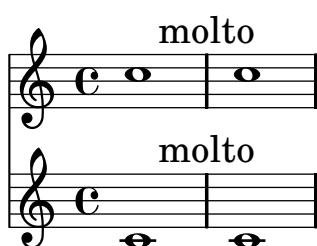
```



Printing marks on every staff

Although text marks are normally only printed above the topmost staff, they may also be printed on every staff.

```
\score {
  <<
    \new Staff { c''1 \mark "molto" c'' }
    \new Staff { c'1 \mark "molto" c' }
  >>
  \layout {
    \context {
      \Score
      \remove "Mark_engraver"
      \remove "Staff_collecting_engraver"
    }
    \context {
      \Staff
      \consists "Mark_engraver"
      \consists "Staff_collecting_engraver"
    }
  }
}
```



Printing text from right to left

It is possible to print text from right to left in a markup object, as demonstrated here.

```
{
  b1^\markup {
    \line { i n g i r u m i m u s n o c t e }
  }
  f'_\markup {
    \override #'(text-direction . -1)
    \line { i n g i r u m i m u s n o c t e }
  }
}
```



Stand-alone two-column markup

Stand-alone text may be arranged in several columns using `\markup` commands:

```
\markup {
  \fill-line {
    \hspace #1
    \column {
      \line { 0 sacrum convivium }
      \line { in quo Christus sumitur, }
      \line { recolitur memoria passionis ejus, }
      \line { mens impletur gratia, }
      \line { futurae gloriae nobis pignus datur. }
      \line { Amen. }
    }
    \hspace #2
    \column {
      \line { \italic { 0 sacred feast } }
      \line { \italic { in which Christ is received, } }
      \line { \italic { the memory of His Passion is renewed, } }
      \line { \italic { the mind is filled with grace, } }
      \line { \italic { and a pledge of future glory is given to us. } }
      \line { \italic { Amen. } }
    }
  }
  \hspace #1
}
```

O sacrum convivium
in quo Christus sumitur,
recolitur memoria passionis ejus,
mens impletur gratia,
futurae gloriae nobis pignus datur.
Amen.

*O sacred feast
in which Christ is received,
the memory of His Passion is renewed,
the mind is filled with grace,
and a pledge of future glory is given to us.
Amen.*

String number extender lines

Make an extender line for string number indications, showing that a series of notes is supposed to be played all on the same string.

```
stringNumberSpanner =
#(define-music-function (parser location StringNumber) (string?)
  #{
    \override TextSpanner #'style = #'solid
    \override TextSpanner #'font-size = #-5
    \override TextSpanner #'(bound-details left stencil-align-dir-y) = #CENTER
    \override TextSpanner #'(bound-details left text) = \markup { \circle \number $StringNumber }
  })

\relative c {
```

```

\clef "treble_8"
\stringNumberSpanner "5"
\textSpannerDown
a8\startTextSpan
b c d e f\stopTextSpan
\stringNumberSpanner "4"
g\startTextSpan a
bes4 a g2\stopTextSpan
}

```



Three-sided box

This example shows how to add a markup command to get a three sided box around some text (or other markup).

```

% New command to add a three sided box, with sides north, west and south
% Based on the box-stencil command defined in scm/stencil.scm
% Note that ";;" is used to comment a line in Scheme
#(define-public (NWS-box-stencil stencil thickness padding)
  "Add a box around STENCIL, producing a new stencil."
  (let* ((x-ext (interval-widen (ly:stencil-extent stencil X) padding))
        (y-ext (interval-widen (ly:stencil-extent stencil Y) padding))
        (y-rule (make-filled-box-stencil (cons 0 thickness) y-ext))
        (x-rule (make-filled-box-stencil
                  (interval-widen x-ext thickness) (cons 0 thickness))))
    ;; (set! stencil (ly:stencil-combine-at-edge stencil X 1 y-rule padding))
    (set! stencil (ly:stencil-combine-at-edge stencil X LEFT y-rule padding))
    (set! stencil (ly:stencil-combine-at-edge stencil Y UP x-rule 0.0))
    (set! stencil (ly:stencil-combine-at-edge stencil Y DOWN x-rule 0.0))
    stencil))

% The corresponding markup command, based on the \box command defined
% in scm/define-markup-commands.scm
#(define-markup-command (NWS-box layout props arg) (markup?)
  "Draw a box round @var{arg}. Looks at @code{thickness},
@code{box-padding} and @code{font-size} properties to determine line
thickness and padding around the markup."
  (let* ((th (chain-assoc-get 'thickness props 0.1))
        (size (chain-assoc-get 'font-size props 0))
        (pad (* (magstep size)
                 (chain-assoc-get 'box-padding props 0.2)))
        (m (interpret-markup layout props arg)))
    (NWS-box-stencil m th pad)))

% Test it:

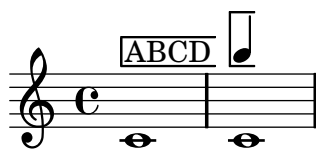
\relative c' {

```

```

c1^\markup { \NWS-box ABCD }
c1^\markup { \NWS-box \note #"4" #1.0 }
}

```



UTF-8

Various scripts may be used for texts (like titles and lyrics) by entering them in UTF-8 encoding, and using a Pango based backend. Depending on the fonts installed, this fragment will render Bulgarian (Cyrillic), Hebrew, Japanese and Portuguese.

```

% end verbatim - this comment is a hack to prevent texinfo.tex
% from choking on non-European UTF-8 subsets

```

```

%% Edit this file using a Unicode aware editor, such as GVIM, GEDIT, Emacs

```

```

%{

```

You may have to install additional fonts.

Red Hat Fedora

```

    taipeifonts fonts-xorg-truetype ttfonts-ja fonts-arabic \
    ttfonts-zh_CN fonts-ja fonts-hebrew

```

Debian GNU/Linux

```

    apt-get install emacs-intl-fonts xfonts-intl-.* \
    ttf-kochi-gothic ttf-kochi-mincho \
    xfonts-bolkhov-75dpi xfonts-cronyx-100dpi xfonts-cronyx-75dpi
}%

```

```

% Cyrillic font

```

```

bulgarian = \lyricmode {
    , , , .
}

```

```

hebrew = \lyricmode {
    .
}

```

```

japanese = \lyricmode {

```

```

}

```

```
% "a legal song to you"
portuguese = \lyricmode {
  à vo -- cê uma can -- ção legal
}
```

```
\relative c' {
  c2 d
  e2 f
  g2 f
  e1
}
\addlyrics { \bulgarian }
\addlyrics { \hebrew }
\addlyrics { \japanese }
\addlyrics { \portuguese }
```

זֶה כִּי סתם לשמוע '

à vo - - cê uma

אֵיךְ תנצח ' קרפד

can - - ção legal

Vocal ensemble template with lyrics aligned below and above the staves

This template is basically the same as the simple “Vocal ensemble” template, with the exception that here all the lyrics lines are placed using `alignAboveContext` and `alignBelowContext`.

```
global = {
  \key c \major
  \time 4/4
}

sopMusic = \relative c'' {
  c4 c c8[( b)] c4
}
sopWords = \lyricmode {
  hi hi hi hi
}

altoMusic = \relative c' {
  e4 f d e
}
```

```

altoWords = \lyricmode {
  ha ha ha ha
}

tenorMusic = \relative c' {
  g4 a f g
}
tenorWords = \lyricmode {
  hu hu hu hu
}

bassMusic = \relative c {
  c4 c g c
}
bassWords = \lyricmode {
  ho ho ho ho
}

\score {
  \new ChoirStaff <<
    \new Staff = women <<
      \new Voice = "sopranos" { \voiceOne << \global \sopMusic >> }
      \new Voice = "altos" { \voiceTwo << \global \altoMusic >> }
    >>
    \new Lyrics \with { alignAboveContext = women } \lyricsto sopranos \sopWords
    \new Lyrics \with { alignBelowContext = women } \lyricsto altos \altoWords
    % we could remove the line about this with the line below, since we want
    % the alto lyrics to be below the alto Voice anyway.
    % \new Lyrics \lyricsto altos \altoWords

    \new Staff = men <<
      \clef bass
      \new Voice = "tenors" { \voiceOne << \global \tenorMusic >> }
      \new Voice = "basses" { \voiceTwo << \global \bassMusic >> }
    >>
    \new Lyrics \with { alignAboveContext = men } \lyricsto tenors \tenorWords
    \new Lyrics \with { alignBelowContext = men } \lyricsto basses \bassWords
    % again, we could replace the line above this with the line below.
    % \new Lyrics \lyricsto basses \bassWords
  >>
  \layout {
    \context {
      % a little smaller so lyrics
      % can be closer to the staff
      \Staff
      \override VerticalAxisGroup #'minimum-Y-extent = #'(-3 . 3)
    }
  }
}

```



Volta text markup using repeatCommands

Though volta are best specified using `\repeat volta`, the context property `repeatCommands` must be used in cases where the volta text needs more advanced formatting with `\markup`.

Since `repeatCommands` takes a list, the simplest method of including markup is to use an identifier for the text and embed it in the command list using the Scheme syntax `#(list (list 'volta textIdentifier))`. Start- and end-repeat commands can be added as separate list elements:

```
voltaAdLib = \markup { 1. 2. 3... \text \italic { ad lib. } }
```

```
\relative c'' {
  c1
  \set Score.repeatCommands = #(list (list 'volta voltaAdLib) 'start-repeat)
  c4 b d e
  \set Score.repeatCommands = #'((volta #f) (volta "4.") end-repeat)
  f1
  \set Score.repeatCommands = #'((volta #f))
}
```



Vocal music

Section “Vocal music” in *Notation Reference*

Adding ambitus per voice

Ambitus can be added per voice. In this case, the ambitus must be moved manually to prevent collisions.

```
\new Staff <<
  \new Voice \with {
    \consists "Ambitus_engraver"
  } \relative c'' {
    \override Ambitus #'X-offset = #2.0
    \voiceOne
    c4 a d e
    f1
  }
  \new Voice \with {
    \consists "Ambitus_engraver"
  } \relative c' {
    \voiceTwo
    es4 f g as
    b1
  }
}>>
```



Adding indicators to staves which get split after a break

This snippet defines the `\splitStaffBarLine` command, which adds arrows in north-east and south-east directions at a bar line, to denote that several voices sharing a staff will each continue on a staff of their own in the next system.

```
#(define-markup-command (arrow-at-angle layout props angle-deg length fill)
  (number? number? boolean?)
  (let* (
    ;; PI-OVER-180 and degrees->radians are taken from flag-styles.scm
    (PI-OVER-180 (/ (atan 1 1) 45))
    (degrees->radians (lambda (degrees) (* degrees PI-OVER-180)))
    (angle-rad (degrees->radians angle-deg))
    (target-x (* length (cos angle-rad)))
    (target-y (* length (sin angle-rad))))
    (interpret-markup layout props
      (markup
        #:translate (cons (/ target-x 2) (/ target-y 2))
        #:rotate angle-deg
        #:translate (cons (/ length -2) 0)
        #:concat (cons length 0)
        #:arrow-head X RIGHT fill))))))
```

```

splitStaffBarLineMarkup = \markup \with-dimensions #'(0 . 0) #'(0 . 0) {
  \combine
    \arrow-at-angle #45 #(sqrt 8) ##f
    \arrow-at-angle #-45 #(sqrt 8) ##f
}

splitStaffBarLine = {
  \once \override Staff.BarLine #'stencil =
    #(lambda (grob)
      (ly:stencil-combine-at-edge
        (ly:bar-line::print grob)
        X RIGHT
        (grob-interpret-markup grob splitStaffBarLineMarkup)
        0 0))
  \break
}

\paper {
  ragged-right = ##t
  short-indent = 5\mm
}

\score {
  <<
    \new ChoirStaff <<
      \new Staff \with { instrumentName = #"High I + II" } {
        <<
          \repeat unfold 4 f''1
          \\
          \repeat unfold 4 d''1
        >>
        \splitStaffBarLine
      }
      \new Staff \with { instrumentName = #"Low" } {
        <<
          \repeat unfold 4 b'1
          \\
          \repeat unfold 4 g'1
        >>
      }

      \new Staff \with { shortInstrumentName = #"H I" } {
        R1*4
        \repeat unfold 2 { r4 f''2 r4 } \repeat unfold 2 e''1
      }
      \new Staff \with { shortInstrumentName = #"H II" } {
        R1*4
        \repeat unfold 4 b'2 \repeat unfold 2 c''1
      }
      \new Staff \with { shortInstrumentName = #"L" } {
        R1*4

```

```

    <<
      \repeat unfold 4 g'1
      \\
      \repeat unfold 4 c'1
    >>
  }
  >>
>>
\layout {
  \context {
    \Staff \RemoveEmptyStaves
    \override VerticalAxisGroup #'remove-first = ##t
  }
}
}

```

High I + II

Low

H I

H II

L

Adjusting lyrics vertical spacing

This snippet shows how to bring the lyrics line closer to the staff.

% Default layout:

```

<<
  \new Staff \new Voice = melody \relative c' {
    c4 d e f
    g4 f e d
    c1
  }
  \new Lyrics \lyricsto melody { aa aa aa aa aa aa aa aa aa }

  % Reducing the minimum space below the staff and above the lyrics:
  \new Staff {
    \new Voice = melody \relative c' {
      c4 d e f
    }
  }

```

```

      g4 f e d
      c1
    }
  }
  \new Lyrics \with {
    \override VerticalAxisGroup #'inter-staff-spacing = #'((space . 1))
  }
  \lyricsto melody { aa aa aa aa aa aa aa aa aa }
>>

```



Ambitus with multiple voices

Adding the `Ambitus_engraver` to the `Staff` context creates a single ambitus per staff, even in the case of staves with multiple voices.

```

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

```



Ambitus

Ambitus indicate pitch ranges for voices.

Accidentals only show up if they are not part of the key signature. `AmbitusNoteHead` grobs also have ledger lines.

```

\layout {
  \context {
    \Voice

```

```

    \consists "Ambitus_engraver"
  }
}

<<
  \new Staff {
    \relative c' {
      \time 2/4
      c4 f'
    }
  }
  \new Staff {
    \relative c' {
      \time 2/4
      \key d \major
      cis4 as'
    }
  }
}
>>

```



Changing stanza fonts

Fonts can be changed independently for each stanza, including the font used for printing the stanza number.

```

\new Voice {
  \time 3/4
  g2 e4
  a2 f4
  g2.
}
\addlyrics {
  \set stanza = #"1. "
  Hi, my name is Bert.
}
\addlyrics {
  \override StanzaNumber #'font-name = #"DejaVu"
  \set stanza = #"2. "
  \override LyricText #'font-family = #'typewriter
  Oh, ché -- ri, je t'aime
}

```



Chant or psalms notation

This form of notation is used for the chant of the Psalms, where verses aren't always the same length.

```
stemOn = { \revert Staff.Stem #'transparent }
stemOff = { \override Staff.Stem #'transparent = ##t }

\score {
  \new Staff \with { \remove "Time_signature_engraver" }
  {
    \key g \minor
    \cadenzaOn
    \stemOff a'\breve bes'4 g'4
    \stemOn a'2 \bar "||"
    \stemOff a'\breve g'4 a'4
    \stemOn f'2 \bar "||"
    \stemOff a'\breve^{\markup { \italic flexe }}
    \stemOn g'2 \bar "||"
  }
}
```



Forcing hyphens to be shown

If LilyPond does not think there is space for a hyphen, it will be omitted. The behaviour can be overridden with the `minimum-distance` property of `LyricHyphen`.

```
\relative c'' {
  c32 c c c
  c32 c c c
  c32 c c c
  c32 c c c
}

\addlyrics {
  syl -- lab word word
  \override LyricHyphen #'minimum-distance = #1.0
  syl -- lab word word
  \override LyricHyphen #'minimum-distance = #2.0
  syl -- lab word word
  \revert LyricHyphen #'minimum-distance
  syl -- lab word word
}
```



Formatting lyrics syllables

To format individual syllables in lyrics, use `\markup { }` on these lyrics.

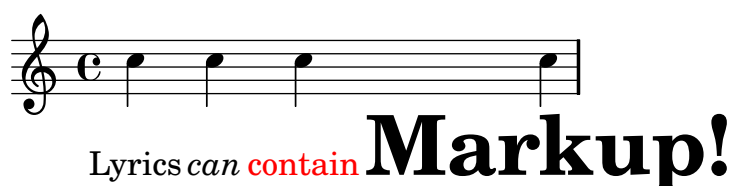
% Tip taken from <http://lists.gnu.org/archive/html/lilypond-user/2007-12/msg00215.html>

```
\header {
  title = "Markup can be used inside lyrics!"
}

mel = \relative c'' { c4 c c c }
lyr = \lyricmode {
  Lyrics \markup { \italic "can" } \markup {\with-color #red "contain" }
  \markup {\fontsize #8 \bold "Markup!" }
}

<<
  \context Voice = melody \mel
  \context Lyrics \lyricsto melody \lyr
>>
```

Markup can be used inside lyrics!



How to put ties between syllables in lyrics

This can be achieved by separating those syllables by tildes.

```
\lyrics {
  wa~o~a
}
```

wa~o~a

Lyrics alignment

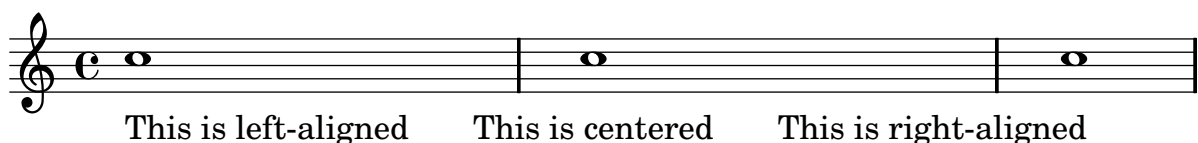
Horizontal alignment for lyrics can be set by overriding the `self-alignment-X` property of the `LyricText` object. #-1 is left, #0 is center and #1 is right; however, you can use `#LEFT`, `#CENTER` and `#RIGHT` as well.

```
\layout { ragged-right = ##f }
\relative c'' {
  c1
  c1
  c1
}
```

```

}
\addlyrics {
  \once \override LyricText #'self-alignment-X = #LEFT
  "This is left-aligned"
  \once \override LyricText #'self-alignment-X = #CENTER
  "This is centered"
  \once \override LyricText #'self-alignment-X = #1
  "This is right-aligned"
}

```



Marking notes of spoken parts with a cross on the stem

This example shows how to put crosses on stems. Mark the beginning of a spoken section with the `\speakOn` keyword, and end it with the `\speakOff` keyword.

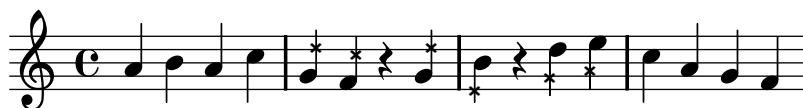
```

\speakOn = {
  \override Stem #'stencil = #(lambda (grob)
    (let* ((x-parent (ly:grob-parent grob X))
      (is-rest? (ly:grob? (ly:grob-object x-parent 'rest))))
    (if is-rest?
      empty-stencil
      (ly:stencil-combine-at-edge
        (ly:stem::print grob)
        Y
        (- (ly:grob-property grob 'direction))
        (grob-interpret-markup grob
          (markup #:hspace -1.025 #:fontsize -4
            #:musicglyph "noteheads.s2cross"))
        -2.3 0))))
}

\speakOff = {
  \revert Stem #'stencil
}

\score {
  \new Staff {
    \relative c'' {
      a4 b a c
      \speakOn
      g4 f r g
      b4 r d e
      \speakOff
      c4 a g f
    }
  }
}

```



Piano template with melody and lyrics

Here is a typical song format: one staff with the melody and lyrics, with piano accompaniment underneath.

```
melody = \relative c'' {
  \clef treble
  \key c \major
  \time 4/4

  a b c d
}

text = \lyricmode {
  Aaa Bee Cee Dee
}

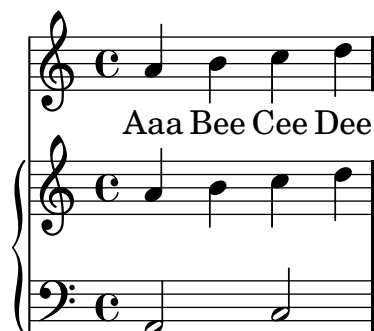
upper = \relative c'' {
  \clef treble
  \key c \major
  \time 4/4

  a4 b c d
}

lower = \relative c {
  \clef bass
  \key c \major
  \time 4/4

  a2 c
}

\score {
  <<
    \new Voice = "mel" { \autoBeamOff \melody }
    \new Lyrics \lyricsto mel \text
    \new PianoStaff <<
      \new Staff = "upper" \upper
      \new Staff = "lower" \lower
    >>
  >>
  \layout {
    \context { \Staff \RemoveEmptyStaves }
  }
  \midi { }
}
```



Single staff template with notes lyrics and chords

This template allows the preparation of a song with melody, words, and chords.

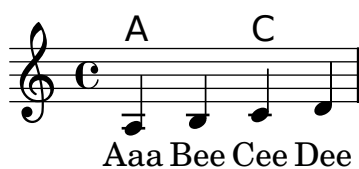
```
melody = \relative c' {
  \clef treble
  \key c \major
  \time 4/4

  a4 b c d
}

text = \lyricmode {
  Aaa Bee Cee Dee
}

harmonies = \chordmode {
  a2 c
}

\score {
  <<
    \new ChordNames {
      \set chordChanges = ##t
      \harmonies
    }
    \new Voice = "one" { \autoBeamOff \melody }
    \new Lyrics \lyricsto "one" \text
  >>
  \layout { }
  \midi { }
}
```



Single staff template with notes lyrics chords and frets

Here is a simple lead sheet template with melody, lyrics, chords and fret diagrams.

```

verseI = \lyricmode {
  \set stanza = #"1."
  This is the first verse
}

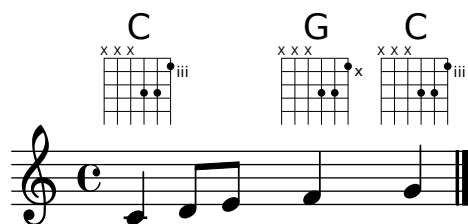
verseII = \lyricmode {
  \set stanza = #"2."
  This is the second verse.
}

theChords = \chordmode {
  % insert chords for chordnames and fretboards here
  c2 g4 c
}

staffMelody = \relative c' {
  \key c \major
  \clef treble
  % Type notes for melody here
  c4 d8 e f4 g
  \bar "|"
}

\score {
  <<
    \context ChordNames { \theChords }
    \context FretBoards { \theChords }
    \new Staff {
      \context Voice = "voiceMelody" { \staffMelody }
    }
    \new Lyrics = "lyricsI" {
      \lyricsto "voiceMelody" \verseI
    }
    \new Lyrics = "lyricsII" {
      \lyricsto "voiceMelody" \verseII
    }
  >>
  \layout { }
  \midi { }
}

```



1. This is the first verse
2. This is the second verse.

Single staff template with notes and lyrics

This small template demonstrates a simple melody with lyrics. Cut and paste, add notes, then words for the lyrics. This example turns off automatic beaming, which is common for vocal parts. To use automatic beaming, change or comment out the relevant line.

```
melody = \relative c' {
  \clef treble
  \key c \major
  \time 4/4

  a4 b c d
}

text = \lyricmode {
  Aaa Bee Cee Dee
}

\score{
  <<
    \new Voice = "one" {
      \autoBeamOff
      \melody
    }
    \new Lyrics \lyricsto "one" \text
  >>
  \layout { }
  \midi { }
}
```



Skips in lyric mode (2)

Although `s` skips cannot be used in `\lyricmode` (it is taken to be a literal “s”, not a space), double quotes (“”) or underscores (_) are available. So for example:

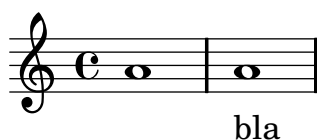
```
<<
  \relative c'' { a4 b c d }
  \new Lyrics \lyricmode { a4 "" _ gap }
>>
```



Skips in lyric mode

The `s` syntax for skips is only available in note mode and chord mode. In other situations, for example, when entering lyrics, using the `\skip` command is recommended.

```
<<
\relative c' { a1 | a }
\new Lyrics \lyricmode { \skip 1 bla1 }
>>
```



Vertically aligning ossia and lyrics

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

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

\relative c' <<
\new Staff = "1" { c4 c s2 }
\new Staff = "2" { c4 c s2 }
\new Staff = "3" { c4 c s2 }
{ \skip 2
  <<
    \lyrics {
      \set alignBelowContext = #"1"
      lyrics4 below
    }
    \new Staff \with {
      alignAboveContext = #"3"
      fontSize = #-2
      \override StaffSymbol #'staff-space = #(magstep -2)
      \remove "Time_signature_engraver"
    } {
      \times 4/6 {
        \override TextScript #'padding = #3
        c8[~"ossia above" d e d e f]
      }
    }
  }
  >>
}
>>
```



Vertically centered common lyrics

In a vocal piece where there are several (two, four or more) lines of lyrics, and common lyrics for all voices at some point, these common lyrics may be vertically centered regardingly, as shown in the following example:

```
\include "english.ly"
leftbrace = \markup { \override #'(font-encoding . fetaBraces) \lookup #"brace240" }
rightbrace = \markup { \rotate #180 \leftbrace }

dropLyrics =
{
  \override LyricText #'extra-offset = #'(0 . -5)
  \override LyricHyphen #'extra-offset = #'(0 . -5)
  \override LyricExtender #'extra-offset = #'(0 . -5)
}

raiseLyrics =
{
  \revert LyricText #'extra-offset
  \revert LyricHyphen #'extra-offset
  \revert LyricExtender #'extra-offset
}

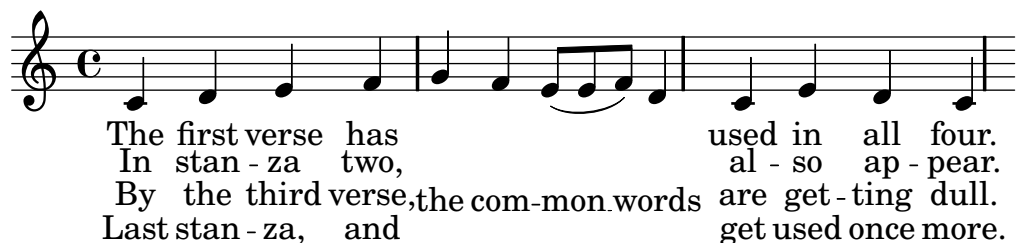
skipFour = \repeat unfold 4 { \skip 8 }

lyricsA = \lyricmode { The first verse has \dropLyrics the com -- mon
  __ words \raiseLyrics used in all four. }
lyricsB = \lyricmode { In stan -- za two, \skipFour al -- so ap -- pear. }
lyricsC = \lyricmode { By the third verse, \skipFour are get -- ting dull. }
lyricsD = \lyricmode { Last stan -- za, and \skipFour get used once more. }

melody = \relative c' { c4 d e f g f e8( e f) d4 c e d c }

\score
{
  <<
    \new Voice = m \melody
    \new Lyrics \lyricsto m \lyricsA
    \new Lyrics \lyricsto m \lyricsB
    \new Lyrics \lyricsto m \lyricsC
    \new Lyrics \lyricsto m \lyricsD
  >>
}
```

}



Vocal ensemble template with automatic piano reduction

This template adds an automatic piano reduction to the standard SATB vocal score demonstrated in “Vocal ensemble template”. This demonstrates one of the strengths of LilyPond – you can use a music definition more than once. If any changes are made to the vocal notes (say, `tenorMusic`), then the changes will also apply to the piano reduction.

```
global = {
  \key c \major
  \time 4/4
}

sopMusic = \relative c' {
  c4 c c8[( b)] c4
}
sopWords = \lyricmode {
  hi hi hi hi
}

altoMusic = \relative c' {
  e4 f d e
}
altoWords = \lyricmode {
  ha ha ha ha
}

tenorMusic = \relative c' {
  g4 a f g
}
tenorWords = \lyricmode {
  hu hu hu hu
}

bassMusic = \relative c {
  c4 c g c
}
bassWords = \lyricmode {
  ho ho ho ho
}

\score {
  <<
```

```

\new ChoirStaff <<
  \new Lyrics = sopranos { s1 }
  \new Staff = women <<
    \new Voice = sopranos { \voiceOne << \global \sopMusic >> }
    \new Voice = altos { \voiceTwo << \global \altoMusic >> }
  >>
  \new Lyrics = altos { s1 }
  \new Lyrics = tenors { s1 }
  \new Staff = men <<
    \clef bass
    \new Voice = tenors { \voiceOne <<\global \tenorMusic >> }
    \new Voice = basses { \voiceTwo <<\global \bassMusic >> }
  >>
  \new Lyrics = basses { s1 }
  \context Lyrics = sopranos \lyricsto sopranos \sopWords
  \context Lyrics = altos \lyricsto altos \altoWords
  \context Lyrics = tenors \lyricsto tenors \tenorWords
  \context Lyrics = basses \lyricsto basses \bassWords
>>
\new PianoStaff <<
  \new Staff <<
    \set Staff.printPartCombineTexts = ##f
    \partcombine
    << \global \sopMusic >>
    << \global \altoMusic >>
  >>
  \new Staff <<
    \clef bass
    \set Staff.printPartCombineTexts = ##f
    \partcombine
    << \global \tenorMusic >>
    << \global \bassMusic >>
  >>
>>
>>
\layout {
  \context {
    % a little smaller so lyrics
    % can be closer to the staff
    \Staff
    \override VerticalAxisGroup #'minimum-Y-extent = #'(-3 . 3)
  }
}
}

```

The image shows a musical score for a vocal ensemble in 4/4 time. It consists of three systems of staves. The first system has a soprano staff with lyrics 'hi hi hi hi' above it and an alto staff with lyrics 'ha ha ha ha' above and 'hu hu hu hu' below it. The second system has a tenor staff with lyrics 'ho ho ho ho' above it. The third system has a bass staff with no lyrics. The music is written in C major and 4/4 time.

Vocal ensemble template with lyrics aligned below and above the staves

This template is basically the same as the simple “Vocal ensemble” template, with the exception that here all the lyrics lines are placed using `alignAboveContext` and `alignBelowContext`.

```
global = {
  \key c \major
  \time 4/4
}

sopMusic = \relative c'' {
  c4 c c8[( b)] c4
}
sopWords = \lyricmode {
  hi hi hi hi
}

altoMusic = \relative c' {
  e4 f d e
}
altoWords = \lyricmode {
  ha ha ha ha
}

tenorMusic = \relative c' {
  g4 a f g
}
tenorWords = \lyricmode {
  hu hu hu hu
}

bassMusic = \relative c {
  c4 c g c
}
bassWords = \lyricmode {
  ho ho ho ho
}
```

```

}

\score {
  \new ChoirStaff <<
    \new Staff = women <<
      \new Voice = "sopranos" { \voiceOne << \global \sopMusic >> }
      \new Voice = "altos" { \voiceTwo << \global \altoMusic >> }
    >>
    \new Lyrics \with { alignAboveContext = women } \lyricsto sopranos \sopWords
    \new Lyrics \with { alignBelowContext = women } \lyricsto altos \altoWords
    % we could remove the line about this with the line below, since we want
    % the alto lyrics to be below the alto Voice anyway.
    % \new Lyrics \lyricsto altos \altoWords

    \new Staff = men <<
      \clef bass
      \new Voice = "tenors" { \voiceOne << \global \tenorMusic >> }
      \new Voice = "basses" { \voiceTwo << \global \bassMusic >> }
    >>
    \new Lyrics \with { alignAboveContext = men } \lyricsto tenors \tenorWords
    \new Lyrics \with { alignBelowContext = men } \lyricsto basses \bassWords
    % again, we could replace the line above this with the line below.
    % \new Lyrics \lyricsto basses \bassWords
  >>
  \layout {
    \context {
      % a little smaller so lyrics
      % can be closer to the staff
      \Staff
      \override VerticalAxisGroup #'minimum-Y-extent = #'(-3 . 3)
    }
  }
}

```



Vocal ensemble template with verse and refrain

This template creates a score which starts with a solo verse and continues into a refrain for two voices. It also demonstrates the use of spacer rests within the `\global` variable to define meter changes (and other elements common to all parts) throughout the entire score.

```

global = {
  \key g \major

  % verse
  \time 3/4
  s2.*2
  \break

  % refrain
  \time 2/4
  s2*2
  \bar "|."
}

SoloNotes = \relative g' {
  \clef "treble"

  % verse
  g4 g g |
  b4 b b |

  % refrain
  R2*2 |
}

SoloLyrics = \lyricmode {
  One two three |
  four five six |
}

SopranoNotes = \relative c'' {
  \clef "treble"

  % verse
  R2.*2 |

  % refrain
  c4 c |
  g4 g |
}

SopranoLyrics = \lyricmode {
  la la |
  la la |
}

BassNotes = \relative c {
  \clef "bass"

  % verse
  R2.*2 |
}

```

```

% refrain
c4 e |
d4 d |
}

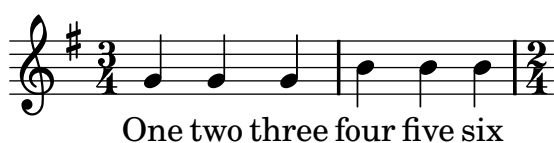
BassLyrics = \lyricmode {
  dum dum |
  dum dum |
}

\score {
  <<
    \new Voice = "SoloVoice" << \global \SoloNotes >>
    \new Lyrics \lyricsto "SoloVoice" \SoloLyrics

    \new ChoirStaff <<
      \new Voice = "SopranoVoice" << \global \SopranoNotes >>
      \new Lyrics \lyricsto "SopranoVoice" \SopranoLyrics

      \new Voice = "BassVoice" << \global \BassNotes >>
      \new Lyrics \lyricsto "BassVoice" \BassLyrics
    >>
  >>
  \layout {
    ragged-right = ##t
    \context { \Staff
      % these lines prevent empty staves from being printed
      \RemoveEmptyStaves
      \override VerticalAxisGroup #'remove-first = ##t
    }
  }
}

```



Vocal ensemble template

Here is a standard four-part SATB vocal score. With larger ensembles, it is often useful to include a section which is included in all parts. For example, the time signature and key signature are

almost always the same for all parts. Like in the “Hymn” template, the four voices are regrouped on only two staves.

```

global = {
  \key c \major
  \time 4/4
}

sopMusic = \relative c'' {
  c4 c c8[( b)] c4
}
sopWords = \lyricmode {
  hi hi hi hi
}

altoMusic = \relative c' {
  e4 f d e
}
altoWords = \lyricmode {
  ha ha ha ha
}

tenorMusic = \relative c' {
  g4 a f g
}
tenorWords = \lyricmode {
  hu hu hu hu
}

bassMusic = \relative c {
  c4 c g c
}
bassWords = \lyricmode {
  ho ho ho ho
}

\score {
  \new ChoirStaff <<
    \new Lyrics = sopranos { s1 }
    \new Staff = women <<
      \new Voice = "sopranos" {
        \voiceOne
        << \global \sopMusic >>
      }
      \new Voice = "altos" {
        \voiceTwo
        << \global \altoMusic >>
      }
    >>
    \new Lyrics = "altos" { s1 }
    \new Lyrics = "tenors" { s1 }
    \new Staff = men <<

```

```

\clef bass
\new Voice = "tenors" {
  \voiceOne
  << \global \tenorMusic >>
}
\new Voice = "basses" {
  \voiceTwo << \global \bassMusic >>
}
>>
\new Lyrics = basses { s1 }
\context Lyrics = sopranos \lyricsto sopranos \sopWords
\context Lyrics = altos \lyricsto altos \altoWords
\context Lyrics = tenors \lyricsto tenors \tenorWords
\context Lyrics = basses \lyricsto basses \bassWords
>>
\layout {
  \context {
    % a little smaller so lyrics
    % can be closer to the staff
    \Staff
    \override VerticalAxisGroup #'minimum-Y-extent = #'(-3 . 3)
  }
}
}

```



Chords

Section “Chord notation” in *Notation Reference*

Adding a figured bass above or below the notes

When writing a figured bass, here’s a way to specify if you want your figures to be placed above or below the bass notes, by defining the `BassFigureAlignmentPositioning` `#'direction` property (exclusively in a `Staff` context). Choices are `#UP` (or `#1`), `#CENTER` (or `#0`) and `#DOWN` (or `#-1`).

As you can see here, this property can be changed as many times as you wish. Use `\once` `\override` if you don’t want the tweak to apply to the whole score.

```
bass = {
  \clef bass
  g4 b, c d
  e d8 c d2
}
continuo = \figuremode {
  <_>4 <6>8
  \once \override Staff.BassFigureAlignmentPositioning #'direction = #CENTER
  <5/>8 <_>4
  \override Staff.BassFigureAlignmentPositioning #'direction = #UP
  <_+>4 <6>
  \set Staff.useBassFigureExtenders = ##t
  \override Staff.BassFigureAlignmentPositioning #'direction = #DOWN
  <4>4. <4>8 <_+>4
}
\score {
  <<
    \new Staff = bassStaff \bass
    \context Staff = bassStaff \continuo
  >>
}
```



Adding bar lines to ChordNames context

To add bar line indications in the `ChordNames` context, add the `Bar_engraver`.

```
\new ChordNames \with {
  \override BarLine #'bar-size = #4
  \consists "Bar_engraver"
}
\chordmode {
  f1:maj7 f:7 bes:7
}
```

$F^{\triangle} \mid F^7 \mid B\flat^7 \mid$

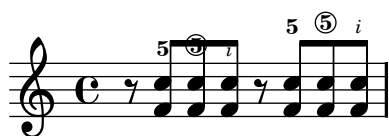
Avoiding collisions with chord fingerings

Fingerings and string numbers applied to individual notes will automatically avoid beams and stems, but this is not true by default for fingerings and string numbers applied to the individual notes of chords. The following example shows how this default behavior can be overridden.

```
\relative c' {
  \set fingeringOrientations = #'(up)
  \set stringNumberOrientations = #'(up)
  \set strokeFingerOrientations = #'(up)

  % Default behavior
  r8
  <f c'-5>8
  <f c'\5>8
  <f c'-\rightHandFinger #2 >8

  % Corrected to avoid collisions
  r8
  \override Fingering #'add-stem-support = ##t
  <f c'-5>8
  \override StringNumber #'add-stem-support = ##t
  <f c'\5>8
  \override StrokeFinger #'add-stem-support = ##t
  <f c'-\rightHandFinger #2 >8
}
```



Bar chords notation for Guitar (with Text Spanner)

Here is how to print bar chords, or half-bar chords (just uncomment the appropriate line for to select either one).

```
The syntax is \bbarre #'fret_number' { notes }
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% %%%%%%%%% Cut here ----- Start 'bbarred.ly'

%% PostScript -----
pScript= \markup {
  \with-dimensions #'(0 . 0.8) #'(0 . 2.0)
  \postscript      #"
  0.15 setlinewidth
  /Times-Roman findfont
  2.0 scalefont
```

```

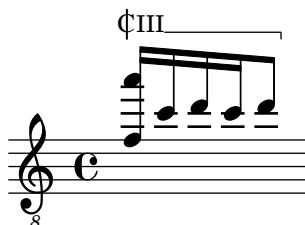
    setfont
    (C)show %%change with B if you prefer
    %(B)show %%change with C if you prefer
    stroke
    0.7 -0.5 moveto
    0.7 1.7 lineto
    stroke"
}
%% Span -----
%% Syntax: \bbarre #"text" { notes } - text = any number of box
bbarre= #(define-music-function (barre location str music) (string? ly:music?)
  (let ((spanned-music
        (let ((first-element #f)
              (last-element #f)
              (first-found? #f))
          (music-map (lambda (m)
                      (if (eqv? (ly:music-property m 'name) 'EventChord)
                          (begin
                           (if (not first-found?)
                               (begin
                                (set! first-found? #t)
                                (set! first-element m)))
                           (set! last-element m)))
                      m)
                    music)
          (if first-found?
              (begin
               (set! (ly:music-property first-element 'elements)
                     (cons (make-music 'TextSpanEvent 'span-direction -1)
                           (ly:music-property first-element 'elements))))
               (set! (ly:music-property last-element 'elements)
                     (cons (make-music 'TextSpanEvent 'span-direction 1)
                           (ly:music-property last-element 'elements))))))
        music)))
  (make-music 'SequentialMusic
    'origin location
    'elements (list #{
      \once \override TextSpanner #'font-size = #-2
      \once \override TextSpanner #'font-shape = #'upright
      \once \override TextSpanner #'staff-padding = #3
      \once \override TextSpanner #'style = #'line
      \once \override TextSpanner #'to-barline = ##f
      \once \override TextSpanner #'bound-details = #'((left (Y . 0) (pa
      \once \override TextSpanner #'bound-details #'right #'text = \marku
      \once \override TextSpanner #'bound-details #'left #'text = \marku
%% uncomment this line for make full barred
    % \once \override TextSpanner #'bound-details #'left #'text = \marl
      #}
    spanned-music))))

%% %%%%%%%%% Cut here ----- End 'bbarred.ly'
%% Copy and change the last line for full barred. Rename in 'fbarred.ly'

```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% Syntax: \bbarre #"text" { notes } - text = any number of box
\relative c' { \clef "G_8" \stemUp \bbarre #"III" { <f a'>16[ c' d c d8] } }
```



Changing chord separator

The separator between different parts of a chord name can be set to any markup.

```
\chords {
  c:7sus4
  \set chordNameSeparator
    = \markup { \typewriter | }
  c:7sus4
}
```

$C^{7/sus4} C^7|sus4$

Changing the chord names to German or semi-German notation

The english naming of chords (default) can be changed to german (`\germanChords` replaces B and Bes with H and B) or semi-german (`\semiGermanChords` replaces B and Bes with H and Bb).

```
music = \chordmode {
  c1/c | cis/cis
  b1/b | bis/bis | bes/bes
}
```

```
%% The following is only here to print the names of the
%% chords styles; it can be removed if you do not need to
%% print them.
```

```
\layout {
  \context {
    \ChordNames
    \consists "Instrument_name_engraver"
  }
}
```

```
<<
\new ChordNames {
  \set ChordNames.instrumentName = #"default"
  \music
```

```

}
\new ChordNames {
  \set ChordNames.instrumentName = #"german"
  \germanChords \music }
\new ChordNames {
  \set ChordNames.instrumentName = #"semi-german"
  \semiGermanChords \music }
\context Voice { \music }
>>

```

default	C/C	C#/C#	B/B	B#/B#	Bb/Bb
german	C/c	C#/cis	H/h	H#/his	B/b
semi-german	C/c	C#/cis	H/h	H#/his	Bb/b

Changing the positions of figured bass alterations

Accidentals and plus signs can appear before or after the numbers, depending on the `figuredBassAlterationDirection` and `figuredBassPlusDirection` properties.

```

\figures {
  <6\+> <5+> <6 4-> r
  \set figuredBassAlterationDirection = #RIGHT
  <6\+> <5+> <6 4-> r
  \set figuredBassPlusDirection = #RIGHT
  <6\+> <5+> <6 4-> r
  \set figuredBassAlterationDirection = #LEFT
  <6\+> <5+> <6 4-> r
}

```

+6 #5 6 +6 5# 6 6+ 5# 6 6+ #5 6

_{b4} _{4b} _{4b} _{b4}

Chord name exceptions

The property `chordNameExceptions` can be used to store a list of special notations for specific chords.

```

% modify maj9 and 6(add9)
% Exception music is chords with markups
chExceptionMusic = {
  <c e g b d'>1-\markup { \super "maj9" }
  <c e g a d'>1-\markup { \super "6(add9)" }
}

```

```

% Convert music to list and prepend to existing exceptions.
chExceptions = #( append

```

```

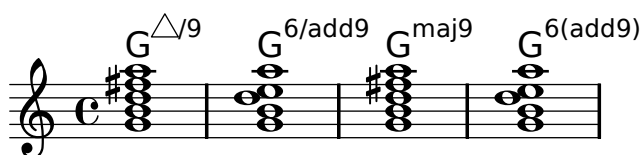
( sequential-music-to-chord-exceptions chExceptionMusic #t)
ignatzekExceptions)

theMusic = \chordmode {
  g1:maj9 g1:6.9
  \set chordNameExceptions = #chExceptions
  g1:maj9 g1:6.9
}

\layout {
  ragged-right = ##t
}

<< \context ChordNames \theMusic
    \context Voice \theMusic
>>

```



chord name major7

The layout of the major 7 can be tuned with `majorSevenSymbol`.

```

\chords {
  c:7+
  \set majorSevenSymbol = \markup { j7 }
  c:7+
}

```

$C^{\Delta}C^{j7}$

Clusters

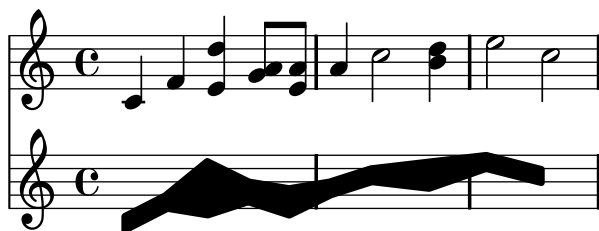
Clusters are a device to denote that a complete range of notes is to be played.

```

fragment = \relative c' {
  c4 f <e d'>4
  <g a>8 <e a> a4 c2 <d b>4
  e2 c
}

<<
  \new Staff \fragment
  \new Staff \makeClusters \fragment
>>

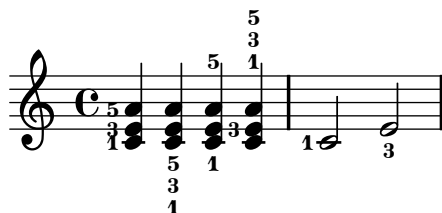
```



Controlling the placement of chord fingerings

The placement of fingering numbers can be controlled precisely. For fingering orientation to apply, you must use a chord construct `<>` even if it is a single note.

```
\relative c' {
  \set fingeringOrientations = #'(left)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(down)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(down right up)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(up)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(left)
  <c-1>2
  \set fingeringOrientations = #'(down)
  <e-3>2
}
```



Displaying complex chords

Here is a way to display a chord where the same note is played twice with different accidentals.

```
fixA = {
  \once \override Stem #'length = #9
}
fixB = {
  \once \override NoteHead #'X-offset = #1.7
  \once \override Stem #'rotation = #'(45 0 0)
  \once \override Stem #'extra-offset = #'(-0.2 . -0.2)
  \once \override Stem #'flag-style = #'no-flag
  \once \override Accidental #'extra-offset = #'(4 . 0)
}

\relative c' {
  << { \fixA <b d!>8 } \ { \voiceThree \fixB dis } >> s
}
```

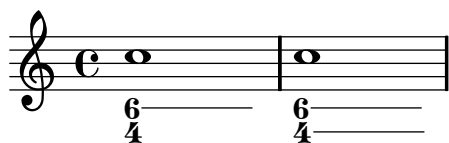


Manually break figured bass extenders for only some numbers

Figured bass often uses extenders to indicate continuation of the corresponding step. However, in this case lilypond is in greedy-mode and uses extenders whenever possible. To break individual extenders, one can simply use a modifier \! to a number, which breaks any extender attributed to that number right before the number.

```
bassfigures = \figuremode {
  \set useBassFigureExtenders = ##t
  <6 4>4 <6 4\!> <6 4\!> <6 4\!> | <6\! 4\!> <6 4> <6 4\!> <6 4>
}

<<
  \new Staff \relative c'' { c1 c1 }
  \new FiguredBass \bassfigures
>>
```



Showing chords at changes

Chord names can be displayed only at the start of lines and when the chord changes.

```
harmonies = \chordmode {
  c1:m c:m \break c:m c:m d
}

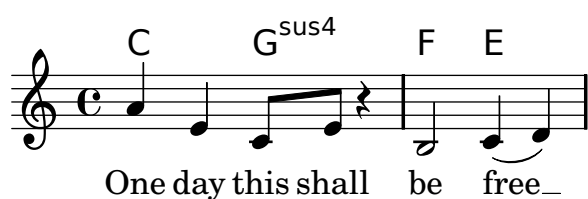
<<
  \new ChordNames {
    \set chordChanges = ##t
    \harmonies
  }
  \new Staff {
    \relative c' { \harmonies }
  }
>>
```



Simple lead sheet

When put together, chord names, a melody, and lyrics form a lead sheet:

```
<<
\chords { c2 g:sus4 f e }
\relative c'' {
  a4 e c8 e r4
  b2 c4( d)
}
\addlyrics { One day this shall be free __ }
>>
```



Single staff template with notes lyrics and chords

This template allows the preparation of a song with melody, words, and chords.

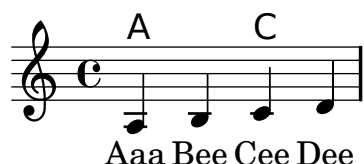
```
melody = \relative c' {
  \clef treble
  \key c \major
  \time 4/4

  a4 b c d
}

text = \lyricmode {
  Aaa Bee Cee Dee
}

harmonies = \chordmode {
  a2 c
}

\score {
  <<
    \new ChordNames {
      \set chordChanges = ##t
      \harmonies
    }
    \new Voice = "one" { \autoBeamOff \melody }
    \new Lyrics \lyricsto "one" \text
  >>
  \layout { }
  \midi { }
}
```



Single staff template with notes lyrics chords and frets

Here is a simple lead sheet template with melody, lyrics, chords and fret diagrams.

```

verseI = \lyricmode {
  \set stanza = #"1."
  This is the first verse
}

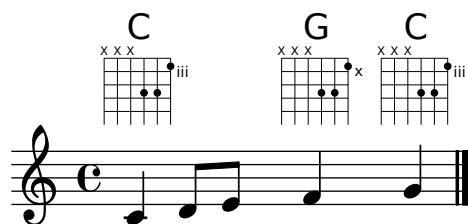
verseII = \lyricmode {
  \set stanza = #"2."
  This is the second verse.
}

theChords = \chordmode {
  % insert chords for chordnames and fretboards here
  c2 g4 c
}

staffMelody = \relative c' {
  \key c \major
  \clef treble
  % Type notes for melody here
  c4 d8 e f4 g
  \bar "|"
}

\score {
  <<
    \context ChordNames { \theChords }
    \context FretBoards { \theChords }
    \new Staff {
      \context Voice = "voiceMelody" { \staffMelody }
    }
    \new Lyrics = "lyricsI" {
      \lyricsto "voiceMelody" \verseI
    }
    \new Lyrics = "lyricsII" {
      \lyricsto "voiceMelody" \verseII
    }
  >>
  \layout { }
  \midi { }
}

```



1. This is the first verse
2. This is the second verse.

Single staff template with notes and chords

Want to prepare a lead sheet with a melody and chords? Look no further!

```
melody = \relative c' {
  \clef treble
  \key c \major
  \time 4/4

  f4 e8[ c] d4 g
  a2 ~ a
}

harmonies = \chordmode {
  c4:m f:min7 g:maj c:aug
  d2:dim b:sus
}

\score {
  <<
    \new ChordNames {
      \set chordChanges = ##t
      \harmonies
    }
    \new Staff \melody
  >>
  \layout{ }
  \midi { }
}
```



Vertically centering paired figured bass extenders

Where figured bass extender lines are being used by setting `useBassFigureExtenders` to true, pairs of congruent figured bass extender lines are vertically centered if `figuredBassCenterContinuations` is set to true.

```
<<
  \relative c' {
    c8 c b b a a c16 c b b
    c8 c b b a a c16 c b b
    c8 c b b a a c c b b
  }
}
```

```

\figures {
  \set useBassFigureExtenders = ##t
  <6+ 4 3>4 <6 4 3>8 r
  <6+ 4 3>4 <6 4 3>8 <4 3+>16 r
  \set figuredBassCenterContinuations = ##t
  <6+ 4 3>4 <6 4 3>8 r
  <6+ 4 3>4 <6 4 3>8 <4 3+>16 r
  \set figuredBassCenterContinuations = ##f
  <6+ 4 3>4 <6 4 3>8 r
  <6+ 4 3>4 <6 4 3>8 <4 3+>8
}
>>

```



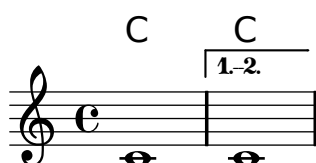
Volta below chords

By adding the `Volta_engraver` to the relevant staff, volte can be put under chords.

```

\score {
  <<
    \chords {
      c1
      c1
    }
    \new Staff \with {
      \consists "Volta_engraver"
    }
    {
      \repeat volta 2 { c'1 }
      \alternative { c' }
    }
  >>
  \layout {
    \context {
      \Score
      \remove "Volta_engraver"
    }
  }
}

```



Keyboards

Section “Keyboard and other multi-staff instruments” in *Notation Reference*

Accordion-discant symbols

Accordion discant-specific symbols are added using `\markup`. The vertical placement of the symbols can be tweaked by changing the `\raise` arguments.

```
discant = \markup {
  \musicglyph #"accordion.accDiscant"
}
dot = \markup {
  \musicglyph #"accordion.accDot"
}

\layout { ragged-right = ##t }

% 16 voets register
accBasson = ^\markup {
  \combine
  \discant
  \raise #0.5 \dot
}

% een korig 8 en 16 voets register
accBandon = ^\markup {
  \combine
  \discant
  \combine
  \raise #0.5 \dot
  \raise #1.5 \dot
}

accVCello = ^\markup {
  \combine
  \discant
  \combine
  \raise #0.5 \dot
  \combine
  \raise #1.5 \dot
  \translate #'(1 . 0) \raise #1.5 \dot
}

% 4-8-16 voets register
accHarmon = ^\markup {
  \combine
  \discant
  \combine
  \raise #0.5 \dot
  \combine
  \raise #1.5 \dot
  \raise #2.5 \dot
}
```

```
}
```

```
accTrombon = ^\markup {
  \combine
  \discant
  \combine
  \raise #0.5 \dot
  \combine
  \raise #1.5 \dot
  \combine
  \translate #'(1 . 0) \raise #1.5 \dot
  \translate #'(-1 . 0) \raise #1.5 \dot
}
```

```
% eenkorig 4 en 16 voets register
```

```
accOrgan = ^\markup {
  \combine
  \discant
  \combine
  \raise #0.5 \dot
  \raise #2.5 \dot
}
```

```
accMaster = ^\markup {
  \combine
  \discant
  \combine
  \raise #0.5 \dot
  \combine
  \raise #1.5 \dot
  \combine
  \translate #'(1 . 0) \raise #1.5 \dot
  \combine
  \translate #'(-1 . 0) \raise #1.5 \dot
  \raise #2.5 \dot
}
```

```
accAccord = ^\markup {
  \combine
  \discant
  \combine
  \raise #1.5 \dot
  \combine
  \translate #'(1 . 0) \raise #1.5 \dot
  \combine
  \translate #'(-1 . 0) \raise #1.5 \dot
  \raise #2.5 \dot
}
```

```
accMusette = ^\markup {
  \combine
  \discant
```

```

    \combine
    \raise #1.5 \dot
    \combine
    \translate #'(1 . 0) \raise #1.5 \dot
    \translate #'(-1 . 0) \raise #1.5 \dot
}

```

```

accCeleste = ^\markup {
  \combine
  \discant
  \combine
  \raise #1.5 \dot
  \translate #'(-1 . 0) \raise #1.5 \dot
}

```

```

accOboe = ^\markup {
  \combine
  \discant
  \combine
  \raise #1.5 \dot
  \raise #2.5 \dot
}

```

```

accClarin = ^\markup {
  \combine
  \discant
  \raise #1.5 \dot
}

```

```

accPiccolo = ^\markup {
  \combine
  \discant
  \raise #2.5 \dot
}

```

```

accViolin = ^\markup {
  \combine
  \discant
  \combine
  \raise #1.5 \dot
  \combine
  \translate #'(1 . 0) \raise #1.5 \dot
  \raise #2.5 \dot
}

```

```

\relative c'' {
  c4 d\accBasson e f
  c4 d\accBandon e f
  c4 d\accVCello e f
  c4 d\accHarmon e f
  c4 d\accTrombon e f
  \break
}

```

```

c4 d\accOrgan e f
c4 d\accMaster e f
c4 d\accAccord e f
c4 d\accMusette e f
c4 d\accCeleste e f
\break
c4 d\accOboe e f
c4 d\accClarinet e f
c4 d\accPiccolo e f
c4 d\accViolin e f
}

```

The image shows three staves of musical notation. The first staff contains five measures, the second contains five measures, and the third contains four measures. Each measure consists of a half note followed by a whole note. Above the notes, there are various symbols: circles with dots inside, and some with horizontal lines. These symbols likely represent different keyboard instruments or articulation marks as defined in the code above.

Changing the text for sustain markings

`Staff.pedalSustainStrings` can be used to set the text used for pedal down and up. Note that the only valid strings are those found in the list of pedal glyphs - the values used in this snippet constitute an exhaustive list.

```
sustainNotes = { c4\sustainOn d e\sustainOff\sustainOn f\sustainOff }
```

```

\relative c' {
  \sustainNotes
  \set Staff.pedalSustainStrings = #("P" "P-" "-")
  \sustainNotes
  \set Staff.pedalSustainStrings = #("d" "de" "e")
  \sustainNotes
  \set Staff.pedalSustainStrings = #("M" "M-" "-")
  \sustainNotes
  \set Staff.pedalSustainStrings = #("Ped" "*Ped" "*")
  \sustainNotes
}

```

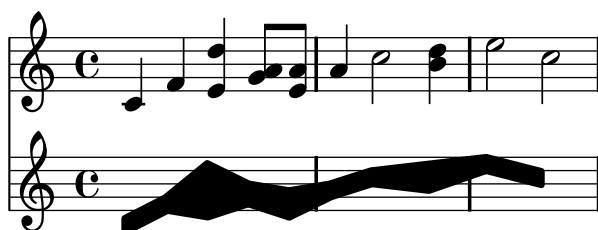
The image shows a single staff of musical notation in treble clef with a common time signature. The staff contains a sequence of notes with various accidentals and articulation marks below them. The marks include symbols like 'Ped', 'P', 'd', 'de', 'e', and others, which correspond to the sustain markings defined in the code above.

Clusters

Clusters are a device to denote that a complete range of notes is to be played.

```
fragment = \relative c' {
  c4 f <e d'>4
  <g a>8 <e a> a4 c2 <d b>4
  e2 c
}

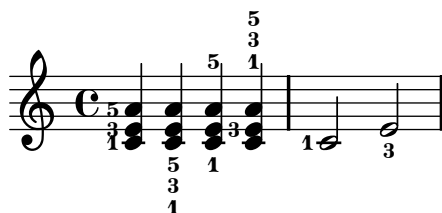
<<
  \new Staff \fragment
  \new Staff \makeClusters \fragment
>>
```



Controlling the placement of chord fingerings

The placement of fingering numbers can be controlled precisely. For fingering orientation to apply, you must use a chord construct <> even if it is a single note.

```
\relative c' {
  \set fingeringOrientations = #'(left)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(down)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(down right up)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(up)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(left)
  <c-1>2
  \set fingeringOrientations = #'(down)
  <e-3>2
}
```



Creating slurs across voices

In some situations, it may be necessary to create slurs between notes from different voices.

The solution is to add invisible notes to one of the voices, using `\hideNotes`.

This example is measure 235 of the Ciaccona from Bach's 2nd Partita for solo violin, BWV 1004.

```
\relative c' {
  <<
  {
    d16( a') s a s a[ s a] s a[ s a]
  }
  \\\
  {
    \slurUp
    bes,16[ s e](
    \hideNotes a)
    \unHideNotes f[(
    \hideNotes a)
    \unHideNotes fis](
    \hideNotes a)
    \unHideNotes g[(
    \hideNotes a)
    \unHideNotes gis](
    \hideNotes a)
  }
  >>
}
```



Cross-staff tremolos

Since `\repeat tremolo` expects exactly two musical arguments for chord tremolos, the note or chord which changes staff within a cross-staff tremolo should be placed inside curly braces together with its `\change Staff` command.

```
\new PianoStaff <<
  \new Staff = "up" \relative c'' {
    \key a \major
    \time 3/8
    s4.
  }
  \new Staff = "down" \relative c'' {
    \key a \major
    \time 3/8
    \voiceOne
    \repeat tremolo 6 {
      <a e'>32
      {

```

```

\change Staff = "up"
\voiceTwo
<cis a' dis>32
}
}
}
>>

```



Fine-tuning pedal brackets

The appearance of pedal brackets may be altered in different ways.

```

\paper { ragged-right = ##f }
\relative c' {
  c2\sostenutoOn c
  c2\sostenutoOff c
  \once \override Staff.PianoPedalBracket #'shorten-pair = #'(-7 . -2)
  c2\sostenutoOn c
  c2\sostenutoOff c
  \once \override Staff.PianoPedalBracket #'edge-height = #'(0 . 3)
  c2\sostenutoOn c
  c2\sostenutoOff c
}

```



Indicating cross-staff chords with arpeggio bracket

An arpeggio bracket can indicate that notes on two different staves are to be played with the same hand. In order to do this, the `PianoStaff` must be set to accept cross-staff arpeggios and the arpeggios must be set to the bracket shape in the `PianoStaff` context.

(Debussy, *Les collines d'Anacapri*, m. 65)

```

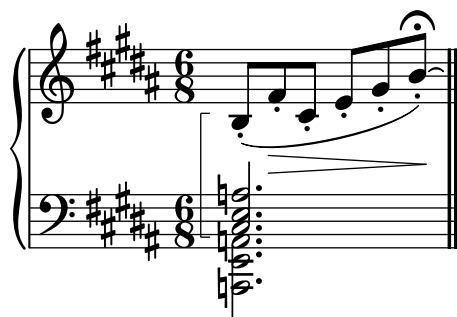
\new PianoStaff <<
  \set PianoStaff.connectArpeggios = ##t
  \override PianoStaff.Arpeggio #'stencil = #ly:arpeggio::brew-chord-bracket
  \new Staff {
    \relative c' {
      \key b \major
      \time 6/8
      b8-.(\arpeggio fis'-.\> cis-. e-. gis-. b-.)\!\fermata^\laissezVibrer
    }
  }
}

```

```

        \bar "||"
    }
}
\new Staff {
    \relative c' {
        \clef bass
        \key b \major
        <<
        {
            <a e cis>2.\arpeggio
        }
        \\\
        {
            <a, e a,>2.
        }
        >>
    }
}
>>

```



Jazz combo template

This is quite an advanced template, for a jazz ensemble. Note that all instruments are notated in `\key c \major`. This refers to the key in concert pitch; the key will be automatically transposed if the music is within a `\transpose` section.

```

\header {
    title = "Song"
    subtitle = "(tune)"
    composer = "Me"
    meter = "moderato"
    piece = "Swing"
    tagline = \markup {
        \column {
            "LilyPond example file by Amelie Zapf,"
            "Berlin 07/07/2003"
        }
    }
}

```

```

%#(set-global-staff-size 16)
\include "english.ly"

```

```

%%%%%%%%%%%%%% Some macros %%%%%%%%%%%%%%%

sl = {
  \override NoteHead #'style = #'slash
  \override Stem #'transparent = ##t
}
nsl = {
  \revert NoteHead #'style
  \revert Stem #'transparent
}
crOn = \override NoteHead #'style = #'cross
crOff = \revert NoteHead #'style

%% insert chord name style stuff here.

jazzChords = { }

%%%%%%%%%%%%%% Keys'n'things %%%%%%%%%%%%%%%

global = { \time 4/4 }

Key = { \key c \major }

% ##### Horns #####

% ----- Trumpet -----
trpt = \transpose c d \relative c' {
  \Key
  c1 | c | c |
}
trpHarmony = \transpose c' d {
  \jazzChords
}
trumpet = {
  \global
  \set Staff.instrumentName = #"Trumpet"
  \clef treble
  <<
  \trpt
  >>
}

% ----- Alto Saxophone -----
alto = \transpose c a \relative c' {
  \Key
  c1 | c | c |
}
altoHarmony = \transpose c' a {
  \jazzChords
}
altoSax = {

```

```

\global
\set Staff.instrumentName = #"Alto Sax"
\clef treble
<<
  \alto
>>
}

% ----- Baritone Saxophone -----
bari = \transpose c a' \relative c {
  \Key
  c1
  c1
  \sl
  d4^"Solo" d d d
  \nsl
}
bariHarmony = \transpose c' a \chordmode {
  \jazzChords s1 s d2:maj e:m7
}
bariSax = {
  \global
  \set Staff.instrumentName = #"Bari Sax"
  \clef treble
  <<
    \bari
  >>
}

% ----- Trombone -----
tbone = \relative c {
  \Key
  c1 | c | c
}
tboneHarmony = \chordmode {
  \jazzChords
}
trombone = {
  \global
  \set Staff.instrumentName = #"Trombone"
  \clef bass
  <<
    \tbone
  >>
}

% ##### Rhythm Section #####

% ----- Guitar -----
gtr = \relative c'' {
  \Key
  c1

```

```

\sl
b4 b b b
\nsl
c1
}
gtrHarmony = \chordmode {
  \jazzChords
  s1 c2:min7+ d2:maj9
}
guitar = {
  \global
  \set Staff.instrumentName = #"Guitar"
  \clef treble
  <<
    \gtr
  >>
}

%% ----- Piano -----
rhUpper = \relative c'' {
  \voiceOne
  \Key
  c1 | c | c
}
rhLower = \relative c' {
  \voiceTwo
  \Key
  e1 | e | e
}

lhUpper = \relative c' {
  \voiceOne
  \Key
  g1 | g | g
}
lhLower = \relative c {
  \voiceTwo
  \Key
  c1 | c | c
}

PianoRH = {
  \clef treble
  \global
  \set Staff.midiInstrument = #"acoustic grand"
  <<
    \new Voice = "one" \rhUpper
    \new Voice = "two" \rhLower
  >>
}
PianoLH = {
  \clef bass

```

```

\global
\set Staff.midiInstrument = #"acoustic grand"
<<
  \new Voice = "one" \lhUpper
  \new Voice = "two" \lhLower
>>
}

piano = {
  <<
    \set PianoStaff.instrumentName = #"Piano"
    \new Staff = "upper" \PianoRH
    \new Staff = "lower" \PianoLH
  >>
}

% ----- Bass Guitar -----
Bass = \relative c {
  \Key
  c1 | c | c
}
bass = {
  \global
  \set Staff.instrumentName = #"Bass"
  \clef bass
  <<
    \Bass
  >>
}

% ----- Drums -----
up = \drummode {
  \voiceOne
  hh4 <hh sn> hh <hh sn>
  hh4 <hh sn> hh <hh sn>
  hh4 <hh sn> hh <hh sn>
}
down = \drummode {
  \voiceTwo
  bd4 s bd s
  bd4 s bd s
  bd4 s bd s
}

drumContents = {
  \global
  <<
    \set DrumStaff.instrumentName = #"Drums"
    \new DrumVoice \up
    \new DrumVoice \down
  >>
}

```

%%%%%%%%%% It All Goes Together Here %%%%%%%%%%%

```
\score {
  <<
    \new StaffGroup = "horns" <<
      \new Staff = "trumpet" \trumpet
      \new Staff = "altosax" \altoSax
      \new ChordNames = "barichords" \bariHarmony
      \new Staff = "barisax" \bariSax
      \new Staff = "trombone" \trombone
    >>

    \new StaffGroup = "rhythm" <<
      \new ChordNames = "chords" \gtrHarmony
      \new Staff = "guitar" \guitar
      \new PianoStaff = "piano" \piano
      \new Staff = "bass" \bass
      \new DrumStaff \drumContents
    >>
  >>
  \layout {
    \context { \Staff \RemoveEmptyStaves }
    \context {
      \Score
      \override BarNumber #'padding = #3
      \override RehearsalMark #'padding = #2
      skipBars = ##t
    }
  }
  \midi { }
}
```

Song (tune)

moderato

Swing

Me

Trumpet

Alto Sax

Bari Sax

Trombone

Guitar

Piano

Bass

Drums

B^{Δ} $C\sharp m^7$
Solo

$Cm^{\Delta} D^{\Delta 9}$

Laissez vibrer ties

Laissez vibrer ties have a fixed size. Their formatting can be tuned using 'tie-configuration.

```
\relative c' {
  <c e g>4\laissezVibrer r <c f g>\laissezVibrer r
  <c d f g>4\laissezVibrer r <c d f g>4.\laissezVibrer r8

  <c d e f>4\laissezVibrer r
  \override LaissezVibrerTieColumn #'tie-configuration
    = \#`((-7 . ,DOWN)
      (-5 . ,DOWN)
      (-3 . ,UP)
      (-1 . ,UP))
  <c d e f>4\laissezVibrer r
}
```



Piano template (simple)

Here is a simple piano staff with some notes.

```
upper = \relative c'' {
  \clef treble
  \key c \major
  \time 4/4

  a4 b c d
}

lower = \relative c {
  \clef bass
  \key c \major
  \time 4/4

  a2 c
}

\score {
  \new PianoStaff <<
    \set PianoStaff.instrumentName = #"Piano  "
    \new Staff = "upper" \upper
    \new Staff = "lower" \lower
  >>
  \layout { }
  \midi { }
}
```



Piano template with centered dynamics

Many piano scores have the dynamics centered between the two staves. The `Dynamics` context, if placed between staves, will automatically position dynamics correctly.

```
global = {
  \key c \major
  \time 4/4
}

upper = \relative c'' {
  \clef treble
  a4 b c d
}

lower = \relative c {
```

```

\clef bass
a2 c
}

dynamics = {
  s2\fff\> s4 s\!\pp
}

pedal = {
  s2\sustainOn s\sustainOff
}

\score {
  \new PianoStaff = "PianoStaff_pf" <<
    \new Staff = "Staff_pfUpper" << \global \upper >>
    \new Dynamics = "Dynamics_pf" \dynamics
    \new Staff = "Staff_pfLower" << \global \lower >>
    \new Dynamics = "pedal" \pedal
  >>
  \layout { }
}

\score {
  \new PianoStaff = "PianoStaff_pf" <<
    \new Staff = "Staff_pfUpper" << \global \upper \dynamics \pedal >>
    \new Staff = "Staff_pfLower" << \global \lower \dynamics \pedal >>
  >>
  \midi { }
}

```



Piano template with centered lyrics

Instead of having a full staff for the melody and lyrics, lyrics can be centered between the staves of a piano staff.

```

upper = \relative c'' {
  \clef treble
  \key c \major
  \time 4/4

  a4 b c d
}

lower = \relative c {

```

```

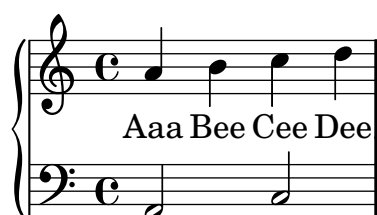
\clef bass
\key c \major
\time 4/4

a2 c
}

text = \lyricmode {
  Aaa Bee Cee Dee
}

\score {
  \new GrandStaff <<
    \new Staff = upper { \new Voice = "singer" \upper }
    \new Lyrics \lyricsto "singer" \text
    \new Staff = lower { \lower }
  >>
  \layout {
    \context {
      \GrandStaff
      \accepts "Lyrics"
    }
    \context {
      \Lyrics
      \consists "Bar_engraver"
    }
  }
  \midi { }
}

```



Piano template with melody and lyrics

Here is a typical song format: one staff with the melody and lyrics, with piano accompaniment underneath.

```

melody = \relative c'' {
  \clef treble
  \key c \major
  \time 4/4

  a b c d
}

text = \lyricmode {
  Aaa Bee Cee Dee
}

```

```

}

upper = \relative c'' {
  \clef treble
  \key c \major
  \time 4/4

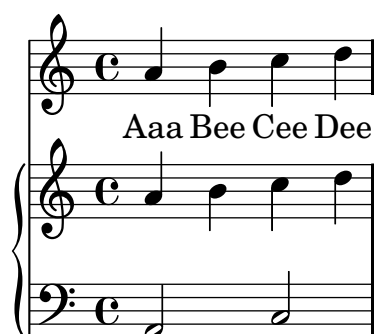
  a4 b c d
}

lower = \relative c {
  \clef bass
  \key c \major
  \time 4/4

  a2 c
}

\score {
  <<
    \new Voice = "mel" { \autoBeamOff \melody }
    \new Lyrics \lyricsto mel \text
    \new PianoStaff <<
      \new Staff = "upper" \upper
      \new Staff = "lower" \lower
    >>
  >>
  \layout {
    \context { \Staff \RemoveEmptyStaves }
  }
  \midi { }
}

```



Vocal ensemble template with automatic piano reduction

This template adds an automatic piano reduction to the standard SATB vocal score demonstrated in “Vocal ensemble template”. This demonstrates one of the strengths of LilyPond – you can use a music definition more than once. If any changes are made to the vocal notes (say, `tenorMusic`), then the changes will also apply to the piano reduction.

```

global = {
  \key c \major

```

```

\time 4/4
}

sopMusic = \relative c'' {
  c4 c c8[( b)] c4
}
sopWords = \lyricmode {
  hi hi hi hi
}

altoMusic = \relative c' {
  e4 f d e
}
altoWords = \lyricmode {
  ha ha ha ha
}

tenorMusic = \relative c' {
  g4 a f g
}
tenorWords = \lyricmode {
  hu hu hu hu
}

bassMusic = \relative c {
  c4 c g c
}
bassWords = \lyricmode {
  ho ho ho ho
}

\score {
  <<
    \new ChoirStaff <<
      \new Lyrics = sopranos { s1 }
      \new Staff = women <<
        \new Voice = sopranos { \voiceOne << \global \sopMusic >> }
        \new Voice = altos { \voiceTwo << \global \altoMusic >> }
      >>
      \new Lyrics = altos { s1 }
      \new Lyrics = tenors { s1 }
      \new Staff = men <<
        \clef bass
        \new Voice = tenors { \voiceOne << \global \tenorMusic >> }
        \new Voice = basses { \voiceTwo << \global \bassMusic >> }
      >>
      \new Lyrics = basses { s1 }
      \context Lyrics = sopranos \lyricsto sopranos \sopWords
      \context Lyrics = altos \lyricsto altos \altoWords
      \context Lyrics = tenors \lyricsto tenors \tenorWords
      \context Lyrics = basses \lyricsto basses \bassWords
    >>
  >>
}

```

```

\new PianoStaff <<
  \new Staff <<
    \set Staff.printPartCombineTexts = ##f
    \partcombine
    << \global \sopMusic >>
    << \global \altoMusic >>
  >>
  \new Staff <<
    \clef bass
    \set Staff.printPartCombineTexts = ##f
    \partcombine
    << \global \tenorMusic >>
    << \global \bassMusic >>
  >>
>>
>>
\layout {
  \context {
    % a little smaller so lyrics
    % can be closer to the staff
    \Staff
    \override VerticalAxisGroup #'minimum-Y-extent = #'(-3 . 3)
  }
}

```

hi hi hi hi

ha ha ha ha

hu hu hu hu

ho ho ho ho

Percussion

Section “Percussion” in *Notation Reference*

Adding drum parts

Using the powerful pre-configured tools such as the `\drummode` function and the `DrumStaff` context, inputting drum parts is quite easy: drums are placed at their own staff positions (with a special clef symbol) and have note heads according to the drum. Attaching an extra symbol to the drum or restricting the number of lines is possible.

```
drh = \drummode { cymc4.^"crash" hhc16^"h.h." hh hhc8 hho hhc8 hh16 hh hhc4 r4 r2 }
drl = \drummode { bd4 sn8 bd bd4 << bd ss >> bd8 tommh tommh bd tom1 tom1 bd tomfh16 tomfh
timb = \drummode { timh4 ssh timl8 ssh r timh r4 ssh8 timl r4 cb8 cb }
```

```
\score {
  <<
    \new DrumStaff \with {
      drumStyleTable = #timbales-style
      \override StaffSymbol #'line-count = #2
      \override BarLine #'bar-size = #2
    } <<
      \set Staff.instrumentName = #"timbales"
      \timb
    >>
    \new DrumStaff <<
      \set Staff.instrumentName = #"drums"
      \new DrumVoice { \stemUp \drh }
      \new DrumVoice { \stemDown \drl }
    >>
  >>
  \layout { }
  \midi {
    \context {
      \Score
      tempoWholesPerMinute = #(ly:make-moment 120 4)
    }
  }
}
```

Heavily customized polymetric time signatures

Though the polymetric time signature shown was not the most essential item here, it has been included to show the beat of this piece (which is the template of a real Balkan song!).

```

#(define plus (markup #:vcenter "+"))
#(define ((custom-time-signature one two three four five six
              seven eight nine ten eleven num) grob)
  (grob-interpret-markup grob
    (markup #:override '(baseline-skip . 0) #:number
      (#:line ((#:column (one num)) plus
                    (#:column (two num)) plus
                    (#:column (three num)) plus
                    (#:column (four num)) plus
                    (#:column (five num)) plus
                    (#:column (six num)) plus
                    (#:column (seven num)) plus
                    (#:column (eight num)) plus
                    (#:column (nine num)) plus
                    (#:column (ten num)) plus
                    (#:column (eleven num)))))))

melody = \relative c'' {
  \set Staff.instrumentName = #"Bb Sop."
  \key g \major
  #(set-time-signature 25 8 '(3 2 2 3 2 2 2 2 3 2 2))
  \override Staff.TimeSignature #'stencil =
    #(custom-time-signature "3" "2" "2" "3" "2" "2"
      "2" "2" "3" "2" "2" "8")
  c8 c c d4 c8 c b c b a4 g fis8 e d c b' c d e4-^ fis8 g \break
  c,4. d4 c4 d4. c4 d c2 d4. e4-^ d4
  c4. d4 c4 d4. c4 d c2 d4. e4-^ d4 \break
  c4. d4 c4 d4. c4 d c2 d4. e4-^ d4
  c4. d4 c4 d4. c4 d c2 d4. e4-^ d4 \break
}

drum = \new DrumStaff \drummode {
  \bar "|:" bd4.^ \markup { Drums } sn4 bd \bar ":" sn4.
  bd4 sn \bar ":" bd sn bd4. sn4 bd \bar ":@"
}

{
  \melody
  \drum
}

```

Bb Sop.



Jazz combo template

This is quite an advanced template, for a jazz ensemble. Note that all instruments are notated in `\key c \major`. This refers to the key in concert pitch; the key will be automatically transposed if the music is within a `\transpose` section.

```
\header {
  title = "Song"
  subtitle = "(tune)"
  composer = "Me"
  meter = "moderato"
  piece = "Swing"
  tagline = \markup {
    \column {
      "LilyPond example file by Amelie Zapf,"
      "Berlin 07/07/2003"
    }
  }
}
```

```
%#(set-global-staff-size 16)
\include "english.ly"
```

```
%%%%%%%%%% Some macros %%%%%%%%%%
```

```
sl = {
  \override NoteHead #'style = #'slash
  \override Stem #'transparent = ##t
}
nsl = {
  \revert NoteHead #'style
  \revert Stem #'transparent
}
crOn = \override NoteHead #'style = #'cross
crOff = \revert NoteHead #'style
```

```
% insert chord name style stuff here.
```

```
jazzChords = { }
```

```
%%%%%%%%%% Keys'n'things %%%%%%%%%%
```

```
global = { \time 4/4 }
```

```
Key = { \key c \major }
```

```
% ##### Horns #####
```

```
% ----- Trumpet -----
```

```
trpt = \transpose c d \relative c' {
  \Key
```

```

    c1 | c | c |
}
trpHarmony = \transpose c' d {
  \jazzChords
}
trumpet = {
  \global
  \set Staff.instrumentName = #"Trumpet"
  \clef treble
  <<
    \trpt
  >>
}

% ----- Alto Saxophone -----
alto = \transpose c a \relative c' {
  \Key
  c1 | c | c |
}
altoHarmony = \transpose c' a {
  \jazzChords
}
altoSax = {
  \global
  \set Staff.instrumentName = #"Alto Sax"
  \clef treble
  <<
    \alto
  >>
}

% ----- Baritone Saxophone -----
bari = \transpose c a' \relative c {
  \Key
  c1
  c1
  \sl
  d4~"Solo" d d d
  \nsl
}
bariHarmony = \transpose c' a \chordmode {
  \jazzChords s1 s d2:maj e:m7
}
bariSax = {
  \global
  \set Staff.instrumentName = #"Bari Sax"
  \clef treble
  <<
    \bari
  >>
}

```

```

% ----- Trombone -----
tbone = \relative c {
  \Key
  c1 | c | c
}
tboneHarmony = \chordmode {
  \jazzChords
}
trombone = {
  \global
  \set Staff.instrumentName = #"Trombone"
  \clef bass
  <<
    \tbone
  >>
}

% ##### Rhythm Section #####

% ----- Guitar -----
gtr = \relative c' {
  \Key
  c1
  \sl
  b4 b b b
  \nsl
  c1
}
gtrHarmony = \chordmode {
  \jazzChords
  s1 c2:min7+ d2:maj9
}
guitar = {
  \global
  \set Staff.instrumentName = #"Guitar"
  \clef treble
  <<
    \gtr
  >>
}

%% ----- Piano -----
rhUpper = \relative c' {
  \voiceOne
  \Key
  c1 | c | c
}
rhLower = \relative c' {
  \voiceTwo
  \Key
  e1 | e | e
}

```

```

lhUpper = \relative c' {
  \voiceOne
  \Key
  g1 | g | g
}
lhLower = \relative c {
  \voiceTwo
  \Key
  c1 | c | c
}

PianoRH = {
  \clef treble
  \global
  \set Staff.midiInstrument = #"acoustic grand"
  <<
    \new Voice = "one" \rhUpper
    \new Voice = "two" \rhLower
  >>
}
PianoLH = {
  \clef bass
  \global
  \set Staff.midiInstrument = #"acoustic grand"
  <<
    \new Voice = "one" \lhUpper
    \new Voice = "two" \lhLower
  >>
}

piano = {
  <<
    \set PianoStaff.instrumentName = #"Piano"
    \new Staff = "upper" \PianoRH
    \new Staff = "lower" \PianoLH
  >>
}

% ----- Bass Guitar -----
Bass = \relative c {
  \Key
  c1 | c | c
}
bass = {
  \global
  \set Staff.instrumentName = #"Bass"
  \clef bass
  <<
    \Bass
  >>
}

```

```

% ----- Drums -----
up = \drummode {
  \voiceOne
  hh4 <hh sn> hh <hh sn>
  hh4 <hh sn> hh <hh sn>
  hh4 <hh sn> hh <hh sn>
}
down = \drummode {
  \voiceTwo
  bd4 s bd s
  bd4 s bd s
  bd4 s bd s
}

drumContents = {
  \global
  <<
    \set DrumStaff.instrumentName = #"Drums"
    \new DrumVoice \up
    \new DrumVoice \down
  >>
}

%%%%%%%%%% It All Goes Together Here %%%%%%%%%%%%%%%

\score {
  <<
    \new StaffGroup = "horns" <<
      \new Staff = "trumpet" \trumpet
      \new Staff = "altosax" \altoSax
      \new ChordNames = "barichords" \bariHarmony
      \new Staff = "barisax" \bariSax
      \new Staff = "trombone" \trombone
    >>

    \new StaffGroup = "rhythm" <<
      \new ChordNames = "chords" \gtrHarmony
      \new Staff = "guitar" \guitar
      \new PianoStaff = "piano" \piano
      \new Staff = "bass" \bass
      \new DrumStaff \drumContents
    >>
  >>
  \layout {
    \context { \Staff \RemoveEmptyStaves }
    \context {
      \Score
      \override BarNumber #'padding = #3
      \override RehearsalMark #'padding = #2
      skipBars = ##t
    }
  }
}

```

```

}
\midi { }
}

```

Song

(tune)

moderato

Swing

Trumpet

Alto Sax

Bari Sax

Trombone

Guitar

Piano

Bass

Drums

B^{Δ} $C\sharp m^7$

Solo

Cm^{Δ} $D^{\Delta 9}$

Percussion beaters

Graphic symbols for percussion instruments are not natively supported; however it is possible to include such symbols, either as an external EPS file or as embedded PostScript code inside a markup, as demonstrated in this example.

```

stick = \markup {
  \with-dimensions #'(0 . 5) #'(0 . 5)
  \postscript #"
    0 6 translate
    0.8 -0.8 scale
    0 0 0 setrgbcolor
    [] 0 setdash
    1 setlinewidth
    0 setlinejoin
    0 setlinecap
    gsave [1 0 0 1 0 0] concat
    gsave [1 0 0 1 -3.5406095 -199.29342] concat
    gsave
    0 0 0 setrgbcolor
    newpath
    7.1434065 200.94354 moveto
    7.2109628 200.90454 7.2785188 200.86554 7.3460747 200.82654 curveto
    8.2056347 202.31535 9.0651946 203.80414 9.9247546 205.29295 curveto
    9.8571989 205.33195 9.7896429 205.37095 9.7220864 205.40996 curveto
    8.8625264 203.92115 8.0029664 202.43233 7.1434065 200.94354 curveto
    closepath
    eofill
    grestore
    gsave
    0 0 0 setrgbcolor
    newpath
    4.9646672 203.10444 moveto
    5.0036707 203.03688 5.0426744 202.96933 5.0816777 202.90176 curveto
    6.5704792 203.76133 8.0592809 204.6209 9.5480824 205.48045 curveto
    9.5090791 205.54801 9.4700754 205.61556 9.4310717 205.68311 curveto
    7.94227 204.82356 6.4534687 203.96399 4.9646672 203.10444 curveto
    closepath
    eofill
    grestore
    gsave
    <<
    /ShadingType 3
    /ColorSpace /DeviceRGB
    /Coords [113.13708 207.87465 0 113.13708 207.87465 16.162441]
    /Extend [true true]
    /Domain [0 1]
    /Function <<
    /FunctionType 3
    /Functions
    [
    <<
    /FunctionType 2
    /Domain [0 1]
    /C0 [1 1 1]
    /C1 [0.72941178 0.72941178 0.72941178]
    /N 1
    >>
  >>

```

```

]
/Domain [0 1]
/Bounds [ ]
/Encode [ 0 1 ]
>>
>>
newpath
7.6422017 200.76488 moveto
7.6505696 201.02554 7.3905363 201.24867 7.1341335 201.20075 curveto
6.8759501 201.16916 6.6949602 200.87978 6.7801462 200.63381 curveto
6.8480773 200.39155 7.1438307 200.25377 7.3728389 200.35861 curveto
7.5332399 200.42458 7.6444521 200.59122 7.6422017 200.76488 curveto
closepath
clip
gsave [0.052859054 0.063089841 -0.020912282 0.017521108 5.7334261 189.76443] concat
shfill
grestore
grestore
0 0 0 setrgbcolor
[] 0 setdash
0.027282091 setlinewidth
0 setlinejoin
0 setlinecap
newpath
7.6422017 200.76488 moveto
7.6505696 201.02554 7.3905363 201.24867 7.1341335 201.20075 curveto
6.8759501 201.16916 6.6949602 200.87978 6.7801462 200.63381 curveto
6.8480773 200.39155 7.1438307 200.25377 7.3728389 200.35861 curveto
7.5332399 200.42458 7.6444521 200.59122 7.6422017 200.76488 curveto
closepath
stroke
gsave
<<
/ShadingType 3
/ColorSpace /DeviceRGB
/Coords [113.13708 207.87465 0 113.13708 207.87465 16.162441]
/Extend [true true]
/Domain [0 1]
/Function <<
/FunctionType 3
/Functions
[
<<
/FunctionType 2
/Domain [0 1]
/C0 [1 1 1]
/C1 [0.72941178 0.72941178 0.72941178]
/N 1
>>
]
/Domain [0 1]
/Bounds [ ]

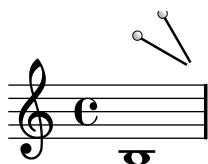
```

```

/Encode [ 0 1 ]
>>
>>
newpath
5.2721217 202.83181 moveto
5.2804896 203.09247 5.0204563 203.3156 4.7640539 203.26768 curveto
4.5058701 203.23609 4.3248803 202.94671 4.4100662 202.70074 curveto
4.4779975 202.45848 4.7737511 202.3207 5.0027593 202.42554 curveto
5.1631598 202.49149 5.2743721 202.65813 5.2721217 202.83181 curveto
closepath
clip
gsave [0.052859054 0.063089841 -0.020912282 0.017521108 3.363346 191.83136] concat
shfill
grestore
grestore
0 0 0 setrgbcolor
[] 0 setdash
0.027282091 setlinewidth
0 setlinejoin
0 setlinecap
newpath
5.2721217 202.83181 moveto
5.2804896 203.09247 5.0204563 203.3156 4.7640539 203.26768 curveto
4.5058701 203.23609 4.3248803 202.94671 4.4100662 202.70074 curveto
4.4779975 202.45848 4.7737511 202.3207 5.0027593 202.42554 curveto
5.1631598 202.49149 5.2743721 202.65813 5.2721217 202.83181 curveto
closepath
stroke
grestore
grestore
"
}

\score {
  b1^\stick
}

```



Printing music with different time signatures

In the following snippet, two parts have a completely different time signature, yet remain synchronized. The bar lines can no longer be printed at the **Score** level; to allow independent bar lines in each part, the **Default_barline_engraver** and **Timing_translator** are moved from the **Score** context to the **Staff** context.

If bar numbers are required, the **Bar_number_engraver** should also be moved, since it relies on properties set by the **Timing_translator**; a **\with** block can be used to add bar numbers to the relevant staff.

```

\paper {
  indent = #0
  ragged-right = ##t
}

global = { \time 3/4 { s2.*3 } \bar "" \break { s2.*3 } }

\layout {
  \context {
    \Score
    \remove "Timing_translator"
    \remove "Default_bar_line_engraver"
    \remove "Bar_number_engraver"
    \override SpacingSpanner #'uniform-stretching = ##t
    \override SpacingSpanner #'strict-note-spacing = ##t
    proportionalNotationDuration = #(ly:make-moment 1 64)
  }
  \context {
    \Staff
    \consists "Timing_translator"
    \consists "Default_bar_line_engraver"
  }
  \context {
    \Voice
    \remove "Forbid_line_break_engraver"
    tupletFullLength = ##t
  }
}

Bassklarinette = \new Staff \with {
  \consists "Bar_number_engraver"
  barNumberVisibility = #(every-nth-bar-number-visible 2)
  \override BarNumber #'break-visibility = #end-of-line-invisible
} <<

\global {
  \bar "|"
  \clef treble
  \time 3/8
  d''4.

  \bar "|"
  \time 3/4
  r8 des''2( c''8)

  \bar "|"
  \time 7/8
  r4. ees''2 ~

  \bar "|"
  \time 2/4
  \tupletUp
  \times 2/3 { ees''4 r4 d''4 ~ }

```

```

\bar "|"
\time 3/8
\tupletUp
\times 3/4 { d''4 r4 }

\bar "|"
\time 2/4
e''2

\bar "|"
\time 3/8
es''4.

\bar "|"
\time 3/4
r8 d''2 r8
\bar "|"
}
>>

Percussion = \new StaffGroup <<
  \new Staff <<
    \global {
      \bar "|"
      \clef percussion
      \time 3/4
      r4 c'2 ~

      \bar "|"
      c'2.

      \bar "|"
      R2.

      \bar "|"
      r2 g'4 ~

      \bar "|"
      g'2. ~

      \bar "|"
      g'2.
    }
  >>
  \new Staff <<
    \global {
      \bar "|"
      \clef percussion
      \time 3/4
      R2.
    }
  >>

```


8

The musical score for Percussion consists of three staves. The top staff is in treble clef with a 3/4 time signature. It contains three measures: a quarter rest, a half note on the middle line (G4), and a quarter rest. The middle and bottom staves are in bass clef and contain two measures each. The first measure of the middle staff has a half note on the second line (F3) with a fermata. The first measure of the bottom staff has a half note on the second line (F2) with a fermata. The second measures of both the middle and bottom staves are empty. The score ends with a double bar line.

Fretted strings

Section “Fretted string instruments” in *Notation Reference*

Adding fingerings to a score

Fingering instructions can be entered using a simple syntax.

```
\relative c'' {
  c4-1 d-2 f-4 e-3
}
```

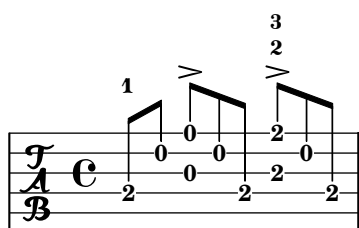


Adding fingerings to tablatures

To add fingerings to tablatures, use a combination of `\markup` and `\finger`.

```
one = \markup { \finger 1 }
two = \markup { \finger 2 }
threeTwo = \markup {
  \override #'(baseline-skip . 2)
  \column {
    \finger 3
    \finger 2
  }
}
threeFour = \markup {
  \override #'(baseline-skip . 2)
  \column {
    \finger 3
    \finger 4
  }
}

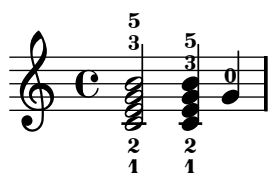
\score {
  \new TabStaff {
    \tabFullNotation
    \stemUp
    e8\4^\one b\2 <g\3 e'\1>^>[ b\2 e\4]
    <a\3 fis'\1>^>^\threeTwo[ b\2 e\4]
  }
}
```



Allowing fingerings to be printed inside the staff

By default, vertically oriented fingerings are positioned outside the staff. However, this behavior can be canceled. Note: you must use a chord construct <>, even if it is only a single note.

```
\relative c' {
  <c-1 e-2 g-3 b-5>2
  \override Fingering #'staff-padding = #'()
  <c-1 e-2 g-3 b-5>4 <g'-0>
}
```



Bar chords notation for Guitar (with Text Spanner)

Here is how to print bar chords, or half-bar chords (just uncomment the appropriate line for to select either one).

```
The syntax is \bbarre #'fret_number' { notes }
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% %%%%%%%%% Cut here ----- Start 'bbarred.ly'

%% PostScript -----
pScript= \markup {
  \with-dimensions #'(0 . 0.8) #'(0 . 2.0)
  \postscript #"
  0.15 setlinewidth
  /Times-Roman findfont
  2.0 scalefont
  setfont
  (C)show %%change with B if you prefer
  %(B)show %%change with C if you prefer
  stroke
  0.7 -0.5 moveto
  0.7 1.7 lineto
  stroke"
}

%% Span -----
%% Syntax: \bbarre #"text" { notes } - text = any number of box
bbarre= #(define-music-function (barre location str music) (string? ly:music?)
  (let ((spanned-music
    (let ((first-element #f)
      (last-element #f)
      (first-found? #f))
      (music-map (lambda (m)
        (if (eqv? (ly:music-property m 'name) 'EventChord)
          (begin
            (if (not first-found?)
              (begin
```

```

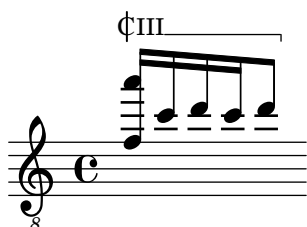
                                (set! first-found? #t)
                                (set! first-element m)))
                                (set! last-element m)))
                                m)
                                music)
    (if first-found?
        (begin
            (set! (ly:music-property first-element 'elements)
                (cons (make-music 'TextSpanEvent 'span-direction -1)
                    (ly:music-property first-element 'elements)))
            (set! (ly:music-property last-element 'elements)
                (cons (make-music 'TextSpanEvent 'span-direction 1)
                    (ly:music-property last-element 'elements)))))
        music)))
    (make-music 'SequentialMusic
        'origin location
        'elements (list #{
            \once \override TextSpanner #'font-size = #-2
            \once \override TextSpanner #'font-shape = #'upright
            \once \override TextSpanner #'staff-padding = #3
            \once \override TextSpanner #'style = #'line
            \once \override TextSpanner #'to-barline = ##f
            \once \override TextSpanner #'bound-details = #'((left (Y . 0) (pa
            \once \override TextSpanner #'bound-details #'right #'text = \mark
            \once \override TextSpanner #'bound-details #'left #'text = \mark

%% uncomment this line for make full barred
% \once \override TextSpanner #'bound-details #'left #'text = \marl
        #}
        spanned-music))))

%% %%%%%%%%% Cut here ----- End 'bbarred.ly'
%% Copy and change the last line for full barred. Rename in 'fbarred.ly'
%% %%%%%%%%%

%% Syntax: \bbarre #"text" { notes } - text = any number of box
\relative c' { \clef "G_8" \stemUp \bbarre #"III" { <f a'>16[ c' d c d8] } }

```



chordChanges for FretBoards

FretBoards can be set to display only when the chord changes or at the beginning of a new line.

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

```
myChords = \chordmode {
    c1 c1 \break
```

```

\set chordChanges = ##t
c1 c1 \break
c1 c1 \break
}

<<
\new ChordNames { \myChords }
\new FretBoards { \myChords }
\new Staff { \myChords }
>>

```

The image displays three musical staves, each representing a different fingering for a C major chord. Each staff begins with a treble clef, a common time signature 'C', and a C major chord symbol. Above each staff is a fretboard diagram for a C major chord. The first staff has a fingering of 3 2 1. The second staff has a fingering of 3 2 1. The third staff has a fingering of 5 3 2 1.

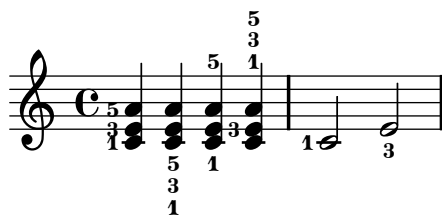
Controlling the placement of chord fingerings

The placement of fingering numbers can be controlled precisely. For fingering orientation to apply, you must use a chord construct `<>` even if it is a single note.

```

\relative c' {
  \set fingeringOrientations = #'(left)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(down)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(down right up)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(up)
  <c-1 e-3 a-5>4
  \set fingeringOrientations = #'(left)
  <c-1>2
  \set fingeringOrientations = #'(down)
  <e-3>2
}

```



Customizing fretboard fret diagrams

Fret diagram properties can be set through 'fret-diagram-details. For FretBoard fret diagrams, overrides are applied to the FretBoards.FretBoard object. Like Voice, FretBoards is a bottom level context, therefore can be omitted in property overrides.

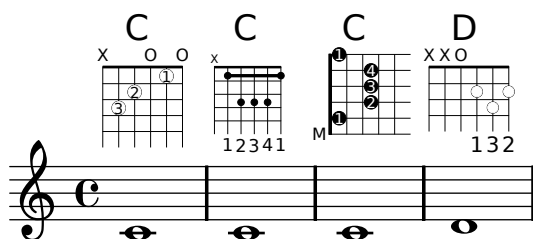
```
\include "predefined-guitar-fretboards.ly"
\storePredefinedDiagram \chordmode { c' }
                                #guitar-tuning
                                #"x;1-1-(;3-2;3-3;3-4;1-1-);"

<<
  \new ChordNames {
    \chordmode { c1 | c | c | d }
  }
  \new FretBoards {
    % Set global properties of fret diagram
    \override FretBoards.FretBoard #'size = #'1.2
    \override FretBoard
      #'(fret-diagram-details finger-code) = #'in-dot
    \override FretBoard
      #'(fret-diagram-details dot-color) = #'white
    \chordmode {
      c
      \once \override FretBoard #'size = #'1.0
      \once \override FretBoard
        #'(fret-diagram-details barre-type) = #'straight
      \once \override FretBoard
        #'(fret-diagram-details dot-color) = #'black
      \once \override FretBoard
        #'(fret-diagram-details finger-code) = #'below-string
      c'
      \once \override FretBoard
        #'(fret-diagram-details barre-type) = #'none
      \once \override FretBoard
        #'(fret-diagram-details number-type) = #'arabic
      \once \override FretBoard
        #'(fret-diagram-details orientation) = #'landscape
      \once \override FretBoard
        #'(fret-diagram-details mute-string) = #'"M"
      \once \override FretBoard
        #'(fret-diagram-details label-dir) = #LEFT
      \once \override FretBoard
        #'(fret-diagram-details dot-color) = #'black
```

```

c'
\once \override FretBoard
  #'(fret-diagram-details finger-code) = #'below-string
\once \override FretBoard
  #'(fret-diagram-details dot-radius) = #0.35
\once \override FretBoard
  #'(fret-diagram-details dot-position) = #0.5
\once \override FretBoard
  #'(fret-diagram-details fret-count) = #3
d
}
}
\new Voice {
  c'1 | c' | c' | d'
}
>>

```



Customizing markup fret diagrams

Fret diagram properties can be set through 'fret-diagram-details. For markup fret diagrams, overrides can be applied to the Voice.TextScript object or directly to the markup.

```

<<
\chords { c1 | c | c | d }

\new Voice = "mel" {
  \textLengthOn
  % Set global properties of fret diagram
  \override TextScript #'size = #'1.2
  \override TextScript
    #'(fret-diagram-details finger-code) = #'in-dot
  \override TextScript
    #'(fret-diagram-details dot-color) = #'white

  %% C major for guitar, no barre, using defaults
  % terse style
  c'1^\markup { \fret-diagram-terse #"x;3-3;2-2;o;1-1;o;" }

  %% C major for guitar, barred on third fret
  % verbose style
  % size 1.0
  % roman fret label, finger labels below string, straight barre
  c'1^\markup {
    % standard size
    \override #'(size . 1.0) {

```

```

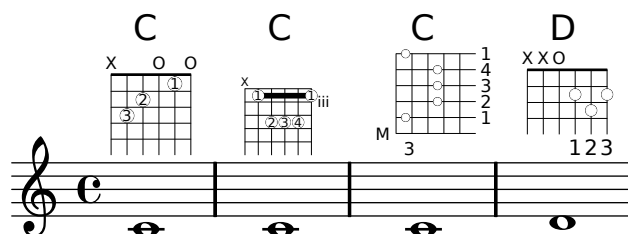
\override #'(fret-diagram-details . (
  (number-type . roman-lower)
  (finger-code . in-dot)
  (barre-type . straight))) {
  \fret-diagram-verbose #'((mute 6)
    (place-fret 5 3 1)
    (place-fret 4 5 2)
    (place-fret 3 5 3)
    (place-fret 2 5 4)
    (place-fret 1 3 1)
    (barre 5 1 3))
  }
}
}

%% C major for guitar, barred on third fret
% verbose style
% landscape orientation, arabic numbers, M for mute string
% no barre, fret label down or left, small mute label font
c'1^\markup {
  \override #'(fret-diagram-details . (
    (finger-code . below-string)
    (number-type . arabic)
    (label-dir . -1)
    (mute-string . "M")
    (orientation . landscape)
    (barre-type . none)
    (xo-font-magnification . 0.4)
    (xo-padding . 0.3))) {
    \fret-diagram-verbose #'((mute 6)
      (place-fret 5 3 1)
      (place-fret 4 5 2)
      (place-fret 3 5 3)
      (place-fret 2 5 4)
      (place-fret 1 3 1)
      (barre 5 1 3))
    }
  }
}

%% simple D chord
% terse style
% larger dots, centered dots, fewer frets
% label below string
d'1^\markup {
  \override #'(fret-diagram-details . (
    (finger-code . below-string)
    (dot-radius . 0.35)
    (dot-position . 0.5)
    (fret-count . 3))) {
    \fret-diagram-terse #"x;x;o;2-1;3-2;2-3;"
  }
}
}

```

```
}
>>
```



Defining predefined fretboards for other instruments

Predefined fret diagrams can be added for new instruments in addition to the standards used for guitar. This file shows how this is done by defining a new string-tuning and a few predefined fretboards for the Venezuelan cuatro.

This file also shows how fingerings can be included in the chords used as reference points for the chord lookup, and displayed in the fret diagram and the `TabStaff`, but not the music.

These fretboards are not transposable because they contain string information. This is planned to be corrected in the future.

```
% add FretBoards for the Cuatro
% Note: This section could be put into a separate file
% predefined-cuatro-fretboards.ly
% and \included into each of your compositions
```

```
cuatroTuning = #'(11 18 14 9)
```

```
dSix = { <a\4 b\1 d\3 fis\2> }
dMajor = { <a\4 d\1 d\3 fis \2> }
aMajSeven = { <a\4 cis\1 e\3 g\2> }
dMajSeven = { <a\4 c\1 d\3 fis\2> }
gMajor = { <b\4 b\1 d\3 g\2> }
```

```
\storePredefinedDiagram \dSix
    #cuatroTuning
    #"o;o;o;o;"
\storePredefinedDiagram \dMajor
    #cuatroTuning
    #"o;o;o;3-3;"
\storePredefinedDiagram \aMajSeven
    #cuatroTuning
    #"o;2-2;1-1;2-3;"
\storePredefinedDiagram \dMajSeven
    #cuatroTuning
    #"o;o;o;1-1;"
\storePredefinedDiagram \gMajor
    #cuatroTuning
    #"2-2;o;1-1;o;"
```

```
% end of potential include file /predefined-cuatro-fretboards.ly
```

```

#(set-global-staff-size 16)

primerosNames = \chordmode {
  d:6 d a:maj7 d:maj7
  g
}
primeros = {
  \dSix \dMajor \aMajSeven \dMajSeven
  \gMajor
}

\score {
  <<
    \new ChordNames {
      \set chordChanges = ##t
      \primerosNames
    }

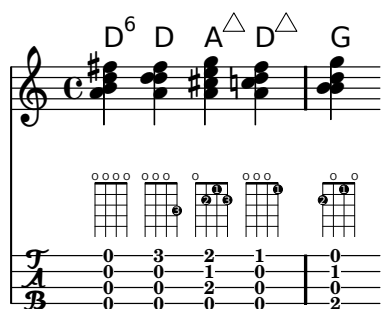
    \new Staff {
      \new Voice \with {
        \remove "New_fingering_engraver"
      }
      \relative c'' {
        \primeros
      }
    }

    \new FretBoards {
      \set stringTunings = #cuatroTuning
      \override FretBoard
        #'(fret-diagram-details string-count) = #'4
      \override FretBoard
        #'(fret-diagram-details finger-code) = #'in-dot
      \primeros
    }

    \new TabStaff \relative c'' {
      \set TabStaff.stringTunings = #cuatroTuning
      \primeros
    }
  >>

  \layout {
    \context {
      \Score
      \override SpacingSpanner
        #'base-shortest-duration = #(ly:make-moment 1 16)
    }
  }
  \midi { }
}

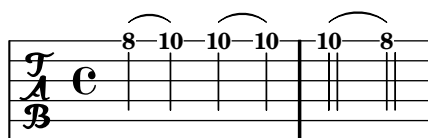
```



Faking a hammer in tablatures

A hammer in tablature can be faked with slurs.

```
\score {
  \new TabStaff {
    \relative c'' {
      \tabFullNotation
      c4( d) d( d)
      d2( c)
    }
  }
}
```

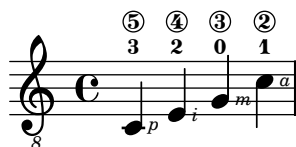


Fingerings string indications and right-hand fingerings

This example combines left-hand fingering, string indications, and right-hand fingering.

```
#(define RH rightHandFinger)
```

```
\relative c {
  \clef "treble_8"
  <c-3\5-\RH #1 >4
  <e-2\4-\RH #2 >4
  <g-0\3-\RH #3 >4
  <c-1\2-\RH #4 >4
}
```



Flamenco notation

For flamenco guitar, special notation is used:

- a golpe symbol to indicate a slap on the guitar body with the nail of the ring finger
- an arrow to indicate (the direction of) strokes
- different letters for fingering (“p”: thumb, “i”: index finger, “m”: middle finger, “a”: ring finger and “x”: little finger)


```

"}

strokeUpGolpe = \markup { \column { \golpe \line { \strokeUp } } }
iUpGolpe = \markup { \column { \golpe \line { \small i } \line { \strokeUp } } }

% Strokes for all fingers
pUp = \markup { \column { \small p \line { \strokeUp } } }
pDown = \markup { \column { \small p \line { \strokeDown } } }
iUp = \markup { \column { \small i \line { \strokeUp } } }
iDown = \markup { \column { \small i \line { \strokeDown } } }
mUp = \markup { \column { \small m \line { \strokeUp } } }
mDown = \markup { \column { \small m \line { \strokeDown } } }
aUp = \markup { \column { \small a \line { \strokeUp } } }
aDown = \markup { \column { \small a \line { \strokeDown } } }
xUp = \markup { \column { \small x \line { \strokeUp } } }
xDown = \markup { \column { \small x \line { \strokeDown } } }

% Just handy :)
tupletoff = {
  \once \override TupletNumber #'stencil = ##f
  \once \override TupletBracket #'stencil = ##f
}

tupletoffs = {
  \override TupletNumber #'stencil = ##f
  \override TupletBracket #'bracket-visibility = #'if-no-beam
}

tupleffson = {
  \override TupletBracket #'bracket-visibility = #'default
  \revert TupletNumber #'stencil
}

headsOff = {
  \override TabNoteHead #'transparent = ##t
  \override NoteHead #'transparent = ##t
  \override NoteHead #'no-ledgers = ##t
}

headsOn = {
  \override TabNoteHead #'transparent = ##f
  \override NoteHead #'transparent = ##f
  \override NoteHead #'no-ledgers = ##f
}

%%%%%%%% Cut here ----- End 'flamenco.ly'
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

part = \relative c' {
  <a, e' a cis e>8^iUp
  <a e' a cis e>8^iDown

```

```

r4
r2^\golpe

<a e' a cis e>8^\iUp
<a e' a cis e>8^\iDown
<a e' a cis e>8^\iUpGolpe
<a e' a cis e>8^\iDown
r2

<a e' a cis e>16^\aUp
\headsOff
<a e' a cis e>^\mUp
<a e' a cis e>^\iUp
<a e' a cis e>^\iDown~
\headsOn
<a e' a cis e>2
r4

\tupletOff
\times 4/5 {
  <a e' a cis e>16^\xUp
  \headsOff
  <a e' a cis e>^\aUp
  <a e' a cis e>^\mUp
  <a e' a cis e>^\iUp
  <a e' a cis e>^\iDown~
  \headsOn
}
<a e' a cis e>2
r4

\tupletsOff
\times 2/3 {
  <a e' a cis e>8^\pDown
  \headsOff
  <a e' a cis e>^\xUp
  <a e' a cis e>^\iUp
  \headsOn
}

\times 2/3 {
  <a e' a cis e>8^\pDown
  \headsOff
  <a e' a cis e>^\xUp
  <a e' a cis e>^\iUp
  \headsOn
}

\times 2/3 {
  <a e' a cis e>8^\pDown
  \headsOff
  <a e' a cis e>^\xUp

```

```

    <a e' a cis e>^\iUp
    \headsOn
  }
  \times 2/3 {
    <a e' a cis e>8^\pDown
    \headsOff
    <a e' a cis e>^\xUp
    <a e' a cis e>^\iUp
    \headsOn
  }

  \tupletsOff
  \override Beam #'positions = #'(2 . 2)
  \times 2/3 {
    a8^\markup{ \small p }
    <e' a>^\strokeUpGolpe
    <e a>^\strokeDown
  }
  \times 2/3 {
    a,8^\markup{ \small p }
    <e' a>^\strokeUpGolpe
    <e a>^\strokeDown
  }
  \times 2/3 {
    a,8^\markup{ \small p }
    <e' a>^\strokeUpGolpe
    <e a>^\strokeDown
  }
  \times 2/3 {
    a,8^\markup{ \small p }
    <e' a>^\strokeUpGolpe
    <e a>^\strokeDown
  }
  \tupletsOn

  \once \override TextScript #'extra-offset = #'(0 . -1)
  <g, b f'>1_\golpe^\mUp
  \bar "|."
}

\score {
  \new StaffGroup <<
    \context Staff = "part" <<
      \clef G
      \transpose c c'
      {
        \part
      }
    >>
    \context TabStaff {
      \part
    }
  }

```

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

Fret diagrams explained and developed

This snippet shows many possibilities for obtaining and tweaking fret diagrams.

```
<<
\chords {
  a2 a
  \repeat unfold 3 {
    c c c d d
  }
}

\new Voice = "mel" {
  \textLengthOn
  % Set global properties of fret diagram
  \override TextScript #'size = #1.2
  \override TextScript
    #'fret-diagram-details #'finger-code = #'below-string
  \override TextScript #'fret-diagram-details #'dot-color = #'black

  %% A chord for ukulele
  a'2^\markup {
    \override #'(fret-diagram-details . (
      (string-count . 4)
      (dot-color . white)
    ))
  }
}
```

```

        (finger-code . in-dot))) {
      \fret-diagram #"4-2-2;3-1-1;2-o;1-o;"
    }
  }

%% A chord for ukulele, with formatting defined in definition string
% 1.2 * size, 4 strings, 4 frets, fingerings below string
% dot radius .35 of fret spacing, dot position 0.55 of fret spacing
a'2^\markup {
  \override #'(fret-diagram-details . (
    (dot-color . white)
    (open-string . "o"))) {
    \fret-diagram #"s:1.2;w:4;h:3;f:2;d:0.35;p:0.55;4-2-2;3-1-1;2-o;1-o;"
  }
}

%% These chords will be in normal orientation

%% C major for guitar, barred on third fret
% verbose style
% roman fret label, finger labels below string, straight barre
c'2^\markup {
  % 110% of default size
  \override #'(size . 1.1) {
    \override #'(fret-diagram-details . (
      (number-type . roman-lower)
      (finger-code . below-string)
      (barre-type . straight))) {
      \fret-diagram-verbose #'((mute 6)
        (place-fret 5 3 1)
        (place-fret 4 5 2)
        (place-fret 3 5 3)
        (place-fret 2 5 4)
        (place-fret 1 3 1)
        (barre 5 1 3))
    }
  }
}

%% C major for guitar, barred on third fret
%% Double barre used to test barre function
% verbose style
c'2^\markup {
  % 110% of default size
  \override #'(size . 1.1) {
    \override #'(fret-diagram-details . (
      (number-type . arabic)
      (dot-label-font-mag . 0.9)
      (finger-code . in-dot)
      (fret-label-font-mag . 0.6)
      (fret-label-vertical-offset . 0)
      (label-dir . -1)
    )
  }
}

```

```

        (mute-string . "M")
        (xo-font-magnification . 0.4)
        (xo-padding . 0.3))) {
\ fret-diagram-verbose #'((mute 6)
                          (place-fret 5 3 1)
                          (place-fret 4 5 2)
                          (place-fret 3 5 3)
                          (place-fret 2 5 4)
                          (place-fret 1 3 1)
                          (barre 4 2 5)
                          (barre 5 1 3))
    }
  }
}

%% C major for guitar, with capo on third fret
% verbose style
c'2^\markup {
  % 110% of default size
  \override #'(size . 1.1) {
    \override #'(fret-diagram-details . (
      (number-type . roman-upper)
      (dot-label-font-mag . 0.9)
      (finger-code . none)
      (fret-label-vertical-offset . 0.5)
      (xo-font-magnification . 0.4)
      (xo-padding . 0.3))) {
      \ fret-diagram-verbose #'((mute 6)
                                (capo 3)
                                (open 5)
                                (place-fret 4 5 1)
                                (place-fret 3 5 2)
                                (place-fret 2 5 3)
                                (open 1))
    }
  }
}

%% simple D chord
d'2^\markup {
  \override #'(fret-diagram-details . (
    (finger-code . below-string)
    (dot-radius . 0.35)
    (string-thickness-factor . 0.3)
    (dot-position . 0.5)
    (fret-count . 3))) {
    \ fret-diagram-terse #"x;x;o;2-1;3-2;2-3;"
  }
}

%% simple D chord, large top fret thickness
d'2^\markup {

```

```

\override #'(fret-diagram-details . (
  (finger-code . below-string)
  (dot-radius . 0.35)
  (dot-position . 0.5)
  (top-fret-thickness . 7)
  (fret-count . 3))) {
  \fret-diagram-terse #"x;x;o;2-1;3-2;2-3;"
}
}

% These chords will be in landscape orientation
\override TextScript
  #'fret-diagram-details #'orientation = #'landscape

%% C major for guitar, barred on third fret
% verbose style
% roman fret label, finger labels below string, straight barre
c'2^\markup {
  % 110% of default size
  \override #'(size . 1.1) {
    \override #'(fret-diagram-details . (
      (number-type . roman-lower)
      (finger-code . below-string)
      (barre-type . straight))) {
      \fret-diagram-verbose #'((mute 6)
        (place-fret 5 3 1)
        (place-fret 4 5 2)
        (place-fret 3 5 3)
        (place-fret 2 5 4)
        (place-fret 1 3 1)
        (barre 5 1 3))
    }
  }
}

%% C major for guitar, barred on third fret
%% Double barre used to test barre function
% verbose style
c'2^\markup {
  % 110% of default size
  \override #'(size . 1.1) {
    \override #'(fret-diagram-details . (
      (number-type . arabic)
      (dot-label-font-mag . 0.9)
      (finger-code . in-dot)
      (fret-label-font-mag . 0.6)
      (fret-label-vertical-offset . 0)
      (label-dir . -1)
      (mute-string . "M")
      (xo-font-magnification . 0.4)
      (xo-padding . 0.3))) {
      \fret-diagram-verbose #'((mute 6)

```

```

        (place-fret 5 3 1)
        (place-fret 4 5 2)
        (place-fret 3 5 3)
        (place-fret 2 5 4)
        (place-fret 1 3 1)
        (barre 4 2 5)
        (barre 5 1 3))
    }
  }
}

%% C major for guitar, with capo on third fret
% verbose style
c'2^\markup {
  % 110% of default size
  \override #'(size . 1.1) {
    \override #'(fret-diagram-details . (
      (number-type . roman-upper)
      (dot-label-font-mag . 0.9)
      (finger-code . none)
      (fret-label-vertical-offset . 0.5)
      (xo-font-magnification . 0.4)
      (xo-padding . 0.3))) {
      \fret-diagram-verbose #'((mute 6)
        (capo 3)
        (open 5)
        (place-fret 4 5 1)
        (place-fret 3 5 2)
        (place-fret 2 5 3)
        (open 1))
    }
  }
}

%% simple D chord
d'2^\markup {
  \override #'(fret-diagram-details . (
    (finger-code . below-string)
    (dot-radius . 0.35)
    (dot-position . 0.5)
    (fret-count . 3))) {
    \fret-diagram-terse #"x;x;o;2-1;3-2;2-3;"
  }
}

%% simple D chord, large top fret thickness
d'2^\markup {
  \override #'(fret-diagram-details . (
    (finger-code . below-string)
    (dot-radius . 0.35)
    (dot-position . 0.5)
    (top-fret-thickness . 7)

```

```

        (fret-count . 3))) {
    \fret-diagram-terse #"x;x;o;2-1;3-2;2-3;"
  }
}

% These chords will be in opposing-landscape orientation
\override TextScript #'fret-diagram-details
    #'orientation = #'opposing-landscape

%% C major for guitar, barred on third fret
% verbose style
% roman fret label, finger labels below string, straight barre
c'2^\markup {
  % 110% of default size
  \override #'(size . 1.1) {
    \override #'(fret-diagram-details . (
      (number-type . roman-lower)
      (finger-code . below-string)
      (barre-type . straight))) {
      \fret-diagram-verbose #'((mute 6)
        (place-fret 5 3 1)
        (place-fret 4 5 2)
        (place-fret 3 5 3)
        (place-fret 2 5 4)
        (place-fret 1 3 1)
        (barre 5 1 3))
    }
  }
}

%% C major for guitar, barred on third fret
%% Double barre used to test barre function
% verbose style
c'2^\markup {
  % 110% of default size
  \override #'(size . 1.1) {
    \override #'(fret-diagram-details . (
      (number-type . arabic)
      (dot-label-font-mag . 0.9)
      (finger-code . in-dot)
      (fret-label-font-mag . 0.6)
      (fret-label-vertical-offset . 0)
      (label-dir . -1)
      (mute-string . "M")
      (xo-font-magnification . 0.4)
      (xo-padding . 0.3))) {
      \fret-diagram-verbose #'((mute 6)
        (place-fret 5 3 1)
        (place-fret 4 5 2)
        (place-fret 3 5 3)
        (place-fret 2 5 4)
        (place-fret 1 3 1)

```

```

                                (barre 4 2 5)
                                (barre 5 1 3))
      }
    }
  }

%% C major for guitar, with capo on third fret
% verbose style
c'2^\markup {
  % 110% of default size
  \override #'(size . 1.1) {
    \override #'(fret-diagram-details . (
      (number-type . roman-upper)
      (dot-label-font-mag . 0.9)
      (finger-code . none)
      (fret-label-vertical-offset . 0.5)
      (xo-font-magnification . 0.4)
      (xo-padding . 0.3))) {
      \fret-diagram-verbose #'((mute 6)
        (capo 3)
        (open 5)
        (place-fret 4 5 1)
        (place-fret 3 5 2)
        (place-fret 2 5 3)
        (open 1))
    }
  }
}

%% simple D chord
d'2^\markup {
  \override #'(fret-diagram-details . (
    (finger-code . below-string)
    (dot-radius . 0.35)
    (dot-position . 0.5)
    (fret-count . 3))) {
    \fret-diagram-terse #"x;x;o;2-1;3-2;2-3;"
  }
}

%% simple D chord, large top fret thickness
d'2^\markup {
  \override #'(fret-diagram-details . (
    (finger-code . below-string)
    (dot-radius . 0.35)
    (dot-position . 0.5)
    (top-fret-thickness . 7)
    (fret-count . 3))) {
    \fret-diagram-terse #"x;x;o;2-1;3-2;2-3;"
  }
}

```

>>

Guitar slides

Unlike glissandos, slides may go from an imprecise point of the fretboard to a specific fret. A good way to do that is to add a grace hidden note before the note which is actually played, as demonstrated in the following example.

```
%% Hide fret number: useful to draw slide into/from a casual point of
%% the fretboard.
```

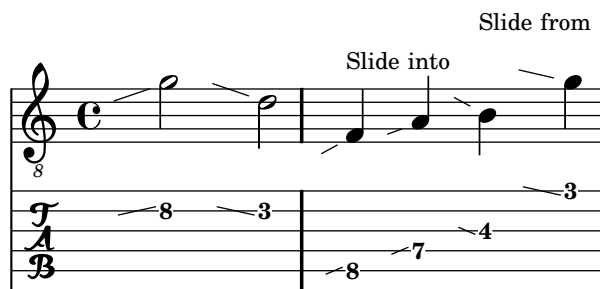
```
hideFretNumber = {
  \once \override TabNoteHead #'transparent = ##t
  \once \override NoteHead #'transparent = ##t
  \once \override Stem #'transparent = ##t
  \once \override NoteHead #'no-ledgers = ##t
  \once \override Glissando #'(bound-details left padding) = #0.3
}

music= \relative c' {
  \grace { \hideFretNumber d8\2 \glissando s2 } g2\2
  \grace { \hideFretNumber g8\2 \glissando s2 } d2 |

  \grace { \hideFretNumber c,8 \glissando s } f4\5^\markup \tiny { Slide into }
  \grace { \hideFretNumber f8 \glissando s } a4\4
  \grace { \hideFretNumber e'8\3 \glissando s } b4\3^\markup \tiny { Slide from }
  \grace { \hideFretNumber b'8 \glissando s2 } g4 |
}

\score {
  <<
    \new Staff {
      \clef "G_8"
      \music
    }
    \new TabStaff {
      \music
    }
  >>
}
```

}



Guitar strum rhythms

For guitar music, it is possible to show strum rhythms, along with melody notes, chord names and fret diagrams.

```
\include "predefined-guitar-fretboards.ly"
<<
\new ChordNames {
  \chordmode {
    c1 | f | g | c
  }
}
\new FretBoards {
  \chordmode {
    c1 | f | g | c
  }
}
\new Voice \with {
  \consists "Pitch_squash_engraver"
} {
  \relative c'' {
    \improvisationOn
    c4 c8 c c4 c8 c
    f4 f8 f f4 f8 f
    g4 g8 g g4 g8 g
    c4 c8 c c4 c8 c
  }
}
\new Voice = "melody" {
  \relative c'' {
    c2 e4 e4
    f2. r4
    g2. a4
    e4 c2.
  }
}
\new Lyrics {
  \lyricsto "melody" {
    This is my song.
    I like to sing.
  }
}
```

>>

The image shows a musical score for a song. The first staff is a guitar-like accompaniment, and the second staff is a vocal line. The lyrics are "This is my song. I like to sing." Above the first staff, there are four fret diagrams for chords C, F, G, and C. The fret diagrams are positioned above the first, second, third, and fourth measures of the first staff. The fret diagrams for C are at measures 1 and 4, and the fret diagrams for F and G are at measures 2 and 3. The fret diagrams for C are at measures 1 and 4, and the fret diagrams for F and G are at measures 2 and 3. The fret diagrams for C are at measures 1 and 4, and the fret diagrams for F and G are at measures 2 and 3. The fret diagrams for C are at measures 1 and 4, and the fret diagrams for F and G are at measures 2 and 3.

How to change fret diagram position

If you want to move the position of a fret diagram, for example, to avoid collision, or to place it between two notes, you have various possibilities:

- 1) modify #'padding or #'extra-offset values (as shown in the first snippet)
- 2) you can add an invisible voice and attach the fret diagrams to the invisible notes in that voice (as shown in the second example).

If you need to move the fret according with a rhythmic position inside the bar (in the example, the third beat of the measure) the second example is better, because the fret is aligned with the third beat itself.

```

harmonies = \chordmode
{
  a8:13
  % THE FOLLOWING IS THE COMMAND TO MOVE THE CHORD NAME
  \once \override ChordNames.ChordName #'extra-offset = #'(10 . 0)
  b8:13 s2.
  % THIS LINE IS THE SECOND METHOD
  s4 s4 b4:13
}

\score
{
  <<
    \context ChordNames \harmonies
    \context Staff
    {a8^\markup { \fret-diagram #"6-x;5-0;4-2;3-0;2-0;1-2;" }
  % THE FOLLOWING IS THE COMMAND TO MOVE THE FRET DIAGRAM
  \once \override TextScript #'extra-offset = #'(10 . 0)
  b4.~^\markup { \fret-diagram #"6-x;5-2;4-4;3-2;2-2;1-4;" } b4. a8\break
  % HERE IS THE SECOND METHOD
  <<
    { a8 b4.~ b4. a8}
    { s4 s4 s4^\markup { \fret-diagram #"6-x;5-2;4-4;3-2;2-2;1-4;" }
    }
  >>
  }
  >>

```

}

Jazz combo template

This is quite an advanced template, for a jazz ensemble. Note that all instruments are notated in `\key c \major`. This refers to the key in concert pitch; the key will be automatically transposed if the music is within a `\transpose` section.

```
\header {
  title = "Song"
  subtitle = "(tune)"
  composer = "Me"
  meter = "moderato"
  piece = "Swing"
  tagline = \markup {
    \column {
      "LilyPond example file by Amelie Zapf,"
      "Berlin 07/07/2003"
    }
  }
}
```

```
%#(set-global-staff-size 16)
\include "english.ly"
```

```
%%%%%%%%%% Some macros %%%%%%%%%%
```

```
sl = {
  \override NoteHead #'style = #'slash
  \override Stem #'transparent = ##t
}
nsl = {
  \revert NoteHead #'style
  \revert Stem #'transparent
```

```

}
crOn = \override NoteHead #'style = #'cross
crOff = \revert NoteHead #'style

%% insert chord name style stuff here.

jazzChords = { }

%%%%%%%%%%%% Keys'n'things %%%%%%%%%%%%%%

global = { \time 4/4 }

Key = { \key c \major }

% ##### Horns #####

% ----- Trumpet -----
trpt = \transpose c d \relative c' {
  \Key
  c1 | c | c |
}
trpHarmony = \transpose c' d {
  \jazzChords
}
trumpet = {
  \global
  \set Staff.instrumentName = #"Trumpet"
  \clef treble
  <<
    \trpt
  >>
}

% ----- Alto Saxophone -----
alto = \transpose c a \relative c' {
  \Key
  c1 | c | c |
}
altoHarmony = \transpose c' a {
  \jazzChords
}
altoSax = {
  \global
  \set Staff.instrumentName = #"Alto Sax"
  \clef treble
  <<
    \alto
  >>
}

% ----- Baritone Saxophone -----
bari = \transpose c a' \relative c {

```

```

\Key
c1
c1
\s1
d4^"Solo" d d d
\ns1
}
bariHarmony = \transpose c' a \chordmode {
  \jazzChords s1 s d2:maj e:m7
}
bariSax = {
  \global
  \set Staff.instrumentName = #"Bari Sax"
  \clef treble
  <<
    \bari
  >>
}

% ----- Trombone -----
tbone = \relative c {
  \Key
  c1 | c | c
}
tboneHarmony = \chordmode {
  \jazzChords
}
trombone = {
  \global
  \set Staff.instrumentName = #"Trombone"
  \clef bass
  <<
    \tbone
  >>
}

% ##### Rhythm Section #####

% ----- Guitar -----
gtr = \relative c'' {
  \Key
  c1
  \s1
  b4 b b b
  \ns1
  c1
}
gtrHarmony = \chordmode {
  \jazzChords
  s1 c2:min7+ d2:maj9
}
guitar = {

```

```

\global
\set Staff.instrumentName = #"Guitar"
\clef treble
<<
  \gtr
>>
}

%% ----- Piano -----
rhUpper = \relative c' {
  \voiceOne
  \Key
  c1 | c | c
}
rhLower = \relative c {
  \voiceTwo
  \Key
  e1 | e | e
}

lhUpper = \relative c' {
  \voiceOne
  \Key
  g1 | g | g
}
lhLower = \relative c {
  \voiceTwo
  \Key
  c1 | c | c
}

PianoRH = {
  \clef treble
  \global
  \set Staff.midiInstrument = #"acoustic grand"
  <<
    \new Voice = "one" \rhUpper
    \new Voice = "two" \rhLower
  >>
}

PianoLH = {
  \clef bass
  \global
  \set Staff.midiInstrument = #"acoustic grand"
  <<
    \new Voice = "one" \lhUpper
    \new Voice = "two" \lhLower
  >>
}

piano = {
  <<

```

```

        \set PianoStaff.instrumentName = #"Piano"
        \new Staff = "upper" \PianoRH
        \new Staff = "lower" \PianoLH
    >>
}

% ----- Bass Guitar -----
Bass = \relative c {
    \Key
    c1 | c | c
}
bass = {
    \global
    \set Staff.instrumentName = #"Bass"
    \clef bass
    <<
        \Bass
    >>
}

% ----- Drums -----
up = \drummode {
    \voiceOne
    hh4 <hh sn> hh <hh sn>
    hh4 <hh sn> hh <hh sn>
    hh4 <hh sn> hh <hh sn>
}
down = \drummode {
    \voiceTwo
    bd4 s bd s
    bd4 s bd s
    bd4 s bd s
}

drumContents = {
    \global
    <<
        \set DrumStaff.instrumentName = #"Drums"
        \new DrumVoice \up
        \new DrumVoice \down
    >>
}

%%%%%%%%%% It All Goes Together Here %%%%%%%%%%%

\score {
    <<
        \new StaffGroup = "horns" <<
            \new Staff = "trumpet" \trumpet
            \new Staff = "altosax" \altoSax
            \new ChordNames = "barichords" \bariHarmony
            \new Staff = "barisax" \bariSax

```

```

\new Staff = "trombone" \trombone
>>

\new StaffGroup = "rhythm" <<
  \new ChordNames = "chords" \gtrHarmony
  \new Staff = "guitar" \guitar
  \new PianoStaff = "piano" \piano
  \new Staff = "bass" \bass
  \new DrumStaff \drumContents
>>
>>
\layout {
  \context { \Staff \RemoveEmptyStaves }
  \context {
    \Score
    \override BarNumber #'padding = #3
    \override RehearsalMark #'padding = #2
    skipBars = ##t
  }
}
\midi { }
}

```

Song

(tune)

moderato

Swing

Me

Laissez vibrer ties

```
\relative c' {
  <c e g>4\laissezVibrer r <c f g>\laissezVibrer r
  <c d f g>4\laissezVibrer r <c d f g>4.\laissezVibrer r8

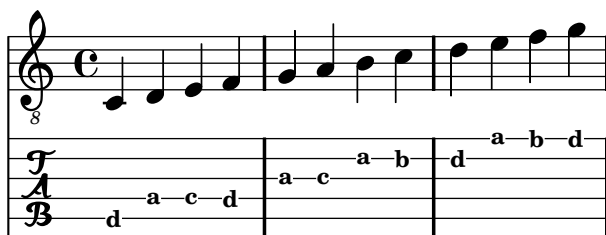
  <c d e f>4\laissezVibrer r
  \override LaissezVibrerTieColumn #'tie-configuration
    = #`((-7 . ,DOWN)
      (-5 . ,DOWN)
      (-3 . ,UP)
      (-1 . ,UP))
  <c d e f>4\laissezVibrer r
}
```

Letter tablature formatting

Tablature can be formatted using letters instead of numbers.

```
music = \relative c {
  c4 d e f
  g4 a b c
  d4 e f g
}

<<
\new Staff {
  \clef "G_8"
  \music
}
\new TabStaff \with {
  tablatureFormat = #fret-letter-tablature-format
}
{
  \music
}
>>
```



Modern TAB text clef

Use a markup text to replace the (TAB) clef glyph with a modern font.

```
\markup {
  This snippet is deprecated as of version 2.13.4 and will be removed
  in version 2.14.
}
```

This snippet is deprecated as of version 2.13.4 and will be removed in version 2.14.

Placement of right-hand fingerings

It is possible to exercise greater control over the placement of right-hand fingerings by setting a specific property, as demonstrated in the following example. Note: you must use a chord construct

```
#(define RH rightHandFinger)

\relative c {
  \clef "treble_8"

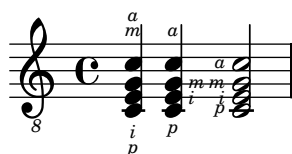
  \set strokeFingerOrientations = #'(up down)
  <c-\RH #1 e-\RH #2 g-\RH #3 c-\RH #4 >4
```

```

\set strokeFingerOrientations = #'(up right down)
<c-\RH #1 e-\RH #2 g-\RH #3 c-\RH #4 >4

\set strokeFingerOrientations = #'(left)
<c-\RH #1 e-\RH #2 g-\RH #3 c-\RH #4 >2
}

```



Polyphony in tablature

Polyphony is created the same way in a `TabStaff` as in a regular staff.

```

upper = \relative c' {
  \time 12/8
  \key e \minor
  \voiceOne
  r4. r8 e, fis g16 b g e e' b c b a g fis e
}

lower = \relative c {
  \key e \minor
  \voiceTwo
  r16 e d c b a g4 fis8 e fis g a b c
}

\score {
  <<
  \new StaffGroup = "tab with traditional" <<
    \new Staff = "guitar traditional" <<
      \clef "treble_8"
      \context Voice = "upper" \upper
      \context Voice = "lower" \lower
    >>
  \new TabStaff = "guitar tab" <<
    \context TabVoice = "upper" \upper
    \context TabVoice = "lower" \lower
  >>
  >>
  >>
}

```

Stem and beam behavior in tablature

The direction of stems is controlled the same way in tablature as in traditional notation. Beams can be made horizontal, as shown in this example.

```
\new TabStaff {
  \relative c {
    \tabFullNotation
    g16 b d g b d g b
    \stemDown
    \override Beam #'damping = #+inf.0
    g,,16 b d g b d g b
  }
}
```

String number extender lines

Make an extender line for string number indications, showing that a series of notes is supposed to be played all on the same string.

```
stringNumberSpanner =
#(define-music-function (parser location StringNumber) (string?)
  #{
    \override TextSpanner #'style = #'solid
    \override TextSpanner #'font-size = #-5
    \override TextSpanner #'(bound-details left stencil-align-dir-y) = #CENTER
    \override TextSpanner #'(bound-details left text) = \markup { \circle \number $StringNumber }
  #})

\relative c {
  \clef "treble_8"
  \stringNumberSpanner "5"
  \textSpannerDown
  a8\startTextSpan
  b c d e f\stopTextSpan
  \stringNumberSpanner "4"
  g\startTextSpan a
  bes4 a g2\stopTextSpan
}
```


Unfretted strings

Section “Unfretted string instruments” in *Notation Reference*

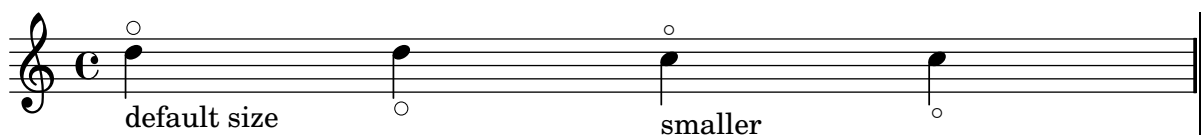
Changing \flageolet mark size

To make the \flageolet circle smaller use the following Scheme function.

```
smallFlageolet =
#(let ((m (make-articulation "flageolet")))
  (set! (ly:music-property m 'tweaks)
    (acons 'font-size -3
      (ly:music-property m 'tweaks)))
  m)

\layout { ragged-right = ##f }

\relative c' {
  d4~\flageolet_\markup { default size } d_\flageolet
  c4~\smallFlageolet_\markup { smaller } c_\smallFlageolet
}
```



Creating slurs across voices

In some situations, it may be necessary to create slurs between notes from different voices.

The solution is to add invisible notes to one of the voices, using \hideNotes.

This example is measure 235 of the Ciaccona from Bach’s 2nd Partita for solo violin, BWV 1004.

```
\relative c' {
  <<
  {
    d16( a') s a s a[ s a] s a[ s a]
  }
  \\\
  {
    \slurUp
    bes,16[ s e](
    \hideNotes a)
    \unHideNotes f[(
    \hideNotes a)
    \unHideNotes fis](
    \hideNotes a)
    \unHideNotes g[(
    \hideNotes a)
    \unHideNotes gis](
    \hideNotes a)
  }
}
```

```
>>
}
```



Dotted harmonics

Artificial harmonics using `\harmonic` do not show dots. To override this behavior, set the context property `harmonicDots`.

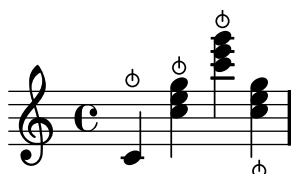
```
\relative c' {} {
  \time 3/4
  \key f \major
  \set harmonicDots = ##t
  <bes f'\harmonic>2. ~
  <bes f'\harmonic>4. <a e'\harmonic>8( <gis dis'\harmonic> <g d'\harmonic>)
  <fis cis'\harmonic>2.
  <bes f'\harmonic>2.
}
```



Snap-pizzicato ("Bartok pizzicato")

A snap-pizzicato (also known as “Bartok pizzicato”) is a “strong pizzicato where the string is plucked vertically by snapping and rebounds off the fingerboard of the instrument” (Wikipedia). It is denoted by a circle with a vertical line going from the center upwards outside the circle.

```
\relative c' {} {
  c4\snappizzicato
  <c' e g>4\snappizzicato
  <c' e g>4^\snappizzicato
  <c, e g>4_\snappizzicato
}
```



String quartet template (simple)

This template demonstrates a simple string quartet. It also uses a `\global` section for time and key signatures

```

global= {
  \time 4/4
  \key c \major
}

violinOne = \new Voice \relative c' {
  \set Staff.instrumentName = #"Violin 1 "

  c2 d
  e1

  \bar "|."
}

violinTwo = \new Voice \relative c' {
  \set Staff.instrumentName = #"Violin 2 "

  g2 f
  e1

  \bar "|."
}

viola = \new Voice \relative c' {
  \set Staff.instrumentName = #"Viola "
  \clef alto

  e2 d
  c1

  \bar "|."
}

cello = \new Voice \relative c' {
  \set Staff.instrumentName = #"Cello "
  \clef bass

  c2 b
  a1

  \bar "|."
}

\score {
  \new StaffGroup <<
    \new Staff << \global \violinOne >>
    \new Staff << \global \violinTwo >>
    \new Staff << \global \viola >>
    \new Staff << \global \cello >>
  >>
  \layout { }
  \midi { }
}

```

}

String quartet template with separate parts

The “String quartet template” snippet produces a nice string quartet, but what if you needed to print parts? This new template demonstrates how to use the `\tag` feature to easily split a piece into individual parts.

You need to split this template into separate files; the filenames are contained in comments at the beginning of each file. `piece.ly` contains all the music definitions. The other files – `score.ly`, `vn1.ly`, `vn2.ly`, `vla.ly`, and `vlc.ly` – produce the appropriate part.

Do not forget to remove specified comments when using separate files!

```

%% piece.ly
%% (This is the global definitions file)

global= {
  \time 4/4
  \key c \major
}

Violinone = \new Voice { \relative c' {
  \set Staff.instrumentName = #"Violin 1 "

  c2 d e1

  \bar "|" } } %*****
Violintwo = \new Voice { \relative c' {
  \set Staff.instrumentName = #"Violin 2 "

  g2 f e1

  \bar "|" } } %*****
Viola = \new Voice { \relative c' {
  \set Staff.instrumentName = #"Viola "
  \clef alto

  e2 d c1

```

```
\bar "|" } } %*****
Cello = \new Voice { \relative c' {
  \set Staff.instrumentName = #"Cello "
  \clef bass
```

```
c2 b a1
```

```
\bar "|" } } %*****
```

```
music = {
  <<
    \tag #'score \tag #'vn1 \new Staff { << \global \Violinone >> }
    \tag #'score \tag #'vn2 \new Staff { << \global \Violintwo>> }
    \tag #'score \tag #'vla \new Staff { << \global \Viola>> }
    \tag #'score \tag #'vlc \new Staff { << \global \Cello>> }
  >>
}
```

```
%%% These are the other files you need to save on your computer
```

```
%%%%% score.ly
%%%%% (This is the main file)
```

```
%\include "piece.ly" %%% uncomment this line when using a separate file
#(set-global-staff-size 14)
\score {
  \new StaffGroup \keepWithTag #'score \music
  \layout { }
  \midi { }
}
```

```
%{ Uncomment this block when using separate files
```

```
%%%%% vn1.ly
%%%%% (This is the Violin 1 part file)
```

```
\include "piece.ly"
\score {
  \keepWithTag #'vn1 \music
  \layout { }
}
```

```
%%%%% vn2.ly
%%%%% (This is the Violin 2 part file)
```

```
\include "piece.ly"
\score {
  \keepWithTag #'vn2 \music
  \layout { }
}
```

```
}
```

```
%%%%% vla.ly  
%%%%% (This is the Viola part file)
```

```
\include "piece.ly"  
\score {  
  \keepWithTag #'vla \music  
  \layout { }  
}
```

```
%%%%% vlc.ly  
%%%%% (This is the Cello part file)
```

```
\include "piece.ly"  
\score {  
  \keepWithTag #'vlc \music  
  \layout { }  
}
```


```
%}
```

Violin 1

Violin 2

Viola

Cello



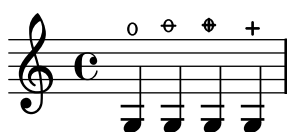
Winds

Section “Wind instruments” in *Notation Reference*

Fingering symbols for wind instruments

Special symbols can be achieved by combining existing glyphs, which is useful for wind instruments.

```
centermarkup = {
  \once \override TextScript #'self-alignment-X = #CENTER
  \once \override TextScript #'X-offset = #(ly:make-simple-closure
    `(+
      ,(ly:make-simple-closure (list
        ly:self-alignment-interface::centered-on-x-parent))
      ,(ly:make-simple-closure (list
        ly:self-alignment-interface::x-aligned-on-self))))
}
\score
{\relative c'
  {
    g\open
    \once \override TextScript #'staff-padding = #-1.0 \centermarkup
    g^\markup{\combine \musicglyph #"scripts.open" \musicglyph
      #"scripts.tenuto"}
    \centermarkup g^\markup{\combine \musicglyph #"scripts.open"
      \musicglyph #"scripts.stopped"}
    g\stopped
  }
}
```



Flute slap notation

It is possible to indicate special articulation techniques such as a flute “tongue slap” by replacing the note head with the appropriate glyph.

```
slap =
#(define-music-function (parser location music) (ly:music?)
#{
  \override NoteHead #'stencil =
    #(lambda (grob)
      (grob-interpret-markup grob
        (markup #:musicglyph "scripts.sforzato")))
  \override NoteHead #'stem-attachment =
    #(lambda (grob)
      (let ((thickness (ly:staff-symbol-line-thickness grob)))
        (cons 1 (/ thickness 2))))
  $music
```

```

\revert NoteHead #'stencil
\revert NoteHead #'stem-attachment
#})

```

```

\relative c' {
  c4 \slap c d r \slap { g a } b r
}

```



Recorder fingering chart

The following example demonstrates how fingering charts for wind instruments can be realized.

% range chart for paetzold contrabass recorder

```

centermarkup = {
  \once \override TextScript #'self-alignment-X = #CENTER
  \once \override TextScript #'X-offset = #(ly:make-simple-closure
    `(+
      ,(ly:make-simple-closure (list
        ly:self-alignment-interface::centered-on-x-parent))
      ,(ly:make-simple-closure (list
        ly:self-alignment-interface::x-aligned-on-self))))
}

\score {
  \new Staff \with {
    \remove "Time_signature_engraver"
    \override Stem #'stencil = ##f
    \consists "Horizontal_bracket_engraver"
  }
  {
    \clef bass
    \set Score.timing = ##f
    f,1*1/4 \glissando
    \clef violin
    gis'1*1/4
    \stemDown a'4^\markup{1)}
    \centermarkup
    \once \override TextScript #'padding = #2
    bes'1*1/4_\markup{\override #'(baseline-skip . 1.7) \column
      { \fontsize #-5 \slashed-digit #0 \finger 1 \finger 2 \finger 3 \finger 4
        \finger 5 \finger 6 \finger 7} }
    b'1*1/4
    c''4^\markup{1)}
    \centermarkup
    \once \override TextScript #'padding = #2
    cis''1*1/4
    deh''1*1/4
  }
}

```

```

\centermarkup
\once \override TextScript #'padding = #2
\once \override Staff.HorizontalBracket #'direction = #UP
e''1*1/4\_markup{\override #'(baseline-skip . 1.7) \column
  { \fontsize #-5 \slashed-digit #0 \finger 1 \finger 2 \finger 4
    \finger 5} }\startGroup
f''1*1/4^\markup{2)}\stopGroup
}
}

```

The image shows a musical staff in bass clef with a key signature of one sharp (F#). The staff contains the following notes and markings:

- First measure: A half note on G2, followed by a half note on A2. Above the first note is a slur with "1)" above it. Below the staff, under the first note, is a vertical list of numbers: 0, 1, 2, 3, 4, 5, 6, 7.
- Second measure: A half note on B2, followed by a half note on C3. Above the first note is a slur with "1)" above it. Below the staff, under the first note, is a vertical list of numbers: 0, 1, 2, 4, 5.
- Third measure: A half note on D3, followed by a half note on E3. Above the first note is a slur with "2)" above it. Below the staff, under the first note, is a vertical list of numbers: 0, 1, 2, 4, 5.

Ancient notation

Section “Ancient notation” in *Notation Reference*

Adding a figured bass above or below the notes

When writing a figured bass, here’s a way to specify if you want your figures to be placed above or below the bass notes, by defining the `BassFigureAlignmentPositioning` `#'direction` property (exclusively in a `Staff` context). Choices are `#UP` (or `#1`), `#CENTER` (or `#0`) and `#DOWN` (or `#-1`).

As you can see here, this property can be changed as many times as you wish. Use `\once \override` if you don’t want the tweak to apply to the whole score.

```
bass = {
  \clef bass
  g4 b, c d
  e d8 c d2
}
continuo = \figuremode {
  <_>4 <6>8
  \once \override Staff.BassFigureAlignmentPositioning #'direction = #CENTER
  <5/>8 <_>4
  \override Staff.BassFigureAlignmentPositioning #'direction = #UP
  <_+>4 <6>
  \set Staff.useBassFigureExtenders = ##t
  \override Staff.BassFigureAlignmentPositioning #'direction = #DOWN
  <4>4. <4>8 <_+>4
}
\score {
  <<
    \new Staff = bassStaff \bass
    \context Staff = bassStaff \continuo
  >>
}
```



Ancient fonts

Shown here are many (all?) of the symbols that are included in LilyPond’s support for ancient notation.

```
upperStaff = \new VaticanaStaff = "upperStaff" <<
  \context VaticanaVoice <<
    \transpose c c {

      \override NoteHead #'style = #'vaticana.punctum
      \key es \major
      \clef "vaticana-fa2"
```

c1 des e f ges

```
\override NoteHead #'style = #'vaticana.inclinatum
a! b ces'
\bar "|"
% \break % 1 (8*1)
```

```
\override NoteHead #'style = #'vaticana.quilisma
b! des'! ges! fes!
\breathes
\clef "vaticana-fa1"
\override NoteHead #'style = #'vaticana.plica
es d
\override NoteHead #'style = #'vaticana.reverse.plica
c d
\bar "|"
% \break %2 (8*1)
```

```
\override NoteHead #'style = #'vaticana.punctum.cavum
es f
\override NoteHead #'style = #'vaticana.lpes
g as
\override NoteHead #'style = #'vaticana.upes
bes as
\override NoteHead #'style = #'vaticana.vupes
g f
\override NoteHead #'style = #'vaticana.linea.punctum
\once \override Staff.BarLine #'bar-size = #2.0 \bar "|"
% \break % 3 (8*1)
```

```
es d
\override NoteHead #'style = #'vaticana.epiphonus
c d
\override NoteHead #'style = #'vaticana.cephalicus
es f
```

```
\override Staff.KeySignature #'glyph-name-alist = #alteration-medicaea-glyph-name-alist
\override Staff.Accidental #'glyph-name-alist = #alteration-medicaea-glyph-name-alist
\override Staff.Custos #'style = #'medicaea
\override NoteHead #'style = #'medicaea.punctum
\clef "medicaea-fa2"
ces des
\bar "|"
% \break % 4 (8*1)
```

```
e! f! ges
\clef "medicaea-do2"
\override NoteHead #'style = #'medicaea.inclinatum
a! b! ces'
\override NoteHead #'style = #'medicaea.virga
b! a!
\bar "|"
```

```

% \break % 5 (8*1)

ges fes
\clef "medicaea-fa1"
\override NoteHead #'style = #'medicaea.rvirga
e des ces

\override Staff.KeySignature #'glyph-name-alist = #alteration-hufnagel-glyph-name-alist
\override Staff.Accidental #'glyph-name-alist = #alteration-hufnagel-glyph-name-alist
\override Staff.Custos #'style = #'hufnagel
\override NoteHead #'style = #'hufnagel.punctum
\clef "hufnagel-fa2"
ces des es
\bar "|"
% \break % 6 (8*1)

fes ges
\clef "hufnagel-do2"
\override NoteHead #'style = #'hufnagel.lpes
as! bes! ces'
\override NoteHead #'style = #'hufnagel.virga
bes! as!
\bar "|"
% \break % 7 (8*1)

ges! fes!
\clef "hufnagel-do-fa"
\override NoteHead #'style = #'hufnagel.punctum
es! des ces des! es! fes!
\bar "||"
% \break % 8 (8*1)

s32*1
% \break % 12 (32*1)
}
>>
>>

lowerStaff = \new MensuralStaff = "lowerStaff" <<
\context MensuralVoice <<
\transpose c c {

\key a \major
cis'1 d'\breve gis'\breve e'\breve \[ e'\longa fis'\longa \]
\set Staff.forceClef = ##t
\clef "neomensural-c2"
cis1
\bar "|"
% \break % 2 (16*1)

\[ g'\breve dis''\longa \]
b'\breve \[ a'\longa d'\longa \]

```

```

\clef "petrucci-c2"
% \break % 4 (16*1)

fis1 ces1
\clef "petrucci-c2"
r\longa
\set Staff.forceClef = ##t
\clef "mensural-c2"
r\breve
\bar "|"
% \break % 5 (8*1)

r2
\clef "mensural-g"
r4 r8 r16 r16
\override NoteHead #'style = #'mensural
\override Rest #'style = #'mensural
\clef "petrucci-f"
c8 b, c16 b, c32 b, c64 b, c64 b,
d8 e d16 e d32 e d64 e d64 e
r\longa
\set Staff.forceClef = ##t
\clef "petrucci-f"
r\breve
\bar "|"
% \break % 6 (8*1)

r\breve
\clef "mensural-f"
r2 r4 r8 r16 r16

\set Staff.forceClef = ##t
\clef "mensural-f"
e\breve f g a1
\clef "mensural-g"
% \break % 7 (8*1)

\[ bes'!\longa a'!\longa c'!\longa \]
e'1 d' c' d' \bar "|"
\bar "|"
% \break % 9 (16*1)

bes'!\longa fis'!1 as'!1 ges'!\longa % lig
\set Staff.forceClef = ##t
\clef "mensural-g"
e'2 d' c' \bar "|"
% \break % 11 (16*1)

\set Staff.forceClef = ##t
\clef "petrucci-g"
c'2 d' e' f'
\clef "petrucci-g"

```

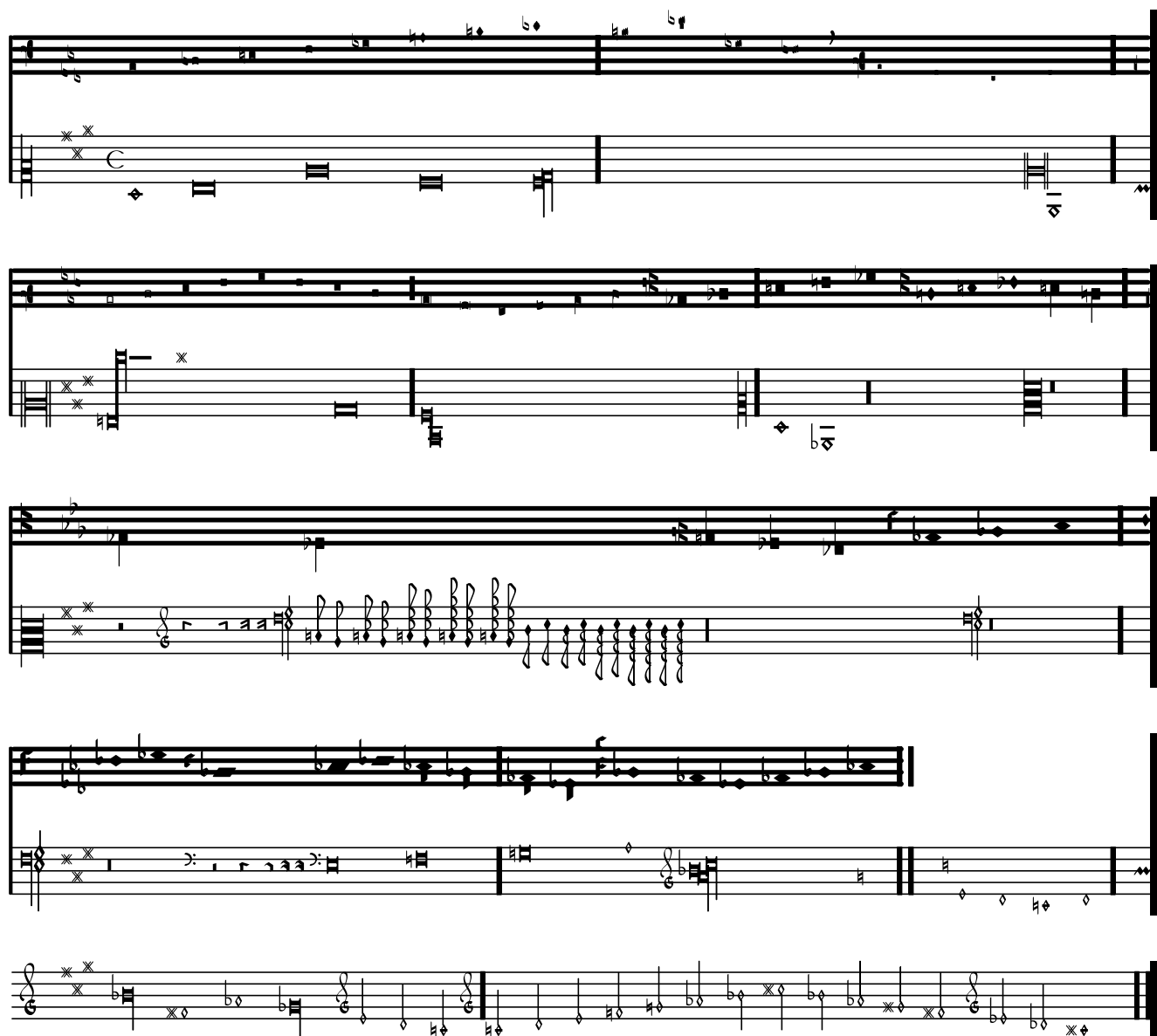
```

        g' as'! bes'! cis'!
        bes'! as'! gis'! fis'!
        \set Staff.forceClef = ##t
        \clef "mensural-g"
        es'! des'! cis'!1 \bar "||"
        % \break % 12 (8*1)
    }
>>
>>

\paper {
  line-thickness = #(/ staff-space 5.0)
}

\score {
  <<
    \upperStaff
    \lowerStaff
  >>
  \layout {
    indent = 0.0
    line-width = 17.25\cm
    \context {
      \Score
      timing = ##f
    }
    \context {
      \MensuralVoice
      \override NoteHead #'style = #'neomensural
      \override Rest #'style = #'neomensural
      \override Stem #'flag-style = #'mensural
      \override Stem #'thickness = #1.0
    }
    \context {
      \MensuralStaff
      \revert BarLine #'transparent
      \override KeySignature #'glyph-name-alist = #alteration-mensural-glyph-name-alist
      clefGlyph = #"clefs.petrucchi.c2"
    }
    \context {
      \VaticanaStaff
      \revert BarLine #'transparent
      \override StaffSymbol #'thickness = #2.0
      \override KeySignature #'glyph-name-alist = #alteration-vaticana-glyph-name-alist
      \override Custos #'neutral-position = #4
    }
  }
}

```



Ancient notation template – modern transcription of gregorian music

This example demonstrates how to do modern transcription of Gregorian music. Gregorian music has no measure, no stems; it uses only half and quarter note heads, and special marks, indicating rests of different length.

```
\include "gregorian.ly"
```

```
chant = \relative c' {
  \set Score.timing = ##f
  f4 a2 \divisioMinima
  g4 b a2 f2 \divisioMaior
  g4( f) f( g) a2 \finalis
}
```

```
verba = \lyricmode {
  Lo -- rem ip -- sum do -- lor sit a -- met
}
```

```
\score {
```

```

\new Staff <<
  \new Voice = "melody" \chant
  \new Lyrics = "one" \lyricsto melody \verba
>>
\layout {
  \context {
    \Staff
    \remove "Time_signature_engraver"
    \remove "Bar_engraver"
    \override Stem #'transparent = ##t
  }
  \context {
    \Voice
    \override Stem #'length = #0
  }
  \context {
    \Score
    barAlways = ##t
  }
}
}

```



Ancient notation template – modern transcription of mensural music

When transcribing mensural music, an incipit at the beginning of the piece is useful to indicate the original key and tempo. While today musicians are used to bar lines in order to faster recognize rhythmic patterns, bar lines were not yet invented during the period of mensural music; in fact, the meter often changed after every few notes. As a compromise, bar lines are often printed between the staves rather than on the staves.

```

global = {
  \set Score.skipBars = ##t

  % incipit
  \once \override Score.SystemStartBracket #'transparent = ##t
  \override Score.SpacingSpanner #'spacing-increment = #1.0 % tight spacing
  \key f \major
  \time 2/2
  \once \override Staff.TimeSignature #'style = #'neomensural
  \override Voice.NoteHead #'style = #'neomensural
  \override Voice.Rest #'style = #'neomensural
  \set Staff.printKeyCancellation = ##f
  \cadenzaOn % turn off bar lines
  \skip 1*10
  \once \override Staff.BarLine #'transparent = ##f
  \bar "||"
}

```

```

\skip 1*1 % need this extra \skip such that clef change comes
          % after bar line
\bar ""

% main
\revert Score.SpacingSpanner #'spacing-increment % CHECK: no effect?
\cadenzaOff % turn bar lines on again
\once \override Staff.Clef #'full-size-change = ##t
\set Staff.forceClef = ##t
\key g \major
\time 4/4
\override Voice.NoteHead #'style = #'default
\override Voice.Rest #'style = #'default

% FIXME: setting printKeyCancellation back to #t must not
% occur in the first bar after the incipit. Dto. for forceClef.
% Therefore, we need an extra \skip.
\skip 1*1
\set Staff.printKeyCancellation = ##t
\set Staff.forceClef = ##f

\skip 1*7 % the actual music

% let finis bar go through all staves
\override Staff.BarLine #'transparent = ##f

% finis bar
\bar "|."
}

discantusNotes = {
  \transpose c' c'' {
    \set Staff.instrumentName = #"Discantus  "

    % incipit
    \clef "neomensural-c1"
    c'1. s2 % two bars
    \skip 1*8 % eight bars
    \skip 1*1 % one bar

    % main
    \clef "treble"
    d'2. d'4 |
    b e' d'2 |
    c'4 e'4.( d'8 c' b |
    a4) b a2 |
    b4.( c'8 d'4) c'4 |
    \once \override NoteHead #'transparent = ##t c'1 |
    b\breve |
  }
}

```

```

discantusLyrics = \lyricmode {
  % incipit
  IV-

  % main
  Ju -- bi -- |
  la -- te De -- |
  o, om --
  nis ter -- |
  ra, __ om- |
  "... " |
  -us. |
}

altusNotes = {
  \transpose c' c' {
    \set Staff.instrumentName = #"Altus  "

    % incipit
    \clef "neomensural-c3"
    r1          % one bar
    f1. s2      % two bars
    \skip 1*7 % seven bars
    \skip 1*1 % one bar

    % main
    \clef "treble"
    r2 g2. e4 fis g | % two bars
    a2 g4 e |
    fis g4.( fis16 e fis4) |
    g1 |
    \once \override NoteHead #'transparent = ##t g1 |
    g\breve |
  }
}

altusLyrics = \lyricmode {
  % incipit
  IV-

  % main
  Ju -- bi -- la -- te | % two bars
  De -- o, om -- |
  nis ter -- ra, |
  "... " |
  -us. |
}

tenorNotes = {
  \transpose c' c' {
    \set Staff.instrumentName = #"Tenor  "

```

```

% incipit
\clef "neomensural-c4"
r\longa    % four bars
r\breve    % two bars
r1         % one bar
c'1. s2    % two bars
\skip 1*1 % one bar
\skip 1*1 % one bar

% main
\clef "treble_8"
R1 |
R1 |
R1 |
r2 d'2. d'4 b e' | % two bars
\once \override NoteHead #'transparent = ##t e'1 |
d'\breve |
}
}

tenorLyrics = \lyricmode {
% incipit
IV-

% main
Ju -- bi -- la -- te | % two bars
"..." |
-us. |
}

bassusNotes = {
\transpose c' c' {
\set Staff.instrumentName = #"Bassus  "

% incipit
\clef "bass"
r\maxima % eight bars
f1. s2   % two bars
\skip 1*1 % one bar

% main
\clef "bass"
R1 |
R1 |
R1 |
R1 |
g2. e4 |
\once \override NoteHead #'transparent = ##t e1 |
g\breve |
}
}

```

```

bassusLyrics = \lyricmode {
  % incipit
  IV-

  % main
  Ju -- bi- |
  "... " |
  -us. |
}

\score {
  \new StaffGroup = choirStaff <<
    \new Voice =
      "discantusNotes" << \global \discantusNotes >>
    \new Lyrics =
      "discantusLyrics" \lyricsto discantusNotes { \discantusLyrics }
    \new Voice =
      "altusNotes" << \global \altusNotes >>
    \new Lyrics =
      "altusLyrics" \lyricsto altusNotes { \altusLyrics }
    \new Voice =
      "tenorNotes" << \global \tenorNotes >>
    \new Lyrics =
      "tenorLyrics" \lyricsto tenorNotes { \tenorLyrics }
    \new Voice =
      "bassusNotes" << \global \bassusNotes >>
    \new Lyrics =
      "bassusLyrics" \lyricsto bassusNotes { \bassusLyrics }
  >>
  \layout {
    \context {
      \Score

      % no bars in staves
      \override BarLine #'transparent = ##t

      % incipit should not start with a start delimiter
      \remove "System_start_delimiter_engraver"
    }
    \context {
      \Voice

      % no slurs
      \override Slur #'transparent = ##t

      % Comment in the below "\remove" command to allow line
      % breaking also at those barlines where a note overlaps
      % into the next bar. The command is commented out in this
      % short example score, but especially for large scores, you
      % will typically yield better line breaking and thus improve
      % overall spacing if you comment in the following command.
      %\remove "Forbid_line_break_engraver"
    }
  }
}

```

}
}
}

Discantus

IV-

Altus

IV-

Tenor

IV-

Bassus

IV-

Ju - bi - la - te De -

Ju - bi - la - te

8

3

o, om - - nis ter - ra, om- ... -us.

De - o, om - nis ter - ra, ... -us.

8

Ju - bi - la - te ... -us.

Ju - bi- ... -us.

Ancient time signatures

Time signatures may also be engraved in an old style.

```
{
  \override Staff.TimeSignature #'style = #'neomensural
  s1
}
```



Chant or psalms notation

This form of notation is used for the chant of the Psalms, where verses aren't always the same length.

```
stemOn = { \revert Staff.Stem #'transparent }
stemOff = { \override Staff.Stem #'transparent = ##t }

\score {
  \new Staff \with { \remove "Time_signature_engraver" }
  {
    \key g \minor
    \cadenzaOn
    \stemOff a'\breve bes'4 g'4
    \stemOn a'2 \bar "||"
    \stemOff a'\breve g'4 a'4
    \stemOn f'2 \bar "||"
    \stemOff a'\breve^\markup { \italic flexe }
    \stemOn g'2 \bar "||"
  }
}
```



Custodes

Custodes may be engraved in various styles.

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

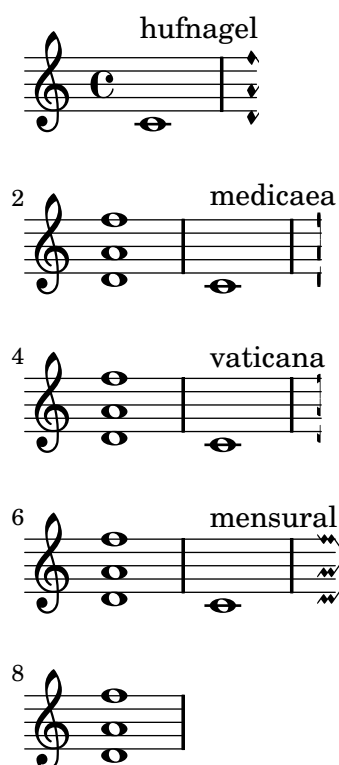
\new Staff \with { \consists "Custos_engraver" } \relative c' {
  \override Staff.Custos #'neutral-position = #4

  \override Staff.Custos #'style = #'hufnagel
  c1^"hufnagel" \break
  <d a' f'>1

  \override Staff.Custos #'style = #'medicaea
  c1^"medicaea" \break
  <d a' f'>1

  \override Staff.Custos #'style = #'vaticana
  c1^"vaticana" \break
  <d a' f'>1

  \override Staff.Custos #'style = #'mensural
  c1^"mensural" \break
  <d a' f'>1
}
```



Incipit

Incipits can be added using the instrument name grob, but keeping separate the instrument name definition and the incipit definition.

```
incipit =
#(define-music-function (parser location incipit-music) (ly:music?)
  #{
    \once \override Staff.InstrumentName #'self-alignment-X = #RIGHT
    \once \override Staff.InstrumentName #'self-alignment-Y = #UP
    \once \override Staff.InstrumentName #'Y-offset =
      #(lambda (grob)
        (+ 4 (system-start-text::calc-y-offset grob)))
    \once \override Staff.InstrumentName #'padding = #0.3
    \once \override Staff.InstrumentName #'stencil =
      #(lambda (grob)
        (let* ((instrument-name (ly:grob-property grob 'long-text))
              (layout (ly:output-def-clone (ly:grob-layout grob)))
              (music (make-sequential-music
                (list (context-spec-music
                  (make-sequential-music
                    (list (make-property-set
                      'instrumentName instrument-name)
                      (make-grob-property-set
                        'VerticalAxisGroup
                        'Y-extent '(-4 . 4))))
                    'MensuralStaff)
                  $incipit-music)))
              (score (ly:make-score music))
              (mm (ly:output-def-lookup layout 'mm))
              (indent (ly:output-def-lookup layout 'indent))
              (width (ly:output-def-lookup layout 'incipit-width))
              (incipit-width (if (number? width)
                                (width)
                                (width))))))
    (score (ly:make-score music))
    (mm (ly:output-def-lookup layout 'mm))
    (indent (ly:output-def-lookup layout 'indent))
    (width (ly:output-def-lookup layout 'incipit-width))
    (incipit-width (if (number? width)
                        (width)
                        (width)))
  )
)
```

```

(* width mm)
(* indent 0.5)))

(ly:output-def-set-variable! layout 'indent (- indent
                                           incipit-width))
(ly:output-def-set-variable! layout 'line-width indent)
(ly:output-def-set-variable! layout 'ragged-right #f)
(ly:output-def-set-variable! layout 'ragged-last #f)
(ly:output-def-set-variable! layout 'system-count 1)
(ly:score-add-output-def! score layout)
(ly:grob-set-property! grob 'long-text
                      (markup #:score score))
(system-start-text::print grob)))
#})

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

global = {
  \set Score.skipBars = ##t
  \key g \major
  \time 4/4

  % the actual music
  \skip 1*8

  % let finis bar go through all staves
  \override Staff.BarLine #'transparent = ##f

  % finis bar
  \bar "|."
}

discantusIncipit = <<
  \new MensuralVoice = "discantusIncipit" <<
    \repeat unfold 9 { s1 \noBreak }
    {
      \clef "neomensural-c1"
      \key f \major
      \time 2/2
      c'1.
    }
  >>
  \new Lyrics \lyricsto discantusIncipit { IV- }
>>

discantusNotes = {
  \transpose c' c'' {
    \clef "treble"
    d'2. d'4 |
    b e' d'2 |
    c'4 e'4.( d'8 c' b |
    a4) b a2 |
  }
}

```

```

    b4.( c'8 d'4) c'4 |
    \once \override NoteHead #'transparent = ##t
    c'1 |
    b\breve |
  }
}

discantusLyrics = \lyricmode {
  Ju -- bi -- |
  la -- te De -- |
  o, om --
  nis ter -- |
  ra, -- om- |
  "... " |
  -us. |
}

altusIncipit = <<
  \new MensuralVoice = "altusIncipit" <<
    \repeat unfold 9 { s1 \noBreak }
    {
      \clef "neomensural-c3"
      \key f \major
      \time 2/2
      r1 f'1.
    }
  >>
  \new Lyrics \lyricsto altusIncipit { IV- }
>>

altusNotes = {
  \transpose c' c'' {
    \clef "treble"
    % two measures
    r2 g2. e4 fis g |
    a2 g4 e |
    fis g4.( fis16 e fis4) |
    g1 |
    \once \override NoteHead #'transparent = ##t
    g1 |
    g\breve |
  }
}

altusLyrics = \lyricmode {
  % two measures
  Ju -- bi -- la -- te |
  De -- o, om -- |
  nis ter -- ra, |
  "... " |
  -us. |
}

```

```

tenorIncipit = <<
  \new MensuralVoice = "tenorIncipit" <<
    \repeat unfold 9 { s1 \noBreak }
    {
      \clef "neomensural-c4"
      \key f \major
      \time 2/2
      r\longa
      r\breve
      r1 c'1.
    }
  >>
  \new Lyrics \lyricsto tenorIncipit { IV- }
>>

tenorNotes = {
  \transpose c' c' {
    \clef "treble_8"
    R1 |
    R1 |
    R1 |
    % two measures
    r2 d'2. d'4 b e' |
    \once \override NoteHead #'transparent = ##t
    e'1 |
    d'\breve |
  }
}

tenorLyrics = \lyricmode {
  % two measures
  Ju -- bi -- la -- te |
  "... " |
  -us.
}

bassusIncipit = <<
  \new MensuralVoice = "bassusIncipit" <<
    \repeat unfold 9 { s1 \noBreak }
    {
      \clef "bass"
      \key f \major
      \time 2/2
      %% incipit
      r\maxima
      f1.
    }
  >>
  \new Lyrics \lyricsto bassusIncipit { IV- }
>>

```

```

bassusNotes = {
  \transpose c' c' {
    \clef "bass"
    R1 |
    R1 |
    R1 |
    R1 |
    g2. e4 |
    \once \override NoteHead #'transparent = ##t
    e1 |
    g\breve |
  }
}

bassusLyrics = \lyricmode {
  Ju -- bi- |
  "... " |
  -us.
}

\score {
  <<
    \new StaffGroup = choirStaff <<
      \new Voice = "discantusNotes" <<
        \global
        \set Staff.instrumentName = #"Discantus"
        \incipit \discantusIncipit
        \discantusNotes
      >>
      \new Lyrics = "discantusLyrics" \lyricsto discantusNotes { \discantusLyrics }
      \new Voice = "altusNotes" <<
        \global
        \set Staff.instrumentName = #"Altus"
        \incipit \altusIncipit
        \altusNotes
      >>
      \new Lyrics = "altusLyrics" \lyricsto altusNotes { \altusLyrics }
      \new Voice = "tenorNotes" <<
        \global
        \set Staff.instrumentName = #"Tenor"
        \incipit \tenorIncipit
        \tenorNotes
      >>
      \new Lyrics = "tenorLyrics" \lyricsto tenorNotes { \tenorLyrics }
      \new Voice = "bassusNotes" <<
        \global
        \set Staff.instrumentName = #"Bassus"
        \incipit \bassusIncipit
        \bassusNotes
      >>
      \new Lyrics = "bassusLyrics" \lyricsto bassusNotes { \bassusLyrics }
    >>
  }
}

```

```

>>
\layout {
  \context {
    \Score
    %% no bar lines in staves or lyrics
    \override BarLine #'transparent = ##t
  }
  %% the next two instructions keep the lyrics between the bar lines
  \context {
    \Lyrics
    \consists "Bar_engraver"
    \consists "Separating_line_group_engraver"
  }
  \context {
    \Voice
    %% no slurs
    \override Slur #'transparent = ##t
    %% Comment in the below "\remove" command to allow line
    %% breaking also at those bar lines where a note overlaps
    %% into the next measure. The command is commented out in this
    %% short example score, but especially for large scores, you
    %% will typically yield better line breaking and thus improve
    %% overall spacing if you comment in the following command.
    %%\remove "Forbid_line_break_engraver"
  }
  indent = 6\cm
  incipit-width = 4\cm
}
}

```

The image displays a musical score for a four-part setting of the text "Jubilate Deo". The score is written for four voices: Discantus, Altus, Tenor, and Bassus. Each voice part is represented by a staff with a treble or bass clef, a key signature of one sharp (F#), and a common time signature (C). The Discantus part begins with a C-clef and a key signature change to one flat (Bb). The lyrics are: "Ju - bi - la - te De -", "Ju", "bi - la - te", and "IV-". The score is presented in two systems, with the first system showing the beginning of the piece and the second system showing the continuation. The lyrics are placed below the staves, and the text "IV-" appears below the Bassus staff in the second system.

Mensurstriche layout (bar lines between the staves)

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

```
global = {
  \override Staff.BarLine #'transparent = ##t
  s1 s
  % the final bar line is not interrupted
  \revert Staff.BarLine #'transparent
  \bar "|."
}
\new StaffGroup \relative c'' {
  <<
    \new Staff { << \global { c1 c } >> }
    \new Staff { << \global { c c } >> }
  >>
}
```

Rest styles

Rests may be used in various styles.

```
\layout {
  indent = 0
  \context {
    \Staff
    \remove "Time_signature_engraver"
  }
}
```

```
\new Staff \relative c {
```

```

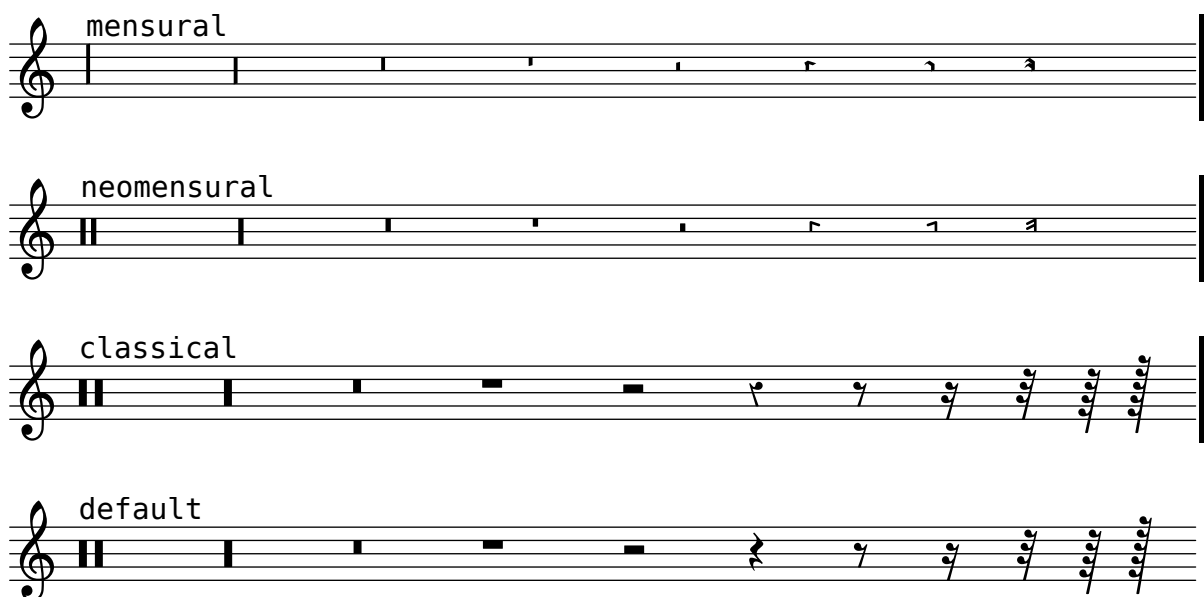
\cadenzaOn
\override Staff.Rest #'style = #'mensural
r\maxima^\markup \typewriter { mensural }
r\longa r\breve r1 r2 r4 r8 r16 s32 s64 s128 s128
\bar ""

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

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

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

```



Transcription of Ancient music with incipit

As a workaround to get real incipits which are independent from the main score these are included as a markup into the field normally used for the instrument name. As for now lyrics can only be added as a direct markup. It doesn't unfortunately conform with the spacing of the main lyrics.

```

global = {
  \set Score.skipBars = ##t
  \key g \major
  \time 4/4
  %make the staff lines invisible on staves
  \override Staff.BarLine #'transparent = ##t
}

```

```

\skip 1*8 % the actual music
% let finis bar go through all staves
\override Staff.BarLine #'transparent = ##f
% finis bar
\bar "|."
}

discantusNotes = {
  \transpose c' c'' {
    \clef treble
    d'2. d'4 |
    b e' d'2 |
    c'4 e'4.( d'8 c' b |
    a4) b a2 |
    b4.( c'8 d'4) c'4 |
    \once \override NoteHead #'transparent = ##t c'1 |
    b\breve |
  }
}

discantusLyrics = \lyricmode {
  Ju -- bi -- |
  la -- te De -- |
  o, om --
  nis ter -- |
  ra, __ om- |
  "... " |
  -us. |
}

altusNotes = {
  \transpose c' c'' {
    \clef treble
    r2 g2. e4 fis g | % two bars
    a2 g4 e |
    fis g4.( fis16 e fis4) |
    g1 |
    \once \override NoteHead #'transparent = ##t g1 |
    g\breve |
  }
}

altusLyrics = \lyricmode {
  Ju -- bi -- la -- te | % two bars
  De -- o, om -- |
  nis ter -- ra, |
  "... " |
  -us. |
}

tenorNotes = {
  \transpose c' c' {

```

```

\clef "treble_8"
R1 |
R1 |
R1 |
r2 d'2. d'4 b e' | % two bars
\once \override NoteHead #'transparent = ##t e'1 |
d'\breve |
}
}

tenorLyrics = \lyricmode {
  Ju -- bi -- la -- te | % two bars
  "... " |
  -us.
}

bassusNotes = {
  \transpose c' c' {
    \clef bass
    R1 |
    R1 |
    R1 |
    R1 |
    g2. e4 |
    \once \override NoteHead #'transparent = ##t e1 |
    g\breve |
  }
}

bassusLyrics = \lyricmode {
  Ju -- bi- |
  "... " |
  -us.
}

incipitDiscantus = \markup {
  \score {
    {
      \set Staff.instrumentName = #"Discantus "
      \override NoteHead #'style = #'neomensural
      \override Rest #'style = #'neomensural
      \override Staff.TimeSignature #'style = #'neomensural
      \cadenzaOn
      \clef "neomensural-c1"
      \key f \major
      \time 2/2
      c''1._"IV-" s2 %two bars
      \skip 1*8 % eight bars
    }
    \layout {
      \context {
        \Voice

```

```

        \remove "Ligature_bracket_engraver"
        \consists "Mensural_ligature_engraver"
    }
    line-width = 4.5\cm
}
}
}

incipitAltus = \markup {
  \score {
    {
      \set Staff.instrumentName = #"Altus "
      \override NoteHead #'style = #'neomensural
      \override Rest #'style = #'neomensural
      \override Staff.TimeSignature #'style = #'neomensural
      \cadenzaOn
      \clef "neomensural-c3"
      \key f \major
      \time 2/2
      r1          % one bar
      f'1._"IV-" s2 % two bars
      \skip 1*7 % seven bars
    }
    \layout {
      \context {
        \Voice
        \remove "Ligature_bracket_engraver"
        \consists "Mensural_ligature_engraver"
      }
      line-width = 4.5\cm
    }
  }
}

incipitTenor = \markup {
  \score {
    {
      \set Staff.instrumentName = #"Tenor "
      \override NoteHead #'style = #'neomensural
      \override Rest #'style = #'neomensural
      \override Staff.TimeSignature #'style = #'neomensural
      \cadenzaOn
      \clef "neomensural-c4"
      \key f \major
      \time 2/2
      r\longa % four bars
      r\breve % two bars
      r1      % one bar
      c'1._"IV-" s2 % two bars
      \skip 1 % one bar
    }
    \layout {

```

```

    \context {
      \Voice
      \remove "Ligature_bracket_engraver"
      \consists "Mensural_ligature_engraver"
    }
    line-width = 4.5\cm
  }
}

incipitBassus = \markup {
  \score {
    {
      \set Staff.instrumentName = #"Bassus "
      \override NoteHead #'style = #'neomensural
      \override Rest #'style = #'neomensural
      \override Staff.TimeSignature #'style = #'neomensural
      \cadenzaOn
      \clef "bass"
      \key f \major
      \time 2/2
      % incipit
      r\maxima % eight bars
      f1._"IV-" s2 % two bars
    }
    \layout {
      \context {
        \Voice
        \remove "Ligature_bracket_engraver"
        \consists "Mensural_ligature_engraver"
      }
      line-width = 4.5\cm
    }
  }
}

```

%StaffGroup is used instead of ChoirStaff to get bar lines between systems

```

\score {
  <<
  \new StaffGroup = choirStaff <<
    \new Voice = "discantusNotes" <<
      \global
      \set Staff.instrumentName = \incipitDiscantus
      \discantusNotes
    >>
  \new Lyrics = "discantusLyrics" \lyricsto discantusNotes { \discantusLyrics }

  \new Voice = "altusNotes" <<
    \global
    \set Staff.instrumentName = \incipitAltus
    \altusNotes
  >>
}

```

```

\new Lyrics = "altusLyrics" \lyricsto altusNotes { \altusLyrics }

\new Voice = "tenorNotes" <<
  \global
  \set Staff.instrumentName = \incipitTenor
  \tenorNotes
>>
\new Lyrics = "tenorLyrics" \lyricsto tenorNotes { \tenorLyrics }

\new Voice = "bassusNotes" <<
  \global
  \set Staff.instrumentName = \incipitBassus
  \bassusNotes
>>
>>
\new Lyrics = "bassusLyrics" \lyricsto bassusNotes { \bassusLyrics }
%Keep the bass lyrics outside of the staff group to avoid bar lines
%between the lyrics.
>>

\layout {
  \context {
    \Score
    % no bars in staves
    \override BarLine #'transparent = ##t
  }
  % the next three instructions keep the lyrics between the bar lines
  \context {
    \Lyrics
    \consists "Bar_engraver"
    \override BarLine #'transparent = ##t
  }
  \context {
    \StaffGroup
    \consists "Separating_line_group_engraver"
  }
  \context {
    \Voice
    % no slurs
    \override Slur #'transparent = ##t
    % Comment in the below "\remove" command to allow line
    % breaking also at those barlines where a note overlaps
    % into the next bar. The command is commented out in this
    % short example score, but especially for large scores, you
    % will typically yield better line breaking and thus improve
    % overall spacing if you comment in the following command.
    %\remove "Forbid_line_break_engraver"
  }
  indent = 5\cm
}
}

```

Discantus

Altus

Tenor

Bassus

IV-

IV-

IV-

IV-

Ju - bi - la - te De - o, om -

Ju - bi - la - te De - o, om -

8

8

4

- nis ter - ra, om - ... -us.

- nis ter - ra, ... -us.

8

Ju - bi - la - te ... -us.

Ju - bi - ... -us.

Vertical line as a baroque articulation mark

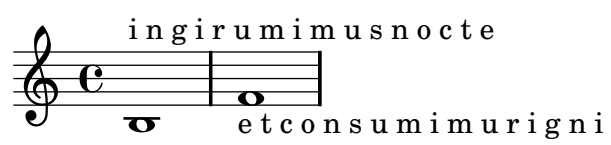
This short vertical line placed above the note is commonly used in baroque music. Its meaning can vary, but generally indicates notes that should be played with more “weight”. The following example demonstrates how to achieve such a notation.

```
upline =
#(let ((m (make-articulation "stopped")))
  (set! (ly:music-property m 'tweaks)
    (acons 'font-size 3
      (acons 'stencil (lambda (grob)
        (grob-interpret-markup
          grob
          (make-draw-line-markup '(0 . 1))))
        (ly:music-property m 'tweaks))))
  m)
```

```
\relative c' {
  a'4^\upline a( c d')_\upline
}
```



}



Contexts and engravers

Section “Changing defaults” in *Notation Reference*

Section “Contexts and engravers” in *Learning Manual*

Adding a figured bass above or below the notes

When writing a figured bass, here’s a way to specify if you want your figures to be placed above or below the bass notes, by defining the `BassFigureAlignmentPositioning` `#'direction` property (exclusively in a `Staff` context). Choices are `#UP` (or `#1`), `#CENTER` (or `#0`) and `#DOWN` (or `#-1`).

As you can see here, this property can be changed as many times as you wish. Use `\once \override` if you don’t want the tweak to apply to the whole score.

```
bass = {
  \clef bass
  g4 b, c d
  e d8 c d2
}
continuo = \figuremode {
  <_>4 <6>8
  \once \override Staff.BassFigureAlignmentPositioning #'direction = #CENTER
  <5/>8 <_>4
  \override Staff.BassFigureAlignmentPositioning #'direction = #UP
  <_+>4 <6>
  \set Staff.useBassFigureExtenders = ##t
  \override Staff.BassFigureAlignmentPositioning #'direction = #DOWN
  <4>4. <4>8 <_+>4
}
\score {
  <<
    \new Staff = bassStaff \bass
    \context Staff = bassStaff \continuo
  >>
}
```



Adding an extra staff at a line break

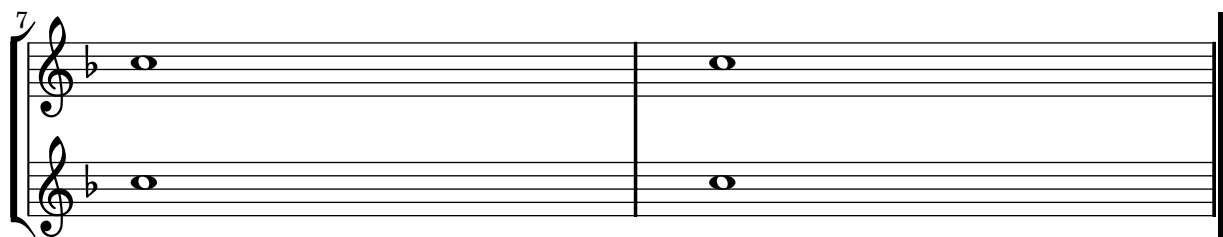
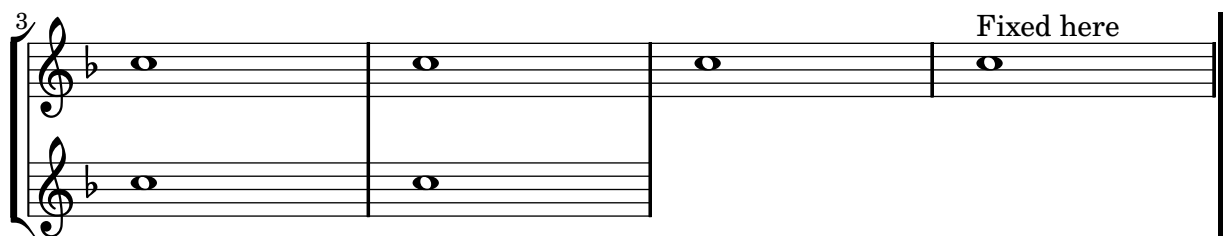
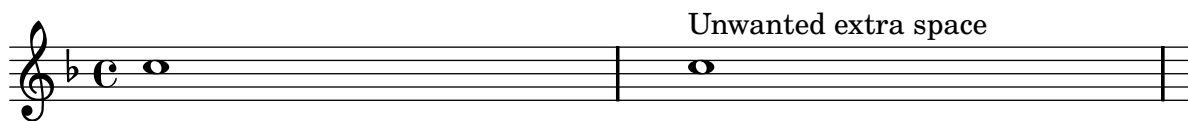
When adding a new staff at a line break, some extra space is unfortunately added at the end of the line before the break (to fit in a key signature change, which will never be printed anyway). The workaround is to add a setting of `Staff.explicitKeySignatureVisibility` as is shown in the example.

```
\score {
  \new StaffGroup \relative c'' {
    \new Staff
```

```

\key f \major
c1 c^"Unwanted extra space" \break
<< { c1 | c }
  \new Staff {
    \key f \major
    \once \override Staff.TimeSignature #'stencil = ##f
    c1 | c
  }
>>
c1 | c^"Fixed here" \break
<< { c1 | c }
  \new Staff {
    \once \set Staff.explicitKeySignatureVisibility = #end-of-line-invisible
    \key f \major
    \once \override Staff.TimeSignature #'stencil = ##f
    c1 | c
  }
>>
}
}

```



Adding an extra staff

An extra staff can be added (possibly temporarily) after the start of a piece.

```

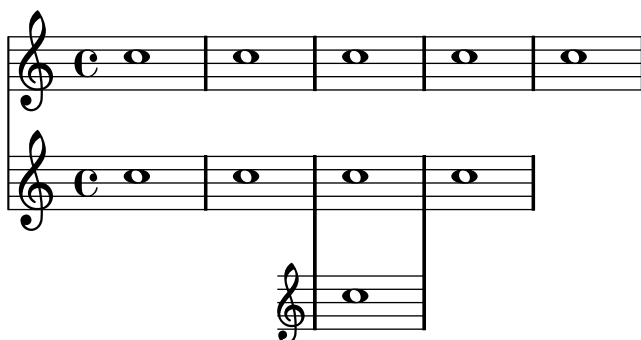
\score {
  <<
    \new Staff \relative c'' { c1 | c | c | c | c }
    \new StaffGroup \relative c'' {
      \new Staff {
        c1 | c <<
        c1

```

```

        \new Staff {
          \once \override Staff.TimeSignature #'stencil = ##f
          c1
        }
      >>
      c1
    }
  }
>>
}

```



Changing MIDI output to one channel per voice

When outputting MIDI, the default behavior is for each staff to represent one MIDI channel, with all the voices on a staff amalgamated. This minimizes the risk of running out of MIDI channels, since there are only 16 available per track.

However, by moving the `Staff_performer` to the `Voice` context, each voice on a staff can have its own MIDI channel, as is demonstrated by the following example: despite being on the same staff, two MIDI channels are created, each with a different `midiInstrument`.

```

\score {
  \new Staff <<
    \new Voice \relative c''' {
      \set midiInstrument = #"flute"
      \voiceOne
      \key g \major
      \time 2/2
      r2 g-"Flute" ~
      g fis ~
      fis4 g8 fis e2 ~
      e4 d8 cis d2
    }
    \new Voice \relative c'' {
      \set midiInstrument = #"clarinet"
      \voiceTwo
      b1-"Clarinet"
      a2. b8 a
      g2. fis8 e
      fis2 r
    }
  >>
}

```

```

\layout { }
\midi {
  \context {
    \Staff
    \remove "Staff_performer"
  }
  \context {
    \Voice
    \consists "Staff_performer"
  }
  \context {
    \Score
    tempoWholesPerMinute = #(ly:make-moment 72 2)
  }
}

```



Changing time signatures inside a polymetric section using `\scaleDurations`

The `measureLength` property, together with `measurePosition`, determines when a bar line is needed. However, when using `\scaleDurations`, the scaling of durations makes it difficult to change time signatures. In this case, `measureLength` should be set manually, using the `ly:make-moment` callback. The second argument must be the same as the second argument of `\scaleDurations`.

```

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

<<
\new Staff {
  \scaleDurations #'(8 . 5) {
    \time 6/8
    \set Timing.measureLength = #(ly:make-moment 6 5)
    b8 b b b b b
    \time 2/4
  }
}

```

```

        \set Timing.measureLength = #(ly:make-moment 4 5)
        b4 b
    }
}
\new Staff {
    \clef bass
    \time 2/4
    c2 d e f
}
>>

```



Chant or psalms notation

This form of notation is used for the chant of the Psalms, where verses aren't always the same length.

```

stemOn = { \revert Staff.Stem #'transparent }
stemOff = { \override Staff.Stem #'transparent = ##t }

\score {
  \new Staff \with { \remove "Time_signature_engraver" }
  {
    \key g \minor
    \cadenzaOn
    \stemOff a'\breve bes'4 g'4
    \stemOn a'2 \bar "||"
    \stemOff a'\breve g'4 a'4
    \stemOn f'2 \bar "||"
    \stemOff a'\breve^{\markup { \italic flexe }}
    \stemOn g'2 \bar "||"
  }
}

```



Creating blank staves

To create blank staves, generate empty measures then remove the `Bar_number_engraver` from the `Score` context, and the `Time_signature_engraver`, `Clef_engraver` and `Bar_engraver` from the `Staff` context.

```

#(set-global-staff-size 20)

```

```

\score {

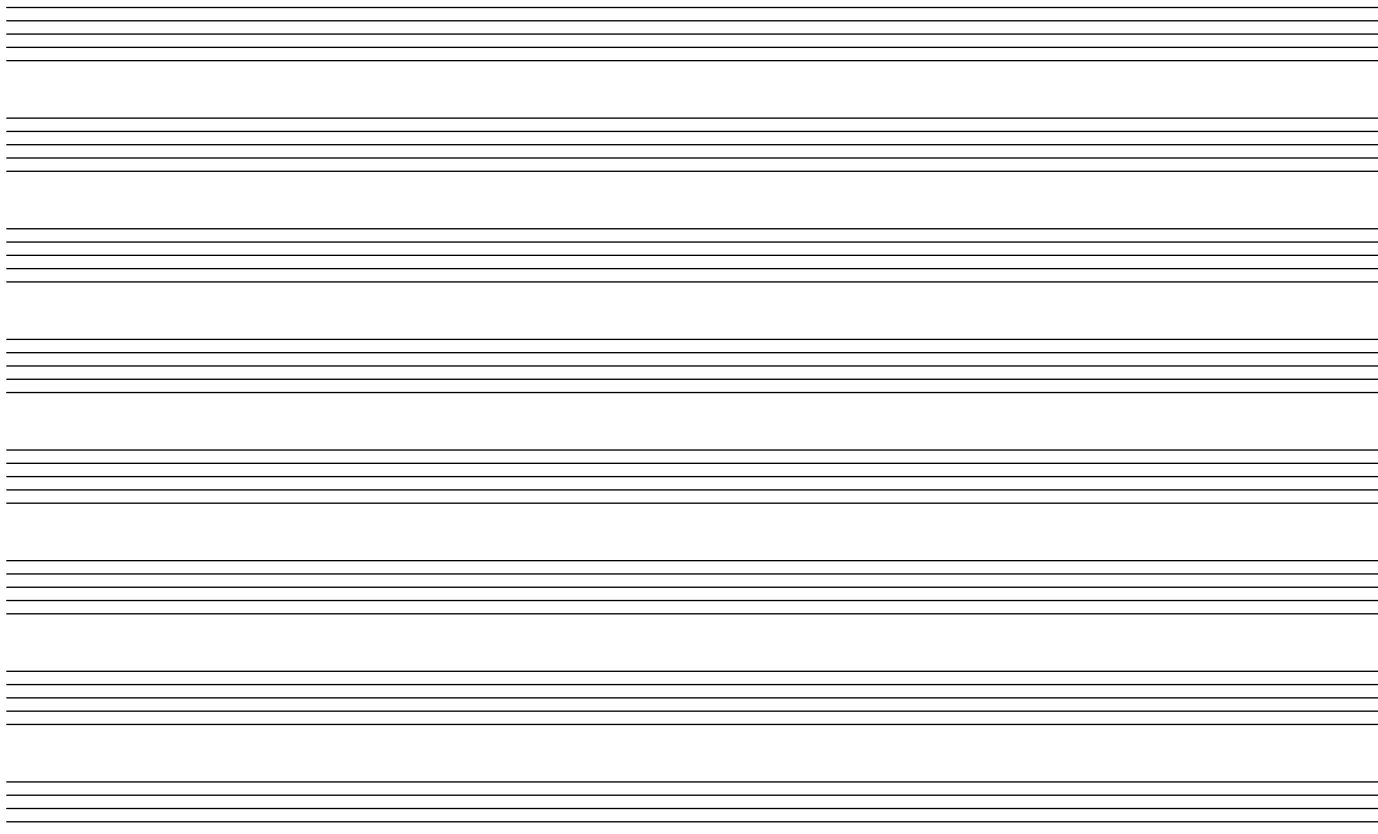
```

```

{
  \repeat unfold 12 { s1 \break }
}
\layout {
  indent = 0\in
  \context {
    \Staff
    \remove "Time_signature_engraver"
    \remove "Clef_engraver"
    \remove "Bar_engraver"
  }
  \context {
    \Score
    \remove "Bar_number_engraver"
  }
}
}

\paper {
  #(set-paper-size "letter")
  ragged-last-bottom = ##f
  line-width = 7.5\in
  left-margin = 0.5\in
  bottom-margin = 0.25\in
  top-margin = 0.25\in
}

```



Defining an engraver in Scheme: ambitus engraver

This example demonstrates how the ambitus engraver may be defined on the user side, with a Scheme engraver.

This is basically a rewrite in Scheme of the code from ‘lily/ambitus-engraver.cc’.

```

#(use-modules (oop goops))

```

```

%%%

```

```

%%% Grob utilities

```

```

%%%

```

```

%%% These are literal rewrites of some C++ methods used by the ambitus engraver.

```

```

#(define (ly:event::in-event-class event class-name)

```

```

  "Check if @var{event} the given class.

```

```

  Rewrite of @code{Stream_event::internal_in_event_class} from @file{lily/stream-event.cc}."

```

```

  (memq class-name (ly:make-event-class (ly:event-property event 'class))))

```

```

#(define (ly:separation-item::add-conditional-item grob grob-item)

```

```

  "Add @var{grob-item} to the array of conditional elements of @var{grob}.

```

```

  Rewrite of @code{Separation_item::add_conditional_item} from @file{lily/separation-item.cc}

```

```

  (ly:pointer-group-interface::add-grob grob 'conditional-elements grob-item))

```

```

#(define (ly:accidental-placement::accidental-pitch accidental-grob)

```

```

  "Get the pitch from the grob cause of @var{accidental-grob}.

```

```

  Rewrite of @code{accidental_pitch} from @file{lily/accidental-placement.cc}."

```

```

  (ly:event-property (ly:grob-property (ly:grob-parent accidental-grob Y) 'cause)
    'pitch))

```

```

#(define (ly:accidental-placement::add-accidental grob accidental-grob)

```

```

  "Add @var{accidental-grob}, an @code{Accidental} grob, to the

```

```

  list of the accidental grobs of @var{grob}, an @code{AccidentalPlacement}

```

```

  grob.

```

```

  Rewrite of @code{Accidental_placement::add_accidental} from @file{lily/accidental-placement

```

```

    (let ((pitch (ly:accidental-placement::accidental-pitch accidental-grob)))

```

```

      (set! (ly:grob-parent accidental-grob X) grob)

```

```

      (set! (ly:grob-property accidental-grob 'X-offset)

```

```

        ly:grob::x-parent-positioning)

```

```

      (let* ((accidentals (ly:grob-object grob 'accidental-grobs))

```

```

        (handle (assq (ly:pitch-notename pitch) accidentals))

```

```

        (entry (if handle (cdr handle) '()))))

```

```

      (set! (ly:grob-object grob 'accidental-grobs)

```

```

        (assq-set! accidentals

```

```

        (ly:pitch-notename pitch)
        (cons accidental-grob entry))))))

%%%
%%% Ambitus data structure
%%%

%%% The <ambitus> class holds the various grobs that are created
%%% to print an ambitus:
%%% - ambitus-group: the grob that groups all the components of an ambitus
%%% (Ambitus grob);
%%% - ambitus-line: the vertical line between the upper and lower ambitus
%%% notes (AmbitusLine grob);
%%% - ambitus-up-note and ambitus-down-note: the note head and accidental
%%% for the lower and upper note of the ambitus (see <ambitus-note> class
%%% below).
%%% The other slots define the key and clef context of the engraver:
%%% - start-c0: position of middle c at the beginning of the piece. It
%%% is used to place the ambitus notes according to their pitch;
%%% - start-key-sig: the key signature at the beginning of the piece. It
%%% is used to determine if accidentals shall be printed next to ambitus
%%% notes.

#(define-class <ambitus> ()
  (ambitus-group #:accessor ambitus-group)
  (ambitus-line #:accessor ambitus-line)
  (ambitus-up-note #:getter ambitus-up-note
    #:init-form (make <ambitus-note>))
  (ambitus-down-note #:getter ambitus-down-note
    #:init-form (make <ambitus-note>))
  (start-c0 #:accessor ambitus-start-c0
    #:init-value #f)
  (start-key-sig #:accessor ambitus-start-key-sig
    #:init-value '()))

%%% Accessor for the lower and upper note data of an ambitus
#(define-method (ambitus-note (ambitus <ambitus>) direction)
  "If @var{direction} is @code{UP}, then return the upper ambitus note
of @var{ambitus}, otherwise return the lower ambitus note."
  (if (= direction UP)
      (ambitus-up-note ambitus)
      (ambitus-down-note ambitus)))

%%% The <ambitus-note> class holds the grobs that are specific to ambitus
%%% (lower and upper) notes:
%%% - head: an AmbitusNoteHead grob;
%%% - accidental: an AmbitusAccidental grob, to be possibly printed next
%%% to the ambitus note head.
%%% Moreover:
%%% - pitch is the absolute pitch of the note
%%% - cause is the note event that causes this ambitus note, i.e. the lower
%%% or upper note of the considered music sequence.

```

```

(define-class <ambitus-note> ()
  (head #:accessor ambitus-note-head
        #:init-value #f)
  (accidental #:accessor ambitus-note-accidental
              #:init-value #f)
  (cause #:accessor ambitus-note-cause
         #:init-value #f)
  (pitch #:accessor ambitus-note-pitch
         #:init-value #f))

%%%
%%% Ambitus engraving logics
%%%
%%% Rewrite of the code from @file{lily/ambitus-engraver.cc}.

(define (make-ambitus translator)
  "Build an ambitus object: initialize all the grobs and their relations.

```

The Ambitus grob contain all other grobs:

```

Ambitus
|- AmbitusLine
|- AmbitusNoteHead   for upper note
|- AmbitusAccidental for upper note
|- AmbitusNoteHead   for lower note
|- AmbitusAccidental for lower note

```

The parent of an accidental is the corresponding note head,
and the accidental is set as the 'accidental-grob of the note head
so that is printed by the function that prints notes."

```

;; make the ambitus object
(let ((ambitus (make <ambitus>)))
  ;; build the Ambitus grob, which will contain all other grobs
  (set! (ambitus-group ambitus) (ly:engraver-make-grob translator 'Ambitus '()))
  ;; build the AmbitusLine grob (line between lower and upper note)
  (set! (ambitus-line ambitus) (ly:engraver-make-grob translator 'AmbitusLine '()))
  ;; build the upper and lower AmbitusNoteHead and AmbitusAccidental
  (for-each (lambda (direction)
              (let ((head (ly:engraver-make-grob translator 'AmbitusNoteHead '()))
                    (accidental (ly:engraver-make-grob translator 'AmbitusAccidental '()))
                    (group (ambitus-group ambitus)))
                ;; The parent of the AmbitusAccidental grob is the
                ;; AmbitusNoteHead grob
                (set! (ly:grob-parent accidental Y) head)
                ;; The AmbitusAccidental grob is set as the accidental-grob
                ;; object of the AmbitusNoteHead. This is later used by the
                ;; function that prints notes.
                (set! (ly:grob-object head 'accidental-grob) accidental)
                ;; both the note head and the accidental grobs are added
                ;; to the main ambitus grob.
                (ly:axis-group-interface::add-element group head)
                (ly:axis-group-interface::add-element group accidental)))
            (list 'upper 'lower)))

```

[illegible]

```

#(define-method (typeset-ambitus (ambitus <ambitus>) translator)
  "Typeset the ambitus:
- place the lower and upper ambitus notes according to their pitch and
  the position of the middle C;
- typeset or delete the note accidentals, according to the key signature.
  An accidental, if it is to be printed, is added to an AccidentalPlacement
  grob (a grob dedicated to the placement of accidentals near a chord);
- both note heads are added to the ambitus line grob, so that a line should
  be printed between them."
  ;; check if there are lower and upper pitches
  (if (and (ambitus-note-pitch (ambitus-note ambitus UP))
           (ambitus-note-pitch (ambitus-note ambitus DOWN)))
      ;; make an AccidentalPlacement grob, for placement of note accidentals
      (let ((accidental-placement (ly:engraver-make-grob
                                   translator
                                   'AccidentalPlacement
                                   (ambitus-note-accidental (ambitus-note ambitus DOWN)))))
        ;; For lower and upper ambitus notes:
        (for-each (lambda (direction)
                     (let ((pitch (ambitus-note-pitch (ambitus-note ambitus direction))))
                       ;; set the cause and the staff position of the ambitus note
                       ;; according to the associated pitch
                       (set! (ly:grob-property (ambitus-note-head (ambitus-note ambitus direction))
                                                'cause)
                             (ambitus-note-cause (ambitus-note ambitus direction)))
                       (set! (ly:grob-property (ambitus-note-head (ambitus-note ambitus direction))
                                                'staff-position)
                             (+ (ambitus-start-c0 ambitus)
                                (ly:pitch-steps pitch)))
                       ;; determine if an accidental shall be printed for this note,
                       ;; according to the key signature
                       (let* ((handle (or (assoc (cons (ly:pitch-octave pitch)
                                                         (ly:pitch-notename pitch))
                                                (ambitus-start-key-sig ambitus))
                                           (assoc (ly:pitch-notename pitch)
                                                  (ambitus-start-key-sig ambitus)))))
                         (sig-alter (if handle (cdr handle) 0)))
                       (cond ((= (ly:pitch-alteration pitch) sig-alter)
                              ;; the note alteration is in the key signature
                              ;; => it does not have to be printed
                              (ly:grob-suicide!
                               (ambitus-note-accidental (ambitus-note ambitus direction)))
                               (set! (ly:grob-object (ambitus-note-head (ambitus-note ambitus direction))
                                                       'accidental-grob)
                                     '()))
                              (else
                               ;; otherwise, the accidental shall be printed
                               (set! (ly:grob-property (ambitus-note-accidental
                                                         (ambitus-note ambitus direction))
                                                         'alteration)
                                     (ly:pitch-alteration pitch)))))))
                     direction)))
  (else
   ;; otherwise, the accidental shall be printed
   (set! (ly:grob-property (ambitus-note-accidental
                           (ambitus-note ambitus direction))
         'alteration)
         (ly:pitch-alteration pitch))))

```

```

;; add the AccidentalPlacement grob to the
;; conditional items of the AmbitusNoteHead
(ly:separation-item::add-conditional-item
  (ambitus-note-head (ambitus-note ambitus direction))
  accidental-placement)
;; add the AmbitusAccidental to the list of the
;; AccidentalPlacement grob accidentals
(ly:accidental-placement::add-accidental
  accidental-placement
  (ambitus-note-accidental (ambitus-note ambitus direction)))■
;; add the AmbitusNoteHead grob to the AmbitusLine grob
(ly:pointer-group-interface::add-grob
  (ambitus-line ambitus)
  'note-heads
  (ambitus-note-head (ambitus-note ambitus direction))))
(list DOWN UP))
;; add the AccidentalPlacement grob to the main Ambitus grob
(ly:axis-group-interface::add-element (ambitus-group ambitus) accidental-placement)
;; no notes ==> suicide the grobs
(begin
  (for-each (lambda (direction)
    (ly:grob-suicide! (ambitus-note-accidental (ambitus-note ambitus direction))
    (ly:grob-suicide! (ambitus-note-head (ambitus-note ambitus direction))))
    (list DOWN UP))
  (ly:grob-suicide! ambitus-line))))

%%%
%%% Ambitus engraver definition
%%%
#(define ambitus-engraver
  (lambda (context)
    (let ((ambitus #f))
      ;; when music is processed: make the ambitus object, if not already built
      `(process-music . ,(lambda (translator)
        (if (not ambitus)
          (set! ambitus (make-ambitus translator))))))
      ;; set the ambitus clef and key signature state
      (stop-translation-timestep . ,(lambda (translator)
        (if ambitus
          (initialize-ambitus-state ambitus translator))))
      ;; when a note-head grob is built, update the ambitus notes
      (acknowledgers
        (note-head-interface . ,(lambda (engraver grob source-engraver)
          (if ambitus
            (update-ambitus-notes ambitus grob))))))
      ;; finally, typeset the ambitus according to its upper and lower notes
      ;; (if any).
      (finalize . ,(lambda (translator)
        (if ambitus
          (typeset-ambitus ambitus translator))))))

%%%

```

```
%%% Example
%%%
```

```
\score {
  \new StaffGroup <<
    \new Staff { c'4 des' e' fis' gis' }
    \new Staff { \clef "bass" c4 des ~ des ees b, }
  >>
  \layout { \context { \Staff \consists #ambitus-engraver } }
}
```



Engravers one-by-one

The notation problem, creating a certain symbol, is handled by plugins. Each plugin is called an Engraver. In this example, engravers are switched on one by one, in the following order:

- note heads,
- staff symbol,
- clef,
- stem,
- beams, slurs, accents,
- accidentals, bar lines, time signature and key signature.

Engravers are grouped. For example, note heads, slurs, beams etc. form a **Voice** context. Engravers for key signature, accidentals, bar line, etc. form a **Staff** context.

```
%% sample music
topVoice = \relative c' {
  \key d \major
  es8([ g] a[ fis])
  b4
  b16[-. b-. b-. cis-.]
  d4->
}
```

```
botVoice = \relative c' {
  \key d \major
  c8([ f] b[ a])
  es4
  es16[-. es-. es-. fis-.]
  b4->
}
```

```
hoom = \relative c {
  \key d \major
  \clef bass
```

```

    g8-. r
    r4
    fis8-.
    r8
    r4
    b'4->
}

pah = \relative c' {
    r8 b-.
    r4
    r8 g8-.
    r16 g-. r8
    \clef treble
    fis'4->
}

%
% setup for Request->Element conversion. Guru-only
%

MyStaff = \context {
    \type "Engraver_group"
    \name Staff

    \description "Handles clefs, bar lines, keys, accidentals. It can contain
@code{Voice} contexts."

    \consists "Output_property_engraver"

    \consists "Font_size_engraver"

    \consists "Volta_engraver"
    \consists "Separating_line_group_engraver"
    \consists "Dot_column_engraver"

    \consists "Ottava_spanner_engraver"
    \consists "Rest_collision_engraver"
    \consists "Piano_pedal_engraver"
    \consists "Piano_pedal_align_engraver"
    \consists "Instrument_name_engraver"
    \consists "Grob_pq_engraver"
    \consists "Forbid_line_break_engraver"
    \consists "Axis_group_engraver"

    \consists "Pitch_squash_engraver"

    \override VerticalAxisGroup #'minimum-Y-extent = #'(-6 . 6)
    localKeySignature = #'()

    % explicitly set instrumentName, so we don't get
    % weird effects when doing instrument names for

```

```

% piano staves

instrumentName = #'()
shortInstrumentName = #'()

\accepts "Voice"
}

MyVoice = \context {
  \type "Engraver_group"
  \name Voice

  \description "
    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."

  localKeySignature = #'()
  \consists "Font_size_engraver"

  % must come before all
  \consists "Output_property_engraver"
  \consists "Arpeggio_engraver"
  \consists "Multi_measure_rest_engraver"
  \consists "Text_spanner_engraver"
  \consists "Grob_pq_engraver"
  \consists "Note_head_line_engraver"
  \consists "Glissando_engraver"
  \consists "Ligature_bracket_engraver"
  \consists "Breathing_sign_engraver"
  % \consists "Rest_engraver"
  \consists "Grace_beam_engraver"
  \consists "New_fingering_engraver"
  \consists "Chord_tremolo_engraver"
  \consists "Percent_repeat_engraver"
  \consists "Slash_repeat_engraver"

  %{
    Must come before text_engraver, but after note_column engraver.
  %}
  \consists "Text_engraver"
  \consists "Dynamic_engraver"
  \consists "Fingering_engraver"

  \consists "Script_column_engraver"
  \consists "Rhythmic_column_engraver"
  \consists "Cluster_spanner_engraver"
  \consists "Tie_engraver"

```

```

\consists "Tie_engraver"
\consists "Tuplet_engraver"
\consists "Note_heads_engraver"
\consists "Rest_engraver"
}

```

```

\score {
  \topVoice
  \layout {
    \context { \MyStaff }
    \context { \MyVoice }
  }
}

```

```

MyStaff = \context {
  \MyStaff
  \consists "Staff_symbol_engraver"
}

```

```

\score {
  \topVoice
  \layout {
    \context { \MyStaff }
    \context { \MyVoice }
  }
}

```

```

MyStaff = \context {
  \MyStaff
  \consists "Clef_engraver"
  \remove "Pitch_squash_engraver"
}

```

```

\score {
  \topVoice
  \layout {
    \context { \MyStaff }
    \context { \MyVoice }
  }
}

```

```

MyVoice = \context {
  \MyVoice
  \consists "Stem_engraver"
}

```

```

\score {
  \topVoice
  \layout {
    \context { \MyStaff }

```

```

    \context { \MyVoice }
  }
}

MyVoice = \context {
  \MyVoice
  \consists "Beam_engraver"
}

\score {
  \topVoice
  \layout {
    \context { \MyStaff }
    \context { \MyVoice }
  }
}

MyVoice = \context {
  \MyVoice
  \consists "Phrasing_slur_engraver"
  \consists "Slur_engraver"
  \consists "Script_engraver"
}

\score {
  \topVoice
  \layout {
    \context { \MyStaff }
    \context { \MyVoice }
  }
}

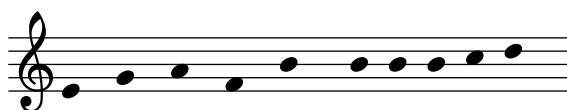
MyStaff = \context {
  \MyStaff
  \consists "Bar_engraver"
  \consists "Time_signature_engraver"
}

\score {
  \topVoice
  \layout {
    \context { \MyStaff }
    \context { \MyVoice }
  }
}

MyStaff = \context {
  \MyStaff
  \consists "Accidental_engraver"
  \consists "Key_engraver"
}

```

```
\score {  
  \topVoice  
  \layout {  
    \context { \MyStaff }  
    \context { \MyVoice }  
  }  
}
```

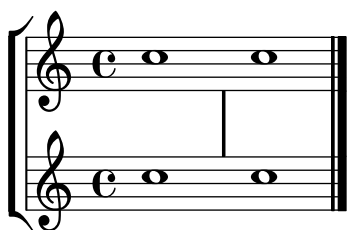




Mensurstriche layout (bar lines between the staves)

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

```
global = {
  \override Staff.BarLine #'transparent = ##t
  s1 s
  % the final bar line is not interrupted
  \revert Staff.BarLine #'transparent
  \bar "|."
}
\new StaffGroup \relative c'' {
  <<
    \new Staff { << \global { c1 c } >> }
    \new Staff { << \global { c c } >> }
  >>
}
```

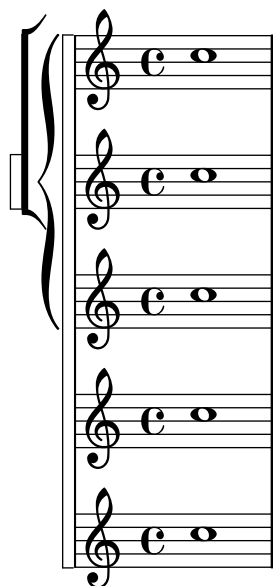


Nesting staves

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

```
\new StaffGroup
\relative c'' <<
  \set StaffGroup.systemStartDelimiterHierarchy
    = #'(SystemStartSquare (SystemStartBrace (SystemStartBracket a
      (SystemStartSquare b) ) c ) d)

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

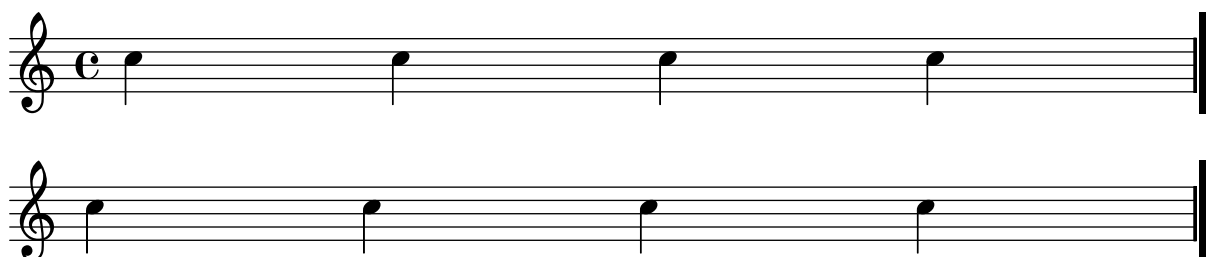


Removing bar numbers from a score

Bar numbers can be removed entirely by removing the `Bar_number_engraver` from the `Score` context.

```
\layout {
  \context {
    \Score
    \remove "Bar_number_engraver"
  }
}
```

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



Use square bracket at the start of a staff group

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

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



Vocal ensemble template with lyrics aligned below and above the staves

This template is basically the same as the simple “Vocal ensemble” template, with the exception that here all the lyrics lines are placed using `alignAboveContext` and `alignBelowContext`.

```
global = {
  \key c \major
  \time 4/4
}

sopMusic = \relative c'' {
  c4 c c8[( b)] c4
}
sopWords = \lyricmode {
  hi hi hi hi
}

altoMusic = \relative c' {
  e4 f d e
}
altoWords = \lyricmode {
  ha ha ha ha
}

tenorMusic = \relative c' {
  g4 a f g
}
tenorWords = \lyricmode {
  hu hu hu hu
}

bassMusic = \relative c {
  c4 c g c
}
bassWords = \lyricmode {
  ho ho ho ho
}

\score {
  \new ChoirStaff <<
    \new Staff = women <<
      \new Voice = "sopranos" { \voiceOne << \global \sopMusic >> }
      \new Voice = "altos" { \voiceTwo << \global \altoMusic >> }
    >>
    \new Lyrics \with { alignAboveContext = women } \lyricsto sopranos \sopWords
    \new Lyrics \with { alignBelowContext = women } \lyricsto altos \altoWords
    % we could remove the line about this with the line below, since we want
```

```

% the alto lyrics to be below the alto Voice anyway.
% \new Lyrics \lyricsto altos \altoWords

\new Staff = men <<
  \clef bass
  \new Voice = "tenors" { \voiceOne << \global \tenorMusic >> }
  \new Voice = "basses" { \voiceTwo << \global \bassMusic >> }
>>
\new Lyrics \with { alignAboveContext = men } \lyricsto tenors \tenorWords
\new Lyrics \with { alignBelowContext = men } \lyricsto basses \bassWords
% again, we could replace the line above this with the line below.
% \new Lyrics \lyricsto basses \bassWords
>>
\layout {
  \context {
    % a little smaller so lyrics
    % can be closer to the staff
    \Staff
    \override VerticalAxisGroup #'minimum-Y-extent = #'(-3 . 3)
  }
}
}

```



Vocal ensemble template with verse and refrain

This template creates a score which starts with a solo verse and continues into a refrain for two voices. It also demonstrates the use of spacer rests within the `\global` variable to define meter changes (and other elements common to all parts) throughout the entire score.

```

global = {
  \key g \major

  % verse
  \time 3/4
  s2.*2
  \break

  % refrain
  \time 2/4
  s2*2
}

```

```

\bar "|."
}

SoloNotes = \relative g' {
  \clef "treble"

  % verse
  g4 g g |
  b4 b b |

  % refrain
  R2*2 |
}

SoloLyrics = \lyricmode {
  One two three |
  four five six |
}

SopranoNotes = \relative c'' {
  \clef "treble"

  % verse
  R2.*2 |

  % refrain
  c4 c |
  g4 g |
}

SopranoLyrics = \lyricmode {
  la la |
  la la |
}

BassNotes = \relative c {
  \clef "bass"

  % verse
  R2.*2 |

  % refrain
  c4 e |
  d4 d |
}

BassLyrics = \lyricmode {
  dum dum |
  dum dum |
}

\score {

```

```

<<
  \new Voice = "SoloVoice" << \global \SoloNotes >>
  \new Lyrics \lyricsto "SoloVoice" \SoloLyrics

  \new ChoirStaff <<
    \new Voice = "SopranoVoice" << \global \SopranoNotes >>
    \new Lyrics \lyricsto "SopranoVoice" \SopranoLyrics

    \new Voice = "BassVoice" << \global \BassNotes >>
    \new Lyrics \lyricsto "BassVoice" \BassLyrics
  >>
>>
\layout {
  ragged-right = ##t
  \context { \Staff
    % these lines prevent empty staves from being printed
    \RemoveEmptyStaves
    \override VerticalAxisGroup #'remove-first = ##t
  }
}
}

```



Tweaks and overrides

Section “Changing defaults” in *Notation Reference*

Section “Tweaking output” in *Learning Manual*

Adjusting grace note spacing

The space given to grace notes can be adjusted using the `spacing-increment` property of `Score.GraceSpacing`.

```
graceNotes = {
  \grace { c4 c8 c16 c32 }
  c8
}

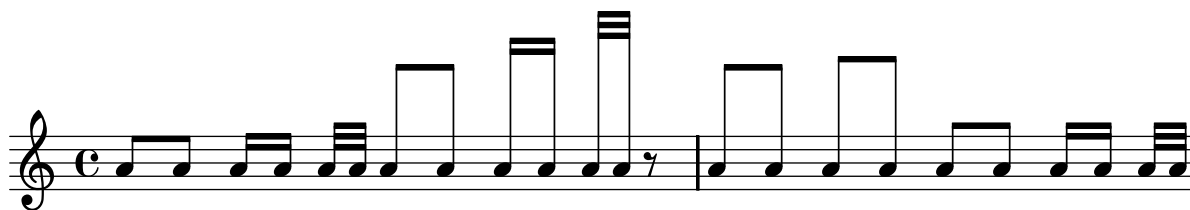
\relative c'' {
  c8
  \graceNotes
  \override Score.GraceSpacing #'spacing-increment = #2.0
  \graceNotes
  \revert Score.GraceSpacing #'spacing-increment
  \graceNotes
}
```



Altering the length of beamed stems

Stem lengths on beamed notes can be varied by overriding the `beamed-lengths` property of the `details` of the `Stem`. If a single value is used as an argument, the length applies to all stems. When multiple arguments are used, the first applies to eighth notes, the second to sixteenth notes and so on. The final argument also applies to all notes shorter than the note length of the final argument. Non-integer arguments may also be used.

```
\relative c'' {
  \override Stem #'details #'beamed-lengths = #'(2)
  a8[ a] a16[ a] a32[ a]
  \override Stem #'details #'beamed-lengths = #'(8 10 12)
  a8[ a] a16[ a] a32[ a] r8
  \override Stem #'details #'beamed-lengths = #'(8)
  a8[ a]
  \override Stem #'details #'beamed-lengths = #'(8.5)
  a8[ a]
  \revert Stem #'details
  a8[ a] a16[ a] a32[ a]
}
```



Analysis brackets above the staff

Simple horizontal analysis brackets are added below the staff by default. The following example shows a way to place them above the staff instead.

```
\layout {
  \context {
    \Voice
    \consists "Horizontal_bracket_engraver"
  }
}
\relative c' {
  \once \override HorizontalBracket #'direction = #UP
  c2\startGroup
  d2\stopGroup
}
```

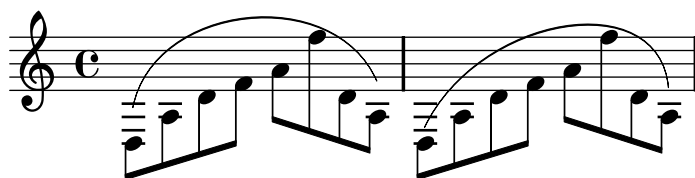


Asymmetric slurs

Slurs can be made asymmetric to match an asymmetric pattern of notes better.

```
slurNotes = { d,8( a' d f a f' d, a) }
```

```
\relative c' {
  \stemDown
  \slurUp
  \slurNotes
  \once \override Slur #'eccentricity = #3.0
  \slurNotes
}
```



Avoiding collisions with chord fingerings

Fingerings and string numbers applied to individual notes will automatically avoid beams and stems, but this is not true by default for fingerings and string numbers applied to the individual notes of chords. The following example shows how this default behavior can be overridden.

```
\relative c' {
  \set fingeringOrientations = #'(up)
}
```

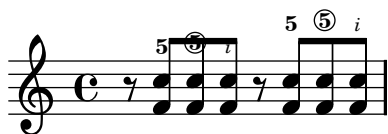
```

\set stringNumberOrientations = #'(up)
\set strokeFingerOrientations = #'(up)

% Default behavior
r8
<f c'-5>8
<f c'\5>8
<f c'-\rightHandFinger #2 >8

% Corrected to avoid collisions
r8
\override Fingering #'add-stem-support = ##t
<f c'-5>8
\override StringNumber #'add-stem-support = ##t
<f c'\5>8
\override StrokeFinger #'add-stem-support = ##t
<f c'-\rightHandFinger #2 >8
}

```



Caesura ("railtracks") with fermata

A caesura is sometimes denoted by a double “railtracks” breath mark with a fermata sign positioned above. This snippet shows an optically pleasing combination of railtracks and fermata.

```

\relative c' {
  c2.
  % construct the symbol
  \override BreathingSign #'text = \markup {
    \line {
      \musicglyph #"scripts.caesura.curved"
      \translate #'(-1.75 . 1.6)
      \musicglyph #"scripts.ufermata"
    }
  }
  \breathe c4
  % set the breathe mark back to normal
  \revert BreathingSign #'text
  c2. \breathe c4
  \bar "|."
}

```



Centering markup on note heads automatically

For technical reasons, text scripts attached to note heads cannot easily be centered on a note head's width, unlike articulations.

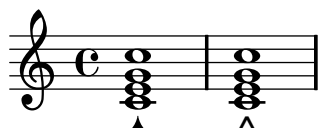
Instead of using trial-and-error offset tweaks, this snippet accesses a note head (or rest) from the `TextScript` object's horizontal parent (a paper column), using its extent to correct the positioning.

```
textScriptCenterOnNote = \override TextScript #'X-offset =
#(lambda (grob)
  (let* ((paper-col (ly:grob-parent grob X))
        (elts (ly:grob-object paper-col 'elements))
        (rhythmic-head grob))

    (for-each
      (lambda (idx)
        (let ((elt (ly:grob-array-ref elts idx)))
          (if (grob::has-interface elt
                                     'rhythmic-grob-interface)
              (set! rhythmic-head elt))))
      (reverse (iota (ly:grob-array-length elts))))

    (+
      (ly:self-alignment-interface::x-aligned-on-self grob)
      (interval-center
        (ly:grob-robust-relative-extent rhythmic-head rhythmic-head X))))

\relative c' {
  \override TextScript #'self-alignment-X = #CENTER
  \textScriptCenterOnNote
  <c e g c>1-\markup \arrow-head #Y #UP ##t
  <c e g c>1-\markup \huge "^"
}
```

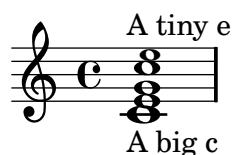


Changing a single note's size in a chord

Individual note heads in a chord can be modified with the `\tweak` command inside a chord, by altering the `font-size` property.

Inside the chord (within the brackets `< >`), before the note to be altered, place the `\tweak` command, followed by `#'font-size` and define the proper size like `#-2` (a tiny note head).

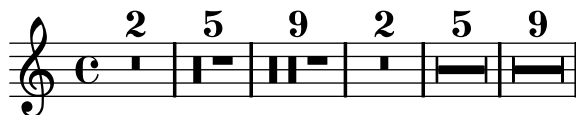
```
\relative c' {
  <\tweak #'font-size #-2 c e g c \tweak #'font-size #-2 e>1-\markup { A tiny e }-\markup {
}
```



Changing form of multi-measure rests

If there are ten or fewer measures of rests, a series of longa and breve rests (called in German “Kirchenpausen” - church rests) is printed within the staff; otherwise a simple line is shown. This default number of ten may be changed by overriding the `expand-limit` property.

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



Changing properties for individual grobs

The `\applyOutput` command allows the tuning of any layout object, in any context. It requires a Scheme function with three arguments.

```
#(define (mc-squared grob grob-origin context)
  (let ((sp (ly:grob-property grob 'staff-position)))
    (if (grob::has-interface grob 'note-head-interface)
        (begin
          (ly:grob-set-property! grob 'stencil
            (grob-interpret-markup grob
              (make-lower-markup 0.5
                (case sp
                  ((-5) "m")
                  ((-3) "c ")
                  ((-2) (make-smaller-markup (make-bold-markup "2"))))
                (else "bla"))))))))

\relative c' {
  <d f g b>2
  \applyOutput #'Voice #mc-squared
  <d f g b>2
}
```



Changing the default text font family

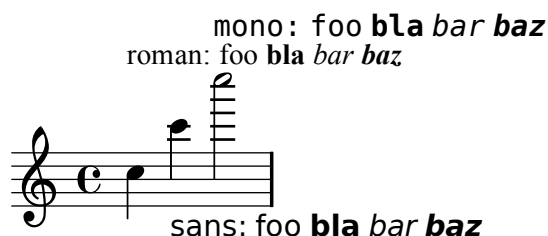
The default font families for text can be overridden with `make-pango-font-tree`.

```
\paper {
  % change for other default global staff size.
  myStaffSize = #20
}
```

```
%{
  run
    lilypond -dshow-available-fonts blabla
  to show all fonts available in the process log.
}%

#(define fonts
  (make-pango-font-tree "Times New Roman"
    "Nimbus Sans"
    "Luxi Mono"
    "Helvetica"
    "Courier"
    (/ myStaffSize 20)))
}

\relative c'' {
  c4^\markup {
    roman: foo \bold bla \italic bar \italic \bold baz
  }
  c'4_\markup {
    \override #'(font-family . sans)
    {
      sans: foo \bold bla \italic bar \italic \bold baz
    }
  }
  c'2^\markup {
    \override #'(font-family . typewriter)
    {
      mono: foo \bold bla \italic bar \italic \bold baz
    }
  }
}
```



Changing the staff size

Though the simplest way to resize staves is to use `#(set-global-staff-size xx)`, an individual staff's size can be changed by scaling the properties `'staff-space` and `fontSize`.

```
<<
  \new Staff {
    \relative c'' {
      \dynamicDown
      c8\ff c c c c c c c
    }
  }
```

```

    }
  }
  \new Staff \with {
    fontSize = #-3
    \override StaffSymbol #'staff-space = #(magstep -3)
  } {
    \clef bass
    c8 c c c c\ff c c c
  }
>>

```



Changing the text for sustain markings

`Staff.pedalSustainStrings` can be used to set the text used for pedal down and up. Note that the only valid strings are those found in the list of pedal glyphs - the values used in this snippet constitute an exhaustive list.

```
sustainNotes = { c4\sustainOn d e\sustainOff\sustainOn f\sustainOff }
```

```

\relative c' {
  \sustainNotes
  \set Staff.pedalSustainStrings = #("P" "P-" "-")
  \sustainNotes
  \set Staff.pedalSustainStrings = #("d" "de" "e")
  \sustainNotes
  \set Staff.pedalSustainStrings = #("M" "M-" "-")
  \sustainNotes
  \set Staff.pedalSustainStrings = #("Ped" "*Ped" "*")
  \sustainNotes
}

```



Controlling spanner visibility after a line break

The visibility of spanners which end on the first note following a line break is controlled by the `after-line-breaking` callback `ly:spanner::kill-zero-spanned-time`.

For objects such as glissandos and hairpins, the default behaviour is to hide the spanner after a break; disabling the callback will allow the left-broken span to be shown.

Conversely, spanners which are usually visible, such as text spans, can be hidden by enabling the callback.

```

\paper { ragged-right = ##t }

\relative c'' {
  \override Hairpin #'to-barline = ##f
  \override Glissando #'breakable = ##t
  % show hairpin
  \override Hairpin #'after-line-breaking = ##t
  % hide text span
  \override TextSpanner #'after-line-breaking =
    #ly:spanner::kill-zero-spanned-time
  e2\<\startTextSpan
  % show glissando
  \override Glissando #'after-line-breaking = ##t
  f2\glissando
  \break
  f,1\!\stopTextSpan
}

```



Controlling the vertical ordering of scripts

The vertical ordering of scripts is controlled with the 'script-priority' property. The lower this number, the closer it will be put to the note. In this example, the `TextScript` (the sharp symbol) first has the lowest priority, so it is put lowest in the first example. In the second, the prall trill (the `Script`) has the lowest, so it is on the inside. When two objects have the same priority, the order in which they are entered determines which one comes first.

```

\relative c''' {
  \once \override TextScript #'script-priority = #-100
  a2^\prall^\markup { \sharp }

  \once \override Script #'script-priority = #-100
  a2^\prall^\markup { \sharp }
}

```



Controlling tuplet bracket visibility

The default behavior of tuplet-bracket visibility is to print a bracket unless there is a beam of the same length as the tuplet. To control the visibility of tuplet brackets, set the property `'bracket-visibility` to either `#t` (always print a bracket), `#f` (never print a bracket) or `#'if-no-beam` (only print a bracket if there is no beam).

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

\new Voice {
  \relative c' {
    << \music s4^"default" >>
    \override TupletBracket #'bracket-visibility = #'if-no-beam
    << \music s4^"'if-no-beam" >>
    \override TupletBracket #'bracket-visibility = ##t
    << \music s4^"#t" >>
    \override TupletBracket #'bracket-visibility = ##f
    << \music s4^"#f" >>
  }
}
```



Creating a delayed turn

Creating a delayed turn, where the lower note of the turn uses the accidental, requires several overrides. The `outside-staff-priority` property must be set to `#f`, as otherwise this would take precedence over the `avoid-slur` property. The value of `halign` is used to position the turn horizontally.

```
\relative c'' {
  \once \override TextScript #'avoid-slur = #'inside
  \once \override TextScript #'outside-staff-priority = ##f
  c2(^\markup \tiny \override #'(baseline-skip . 1) {
    \halign #-4
    \center-column {
      \sharp
      \musicglyph #"scripts.turn"
    }
  }
}
d4.) c8
```

}



Creating simultaneous rehearsal marks

Unlike text scripts, rehearsal marks cannot be stacked at a particular point in a score: only one `RehearsalMark` object is created. Using an invisible measure and bar line, an extra rehearsal mark can be added, giving the appearance of two marks in the same column. This method may also prove useful for placing rehearsal marks at both the end of one system and the start of the following system.

```
{
  \key a \major
  \set Score.markFormatter = #format-mark-box-letters
  \once \override Score.RehearsalMark #'outside-staff-priority = #5000
  \once \override Score.RehearsalMark #'self-alignment-X = #LEFT
  \once \override Score.RehearsalMark #'break-align-symbols = #'(key-signature)
  \mark \markup { \bold { Senza denti } }

  % the hidden measure and bar line
  % \cadenzaOn turns off automatic calculation of bar numbers
  \cadenzaOn
  \once \override Score.TimeSignature #'stencil = ##f
  \time 1/16
  s16 \bar ""
  \cadenzaOff

  \time 4/4
  \once \override Score.RehearsalMark #'self-alignment-X = #LEFT
  \mark \markup { \box \bold Intro }
  d'1
  \mark \default
  d'1
}
```



Creating text spanners

The `\startTextSpan` and `\stopTextSpan` commands allow the creation of text spanners as easily as pedal indications or octavations. Override some properties of the `TextSpanner` object to modify its output.

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

```
\relative c' {
```

```

\override TextSpanner #'(bound-details left text) = #"bla"
\override TextSpanner #'(bound-details right text) = #"blu"
a4 \startTextSpan
b4 c
a4 \stopTextSpan

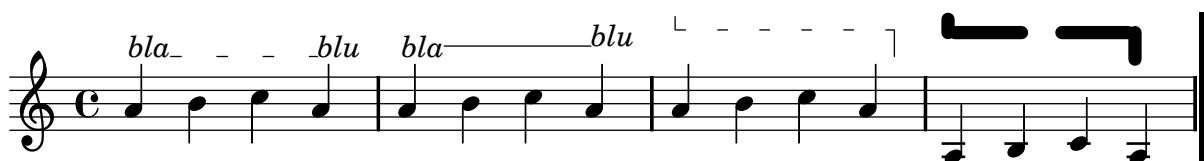
\override TextSpanner #'style = #'line
\once \override TextSpanner
  #'(bound-details left stencil-align-dir-y) = #CENTER
a4 \startTextSpan
b4 c
a4 \stopTextSpan

\override TextSpanner #'style = #'dashed-line
\override TextSpanner #'(bound-details left text) =
  \markup { \draw-line #'(0 . 1) }
\override TextSpanner #'(bound-details right text) =
  \markup { \draw-line #'(0 . -2) }
\once \override TextSpanner #'(bound-details right padding) = #-2

a4 \startTextSpan
b4 c
a4 \stopTextSpan

\set Staff.middleCPosition = #-13
\override TextSpanner #'dash-period = #10
\override TextSpanner #'dash-fraction = #0.5
\override TextSpanner #'thickness = #10
a4 \startTextSpan
b4 c
a4 \stopTextSpan
}

```



Custodes

Custodes may be engraved in various styles.

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

```

\new Staff \with { \consists "Custos_engraver" } \relative c' {
  \override Staff.Custos #'neutral-position = #4

  \override Staff.Custos #'style = #'hufnagel
  c1~"hufnagel" \break
  <d a' f'>1

  \override Staff.Custos #'style = #'medicaea

```

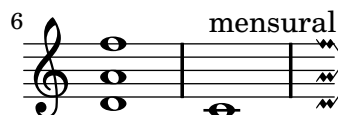
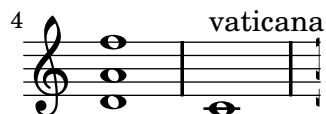
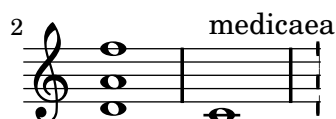
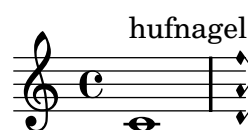
```

c1~"medicaea" \break
<d a' f'>1

\override Staff.Custos #'style = #'vaticana
c1~"vaticana" \break
<d a' f'>1

\override Staff.Custos #'style = #'mensural
c1~"mensural" \break
<d a' f'>1
}

```



Customizing fretboard fret diagrams

Fret diagram properties can be set through 'fret-diagram-details. For FretBoard fret diagrams, overrides are applied to the FretBoards.FretBoard object. Like Voice, FretBoards is a bottom level context, therefore can be omitted in property overrides.

```

\include "predefined-guitar-fretboards.ly"
\storePredefinedDiagram \chordmode { c' }
                                #guitar-tuning
                                #"x;1-1-(;3-2;3-3;3-4;1-1-);"

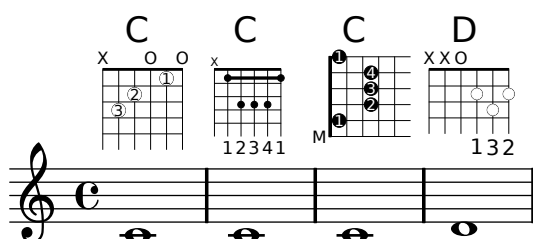
<<
  \new ChordNames {
    \chordmode { c1 | c | c | d }
  }
  \new FretBoards {
    % Set global properties of fret diagram
    \override FretBoards.FretBoard #'size = #'1.2

```

```

\override FretBoard
  #'(fret-diagram-details finger-code) = #'in-dot
\override FretBoard
  #'(fret-diagram-details dot-color) = #'white
\chordmode {
  c
  \once \override FretBoard #'size = #'1.0
  \once \override FretBoard
    #'(fret-diagram-details barre-type) = #'straight
  \once \override FretBoard
    #'(fret-diagram-details dot-color) = #'black
  \once \override FretBoard
    #'(fret-diagram-details finger-code) = #'below-string
  c'
  \once \override FretBoard
    #'(fret-diagram-details barre-type) = #'none
  \once \override FretBoard
    #'(fret-diagram-details number-type) = #'arabic
  \once \override FretBoard
    #'(fret-diagram-details orientation) = #'landscape
  \once \override FretBoard
    #'(fret-diagram-details mute-string) = #'M"
  \once \override FretBoard
    #'(fret-diagram-details label-dir) = #LEFT
  \once \override FretBoard
    #'(fret-diagram-details dot-color) = #'black
  c'
  \once \override FretBoard
    #'(fret-diagram-details finger-code) = #'below-string
  \once \override FretBoard
    #'(fret-diagram-details dot-radius) = #0.35
  \once \override FretBoard
    #'(fret-diagram-details dot-position) = #0.5
  \once \override FretBoard
    #'(fret-diagram-details fret-count) = #3
  d
}
}
\new Voice {
  c'1 | c' | c' | d'
}
>>

```



Customizing markup fret diagrams

Fret diagram properties can be set through 'fret-diagram-details. For markup fret diagrams, overrides can be applied to the Voice.TextScript object or directly to the markup.

```
<<
\chords { c1 | c | c | d }

\new Voice = "mel" {
  \textLengthOn
  % Set global properties of fret diagram
  \override TextScript #'size = #'1.2
  \override TextScript
    #'(fret-diagram-details finger-code) = #'in-dot
  \override TextScript
    #'(fret-diagram-details dot-color) = #'white

  %% C major for guitar, no barre, using defaults
  % terse style
  c'1^\markup { \fret-diagram-terse #"x;3-3;2-2;o;1-1;o;" }

  %% C major for guitar, barred on third fret
  % verbose style
  % size 1.0
  % roman fret label, finger labels below string, straight barre
  c'1^\markup {
    % standard size
    \override #'(size . 1.0) {
      \override #'(fret-diagram-details . (
        (number-type . roman-lower)
        (finger-code . in-dot)
        (barre-type . straight))) {
        \fret-diagram-verbose #'((mute 6)
          (place-fret 5 3 1)
          (place-fret 4 5 2)
          (place-fret 3 5 3)
          (place-fret 2 5 4)
          (place-fret 1 3 1)
          (barre 5 1 3))
        }
      }
    }
  }

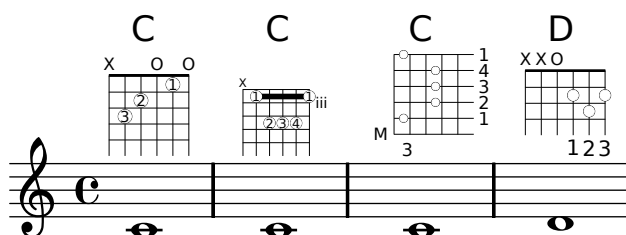
  %% C major for guitar, barred on third fret
  % verbose style
  % landscape orientation, arabic numbers, M for mute string
  % no barre, fret label down or left, small mute label font
  c'1^\markup {
    \override #'(fret-diagram-details . (
      (finger-code . below-string)
      (number-type . arabic)
      (label-dir . -1)
      (mute-string . "M")
    )
  }
}
```

```

        (orientation . landscape)
        (barre-type . none)
        (xo-font-magnification . 0.4)
        (xo-padding . 0.3))) {
\ fret-diagram-verbose #'(mute 6)
        (place-fret 5 3 1)
        (place-fret 4 5 2)
        (place-fret 3 5 3)
        (place-fret 2 5 4)
        (place-fret 1 3 1)
        (barre 5 1 3))
    }
}

%% simple D chord
% terse style
% larger dots, centered dots, fewer frets
% label below string
d'1^\markup {
  \override #'(fret-diagram-details . (
    (finger-code . below-string)
    (dot-radius . 0.35)
    (dot-position . 0.5)
    (fret-count . 3))) {
    \fret-diagram-terse #"x;x;o;2-1;3-2;2-3;"
  }
}
}
>>

```



Display bracket with only one staff in a system

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

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

```

\markup \left-column {
  \score {
    \new StaffGroup <<
      % Must be lower than the actual number of staff lines
      \override StaffGroup.SystemStartBracket #'collapse-height = #1
      \override Score.SystemStartBar #'collapse-height = #1

```

```

    \new Staff {
      c'1
    }
  >>
  \layout { }
}
\null
\score {
  \new PianoStaff <<
    \override PianoStaff.SystemStartBrace #'collapse-height = #1
    \override Score.SystemStartBar #'collapse-height = #1
    \new Staff {
      c'1
    }
  >>
  \layout { }
}
}

```



Displaying grob ancestry

When working with grob callbacks, it can be helpful to understand a grob’s “ancestry”. Most grobs have “parents” which influence the positioning of the grob. X- and Y-parents influence the horizontal and vertical positions for the grob, respectively. Additionally, each parent may have parents of its own.

Unfortunately, there are several aspects of a grob’s ancestry that can lead to confusion:

- The types of parents a grob has may depend on context.
- For some grobs, the X- and Y-parents are the same.
- A particular “ancestor” may be related to a grob in multiple ways.
- The concept of “generations” is misleading.

For example, the `System` grob can be both parent (on the Y-side) and grandparent (twice on the X-side) to a `VerticalAlignment` grob.

This macro prints (to the console) a textual representation of a grob’s ancestry.

When called this way

```

{
  \once \override NoteHead #'before-line-breaking = #display-ancestry
  c4
}

```

The following output is generated:

```

NoteHead X,Y: NoteColumn
  X: PaperColumn
  X,Y: System
Y: VerticalAxisGroup
  X: NonMusicalPaperColumn
  X,Y: System
Y: VerticalAlignment
  X: NonMusicalPaperColumn
  X,Y: System
Y: System

#(define (grob-name grob)
  (if (ly:grob? grob)
      (assoc-ref (ly:grob-property grob 'meta) 'name)
      #f))

#(define (get-ancestry grob)
  (if (not (null? (ly:grob-parent grob X)))
      (list (grob-name grob)
            (get-ancestry (ly:grob-parent grob X))
            (get-ancestry (ly:grob-parent grob Y)))
      (grob-name grob)))

#(define (format-ancestry lst padding)
  (string-append
    (symbol->string (car lst))
    "\n"
    (let ((X-ancestry
          (if (list? (cadr lst))
              (format-ancestry (cadr lst) (+ padding 3))
              (symbol->string (cadr lst))))
          (Y-ancestry
          (if (list? (caddr lst))
              (format-ancestry (caddr lst) (+ padding 3))
              (symbol->string (caddr lst))))))
    (if (equal? X-ancestry Y-ancestry)
        (string-append
          (format #f "~&")
          (make-string padding #\space)
          "X,Y: "
          (if (list? (cadr lst))
              (format-ancestry (cadr lst) (+ padding 5))
              (symbol->string (cadr lst))))
        (string-append
          (format #f "~&")
          (make-string padding #\space)
          "X: " X-ancestry
          "\n"
          (make-string padding #\space)
          "Y: " Y-ancestry
          (format #f "~&"))))
    (format #f "~&"))

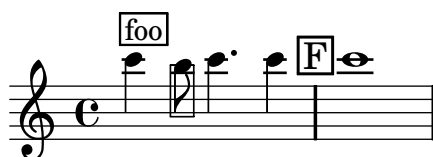
```



```

\revert Stem #'stencil
c4. c4
\mark "F"
c1
}

```



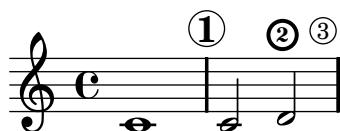
Drawing circles around various objects

The `\circle` markup command draws circles around various objects, for example fingering indications. For other objects, specific tweaks may be required: this example demonstrates two strategies for rehearsal marks and measure numbers.

```

\relative c' {
  c1
  \set Score.markFormatter =
    #(lambda (mark context)
      (make-circle-markup (format-mark-numbers mark context)))
  \mark \default
  c2 d^\markup {
    \override #'(thickness . 3) {
      \circle \finger 2
    }
  }
}
\override Score.BarNumber #'break-visibility = #all-visible
\override Score.BarNumber #'stencil =
  #(make-stencil-circler 0.1 0.25 ly:text-interface::print)
}

```



Dynamics custom text spanner postfix

Postfix functions for custom crescendo text spanners. The spanners should start on the first note of the measure. One has to use `-\mycresc`, otherwise the spanner start will rather be assigned to the next note.

```

% Two functions for (de)crescendo spanners where you can explicitly give the
% spanner text.
mycresc =
  #(define-music-function (parser location mymarkup) (markup?)
    (make-music 'CrescendoEvent
      'span-direction START

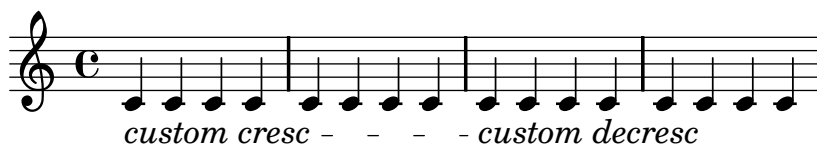
```

```

        'span-type 'text
        'span-text mymarkup))
mydecresc =
#(define-music-function (parser location mymarkup) (markup?)
  (make-music 'DecrescendoEvent
    'span-direction START
    'span-type 'text
    'span-text mymarkup))

\relative c' {
  c4-\mycresc "custom cresc" c4 c4 c4 |
  c4 c4 c4 c4 |
  c4-\mydecresc "custom decresc" c4 c4 c4 |
  c4 c4\! c4 c4
}

```



Dynamics text spanner postfix

The `\cresc`, `\dim` and `\decresc` spanners can now be redefined as postfix operators and produce one text spanner. Defining custom spanners is also easy. Hairpin and text crescendi can be easily mixed. `\<` and `\>` produce hairpins by default, `\cresc` etc. produce text spanners by default.

% Some sample text dynamic spanners, to be used as postfix operators

```

crpoco =
#(make-music 'CrescendoEvent
  'span-direction START
  'span-type 'text
  'span-text "cresc. poco a poco")
% Redefine the existing \cresc, \dim and \decresc commands to use postfix syntax
cresc =
#(make-music 'CrescendoEvent
  'span-direction START
  'span-type 'text
  'span-text "cresc.")

dim =
#(make-music 'DecrescendoEvent
  'span-direction START
  'span-type 'text
  'span-text "dim.")

decresc =
#(make-music 'DecrescendoEvent
  'span-direction START
  'span-type 'text
  'span-text "decresc.")

```

```
\relative c' {
  c4\cresc d4 e4 f4 |
  g4 a4\! b4\crpoco c4 |
  c4 d4 e4 f4 |
  g4 a4\! b4\< c4 |
  g4\dim a4 b4\decresc c4\!
}
```



Fine-tuning pedal brackets

The appearance of pedal brackets may be altered in different ways.

```
\paper { ragged-right = ##f }
\relative c' {
  c2\sostenutoOn c
  c2\sostenutoOff c
  \once \override Staff.PianoPedalBracket #'shorten-pair = #'(-7 . -2)
  c2\sostenutoOn c
  c2\sostenutoOff c
  \once \override Staff.PianoPedalBracket #'edge-height = #'(0 . 3)
  c2\sostenutoOn c
  c2\sostenutoOff c
}
```



Forcing horizontal shift of notes

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

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



Fret diagrams explained and developed

This snippet shows many possibilities for obtaining and tweaking fret diagrams.

```
<<
\chords {
  a2 a
  \repeat unfold 3 {
    c c c d d
  }
}

\new Voice = "mel" {
  \textLengthOn
  % Set global properties of fret diagram
  \override TextScript #'size = #1.2
  \override TextScript
    #'fret-diagram-details #'finger-code = #'below-string
  \override TextScript #'fret-diagram-details #'dot-color = #'black

  %% A chord for ukulele
  a'2^\markup {
    \override #'(fret-diagram-details . (
      (string-count . 4)
      (dot-color . white)
      (finger-code . in-dot))) {
      \fret-diagram #"4-2-2;3-1-1;2-o;1-o;"
    }
  }

  %% A chord for ukulele, with formatting defined in definition string
  % 1.2 * size, 4 strings, 4 frets, fingerings below string
  % dot radius .35 of fret spacing, dot position 0.55 of fret spacing
  a'2^\markup {
    \override #'(fret-diagram-details . (
      (dot-color . white)
      (open-string . "o"))) {
      \fret-diagram #"s:1.2;w:4;h:3;f:2;d:0.35;p:0.55;4-2-2;3-1-1;2-o;1-o;"
    }
  }

  %% These chords will be in normal orientation

  %% C major for guitar, barred on third fret
  % verbose style
  % roman fret label, finger labels below string, straight barre
  c'2^\markup {
    % 110% of default size
    \override #'(size . 1.1) {
      \override #'(fret-diagram-details . (
```

```

        (number-type . roman-lower)
        (finger-code . below-string)
        (barre-type . straight))) {
\ fret-diagram-verbose #'((mute 6)
                        (place-fret 5 3 1)
                        (place-fret 4 5 2)
                        (place-fret 3 5 3)
                        (place-fret 2 5 4)
                        (place-fret 1 3 1)
                        (barre 5 1 3))
    }
}
}

%% C major for guitar, barred on third fret
%% Double barre used to test barre function
% verbose style
c'2^\markup {
  % 110% of default size
  \override #'(size . 1.1) {
    \override #'(fret-diagram-details . (
      (number-type . arabic)
      (dot-label-font-mag . 0.9)
      (finger-code . in-dot)
      (fret-label-font-mag . 0.6)
      (fret-label-vertical-offset . 0)
      (label-dir . -1)
      (mute-string . "M")
      (xo-font-magnification . 0.4)
      (xo-padding . 0.3))) {
      \ fret-diagram-verbose #'((mute 6)
                              (place-fret 5 3 1)
                              (place-fret 4 5 2)
                              (place-fret 3 5 3)
                              (place-fret 2 5 4)
                              (place-fret 1 3 1)
                              (barre 4 2 5)
                              (barre 5 1 3))
    }
  }
}

%% C major for guitar, with capo on third fret
% verbose style
c'2^\markup {
  % 110% of default size
  \override #'(size . 1.1) {
    \override #'(fret-diagram-details . (
      (number-type . roman-upper)
      (dot-label-font-mag . 0.9)
      (finger-code . none)
      (fret-label-vertical-offset . 0.5)

```

```

        (xo-font-magnification . 0.4)
        (xo-padding . 0.3))) {
  \fret-diagram-verbose #'((mute 6)
                           (capo 3)
                           (open 5)
                           (place-fret 4 5 1)
                           (place-fret 3 5 2)
                           (place-fret 2 5 3)
                           (open 1))
    }
  }
}

%% simple D chord
d'2^\markup {
  \override #'(fret-diagram-details . (
    (finger-code . below-string)
    (dot-radius . 0.35)
    (string-thickness-factor . 0.3)
    (dot-position . 0.5)
    (fret-count . 3))) {
    \fret-diagram-terse #"x;x;o;2-1;3-2;2-3;"
  }
}

%% simple D chord, large top fret thickness
d'2^\markup {
  \override #'(fret-diagram-details . (
    (finger-code . below-string)
    (dot-radius . 0.35)
    (dot-position . 0.5)
    (top-fret-thickness . 7)
    (fret-count . 3))) {
    \fret-diagram-terse #"x;x;o;2-1;3-2;2-3;"
  }
}

% These chords will be in landscape orientation
\override TextScript
  #'fret-diagram-details #'orientation = #'landscape

%% C major for guitar, barred on third fret
% verbose style
% roman fret label, finger labels below string, straight barre
c'2^\markup {
  % 110% of default size
  \override #'(size . 1.1) {
    \override #'(fret-diagram-details . (
      (number-type . roman-lower)
      (finger-code . below-string)
      (barre-type . straight))) {
      \fret-diagram-verbose #'((mute 6)

```

```

        (place-fret 5 3 1)
        (place-fret 4 5 2)
        (place-fret 3 5 3)
        (place-fret 2 5 4)
        (place-fret 1 3 1)
        (barre 5 1 3))
    }
  }
}

```

```

%% C major for guitar, barred on third fret
%% Double barre used to test barre function
% verbose style
c'2^\markup {
  % 110% of default size
  \override #'(size . 1.1) {
    \override #'(fret-diagram-details . (
      (number-type . arabic)
      (dot-label-font-mag . 0.9)
      (finger-code . in-dot)
      (fret-label-font-mag . 0.6)
      (fret-label-vertical-offset . 0)
      (label-dir . -1)
      (mute-string . "M")
      (xo-font-magnification . 0.4)
      (xo-padding . 0.3))) {
      \fret-diagram-verbose #'((mute 6)
        (place-fret 5 3 1)
        (place-fret 4 5 2)
        (place-fret 3 5 3)
        (place-fret 2 5 4)
        (place-fret 1 3 1)
        (barre 4 2 5)
        (barre 5 1 3))
      }
    }
  }
}

```

```

%% C major for guitar, with capo on third fret
% verbose style
c'2^\markup {
  % 110% of default size
  \override #'(size . 1.1) {
    \override #'(fret-diagram-details . (
      (number-type . roman-upper)
      (dot-label-font-mag . 0.9)
      (finger-code . none)
      (fret-label-vertical-offset . 0.5)
      (xo-font-magnification . 0.4)
      (xo-padding . 0.3))) {
      \fret-diagram-verbose #'((mute 6)
        (capo 3)

```

```

                                (open 5)
                                (place-fret 4 5 1)
                                (place-fret 3 5 2)
                                (place-fret 2 5 3)
                                (open 1))
      }
    }
  }

%% simple D chord
d'2^\markup {
  \override #'(fret-diagram-details . (
    (finger-code . below-string)
    (dot-radius . 0.35)
    (dot-position . 0.5)
    (fret-count . 3))) {
    \fret-diagram-terse #"x;x;o;2-1;3-2;2-3;"
  }
}

%% simple D chord, large top fret thickness
d'2^\markup {
  \override #'(fret-diagram-details . (
    (finger-code . below-string)
    (dot-radius . 0.35)
    (dot-position . 0.5)
    (top-fret-thickness . 7)
    (fret-count . 3))) {
    \fret-diagram-terse #"x;x;o;2-1;3-2;2-3;"
  }
}

% These chords will be in opposing-landscape orientation
\override TextScript #'fret-diagram-details
  #'orientation = #'opposing-landscape

%% C major for guitar, barred on third fret
% verbose style
% roman fret label, finger labels below string, straight barre
c'2^\markup {
  % 110% of default size
  \override #'(size . 1.1) {
    \override #'(fret-diagram-details . (
      (number-type . roman-lower)
      (finger-code . below-string)
      (barre-type . straight))) {
      \fret-diagram-verbose #'((mute 6)
        (place-fret 5 3 1)
        (place-fret 4 5 2)
        (place-fret 3 5 3)
        (place-fret 2 5 4)
        (place-fret 1 3 1)

```

```

                                (barre 5 1 3))
      }
    }
  }

%% C major for guitar, barred on third fret
%% Double barre used to test barre function
% verbose style
c'2^\markup {
  % 110% of default size
  \override #'(size . 1.1) {
    \override #'(fret-diagram-details . (
      (number-type . arabic)
      (dot-label-font-mag . 0.9)
      (finger-code . in-dot)
      (fret-label-font-mag . 0.6)
      (fret-label-vertical-offset . 0)
      (label-dir . -1)
      (mute-string . "M")
      (xo-font-magnification . 0.4)
      (xo-padding . 0.3))) {
      \fret-diagram-verbose #'((mute 6)
        (place-fret 5 3 1)
        (place-fret 4 5 2)
        (place-fret 3 5 3)
        (place-fret 2 5 4)
        (place-fret 1 3 1)
        (barre 4 2 5)
        (barre 5 1 3))
      }
    }
  }

%% C major for guitar, with capo on third fret
% verbose style
c'2^\markup {
  % 110% of default size
  \override #'(size . 1.1) {
    \override #'(fret-diagram-details . (
      (number-type . roman-upper)
      (dot-label-font-mag . 0.9)
      (finger-code . none)
      (fret-label-vertical-offset . 0.5)
      (xo-font-magnification . 0.4)
      (xo-padding . 0.3))) {
      \fret-diagram-verbose #'((mute 6)
        (capo 3)
        (open 5)
        (place-fret 4 5 1)
        (place-fret 3 5 2)
        (place-fret 2 5 3)
        (open 1))
      }
    }
  }

```

```

    }
  }
}

%% simple D chord
d'2^\markup {
  \override #'(fret-diagram-details . (
    (finger-code . below-string)
    (dot-radius . 0.35)
    (dot-position . 0.5)
    (fret-count . 3))) {
    \fret-diagram-terse #"x;x;o;2-1;3-2;2-3;"
  }
}

%% simple D chord, large top fret thickness
d'2^\markup {
  \override #'(fret-diagram-details . (
    (finger-code . below-string)
    (dot-radius . 0.35)
    (dot-position . 0.5)
    (top-fret-thickness . 7)
    (fret-count . 3))) {
    \fret-diagram-terse #"x;x;o;2-1;3-2;2-3;"
  }
}
}
}
>>

```

The image displays two staves of musical notation, each featuring a series of guitar fret diagrams. The first staff contains eight chords: A, A, C, C, C, D, D, and C. The second staff contains nine chords: C, C, D, D, C, C, C, D, and D. Each chord is represented by a fret diagram showing the fret number, fingerings (e.g., 1, 2, 3, 4, 5), and a corresponding musical note on a staff. The fret diagrams are labeled with their respective chord names (A, C, D) and include fingerings and fret numbers. The musical notation is in treble clef with a key signature of one flat (B-flat).

Hairpins with different line styles

Hairpins can take any style from `line-interface` - dashed-line, dotted-line, line, trill or zigzag.

```

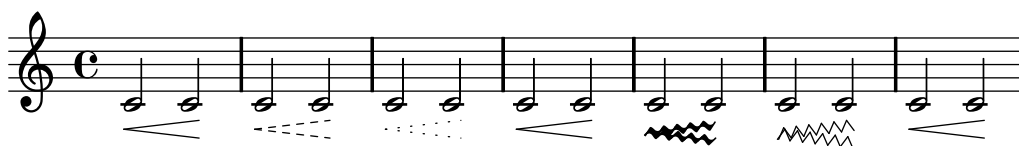
\relative c' {
  c2\< c\!
  \override Hairpin #'style = #'dashed-line
  c2\< c\!
  \override Hairpin #'style = #'dotted-line

```

```

c2\< c\!
\override Hairpin #'style = #'line
c2\< c\!
\override Hairpin #'style = #'trill
c2\< c\!
\override Hairpin #'style = #'zigzag
c2\< c\!
\revert Hairpin #'style
c2\< c\!
}

```



Horizontally aligning custom dynamics (e.g. "sempre pp" "piu f" "subito p")

Some dynamic expressions involve additional text, like “sempre pp”. Since dynamics are usually centered under the note, the `\pp` would be displayed way after the note it applies to.

To correctly align the “sempre pp” horizontally, so that it is aligned as if it were only the `\pp`, there are several approaches:

- * Simply use `\once\override DynamicText #'X-offset = #-9.2` before the note with the dynamics to manually shift it to the correct position. Drawback: This has to be done manually each time you use that dynamic markup... * Add some padding (`#:hspace 7.1`) into the definition of your custom dynamic mark, so that after LilyPond center-aligns it, it is already correctly aligned. Drawback: The padding really takes up that space and does not allow any other markup or dynamics to be shown in that position.

- * Shift the dynamic script `\once\override ... #'X-offset = ...` Drawback: `\once\override` is needed for every invocation!

- * Set the dimensions of the additional text to 0 (using `#:with-dimensions '(0 . 0) '(0 . 0)`). Drawback: To LilyPond “sempre” has no extent, so it might put other stuff there and create collisions (which are not detected by the collision detection!). Also, there seems to be some spacing, so it’s not exactly the same alignment as without the additional text

- * Add an explicit shifting directly inside the scheme function for the dynamic-script.

- * Set an explicit alignment inside the dynamic-script. By default, this won’t have any effect, only if one sets `X-offset`! Drawback: One needs to set `DynamicText #'X-offset`, which will apply to all dynamic texts! Also, it is aligned at the right edge of the additional text, not at the center of pp.

```
\header { title = "Horizontally aligning custom dynamics" }
```

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

```

% Solution 1: Using a simple markup with a particular halign value
% Drawback: It's a markup, not a dynamic command, so \dynamicDown
%           etc. will have no effect
semppMarkup = \markup { \halign #1.4 \italic "sempre" \dynamic "pp" }

```

```
% Solution 2: Using a dynamic script & shifting with
```

```

%           \once \override ... #'X-offset = ..
% Drawback: \once \override needed for every invocation
semppK =
#(make-dynamic-script
  (markup #:line
    (#:normal-text
      #:italic "sempre"
      #:dynamic "pp"))))

% Solution 3: Padding the dynamic script so the center-alignment
%           puts it at the correct position
% Drawback: the padding really reserves the space, nothing else can be there
semppT =
#(make-dynamic-script
  (markup #:line
    (#:normal-text
      #:italic "sempre"
      #:dynamic "pp"
      #:hspace 7.1)))

% Solution 4: Dynamic, setting the dimensions of the additional text to 0
% Drawback: To lilypond "sempre" has no extent, so it might put
%           other stuff there => collisions
% Drawback: Also, there seems to be some spacing, so it's not exactly the
%           same alignment as without the additional text
semppM =
#(make-dynamic-script
  (markup #:line
    (#:with-dimensions '(0 . 0) '(0 . 0)
      #:right-align
      #:normal-text
      #:italic "sempre"
      #:dynamic "pp"))))

% Solution 5: Dynamic with explicit shifting inside the scheme function
semppG =
#(make-dynamic-script
  (markup #:hspace 0
    #:translate '(-18.85 . 0)
    #:line (#:normal-text
      #:italic "sempre"
      #:dynamic "pp"))))

% Solution 6: Dynamic with explicit alignment. This has only effect
%           if one sets X-offset!
% Drawback: One needs to set DynamicText #'X-offset!
% Drawback: Aligned at the right edge of the additional text,
%           not at the center of pp
semppMII =
#(make-dynamic-script
  (markup #:line (#:right-align
    #:normal-text

```

```

        #:italic "sempre"
        #:dynamic "pp"))))

\context StaffGroup <<
  \context Staff = "s" <<
    \set Staff.instrumentName = #"Normal"
    \relative c'' {
      \key es \major
      c4\pp c\p c c | c\ff c c\pp c
    }
  >>
  \context Staff = "sMarkup" <<
    \set Staff.instrumentName = \markup \column { Normal markup }
    \relative c'' {
      \key es \major
      c4-\semppMarkup c\p c c | c\ff c c-\semppMarkup c
    }
  >>
  \context Staff = "sK" <<
    \set Staff.instrumentName = \markup \column { Explicit shifting }
    \relative c'' {
      \key es \major
      \once \override DynamicText #'X-offset = #-9.2
      c4\semppK c\p c c
      c4\ff c
      \once \override DynamicText #'X-offset = #-9.2
      c4\semppK c
    }
  >>
  \context Staff = "sT" <<
    \set Staff.instrumentName = \markup \column { Right padding }
    \relative c'' {
      \key es \major
      c4\semppT c\p c c | c\ff c c\semppT c
    }
  >>
  \context Staff = "sM" <<
    \set Staff.instrumentName = \markup \column { Setting dimension "to zero" }
    \relative c'' {
      \key es \major
      c4\semppM c\p c c | c\ff c c\semppM c
    }
  >>
  \context Staff = "sG" <<
    \set Staff.instrumentName = \markup \column { Shifting inside dynamics }
    \relative c'' {
      \key es \major
      c4\semppG c\p c c | c\ff c c\semppG c
    }
  >>
  \context Staff = "sMII" <<
    \set Staff.instrumentName = \markup \column { Alignment inside dynamics }

```

```

\relative c'' {
  \key es \major
  % Setting to ##f (false) gives the same result
  \override DynamicText #'X-offset = #0
  c4\sempMII c\p c c | c\ff c c\sempMII c
}
>>
>>

```

Horizontally aligning custom dynamics

The image displays seven musical staves, each illustrating a different method for horizontally aligning custom dynamics in a score. The music is in E-flat major (two flats) and 4/4 time. The first measure contains notes with dynamics *pp*, *p*, and *p*. The second measure contains notes with dynamics *ff* and *pp*. The staves are labeled on the left:

- Normal:** Shows the default alignment of dynamics.
- Normal markup:** Shows dynamics aligned with the notes using markup.
- Explicit shifting:** Shows dynamics aligned with the notes using explicit shifting.
- Right padding:** Shows dynamics aligned with the notes using right padding.
- Setting dimension to zero:** Shows dynamics aligned with the notes by setting the dimension to zero.
- Shifting inside dynamics:** Shows dynamics aligned with the notes by shifting inside the dynamic text.
- Alignment inside dynamics:** Shows dynamics aligned with the notes by alignment inside the dynamic text.

How to change fret diagram position

If you want to move the position of a fret diagram, for example, to avoid collision, or to place it between two notes, you have various possibilities:

- 1) modify #'padding or #'extra-offset values (as shown in the first snippet)
- 2) you can add an invisible voice and attach the fret diagrams to the invisible notes in that voice (as shown in the second example).

If you need to move the fret according with a rhythmic position inside the bar (in the example, the third beat of the measure) the second example is better, because the fret is aligned with the third beat itself.

```

harmonies = \chordmode
{
  a8:13
% THE FOLLOWING IS THE COMMAND TO MOVE THE CHORD NAME
  \once \override ChordNames.ChordName #'extra-offset = #'(10 . 0)
  b8:13 s2.
% THIS LINE IS THE SECOND METHOD
  s4 s4 b4:13
}

\score
{
  <<
    \context ChordNames \harmonies
    \context Staff
    {a8^\markup { \fret-diagram #"6-x;5-0;4-2;3-0;2-0;1-2;" }
% THE FOLLOWING IS THE COMMAND TO MOVE THE FRET DIAGRAM
    \once \override TextScript #'extra-offset = #'(10 . 0)
    b4.~\markup { \fret-diagram #"6-x;5-2;4-4;3-2;2-2;1-4;" } b4. a8\break
% HERE IS THE SECOND METHOD
    <<
      { a8 b4.~ b4. a8}
      { s4 s4 s4^\markup { \fret-diagram #"6-x;5-2;4-4;3-2;2-2;1-4;" }
      }
    >>
  }
  >>
}

```

Inserting a caesura

Caesura marks can be created by overriding the 'text property of the `BreathingSign` object. A curved caesura mark is also available.

```

\relative c' {
  \override BreathingSign #'text = \markup {
    \musicglyph #"scripts.caesura.straight"
  }
  c8 e4. \breathe g8. e16 c4

  \override BreathingSign #'text = \markup {
    \musicglyph #"scripts.caesura.curved"
  }
  g8 e'4. \breathe g8. e16 c4
}

```



Keep change clefs full sized

When a clef is changed, the clef sign displayed is smaller than the initial clef. This can be overridden with `full-size-change`.

```

\relative c' {
  \clef "treble"
  c1
  \clef "bass"
  c1
  \clef "treble"
  c1
  \override Staff.Clef #'full-size-change = ##t
  \clef "bass"
  c1
  \clef "treble"
  c1
  \revert Staff.Clef #'full-size-change
  \clef "bass"
  c1
  \clef "treble"
  c1
}

```



Line arrows

Arrows can be applied to text-spanners and line-spanners (such as the Glissando).

```

\relative c' {
  \override TextSpanner #'bound-padding = #1.0
  \override TextSpanner #'style = #'line
  \override TextSpanner #'(bound-details right arrow) = ##t
  \override TextSpanner #'(bound-details left text) = #"f"
}

```

```

\override TextSpanner #'(bound-details right text) = #"gag"
\override TextSpanner #'(bound-details right padding) = #0.6

\override TextSpanner #'(bound-details right stencil-align-dir-y) = #CENTER
\override TextSpanner #'(bound-details left stencil-align-dir-y) = #CENTER

\override Glissando #'(bound-details right arrow) = ##t
\override Glissando #'arrow-length = #0.5
\override Glissando #'arrow-width = #0.25

a8\startTextSpan gis a4 b\glissando b,
g'4 c\stopTextSpan c2
}

```



Making an object invisible with the 'transparent' property

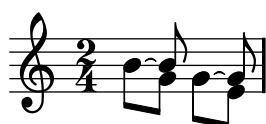
Setting the `transparent` property will cause an object to be printed in “invisible ink”: the object is not printed, but all its other behavior is retained. The object still takes up space, it takes part in collisions, and slurs, ties and beams can be attached to it.

This snippet demonstrates how to connect different voices using ties. Normally, ties only connect two notes in the same voice. By introducing a tie in a different voice, and blanking the first up-stem in that voice, the tie appears to cross voices.

```

\relative c'' {
  \time 2/4
  <<
  {
    \once \override Stem #'transparent = ##t
    \once \override Stem #'length = #8
    b8 ~ b\noBeam
    \once \override Stem #'transparent = ##t
    \once \override Stem #'length = #8
    g8 ~ g\noBeam
  }
  \\
  {
    b8 g g e
  }
  >>
}

```



Manually controlling beam positions

Beam positions may be controlled manually, by overriding the `positions` setting of the `Beam` grob.

```
\relative c' {
  \time 2/4
  % from upper staff-line (position 2) to center (position 0)
  \override Beam #'positions = #'(2 . 0)
  c8 c
  % from center to one above center (position 1)
  \override Beam #'positions = #'(0 . 1)
  c8 c
}
```

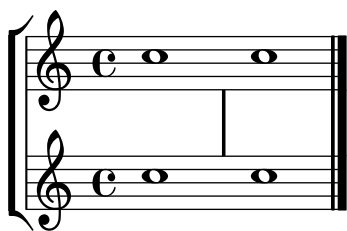


Mensurstriche layout (bar lines between the staves)

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

```
global = {
  \override Staff.BarLine #'transparent = ##t
  s1 s
  % the final bar line is not interrupted
  \revert Staff.BarLine #'transparent
  \bar "|."
}

\new StaffGroup \relative c'' {
  <<
    \new Staff { << \global { c1 c } >> }
    \new Staff { << \global { c c } >> }
  >>
}
```



Moving dotted notes in polyphony

When a dotted note in the upper voice is moved to avoid a collision with a note in another voice, the default is to move the upper note to the right. This behaviour can be over-ridden by using the `prefer-dotted-right` property of `NoteCollision`.

```

\new Staff \relative c' <<
{ f2. f4
  \override Staff.NoteCollision #'prefer-dotted-right = ##f
  f2. f4
  \override Staff.NoteCollision #'prefer-dotted-right = ##t
  f2. f4
}
\\
{ e4 e e e e e e e e e e }
>>

```



Moving slur positions vertically

The vertical position of a slur can be adjusted using the `positions` property of `Slur`. The property has 2 parameters, the first referring to the left end of the slur and the second to the right. The values of the parameters are not used by LilyPond to make an exact movement of the slur - instead it selects what placement of the slur looks best, taking into account the parameter values. Positive values move the slur up, and are appropriate for notes with stems down. Negative values move the slur down.

```

\relative c' {
  \stemDown
  e4( a)
  \override Slur #'positions = #'(1 . 1)
  e4( a)
  \override Slur #'positions = #'(2 . 2)
  e4( a)
  \override Slur #'positions = #'(3 . 3)
  e4( a)
  \override Slur #'positions = #'(4 . 4)
  e4( a)
  \override Slur #'positions = #'(5 . 5)
  e4( a)
  \override Slur #'positions = #'(0 . 5)
  e4( a)
  \override Slur #'positions = #'(5 . 0)
  e4( a)
}

```



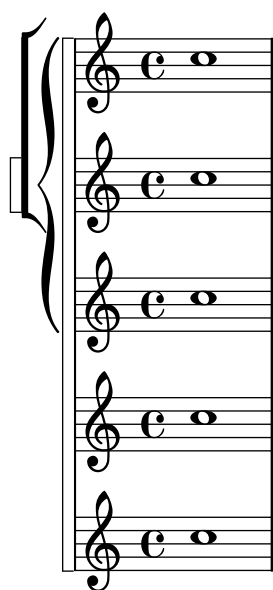
Nesting staves

The property `systemStartDelimiterHierarchy` can be used to make more complex nested staff groups. The command `\set StaffGroup.systemStartDelimiterHierarchy` takes an al-

phabetical list of the number of staves produced. Before each staff a system start delimiter can be given. It has to be enclosed in brackets and takes as much staves as the brackets enclose. Elements in the list can be omitted, but the first bracket takes always the complete number of staves. The possibilities are `SystemStartBar`, `SystemStartBracket`, `SystemStartBrace`, and `SystemStartSquare`.

```
\new StaffGroup
\relative c'' <<
  \set StaffGroup.systemStartDelimiterHierarchy
    = #'(SystemStartSquare (SystemStartBrace (SystemStartBracket a
      (SystemStartSquare b) ) c ) d)

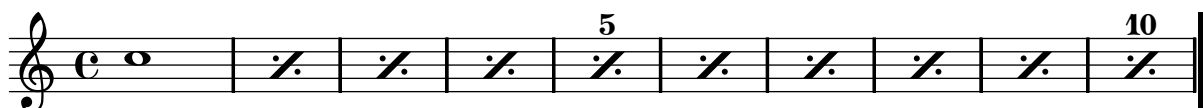
  \new Staff { c1 }
  \new Staff { c1 }
  \new Staff { c1 }
  \new Staff { c1 }
  \new Staff { c1 }
>>
```



Percent repeat count visibility

Percent repeat counters can be shown at regular intervals by setting the context property `repeatCountVisibility`.

```
\relative c'' {
  \set countPercentRepeats = ##t
  \set repeatCountVisibility = #(every-nth-repeat-count-visible 5)
  \repeat percent 10 { c1 } \break
  \set repeatCountVisibility = #(every-nth-repeat-count-visible 2)
  \repeat percent 6 { c1 d1 }
}
```

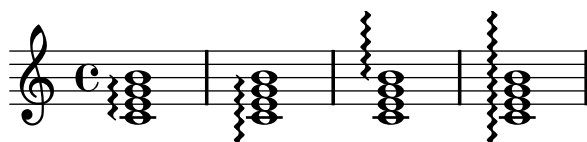




Positioning arpeggios

If you need to extend or shorten an arpeggio, you can modify the upper and lower start positions independently.

```
\relative c' {
  <c e g b>1\arpeggio
  \once \override Arpeggio #'positions = #'(-5 . 0)
  <c e g b>1\arpeggio
  \once \override Arpeggio #'positions = #'(0 . 5)
  <c e g b>1\arpeggio
  \once \override Arpeggio #'positions = #'(-5 . 5)
  <c e g b>1\arpeggio
}
```

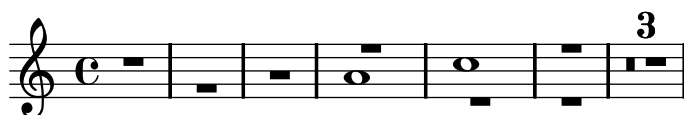


Positioning multi-measure rests

Unlike ordinary rests, there is no predefined command to change the staff position of a multi-measure rest symbol of either form by attaching it to a note. However, in polyphonic music multi-measure rests in odd-numbered and even-numbered voices are vertically separated. The positioning of multi-measure rests can be controlled as follows:

```
\relative c' {
  % MMR - Multi-Measure Rest
  % MMRs by default are set under the fourth line
  R1
  % They can be moved with an override
  \override MultiMeasureRest #'staff-position = #-2
  R1
  % A value of 0 is the default position;
  % the following trick moves the rest to the center line
  \override MultiMeasureRest #'staff-position = #-0.01
  R1
  % MMRs in odd-numbered voices are under the top line
  << { R1 } \ { a1 } >>
  % MMRs in even-numbered voices are under the bottom line
  << { c1 } \ { R1 } >>
  % They remain separated even in empty measures
  << { R1 } \ { R1 } >>
  % This brings them together even though there are two voices
  \compressFullBarRests
  <<
    \revert MultiMeasureRest #'staff-position
    { R1*3 }
    \
    \revert MultiMeasureRest #'staff-position
    { R1*3 }
  >>
```

```
>>
}
```



Positioning text markups inside slurs

Text markups need to have the `outside-staff-priority` property set to `false` in order to be printed inside slurs.

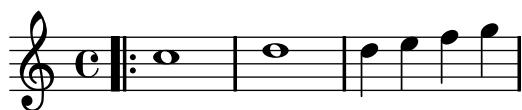
```
\relative c' {
  \override TextScript #'avoid-slur = #'inside
  \override TextScript #'outside-staff-priority = ##f
  c2(^{\markup { \halign #-10 \natural } d4.}) c8
}
```



Printing a repeat sign at the beginning of a piece

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

```
\relative c' {
  \once \override Score.BreakAlignment #'break-align-orders =
    #(make-vector 3 '(instrument-name
                      left-edge
                      ambitus
                      span-bar
                      breathing-sign
                      clef
                      key-signature
                      time-signature
                      staff-bar
                      custos
                      span-bar))
  \once \override Staff.TimeSignature #'space-alist =
    #'((first-note . (fixed-space . 2.0))
       (right-edge . (extra-space . 0.5))
       ;; free up some space between time signature
       ;; and repeat bar line
       (staff-bar . (extra-space . 1)))
  \bar "|:"
  c1
  d1
  d4 e f g
}
```



Printing bar numbers inside boxes or circles

Bar numbers can also be printed inside boxes or circles.

```
\relative c' {
  % Prevent bar numbers at the end of a line and permit them elsewhere
  \override Score.BarNumber #'break-visibility = #end-of-line-invisible
  \set Score.barNumberVisibility = #(every-nth-bar-number-visible 4)

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

  % Draw a box round the following bar number(s)
  \override Score.BarNumber #'stencil
    = #(make-stencil-boxer 0.1 0.25 ly:text-interface::print)
  \repeat unfold 5 { c1 }

  % Draw a circle round the following bar number(s)
  \override Score.BarNumber #'stencil
    = #(make-stencil-circler 0.1 0.25 ly:text-interface::print)
  \repeat unfold 4 { c1 } \bar "|."
}
```



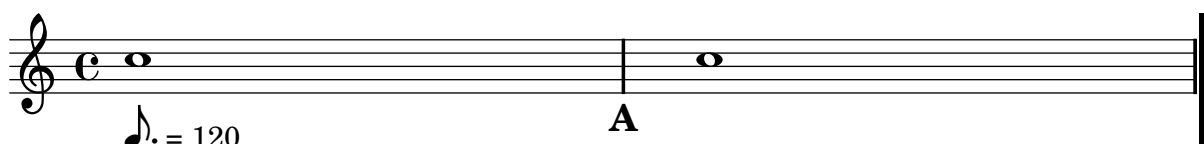
Printing metronome and rehearsal marks below the staff

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

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

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

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



Printing note names with and without an octave marker

The `NoteNames` context can be used to print the text value of notes. The `printOctaveNames` property turns on or off the representation of the octave of the note.

```
scale = \relative c' {
  a4 b c d
  e4 f g a
}

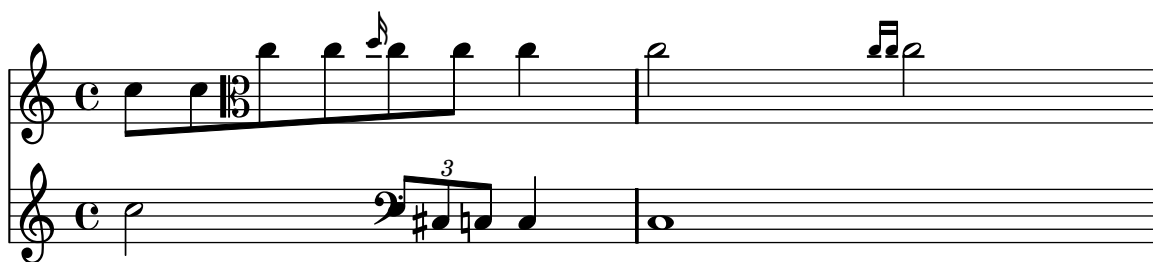
\new Staff {
  <<
    \scale
    \context NoteNames {
      \set printOctaveNames = ##f
    }
  >>
  R1
  <<
    \scale
    \context NoteNames {
      \set printOctaveNames = ##t
    }
  >>
}
```



Proportional strict notespacing

If `strict-note-spacing` is set spacing of notes is not influenced by bars or clefs within a system. Rather, they are placed just before the note that occurs at the same time. This may cause collisions.

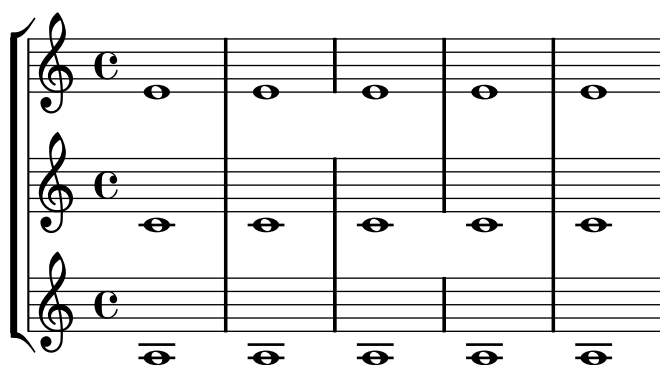
```
\relative c'' <<
  \override Score.SpacingSpanner #'strict-note-spacing = ##t
  \set Score.proportionalNotationDuration = #(ly:make-moment 1 16)
  \new Staff {
    c8[ c \clef alto c c \grace { d16 } c8 c] c4
    c2 \grace { c16[ c16] } c2
  }
  \new Staff {
    c2 \times 2/3 { c8 \clef bass cis,, c } c4
    c1
  }
>>
```



Removing connecting bar lines on StaffGroup PianoStaff or GrandStaff

By default, bar lines in StaffGroup, PianoStaff, or GrandStaff groups are connected between the staves. This behaviour can be overridden on a staff-by-staff basis.

```
\relative c' {
  \new StaffGroup <<
    \new Staff {
      e1 | e
      \once \override Staff.BarLine #'allow-span-bar = ##f
      e1 | e | e
    }
    \new Staff {
      c1 | c | c
      \once \override Staff.BarLine #'allow-span-bar = ##f
      c1 | c
    }
    \new Staff {
      a1 | a | a | a | a
    }
  >>
}
```



Removing the first empty line

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

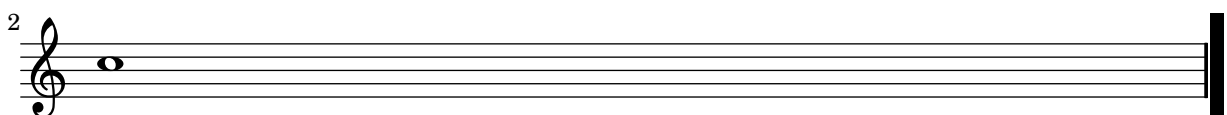
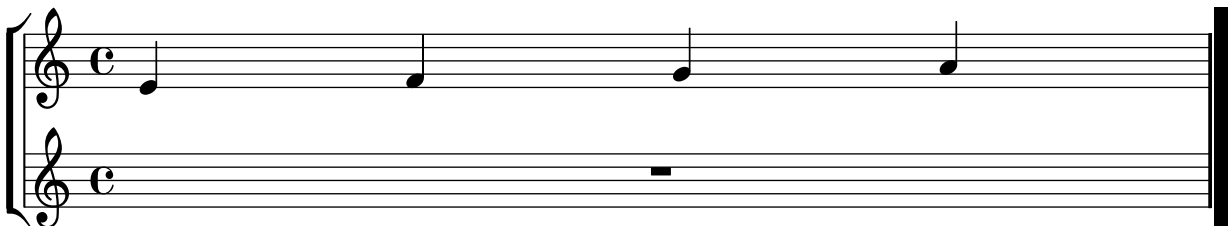
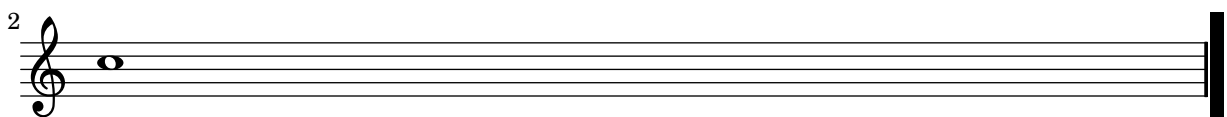
The lower staff of the second staff group is not removed, because the setting applies only to the specific staff inside of which it is written.

```
\layout {
  \context {
    \Staff \RemoveEmptyStaves
```

```

    % To use the setting globally, uncomment the following line:
    % \override VerticalAxisGroup #'remove-first = ##t
  }
}
\new StaffGroup <<
  \new Staff \relative c' {
    e4 f g a \break
    c1
  }
  \new Staff {
    % To use the setting globally, comment this line,
    % uncomment the line in the \layout block above
    \override Staff.VerticalAxisGroup #'remove-first = ##t
    R1 \break
    R
  }
>>
\new StaffGroup <<
  \new Staff \relative c' {
    e4 f g a \break
    c1
  }
  \new Staff {
    R1 \break
    R
  }
}
>>

```



Rest styles

Rests may be used in various styles.

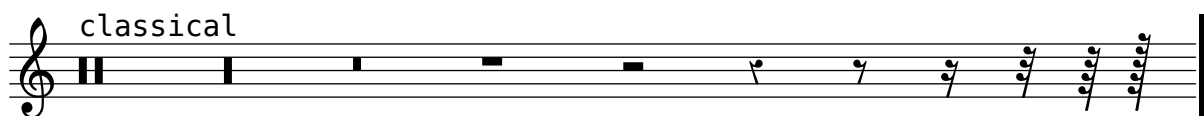
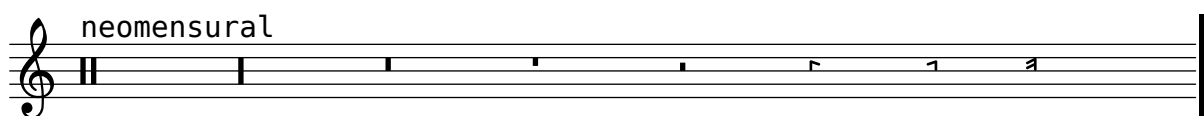
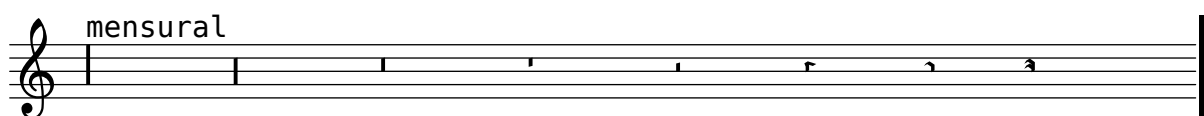
```
\layout {
  indent = 0
  \context {
    \Staff
    \remove "Time_signature_engraver"
  }
}

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

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

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

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



Rhythmic slashes

In “simple” lead-sheets, sometimes no actual notes are written, instead only “rhythmic patterns” and chords above the measures are notated giving the structure of a song. Such a feature is for example useful while creating/transcribing the structure of a song and also when sharing lead sheets with guitarists or jazz musicians. The standard support for this using `\repeat percent` is unsuitable here since the first beat has to be an ordinary note or rest. This example shows two solutions to this problem, by redefining ordinary rests to be printed as slashes. (If the duration of each beat is not a quarter note, replace the `r4` in the definitions with a rest of the appropriate duration).

```
% Macro to print single slash
rs = {
  \once \override Rest #'stencil = #ly:percent-repeat-item-interface::beat-slash
  \once \override Rest #'thickness = #0.48
  \once \override Rest #'slope = #1.7
  r4
}

% Function to print a specified number of slashes
comp = #(define-music-function (parser location count) (integer?)
  #{
    \override Rest #'stencil = #ly:percent-repeat-item-interface::beat-slash
    \override Rest #'thickness = #0.48
    \override Rest #'slope = #1.7
    \repeat unfold $count { r4 }
    \revert Rest #'stencil
  }#)

\score {
  \relative c' {
    c4 d e f |
    \rs \rs \rs \rs |
    \comp #4 |
  }
}
```



Separating key cancellations from key signature changes

By default, the accidentals used for key cancellations are placed adjacent to those for key signature changes. This behavior can be changed by overriding the `'break-align-orders` property of the `BreakAlignment` grob.

The value of `'break-align-orders` is a vector of length 3, with quoted lists of breakable items as elements. This example only modifies the second list, moving `key-cancellation` before `staff-bar`; by modifying the second list, break alignment behavior only changes in the middle of a system, not at the beginning or the end.

```
\new Staff {
```

```

\override Score.BreakAlignment #'break-align-orders =
  #'#((left-edge ambitus breathing-sign clef staff-bar
        key-cancellation key-signature time-signature custos)

        (left-edge ambitus breathing-sign clef key-cancellation
          staff-bar key-signature time-signature custos)

        (left-edge ambitus breathing-sign clef key-cancellation
          key-signature staff-bar time-signature custos))

\key des \major
c'1
\bar "||"
\key bes \major
c'1
}

```



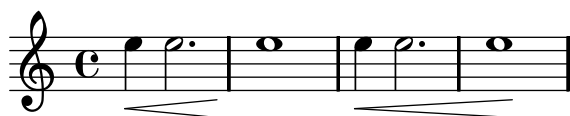
Setting hairpin behavior at bar lines

If the note which ends a hairpin falls on a downbeat, the hairpin stops at the bar line immediately preceding. This behavior can be controlled by overriding the 'to-barline property.

```

\relative c' {
  e4\< e2.
  e1\!
  \override Hairpin #'to-barline = ##f
  e4\< e2.
  e1\!
}

```



Setting system separators

System separators can be inserted between systems. Any markup can be used, but `\slashSeparator` has been provided as a sensible default.

```

\paper {
  system-separator-markup = \slashSeparator
}

```

```

notes = \relative c' {
  c1 | c \break
  c1 | c \break
  c1 | c
}

```

```

\book {

```

```
\score {  
  \new GrandStaff <<  
    \new Staff \notes  
    \new Staff \notes  
  >>  
}
```

The image displays three systems of musical notation, each consisting of a grand staff (treble and bass clefs) and a triplet marking. The first system is marked with a '3' and a triplet bracket. The second system is marked with a '3' and a triplet bracket. The third system is marked with a '5' and a triplet bracket. Each system shows a single note in the treble clef and a single note in the bass clef, with a triplet bracket indicating a triplet of notes. The notes are positioned on the second line of the treble clef and the second space of the bass clef. The first system has a '3' and a triplet bracket. The second system has a '3' and a triplet bracket. The third system has a '5' and a triplet bracket.

Showing the same articulation above and below a note or chord

By default, LilyPond does not allow the same articulation (e.g. an accent, a fermata, a flageolet, etc.) to be displayed above and below a note. For example, `c4_\fermata^\fermata` will only show a fermata below. The fermata above will simply be ignored. However, one can stick scripts (just like fingerings) inside a chord, which means it is possible to have as many articulations as desired. This approach has the advantage that it ignores the stem and positions the articulation relative to the note head. This can be seen in the case of the flageolets in the snippet. To mimic the behaviour of scripts outside a chord, 'add-stem-support' would be required. So, the solution is to write the note as a chord and add the articulations inside the `<...>`. The direction will always be above, but one can tweak this via a `\tweak`: `<c-\tweak #'direction #DOWN-\fermata^\fermata>`

```
% The same as \flageolet, just a little smaller
smallFlageolet =
#(let ((m (make-articulation "flageolet")))
  (set! (ly:music-property m 'tweaks)
    (acons 'font-size -2
      (ly:music-property m 'tweaks)))
  m)

\relative c' {
  s4^"wrong:"
  c_\fermata^\fermata % The second fermata is ignored!
  <e d'>^\smallFlageolet_\smallFlageolet

  % it works only if you wrap the note inside a chord. By default,
  % all articulations will be printed above, so you have to tweak
  % the direction.
  s4^"Works if written inside a chord:"
  <e-\tweak #'direction #DOWN -\smallFlageolet d'^\smallFlageolet>
  <e-\tweak #'direction #DOWN -\flageolet d'^\flageolet>
  <e-\tweak #'direction #DOWN -\smallFlageolet^\smallFlageolet>
  <e-\tweak #'direction #DOWN -\fermata^\fermata>
}
```

wrong: Works if written inside a chord:



String number extender lines

Make an extender line for string number indications, showing that a series of notes is supposed to be played all on the same string.

```
stringNumberSpanner =
#(define-music-function (parser location StringNumber) (string?)
  #{
    \override TextSpanner #'style = #'solid
    \override TextSpanner #'font-size = #-5
    \override TextSpanner #'(bound-details left stencil-align-dir-y) = #CENTER
    \override TextSpanner #'(bound-details left text) = \markup { \circle \number $StringNumber }
```

```
#})
```

```
\relative c {
  \clef "treble_8"
  \stringNumberSpanner "5"
  \textSpannerDown
  a8\startTextSpan
  b c d e f\stopTextSpan
  \stringNumberSpanner "4"
  g\startTextSpan a
  bes4 a g2\stopTextSpan
}
```



Suppressing warnings for clashing note columns

If notes from two voices with stems in the same direction are placed at the same position, and both voices have no shift or the same shift specified, the error message ‘**warning: ignoring too many clashing note columns**’ will appear when compiling the LilyPond file. This message can be suppressed by setting the ‘ignore-collision’ property of the `NoteColumn` object to `#t`.

```
ignore = \override NoteColumn #'ignore-collision = #t
```

```
\relative c' {
  <<
    \ignore
    { \stemDown f2 g }
    \\\
    { c2 c, }
  >>
}
```



Time signature in parentheses

The time signature can be enclosed within parentheses.

```
\relative c' {
  \override Staff.TimeSignature #'stencil = #(lambda (grob)
    (bracketify-stencil (ly:time-signature::print grob) Y 0.1 0.2 0.1))
  \time 2/4
  a4 b8 c
}
```



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

Sometimes, a time signature should not print the whole fraction (e.g. 7/4), but only the numerator (7 in this case). This can be easily done by using `\override Staff.TimeSignature #'style = #'single-digit` to change the style permanently. By using `\revert Staff.TimeSignature #'style`, this setting can be reversed. To apply the single-digit style to only one time signature, use the `\override` command and prefix it with a `\once`.

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



Transcription of Ancient music with incipit

As a workaround to get real incipits which are independent from the main score these are included as a markup into the field normally used for the instrument name. As for now lyrics can only be added as a direct markup. It doesn't unfortunately conform with the spacing of the main lyrics.

```
global = {
  \set Score.skipBars = ##t
  \key g \major
  \time 4/4
  %make the staff lines invisible on staves
  \override Staff.BarLine #'transparent = ##t
  \skip 1*8 % the actual music
  % let finis bar go through all staves
  \override Staff.BarLine #'transparent = ##f
  % finis bar
}
```

```

\bar "|."
}

discantusNotes = {
  \transpose c' c'' {
    \clef treble
    d'2. d'4 |
    b e' d'2 |
    c'4 e'4.( d'8 c' b |
    a4) b a2 |
    b4.( c'8 d'4) c'4 |
    \once \override NoteHead #'transparent = ##t c'1 |
    b\breve |
  }
}

discantusLyrics = \lyricmode {
  Ju -- bi -- |
  la -- te De -- |
  o, om --
  nis ter -- |
  ra, __ om- |
  "... " |
  -us. |
}

altusNotes = {
  \transpose c' c'' {
    \clef treble
    r2 g2. e4 fis g | % two bars
    a2 g4 e |
    fis g4.( fis16 e fis4) |
    g1 |
    \once \override NoteHead #'transparent = ##t g1 |
    g\breve |
  }
}

altusLyrics = \lyricmode {
  Ju -- bi -- la -- te | % two bars
  De -- o, om -- |
  nis ter -- ra, |
  "... " |
  -us. |
}

tenorNotes = {
  \transpose c' c' {
    \clef "treble_8"
    R1 |
    R1 |
    R1 |
  }
}

```

```

    r2 d'2. d'4 b e' | % two bars
    \once \override NoteHead #'transparent = ##t e'1 |
    d'\breve |
  }
}

tenorLyrics = \lyricmode {
  Ju -- bi -- la -- te | % two bars
  "... " |
  -us.
}

bassusNotes = {
  \transpose c' c' {
    \clef bass
    R1 |
    R1 |
    R1 |
    R1 |
    g2. e4 |
    \once \override NoteHead #'transparent = ##t e1 |
    g\breve |
  }
}

bassusLyrics = \lyricmode {
  Ju -- bi- |
  "... " |
  -us.
}

incipitDiscantus = \markup {
  \score {
    {
      \set Staff.instrumentName = #"Discantus "
      \override NoteHead #'style = #'neomensural
      \override Rest #'style = #'neomensural
      \override Staff.TimeSignature #'style = #'neomensural
      \cadenzaOn
      \clef "neomensural-c1"
      \key f \major
      \time 2/2
      c''1._"IV-" s2 %two bars
      \skip 1*8 % eight bars
    }
    \layout {
      \context {
        \Voice
        \remove "Ligature_bracket_engraver"
        \consists "Mensural_ligature_engraver"
      }
      line-width = 4.5\cm
    }
  }
}

```

```

    }
  }
}

incipitAltus = \markup {
  \score {
    {
      \set Staff.instrumentName = #"Altus "
      \override NoteHead #'style = #'neomensural
      \override Rest #'style = #'neomensural
      \override Staff.TimeSignature #'style = #'neomensural
      \cadenzaOn
      \clef "neomensural-c3"
      \key f \major
      \time 2/2
      r1          % one bar
      f'1._"IV-" s2  % two bars
      \skip 1*7 % seven bars
    }
    \layout {
      \context {
        \Voice
        \remove "Ligature_bracket_engraver"
        \consists "Mensural_ligature_engraver"
      }
      line-width = 4.5\cm
    }
  }
}

```

```

incipitTenor = \markup {
  \score {
    {
      \set Staff.instrumentName = #"Tenor "
      \override NoteHead #'style = #'neomensural
      \override Rest #'style = #'neomensural
      \override Staff.TimeSignature #'style = #'neomensural
      \cadenzaOn
      \clef "neomensural-c4"
      \key f \major
      \time 2/2
      r\longa    % four bars
      r\breve    % two bars
      r1         % one bar
      c'1._"IV-" s2  % two bars
      \skip 1    % one bar
    }
    \layout {
      \context {
        \Voice
        \remove "Ligature_bracket_engraver"
        \consists "Mensural_ligature_engraver"
      }
    }
  }
}

```

```

    }
    line-width = 4.5\cm
  }
}

incipitBassus = \markup {
  \score {
    {
      \set Staff.instrumentName = #"Bassus "
      \override NoteHead #'style = #'neomensural
      \override Rest #'style = #'neomensural
      \override Staff.TimeSignature #'style = #'neomensural
      \cadenzaOn
      \clef "bass"
      \key f \major
      \time 2/2
      % incipit
      r\maxima % eight bars
      f1._"IV-" s2 % two bars
    }
    \layout {
      \context {
        \Voice
        \remove "Ligature_bracket_engraver"
        \consists "Mensural_ligature_engraver"
      }
      line-width = 4.5\cm
    }
  }
}

%StaffGroup is used instead of ChoirStaff to get bar lines between systems
\score {
  <<
    \new StaffGroup = choirStaff <<
      \new Voice = "discantusNotes" <<
        \global
        \set Staff.instrumentName = \incipitDiscantus
        \discantusNotes
      >>
      \new Lyrics = "discantusLyrics" \lyricsto discantusNotes { \discantusLyrics }

    \new Voice = "altusNotes" <<
      \global
      \set Staff.instrumentName = \incipitAltus
      \altusNotes
    >>
    \new Lyrics = "altusLyrics" \lyricsto altusNotes { \altusLyrics }

    \new Voice = "tenorNotes" <<
      \global

```

```

    \set Staff.instrumentName = \incipitTenor
    \tenorNotes
  >>
  \new Lyrics = "tenorLyrics" \lyricsto tenorNotes { \tenorLyrics }

  \new Voice = "bassusNotes" <<
    \global
    \set Staff.instrumentName = \incipitBassus
    \bassusNotes
  >>
  >>
  \new Lyrics = "bassusLyrics" \lyricsto bassusNotes { \bassusLyrics }
  %Keep the bass lyrics outside of the staff group to avoid bar lines
  %between the lyrics.
  >>

\layout {
  \context {
    \Score
    % no bars in staves
    \override BarLine #'transparent = ##t
  }
  % the next three instructions keep the lyrics between the bar lines
  \context {
    \Lyrics
    \consists "Bar_engraver"
    \override BarLine #'transparent = ##t
  }
  \context {
    \StaffGroup
    \consists "Separating_line_group_engraver"
  }
  \context {
    \Voice
    % no slurs
    \override Slur #'transparent = ##t
    % Comment in the below "\remove" command to allow line
    % breaking also at those barlines where a note overlaps
    % into the next bar. The command is commented out in this
    % short example score, but especially for large scores, you
    % will typically yield better line breaking and thus improve
    % overall spacing if you comment in the following command.
    %\remove "Forbid_line_break_engraver"
  }
  indent = 5\cm
}
}

```

Discantus

Altus

Tenor

Bassus

IV-

IV-

IV-

IV-

8

8

4

Ju - bi - la - te De - o, om -

Ju - bi - la - te De - o, om -

- nis ter - ra, om - ... -us.

- nis ter - ra, ... -us.

Ju - bi - la - te ... -us.

Ju - bi - ... -us.

Tweaking clef properties

The command `\clef "treble_8"` is equivalent to setting `clefGlyph`, `clefPosition` (which controls the vertical position of the clef), `middleCPosition` and `clefOctavation`. A clef is printed when any of the properties except `middleCPosition` are changed.

Note that changing the glyph, the position of the clef, or the octavation does not in itself change the position of subsequent notes on the staff: the position of middle C must also be specified to do this. The positional parameters are relative to the staff center line, positive numbers displacing upwards, counting one for each line and space. The `clefOctavation` value would normally be set to 7, -7, 15 or -15, but other values are valid.

When a clef change takes place at a line break the new clef symbol is printed at both the end of the previous line and the beginning of the new line by default. If the warning clef at the end of the previous line is not required it can be suppressed by setting the `Staff` property `explicitClefVisibility` to the value `end-of-line-invisible`. The default behavior can be recovered with `\unset Staff.explicitClefVisibility`.

The following examples show the possibilities when setting these properties manually. On the first line, the manual changes preserve the standard relative positioning of clefs and notes, whereas on the second line, they do not.

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

```
{
  % The default treble clef
  c'1
  % The standard bass clef
```

```

\set Staff.clefGlyph = #"clefs.F"
\set Staff.clefPosition = #2
\set Staff.middleCPosition = #6
c'1
% The baritone clef
\set Staff.clefGlyph = #"clefs.C"
\set Staff.clefPosition = #4
\set Staff.middleCPosition = #4
c'1
% The standard choral tenor clef
\set Staff.clefGlyph = #"clefs.G"
\set Staff.clefPosition = #-2
\set Staff.clefOctavation = #-7
\set Staff.middleCPosition = #1
c'1
% A non-standard clef
\set Staff.clefPosition = #0
\set Staff.clefOctavation = #0
\set Staff.middleCPosition = #-4
c'1 \break

% The following clef changes do not preserve
% the normal relationship between notes and clefs:

\set Staff.clefGlyph = #"clefs.F"
\set Staff.clefPosition = #2
c'1
\set Staff.clefGlyph = #"clefs.G"
c'1
\set Staff.clefGlyph = #"clefs.C"
c'1
\set Staff.clefOctavation = #7
c'1
\set Staff.clefOctavation = #0
\set Staff.clefPosition = #0
c'1

% Return to the normal clef:

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

```



Tweaking grace layout within music

The layout of grace expressions can be changed throughout the music using the functions `add-grace-property` and `remove-grace-property`. The following example undefines the `Stem` direction for this grace, so that stems do not always point up, and changes the default note heads to crosses.

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



Using alternative flag styles

Alternative styles of flag on eighth and shorter notes can be displayed by overriding the `flag` property of `Stem`. Valid values are `modern-straight-flag` and `old-straight-flag`.

```
testnotes = {
  \autoBeamOff
  c8 d16 c32 d64 \acciaccatura { c8 } d64 r4
}

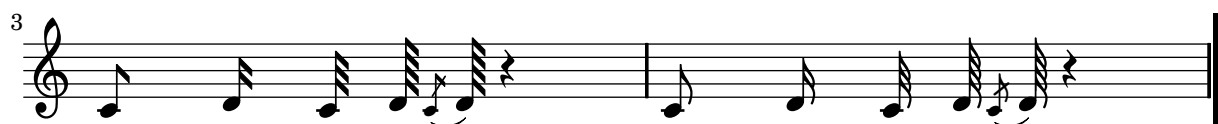
\relative c' {
  \time 2/4
  \testnotes

  \override Stem #'flag = #modern-straight-flag
  \testnotes

  \override Stem #'flag = #old-straight-flag
  \testnotes

  \revert Stem #'flag
  \testnotes
}
```





Using ly:grob-object to access grobs with \tweak

Some grobs can be accessed “laterally” from within another grob’s callback. These are usually listed as “layout objects” in the “Internal properties” section of a grob-interface. The function `ly:grob-object` is used to access these grobs.

Demonstrated below are some ways of accessing grobs from within a `NoteHead` callback, but the technique is not limited to `NoteHeads`. However, the `NoteHead` callback is particularly important, since it is the implicit callback used by the `\tweak` command.

The example function defined below (`"display-grobs"`) is probably not that useful, but it demonstrates that the grobs are indeed being accessed.

Example console output:

```
-----
#-Grob Accidental -
#-Grob Arpeggio -
#-Grob Stem -
#(define (notehead-get-accidental notehead)
  ;; notehead is grob
  (ly:grob-object notehead 'accidental-grob))

#(define (notehead-get-arpeggio notehead)
  ;; notehead is grob
  (let ((notecolumn (notehead-get-notecolumn notehead)))
    (ly:grob-object notecolumn 'arpeggio)))

#(define (notehead-get-notecolumn notehead)
  ;; notehead is grob
  (ly:grob-parent notehead X))

#(define (notehead-get-stem notehead)
  ;; notehead is grob
  (let ((notecolumn (notehead-get-notecolumn notehead)))
    (ly:grob-object notecolumn 'stem)))

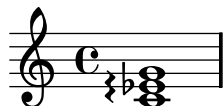
#(define (display-grobs notehead)
  ;; notehead is grob
  (let ((accidental (notehead-get-accidental notehead))
        (arpeggio (notehead-get-arpeggio notehead))
        (stem (notehead-get-stem notehead)))
    (format #t "~2&~a\n" (make-string 20 #\ -))
    (for-each
     (lambda (x) (format #t "~a\n" x))
     (list accidental arpeggio stem))))

\relative c' {
  %% display grobs for each note head:
  \%override NoteHead #'before-line-breaking = #display-grobs
  <c
  %% or just for one:
  \tweak #'before-line-breaking #display-grobs
```

```

es
g>1\arpeggio
}

```



Using PostScript to generate special note head shapes

When a note head with a special shape cannot easily be generated with graphic markup, PostScript code can be used to generate the shape. This example shows how a parallelogram-shaped note head is generated.

```

parallelogram =
  #(ly:make-stencil (list 'embedded-ps
    "gsave
      currentpoint translate
      newpath
      0 0.25 moveto
      1.3125 0.75 lineto
      1.3125 -0.25 lineto
      0 -0.75 lineto
      closepath
      fill
      grestore" )
    (cons 0 1.3125)
    (cons 0 0))

myNoteHeads = \override NoteHead #'stencil = \parallelogram
normalNoteHeads = \revert NoteHead #'stencil

\relative c' {
  \myNoteHeads
  g4 d'
  \normalNoteHeads
  <f, \tweak #'stencil \parallelogram b e>4 d
}

```



Using the \tweak command to tweak individual grobs

With the \tweak command, every grob can be tuned directly. Here are some examples of available tweaks.

```

\relative c' {
  \time 2/4
  \set fingeringOrientations = #'(right)
  <

```

```

\tweak #'font-size #3 c
\tweak #'color #red d-\tweak #'font-size #8 -4
\tweak #'style #'cross g
\tweak #'duration-log #2 a
>2
}

```



Vertically aligned dynamics and textscripts

By setting the 'Y-extent' property to a suitable value, all `DynamicLineSpanner` objects (hairpins and dynamic texts) can be aligned to a common reference point, regardless of their actual extent. This way, every element will be vertically aligned, thus producing a more pleasing output.

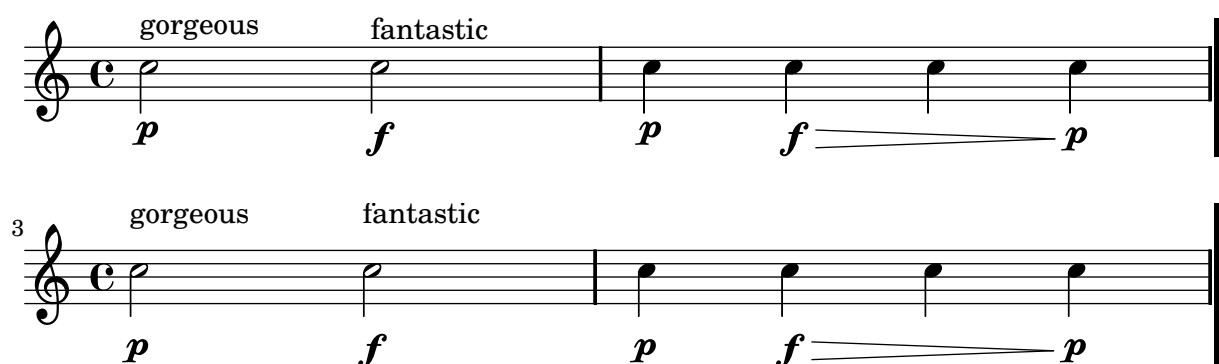
The same idea is used to align the text scripts along their baseline.

```

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

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

```



Vertically aligning ossias and lyrics

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

```

\paper {
  ragged-right = ##t
}

```

```

\relative c' <<

```

```

\new Staff = "1" { c4 c s2 }
\new Staff = "2" { c4 c s2 }
\new Staff = "3" { c4 c s2 }
{ \skip 2
  <<
    \lyrics {
      \set alignBelowContext = #"1"
      lyrics4 below
    }
    \new Staff \with {
      alignAboveContext = #"3"
      fontSize = #-2
      \override StaffSymbol #'staff-space = #(magstep -2)
      \remove "Time_signature_engraver"
    } {
      \times 4/6 {
        \override TextScript #'padding = #3
        c8["ossia above" d e d e f]
      }
    }
  >>
}
>>

```



Vertically centering paired figured bass extenders

Where figured bass extender lines are being used by setting `useBassFigureExtenders` to true, pairs of congruent figured bass extender lines are vertically centered if `figuredBassCenterContinuations` is set to true.

```

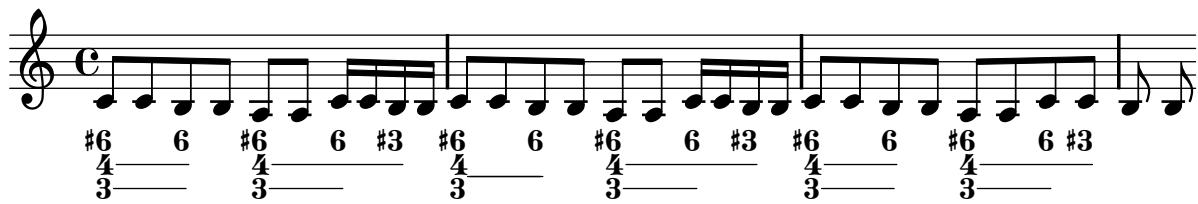
<<
\relative c' {
  c8 c b b a a c16 c b b
  c8 c b b a a c16 c b b
  c8 c b b a a c c b b
}
\figures {
  \set useBassFigureExtenders = ##t
  <6+ 4 3>4 <6 4 3>8 r
}
>>

```

```

<6+ 4 3>4 <6 4 3>8 <4 3+>16 r
\set figuredBassCenterContinuations = ##t
<6+ 4 3>4 <6 4 3>8 r
<6+ 4 3>4 <6 4 3>8 <4 3+>16 r
\set figuredBassCenterContinuations = ##f
<6+ 4 3>4 <6 4 3>8 r
<6+ 4 3>4 <6 4 3>8 <4 3+>8
}
>>

```



Paper and layout

Section “Spacing issues” in *Notation Reference*

Aligning and centering instrument names

The horizontal alignment of instrument names is tweaked by changing the `Staff.InstrumentName #'self-alignment-X` property. The `\layout` variables `indent` and `short-indent` define the space in which the instrument names are aligned before the first and the following systems, respectively.

```
\paper {
  left-margin = 3\cm
}

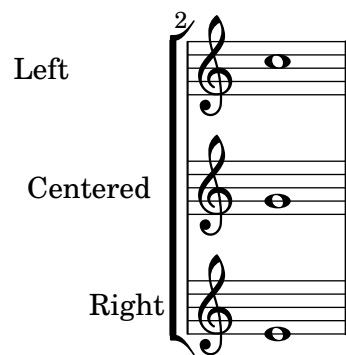
\score {
  \new StaffGroup <<
    \new Staff {
      \override Staff.InstrumentName #'self-alignment-X = #LEFT
      \set Staff.instrumentName = \markup \left-column {
        "Left aligned"
        "instrument name"
      }
      \set Staff.shortInstrumentName = #"Left"
      c'1
      \break
      c'1
    }
    \new Staff {
      \override Staff.InstrumentName #'self-alignment-X = #CENTER
      \set Staff.instrumentName = \markup \center-column {
        Centered
        "instrument name"
      }
      \set Staff.shortInstrumentName = #"Centered"
      g'1
      g'1
    }
    \new Staff {
      \override Staff.InstrumentName #'self-alignment-X = #RIGHT
      \set Staff.instrumentName = \markup \right-column {
        "Right aligned"
        "instrument name"
      }
      \set Staff.shortInstrumentName = #"Right"
      e'1
      e'1
    }
  >>
  \layout {
    ragged-right = ##t
    indent = 4\cm
    short-indent = 2\cm
  }
}
```

```
}
}
```

Left aligned
instrument name

Centered
instrument name

Right aligned
instrument name



Book parts

`\bookpart` can be used to split a book into several parts. Each part last page can be affected by `ragged-last-bottom`. Header and footer markups can detect a part last page, and make the difference with the book last page.

```
#{set-default-paper-size "a6")
```

```
\book {
  %% book paper, which is inherited by all children bookparts
  \paper {
    ragged-last-bottom = ##t
    %% Page footer: add a different part-tagline at part last page
    oddFooterMarkup = \markup {
      \column {
        \fill-line {
          %% Copyright header field only on book first page.
          \on-the-fly #first-page \fromproperty #'header:copyright
        }
        \fill-line {
          %% Part tagline header field only on each part last page.
          \on-the-fly #part-last-page \fromproperty #'header:parttagline
        }
        \fill-line {
          %% Tagline header field only on book last page.
          \on-the-fly #last-page \fromproperty #'header:tagline
        }
      }
    }
  }
}
```

```
    }
  }

%% book header, which is inherited by the first bookpart
\header {
  title = "Book title"
  copyright = "Copyright line on book first page"
  parttagline = "Part tagline"
  tagline = "Book tagline"
}

\bookpart {
  %% a different page breaking function may be used on each part
  \paper { page-breaking = #ly:minimal-breaking }
  \header { subtitle = "First part" }
  \markup { The first book part }
  \markup { a page break }
  \pageBreak
  \markup { first part last page }
  \markup \wordwrap { with ragged-last-bottom (see the space below this text) }
}

\bookpart {
  \header { subtitle = "Second part" }
  { c'4 }
}
}
```

Book title

First part

The first book part

a page break

Copyright line on book first page

2

first part last page

with ragged-last-bottom (see the space below this
text)

Part tagline

Book title

Second part



Part tagline
Book tagline

Changing the staff size

Though the simplest way to resize staves is to use `#{set-global-staff-size xx}`, an individual staff's size can be changed by scaling the properties `'staff-space` and `fontSize`.

```
<<
  \new Staff {
    \relative c'' {
      \dynamicDown
      c8\ff c c c c c c c
    }
  }
  \new Staff \with {
    fontSize = #-3
    \override StaffSymbol #'staff-space = #(magstep -3)
  } {
    \clef bass
    c8 c c c c\f c c c
  }
>>
```



Clip systems

This code shows how to clip (extract) snippets from a full score.

This file needs to be run separately with `-dclip-systems`; the snippets page may not adequately show the results.

The result will be files named `'base-from-start-to-end[-count].eps'`.

If system starts and ends are included, they include extents of the System grob, e.g., instrument names.

Grace notes at the end point of the region are not included.

Regions can span multiple systems. In this case, multiple EPS files are generated.

```
#(ly:set-option 'clip-systems)
```

```
#(define output-suffix "1")
```

```
origScore = \score {
```

```
  \relative c' {
```

```
    \set Staff.instrumentName = #"bla"
```

```
    c1
```

```
    d1
```

```
    \grace c16 e1
```

```
    \key d \major
```

```
    f1 \break
```

```
    \clef bass
```

```
    g,1
```

```
    fis1
```

```
  }
```

```
}
```

```
\book {
```

```
  \score {
```

```
    \origScore
```

```
    \layout {
```

```
      % Each clip-region is a (START . END) pair
```

```
      % where both are rhythmic-locations.
```

```
      % (make-rhythmic-locations BAR-NUMBER NUM DEN)
```

```
      % means NUM/DEN whole-notes into bar numbered BAR-NUMBER
```

```
      clip-regions = #(list
```

```
        (cons
```

```
          (make-rhythmic-location 2 0 1)
```

```
          (make-rhythmic-location 4 0 1))
```

```
        (cons
```

```
          (make-rhythmic-location 0 0 1)
```

```
          (make-rhythmic-location 4 0 1))
```

```

        (cons
          (make-rhythmic-location 0 0 1)
          (make-rhythmic-location 6 0 1))
        )
      }
    }
  }

#(ly:set-option 'clip-systems #f)
#(define output-suffix #f)

\book {
  \score { \origScore }
  \markup { \bold \fontsize #6 clips }
  \score {
    \lyrics {
      \markup { from-2.0.1-to-4.0.1-clip.eps }
      \markup {
        \epsfile #X #30.0 #(format #f "~a-1-from-2.0.1-to-4.0.1-clip.eps"
          (ly:parser-output-name parser)) }
    }
  }
}

```



Creating blank staves

To create blank staves, generate empty measures then remove the `Bar_number_engraver` from the `Score` context, and the `Time_signature_engraver`, `Clef_engraver` and `Bar_engraver` from the `Staff` context.

```

\set-global-staff-size 20)

```

```

\score {
  {
    \repeat unfold 12 { s1 \break }
  }
  \layout {
    indent = 0\in
    \context {
      \Staff
      \remove "Time_signature_engraver"
      \remove "Clef_engraver"
      \remove "Bar_engraver"
    }
    \context {
      \Score
      \remove "Bar_number_engraver"
    }
  }
}

```

```

\paper {
  \set-paper-size "letter")
  ragged-last-bottom = ##f
  line-width = 7.5\in
  left-margin = 0.5\in
  bottom-margin = 0.25\in
  top-margin = 0.25\in
}

```



Demonstrating all headers

All header fields with special meanings.

```
\header {
  copyright = "copyright"
  title = "title"
  subtitle = "subtitle"
  composer = "composer"
  arranger = "arranger"
  instrument = "instrument"
  metre = "metre"
  opus = "opus"
  piece = "piece"
  poet = "poet"
  texidoc = "All header fields with special meanings."
  copyright = "public domain"
  enteredby = "jcn"
  source = "urtext"
}

\layout {
  ragged-right = ##f
}

\score {
  \relative c'' { c1 | c | c | c }
}

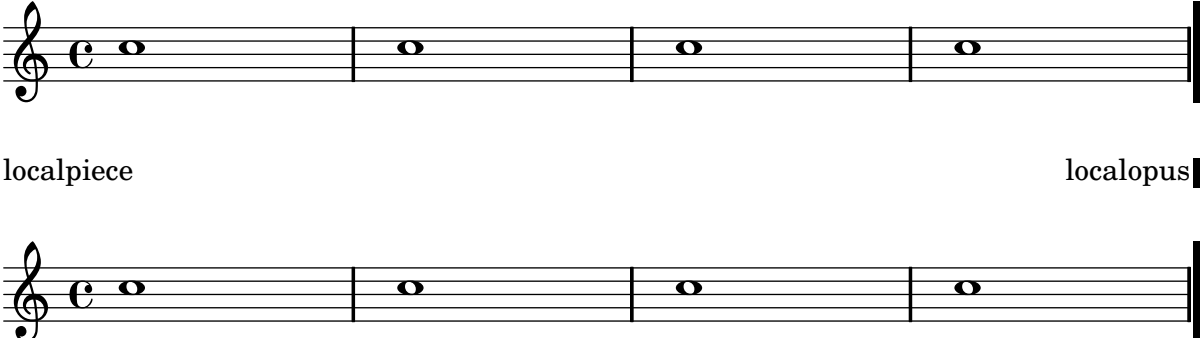
\score {
  \relative c'' { c1 | c | c | c }
  \header {
    title = "localtitle"
    subtitle = "localsubtitle"
    composer = "localcomposer"
    arranger = "localarranger"
```

```

    instrument = "localinstrument"
    metre = "localmetre"
    opus = "localopus"
    piece = "localpiece"
    poet = "localpoet"
    copyright = "localcopyright"
  }
}

```

	title	
	subtitle	
poet	instrument	composer arranger
piece		opus
localpiece		localopus



Setting system separators

System separators can be inserted between systems. Any markup can be used, but `\slashSeparator` has been provided as a sensible default.

```

\paper {
  system-separator-markup = \slashSeparator
}

notes = \relative c' {
  c1 | c \break
  c1 | c \break
  c1 | c
}

\book {
  \score {
    \new GrandStaff <<
      \new Staff \notes
      \new Staff \notes
    >>
  }
}

```


Three systems of musical notation, each consisting of a grand staff (treble and bass clef) and a key signature of one flat (B-flat). The first system has a common time signature 'C'. The second system has a '3' time signature. The third system has a '5' time signature. Each system contains two measures separated by a bar line. The notes are: System 1 (treble: B-flat4, bass: B-flat3), System 2 (treble: B-flat4, bass: B-flat3), System 3 (treble: B-flat4, bass: B-flat3).

Table of contents

A table of contents is included using `\markuplines \table-of-contents`. The TOC items are added with the `\tocItem` command.

```

#(set-default-paper-size "a6")

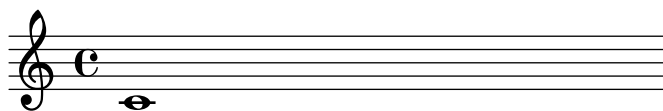
\book {
  \markuplines \table-of-contents
  \pageBreak
  \tocItem \markup { The first score }
  \score {
    {
      c'1 \pageBreak
      \mark "A" \tocItem \markup { Mark A }
      d'1
    }
  }
  \pageBreak
  \tocItem \markup { The second score }
  \score {
    { e'1 }
    \header { piece = "Second score" }
  }
}

```

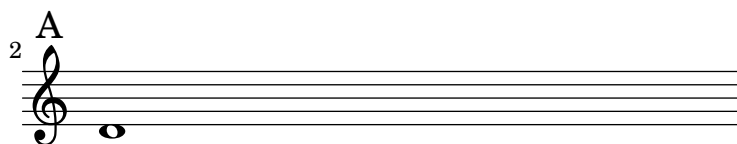
Table of Contents

The first score	2
Mark A	2
The second score	4

2

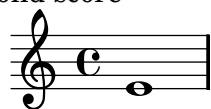


3



4

Second score



Titles

Section “Titles and headers” in *Notation Reference*

Adding the current date to a score

With a little Scheme code, the current date can easily be added to a score.

```
% first, define a variable to hold the formatted date:
date = #(strftime "%d-%m-%Y" (localtime (current-time)))
```

```
% use it in the title block:
\header {
  title = "Including the date!"
  subtitle = \date
}
```

```
\score {
  \relative c'' {
    c4 c c c
  }
}
% and use it in a \markup block:
\markup {
  \date
}
```

Including the date!
06-07-2010



06-07-2010

Aligning and centering instrument names

The horizontal alignment of instrument names is tweaked by changing the `Staff.InstrumentName #'self-alignment-X` property. The `\layout` variables `indent` and `short-indent` define the space in which the instrument names are aligned before the first and the following systems, respectively.

```
\paper {
  left-margin = 3\cm
}

\score {
  \new StaffGroup <<
    \new Staff {
      \override Staff.InstrumentName #'self-alignment-X = #LEFT
      \set Staff.instrumentName = \markup \left-column {
```

```


    "Left aligned"
    "instrument name"
  }
  \set Staff.shortInstrumentName = #"Left"
  c'1
  \break
  c'1
}
\new Staff {
  \override Staff.InstrumentName #'self-alignment-X = #CENTER
  \set Staff.instrumentName = \markup \center-column {
    Centered
    "instrument name"
  }
  \set Staff.shortInstrumentName = #"Centered"
  g'1
  g'1
}
\new Staff {
  \override Staff.InstrumentName #'self-alignment-X = #RIGHT
  \set Staff.instrumentName = \markup \right-column {
    "Right aligned"
    "instrument name"
  }
  \set Staff.shortInstrumentName = #"Right"
  e'1
  e'1
}
>>
\layout {
  ragged-right = ##t
  indent = 4\cm
  short-indent = 2\cm
}
}

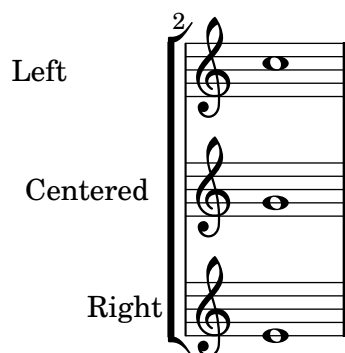
```

Left aligned
instrument name

Centered
instrument name

Right aligned
instrument name





Demonstrating all headers

All header fields with special meanings.

```
\header {
  copyright = "copyright"
  title = "title"
  subtitle = "subtitle"
  composer = "composer"
  arranger = "arranger"
  instrument = "instrument"
  metre = "metre"
  opus = "opus"
  piece = "piece"
  poet = "poet"
  texidoc = "All header fields with special meanings."
  copyright = "public domain"
  enteredby = "jcn"
  source = "urtext"
}

\layout {
  ragged-right = ##f
}

\score {
  \relative c'' { c1 | c | c | c }
}

\score {
  \relative c'' { c1 | c | c | c }
  \header {
    title = "localtitle"
    subtitle = "localsubtitle"
    composer = "localcomposer"
    arranger = "localarranger"
    instrument = "localinstrument"
    metre = "localmetre"
    opus = "localopus"
    piece = "localpiece"
    poet = "localpoet"
    copyright = "localcopyright"
  }
}
```

	title	
	subtitle	
poet	instrument	composer
		arranger
piece		opus
		
localpiece		localopus
		

Outputting the version number

By putting the output of `lilypond-version` into a lyric, it is possible to print the version number of LilyPond in a score, or in a document generated with `lilypond-book`. Another possibility is to append the version number to the doc-string, in this manner:

```
\score {
  \new Lyrics {
    \override Score.RehearsalMark #'self-alignment-X = #LEFT
    \mark #(string-append "Processed with LilyPond version " (lilypond-version))
    s2
  }
}
```

Processed with LilyPond version 2.13.27

Spacing

Section “Spacing issues” in *Notation Reference*

Adjusting lyrics vertical spacing

This snippet shows how to bring the lyrics line closer to the staff.

% Default layout:

```
<<
  \new Staff \new Voice = melody \relative c' {
    c4 d e f
    g4 f e d
    c1
  }
  \new Lyrics \lyricsto melody { aa aa aa aa aa aa aa aa aa }

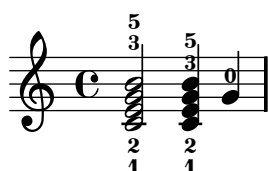
% Reducing the minimum space below the staff and above the lyrics:
\new Staff {
  \new Voice = melody \relative c' {
    c4 d e f
    g4 f e d
    c1
  }
}
\new Lyrics \with {
  \override VerticalAxisGroup #'inter-staff-spacing = #'((space . 1))
}
\lyricsto melody { aa aa aa aa aa aa aa aa aa }
>>
```



Allowing fingerings to be printed inside the staff

By default, vertically oriented fingerings are positioned outside the staff. However, this behavior can be canceled. Note: you must use a chord construct <>, even if it is only a single note.

```
\relative c' {
  <c-1 e-2 g-3 b-5>2
  \override Fingering #'staff-padding = #'()
  <c-1 e-2 g-3 b-5>4 <g'-0>
}
```



Page label

Page labels may be placed inside music or at top-level, and referred to in markups.

```

#(set-default-paper-size "a6")

#(define-markup-command (toc-line layout props label text)
  (symbol? markup?)
  (interpret-markup layout props
    (markup #:fill-line (text #:page-ref label "8" "?"))))

\book {
  \markup \huge \fill-line { \null Title Page \null }

  \pageBreak

  \label #'toc
  \markup \column {
    \large \fill-line { \null Table of contents \null }
    \toc-line #'toc "Table of contents"
    \toc-line #'firstScore "First Score"
    \toc-line #'markA "Mark A"
    \toc-line #'markB "Mark B"
    \toc-line #'markC "Mark C"
    \toc-line #'unknown "Unknown label"
  }

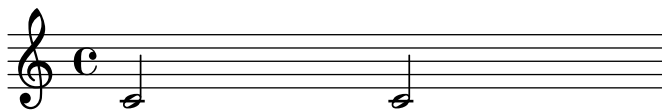
  \pageBreak

  \label #'firstScore
  \score {
    \new Staff \relative c' {
      c2 c
      \mark \markup {
        A (page \concat { \page-ref #'markA "0" "?" ) }
      } \label #'markA
      c2 c
      \pageBreak
      \mark "B" \label #'markB
      d2 d
      d2 d
      \once \override Score.RehearsalMark #'break-visibility =
        #begin-of-line-invisible
      \mark "C" \label #'markC
    }
    \header { piece = "First score" }
  }
}

```

2	Table of contents	
	Table of contents	2
	First Score	3
	Mark A	3
	Mark B	3
	Mark C	4
	Unknown label	?

	3
First score	



A (page 3)



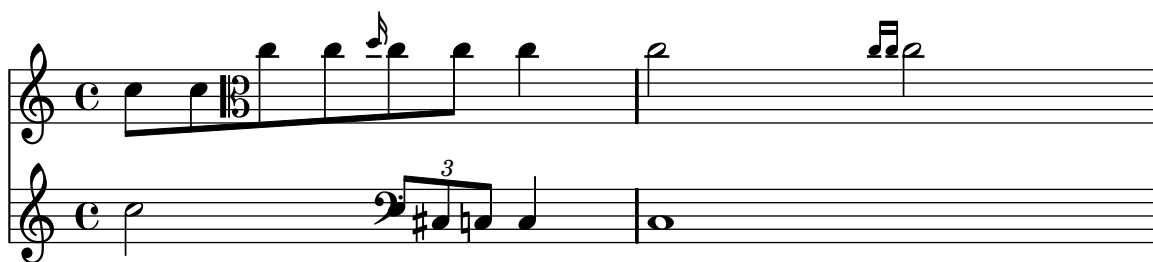


Music engraving by LilyPond 2.13.27—www.lilypond.org

Proportional strict notespacing

If `strict-note-spacing` is set spacing of notes is not influenced by bars or clefs within a system. Rather, they are placed just before the note that occurs at the same time. This may cause collisions.

```
\relative c'' <<
  \override Score.SpacingSpanner #'strict-note-spacing = ##t
  \set Score.proportionalNotationDuration = #(ly:make-moment 1 16)
  \new Staff {
    c8[ c \clef alto c c \grace { d16 } c8 c] c4
    c2 \grace { c16[ c16] } c2
  }
  \new Staff {
    c2 \times 2/3 { c8 \clef bass cis,, c } c4
    c1
  }
  >>
```



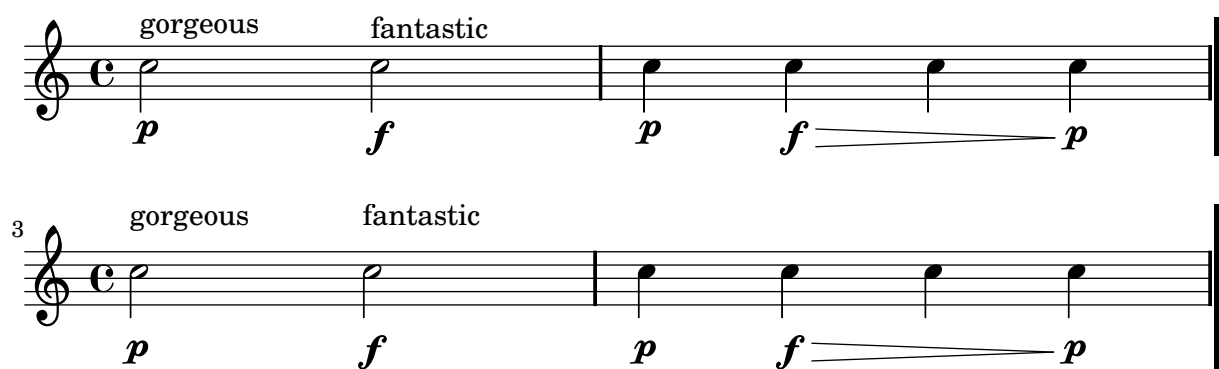
Vertically aligned dynamics and textscripts

By setting the 'Y-extent' property to a suitable value, all `DynamicLineSpanner` objects (hairpins and dynamic texts) can be aligned to a common reference point, regardless of their actual extent. This way, every element will be vertically aligned, thus producing a more pleasing output.

The same idea is used to align the text scripts along their baseline.

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

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



Vertically aligning ossia and lyrics

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

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

\relative c' <<
  \new Staff = "1" { c4 c s2 }
  \new Staff = "2" { c4 c s2 }
  \new Staff = "3" { c4 c s2 }
  { \skip 2
    <<
      \lyrics {
```

```

\set alignBelowContext = #"1"
lyrics4 below
}
\new Staff \with {
  alignAboveContext = #"3"
  fontSize = #-2
  \override StaffSymbol #'staff-space = #(magstep -2)
  \remove "Time_signature_engraver"
} {
  \times 4/6 {
    \override TextScript #'padding = #3
    c8["ossa above" d e d e f]
  }
}
>>
}
>>

```



MIDI

Section “MIDI output” in *Notation Reference*

Changing MIDI output to one channel per voice

When outputting MIDI, the default behavior is for each staff to represent one MIDI channel, with all the voices on a staff amalgamated. This minimizes the risk of running out of MIDI channels, since there are only 16 available per track.

However, by moving the `Staff_performer` to the `Voice` context, each voice on a staff can have its own MIDI channel, as is demonstrated by the following example: despite being on the same staff, two MIDI channels are created, each with a different `midiInstrument`.

```
\score {
  \new Staff <<
    \new Voice \relative c''' {
      \set midiInstrument = #"flute"
      \voiceOne
      \key g \major
      \time 2/2
      r2 g-"Flute" ~
      g fis ~
      fis4 g8 fis e2 ~
      e4 d8 cis d2
    }
    \new Voice \relative c'' {
      \set midiInstrument = #"clarinet"
      \voiceTwo
      b1-"Clarinet"
      a2. b8 a
      g2. fis8 e
      fis2 r
    }
  >>
  \layout { }
  \midi {
    \context {
      \Staff
      \remove "Staff_performer"
    }
    \context {
      \Voice
      \consists "Staff_performer"
    }
    \context {
      \Score
      tempoWholesPerMinute = #(ly:make-moment 72 2)
    }
  }
}
```



Changing the tempo without a metronome mark

To change the tempo in MIDI output without printing anything, make the metronome mark invisible.

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



Demo MidiInstruments

Problem: How to know which midiInstrument would be best for your composition? Solution: A LilyPond demo file.

```
\header {
  title = "Demo of all midi sounds"
  arranger = "Myself "
}

baseMelody = \relative c' {
  c4.\mf g c16 b' c d
  e16 d e f g4 g'4 r
  R1
}

melody = {
  \tempo 4 = 150
  \baseMelody
}

\score {
  \new Staff <<
    \new Voice \melody
  >>
  \layout { }
}
```

```

\score {
  \new Staff <<
    \new Voice {
      r\mf
      \set Staff.midiInstrument = #"acoustic grand" \melody
      \set Staff.midiInstrument = #"bright acoustic" \melody
      \set Staff.midiInstrument = #"electric grand" \melody
      \set Staff.midiInstrument = #"honky-tonk" \melody
      \set Staff.midiInstrument = #"electric piano 1" \melody
      \set Staff.midiInstrument = #"electric piano 2" \melody
      \set Staff.midiInstrument = #"harpsichord" \melody
      \set Staff.midiInstrument = #"clav" \melody
      \set Staff.midiInstrument = #"celesta" \melody
      \set Staff.midiInstrument = #"glockenspiel" \melody
      \set Staff.midiInstrument = #"music box" \melody
      \set Staff.midiInstrument = #"vibraphone" \melody
      \set Staff.midiInstrument = #"marimba" \melody
      \set Staff.midiInstrument = #"xylophone" \melody
      \set Staff.midiInstrument = #"tubular bells" \melody
      \set Staff.midiInstrument = #"dulcimer" \melody
      \set Staff.midiInstrument = #"drawbar organ" \melody
      \set Staff.midiInstrument = #"percussive organ" \melody
      \set Staff.midiInstrument = #"rock organ" \melody
      \set Staff.midiInstrument = #"church organ" \melody
      \set Staff.midiInstrument = #"reed organ" \melody
      \set Staff.midiInstrument = #"accordion" \melody
      \set Staff.midiInstrument = #"harmonica" \melody
      \set Staff.midiInstrument = #"concertina" \melody
      \set Staff.midiInstrument = #"acoustic guitar (nylon)" \melody
      \set Staff.midiInstrument = #"acoustic guitar (steel)" \melody
      \set Staff.midiInstrument = #"electric guitar (jazz)" \melody
      \set Staff.midiInstrument = #"electric guitar (clean)" \melody
      \set Staff.midiInstrument = #"electric guitar (muted)" \melody
      \set Staff.midiInstrument = #"overdriven guitar" \melody
      \set Staff.midiInstrument = #"distorted guitar" \melody
      \set Staff.midiInstrument = #"acoustic bass" \melody
      \set Staff.midiInstrument = #"electric bass (finger)" \melody
      \set Staff.midiInstrument = #"electric bass (pick)" \melody
      \set Staff.midiInstrument = #"fretless bass" \melody
      \set Staff.midiInstrument = #"slap bass 1" \melody
      \set Staff.midiInstrument = #"slap bass 2" \melody
      \set Staff.midiInstrument = #"synth bass 1" \melody
      \set Staff.midiInstrument = #"synth bass 2" \melody
      \set Staff.midiInstrument = #"violin" \melody
      \set Staff.midiInstrument = #"viola" \melody
      \set Staff.midiInstrument = #"cello" \melody
      \set Staff.midiInstrument = #"contrabass" \melody
      \set Staff.midiInstrument = #"tremolo strings" \melody
      \set Staff.midiInstrument = #"pizzicato strings" \melody
      \set Staff.midiInstrument = #"orchestral harp" \melody
      \set Staff.midiInstrument = #"timpani" \melody
      \set Staff.midiInstrument = #"string ensemble 1" \melody
    }
  }
}

```

```
\set Staff.midiInstrument = #"string ensemble 2" \melody
\set Staff.midiInstrument = #"synthstrings 1" \melody
\set Staff.midiInstrument = #"synthstrings 2" \melody
\set Staff.midiInstrument = #"choir aahs" \melody
\set Staff.midiInstrument = #"voice oohs" \melody
\set Staff.midiInstrument = #"synth voice" \melody
\set Staff.midiInstrument = #"orchestra hit" \melody
\set Staff.midiInstrument = #"trumpet" \melody
\set Staff.midiInstrument = #"trombone" \melody
\set Staff.midiInstrument = #"tuba" \melody
\set Staff.midiInstrument = #"muted trumpet" \melody
\set Staff.midiInstrument = #"french horn" \melody
\set Staff.midiInstrument = #"brass section" \melody
\set Staff.midiInstrument = #"synthbrass 1" \melody
\set Staff.midiInstrument = #"synthbrass 2" \melody
\set Staff.midiInstrument = #"soprano sax" \melody
\set Staff.midiInstrument = #"alto sax" \melody
\set Staff.midiInstrument = #"tenor sax" \melody
\set Staff.midiInstrument = #"baritone sax" \melody
\set Staff.midiInstrument = #"oboe" \melody
\set Staff.midiInstrument = #"english horn" \melody
\set Staff.midiInstrument = #"bassoon" \melody
\set Staff.midiInstrument = #"clarinet" \melody
\set Staff.midiInstrument = #"piccolo" \melody
\set Staff.midiInstrument = #"flute" \melody
\set Staff.midiInstrument = #"recorder" \melody
\set Staff.midiInstrument = #"pan flute" \melody
\set Staff.midiInstrument = #"blown bottle" \melody
\set Staff.midiInstrument = #"shakuhachi" \melody
\set Staff.midiInstrument = #"whistle" \melody
\set Staff.midiInstrument = #"ocarina" \melody
\set Staff.midiInstrument = #"lead 1 (square)" \melody
\set Staff.midiInstrument = #"lead 2 (sawtooth)" \melody
\set Staff.midiInstrument = #"lead 3 (calliope)" \melody
\set Staff.midiInstrument = #"lead 4 (chiff)" \melody
\set Staff.midiInstrument = #"lead 5 (charang)" \melody
\set Staff.midiInstrument = #"lead 6 (voice)" \melody
\set Staff.midiInstrument = #"lead 7 (fifths)" \melody
\set Staff.midiInstrument = #"lead 8 (bass+lead)" \melody
\set Staff.midiInstrument = #"pad 1 (new age)" \melody
\set Staff.midiInstrument = #"pad 2 (warm)" \melody
\set Staff.midiInstrument = #"pad 3 (polysynth)" \melody
\set Staff.midiInstrument = #"pad 4 (choir)" \melody
\set Staff.midiInstrument = #"pad 5 (bowed)" \melody
\set Staff.midiInstrument = #"pad 6 (metallic)" \melody
\set Staff.midiInstrument = #"pad 7 (halo)" \melody
\set Staff.midiInstrument = #"pad 8 (sweep)" \melody
\set Staff.midiInstrument = #"fx 1 (rain)" \melody
\set Staff.midiInstrument = #"fx 2 (soundtrack)" \melody
\set Staff.midiInstrument = #"fx 3 (crystal)" \melody
\set Staff.midiInstrument = #"fx 4 (atmosphere)" \melody
\set Staff.midiInstrument = #"fx 5 (brightness)" \melody
```

```

\set Staff.midiInstrument = #"fx 6 (goblins)" \melody
\set Staff.midiInstrument = #"fx 7 (echoes)" \melody
\set Staff.midiInstrument = #"fx 8 (sci-fi)" \melody
\set Staff.midiInstrument = #"sitar" \melody
\set Staff.midiInstrument = #"banjo" \melody
\set Staff.midiInstrument = #"shamisen" \melody
\set Staff.midiInstrument = #"koto" \melody
\set Staff.midiInstrument = #"kalimba" \melody
\set Staff.midiInstrument = #"bagpipe" \melody
\set Staff.midiInstrument = #"fiddle" \melody
\set Staff.midiInstrument = #"shanai" \melody
\set Staff.midiInstrument = #"tinkle bell" \melody
\set Staff.midiInstrument = #"agogo" \melody
\set Staff.midiInstrument = #"steel drums" \melody
\set Staff.midiInstrument = #"woodblock" \melody
\set Staff.midiInstrument = #"taiko drum" \melody
\set Staff.midiInstrument = #"melodic tom" \melody
\set Staff.midiInstrument = #"synth drum" \melody
\set Staff.midiInstrument = #"reverse cymbal" \melody
\set Staff.midiInstrument = #"guitar fret noise" \melody
\set Staff.midiInstrument = #"breath noise" \melody
\set Staff.midiInstrument = #"seashore" \melody
\set Staff.midiInstrument = #"bird tweet" \melody
\set Staff.midiInstrument = #"telephone ring" \melody
\set Staff.midiInstrument = #"helicopter" \melody
\set Staff.midiInstrument = #"applause" \melody
\set Staff.midiInstrument = #"gunshot" \melody
}
>>
\midi { }
}

```

Templates

Ancient notation template – modern transcription of gregorian music

This example demonstrates how to do modern transcription of Gregorian music. Gregorian music has no measure, no stems; it uses only half and quarter note heads, and special marks, indicating rests of different length.

```
\include "gregorian.ly"

chant = \relative c' {
  \set Score.timing = ##f
  f4 a2 \divisioMinima
  g4 b a2 f2 \divisioMaior
  g4( f) f( g) a2 \finalis
}

verba = \lyricmode {
  Lo -- rem ip -- sum do -- lor sit a -- met
}

\score {
  \new Staff <<
    \new Voice = "melody" \chant
    \new Lyrics = "one" \lyricsto melody \verba
  >>
  \layout {
    \context {
      \Staff
      \remove "Time_signature_engraver"
      \remove "Bar_engraver"
      \override Stem #'transparent = ##t
    }
    \context {
      \Voice
      \override Stem #'length = #0
    }
    \context {
      \Score
      barAlways = ##t
    }
  }
}
```



Ancient notation template – modern transcription of mensural music

When transcribing mensural music, an incipit at the beginning of the piece is useful to indicate the original key and tempo. While today musicians are used to bar lines in order to faster recognize rhythmic patterns, bar lines were not yet invented during the period of mensural music; in fact, the meter often changed after every few notes. As a compromise, bar lines are often printed between the staves rather than on the staves.

```
global = {
  \set Score.skipBars = ##t

  % incipit
  \once \override Score.SystemStartBracket #'transparent = ##t
  \override Score.SpacingSpanner #'spacing-increment = #1.0 % tight spacing
  \key f \major
  \time 2/2
  \once \override Staff.TimeSignature #'style = #'neomensural
  \override Voice.NoteHead #'style = #'neomensural
  \override Voice.Rest #'style = #'neomensural
  \set Staff.printKeyCancellation = ##f
  \cadenzaOn % turn off bar lines
  \skip 1*10
  \once \override Staff.BarLine #'transparent = ##f
  \bar "||"
  \skip 1*1 % need this extra \skip such that clef change comes
             % after bar line
  \bar ""

  % main
  \revert Score.SpacingSpanner #'spacing-increment % CHECK: no effect?
  \cadenzaOff % turn bar lines on again
  \once \override Staff.Clef #'full-size-change = ##t
  \set Staff.forceClef = ##t
  \key g \major
  \time 4/4
  \override Voice.NoteHead #'style = #'default
  \override Voice.Rest #'style = #'default

  % FIXME: setting printKeyCancellation back to #t must not
  % occur in the first bar after the incipit. Dto. for forceClef.
  % Therefore, we need an extra \skip.
  \skip 1*1
  \set Staff.printKeyCancellation = ##t
  \set Staff.forceClef = ##f

  \skip 1*7 % the actual music

  % let finis bar go through all staves
  \override Staff.BarLine #'transparent = ##f

  % finis bar
  \bar "|."
}
```

```

discantusNotes = {
  \transpose c' c'' {
    \set Staff.instrumentName = #"Discantus  "

    % incipit
    \clef "neomensural-c1"
    c'1. s2  % two bars
    \skip 1*8 % eight bars
    \skip 1*1 % one bar

    % main
    \clef "treble"
    d'2. d'4 |
    b e' d'2 |
    c'4 e'4.( d'8 c' b |
    a4) b a2 |
    b4.( c'8 d'4) c'4 |
    \once \override NoteHead #'transparent = ##t c'1 |
    b\breve |
  }
}

```

```

discantusLyrics = \lyricmode {
  % incipit
  IV-

  % main
  Ju -- bi -- |
  la -- te De -- |
  o, om --
  nis ter -- |
  ra, __ om- |
  "... " |
  -us. |
}

```

```

altusNotes = {
  \transpose c' c'' {
    \set Staff.instrumentName = #"Altus  "

    % incipit
    \clef "neomensural-c3"
    r1          % one bar
    f1. s2      % two bars
    \skip 1*7 % seven bars
    \skip 1*1 % one bar

    % main
    \clef "treble"
    r2 g2. e4 fis g | % two bars
    a2 g4 e |
  }
}

```

```

        fis g4.( fis16 e fis4) |
        g1 |
        \once \override NoteHead #'transparent = ##t g1 |
        g\breve |
    }
}

altusLyrics = \lyricmode {
    % incipit
    IV-

    % main
    Ju -- bi -- la -- te | % two bars
    De -- o, om -- |
    nis ter -- ra, |
    "... " |
    -us. |
}

tenorNotes = {
    \transpose c' c' {
        \set Staff.instrumentName = #"Tenor  "

        % incipit
        \clef "neomensural-c4"
        r\longa    % four bars
        r\breve    % two bars
        r1         % one bar
        c'1. s2    % two bars
        \skip 1*1 % one bar
        \skip 1*1 % one bar

        % main
        \clef "treble_8"
        R1 |
        R1 |
        R1 |
        r2 d'2. d'4 b e' | % two bars
        \once \override NoteHead #'transparent = ##t e'1 |
        d'\breve |
    }
}

tenorLyrics = \lyricmode {
    % incipit
    IV-

    % main
    Ju -- bi -- la -- te | % two bars
    "... " |
    -us. |
}

```

```

bassusNotes = {
  \transpose c' c' {
    \set Staff.instrumentName = #"Bassus  "

    % incipit
    \clef "bass"
    r\maxima % eight bars
    f1. s2   % two bars
    \skip 1*1 % one bar

    % main
    \clef "bass"
    R1 |
    R1 |
    R1 |
    R1 |
    g2. e4 |
    \once \override NoteHead #'transparent = ##t e1 |
    g\breve |
  }
}

bassusLyrics = \lyricmode {
  % incipit
  IV-

  % main
  Ju -- bi- |
  "... " |
  -us. |
}

\score {
  \new StaffGroup = choirStaff <<
    \new Voice =
      "discantusNotes" << \global \discantusNotes >>
    \new Lyrics =
      "discantusLyrics" \lyricsto discantusNotes { \discantusLyrics }
    \new Voice =
      "altusNotes" << \global \altusNotes >>
    \new Lyrics =
      "altusLyrics" \lyricsto altusNotes { \altusLyrics }
    \new Voice =
      "tenorNotes" << \global \tenorNotes >>
    \new Lyrics =
      "tenorLyrics" \lyricsto tenorNotes { \tenorLyrics }
    \new Voice =
      "bassusNotes" << \global \bassusNotes >>
    \new Lyrics =
      "bassusLyrics" \lyricsto bassusNotes { \bassusLyrics }
  >>

```

```

\layout {
  \context {
    \Score

    % no bars in staves
    \override BarLine #'transparent = ##t

    % incipit should not start with a start delimiter
    \remove "System_start_delimiter_engraver"
  }
  \context {
    \Voice

    % no slurs
    \override Slur #'transparent = ##t

    % Comment in the below "\remove" command to allow line
    % breaking also at those barlines where a note overlaps
    % into the next bar. The command is commented out in this
    % short example score, but especially for large scores, you
    % will typically yield better line breaking and thus improve
    % overall spacing if you comment in the following command.
    %\remove "Forbid_line_break_engraver"
  }
}

```

Discantus

Altus

Tenor

Bassus

IV-

IV-

Ju - bi - la - te De -

Ju - bi - la - te

IV-

3

o, om - nis ter - ra, om- ... -us.

De - o, om - nis ter - ra, ... -us.

8 Ju - bi - la - te ... -us.

Ju - bi- ... -us.

Jazz combo template

This is quite an advanced template, for a jazz ensemble. Note that all instruments are notated in `\key c \major`. This refers to the key in concert pitch; the key will be automatically transposed if the music is within a `\transpose` section.

```
\header {
  title = "Song"
  subtitle = "(tune)"
  composer = "Me"
  meter = "moderato"
  piece = "Swing"
  tagline = \markup {
    \column {
      "LilyPond example file by Amelie Zapf,"
      "Berlin 07/07/2003"
    }
  }
}
```

```
%#(set-global-staff-size 16)
\include "english.ly"
```

```
%%%%%%%%%% Some macros %%%%%%%%%%
```

```
sl = {
  \override NoteHead #'style = #'slash
  \override Stem #'transparent = ##t
}
nsl = {
  \revert NoteHead #'style
  \revert Stem #'transparent
}
crOn = \override NoteHead #'style = #'cross
crOff = \revert NoteHead #'style
```

```
% insert chord name style stuff here.
```

```
jazzChords = { }
```

```
%%%%%%%%%%%% Keys'n'things %%%%%%%%%%%%%%
```

```
global = { \time 4/4 }
```

```
Key = { \key c \major }
```

```
% ##### Horns #####
```

```
% ----- Trumpet -----
```

```
trpt = \transpose c d \relative c' {  
  \Key  
  c1 | c | c |  
}
```

```
trpHarmony = \transpose c' d {  
  \jazzChords  
}
```

```
trumpet = {  
  \global  
  \set Staff.instrumentName = #"Trumpet"  
  \clef treble  
  <<  
    \trpt  
  >>  
}
```

```
% ----- Alto Saxophone -----
```

```
alto = \transpose c a \relative c' {  
  \Key  
  c1 | c | c |  
}
```

```
altoHarmony = \transpose c' a {  
  \jazzChords  
}
```

```
altoSax = {  
  \global  
  \set Staff.instrumentName = #"Alto Sax"  
  \clef treble  
  <<  
    \alto  
  >>  
}
```

```
% ----- Baritone Saxophone -----
```

```
bari = \transpose c a' \relative c {  
  \Key  
  c1  
  c1  
  \sl  
  d4~"Solo" d d d  
  \nsl  
}
```

```
bariHarmony = \transpose c' a \chordmode {
```

```

    \jazzChords s1 s d2:maj e:m7
}
bariSax = {
  \global
  \set Staff.instrumentName = #"Bari Sax"
  \clef treble
  <<
    \bari
  >>
}

% ----- Trombone -----
tbone = \relative c {
  \Key
  c1 | c | c
}
tboneHarmony = \chordmode {
  \jazzChords
}
trombone = {
  \global
  \set Staff.instrumentName = #"Trombone"
  \clef bass
  <<
    \tbone
  >>
}

% ##### Rhythm Section #####

% ----- Guitar -----
gtr = \relative c'' {
  \Key
  c1
  \sl
  b4 b b b
  \nsl
  c1
}
gtrHarmony = \chordmode {
  \jazzChords
  s1 c2:min7+ d2:maj9
}
guitar = {
  \global
  \set Staff.instrumentName = #"Guitar"
  \clef treble
  <<
    \gtr
  >>
}

```

```

%% ----- Piano -----
rhUpper = \relative c'' {
  \voiceOne
  \Key
  c1 | c | c
}
rhLower = \relative c' {
  \voiceTwo
  \Key
  e1 | e | e
}

lhUpper = \relative c' {
  \voiceOne
  \Key
  g1 | g | g
}
lhLower = \relative c {
  \voiceTwo
  \Key
  c1 | c | c
}

PianoRH = {
  \clef treble
  \global
  \set Staff.midiInstrument = #"acoustic grand"
  <<
    \new Voice = "one" \rhUpper
    \new Voice = "two" \rhLower
  >>
}
PianoLH = {
  \clef bass
  \global
  \set Staff.midiInstrument = #"acoustic grand"
  <<
    \new Voice = "one" \lhUpper
    \new Voice = "two" \lhLower
  >>
}

piano = {
  <<
    \set PianoStaff.instrumentName = #"Piano"
    \new Staff = "upper" \PianoRH
    \new Staff = "lower" \PianoLH
  >>
}

% ----- Bass Guitar -----
Bass = \relative c {

```

```

\Key
c1 | c | c
}
bass = {
  \global
  \set Staff.instrumentName = #"Bass"
  \clef bass
  <<
    \Bass
  >>
}

% ----- Drums -----
up = \drummode {
  \voiceOne
  hh4 <hh sn> hh <hh sn>
  hh4 <hh sn> hh <hh sn>
  hh4 <hh sn> hh <hh sn>
}
down = \drummode {
  \voiceTwo
  bd4 s bd s
  bd4 s bd s
  bd4 s bd s
}

drumContents = {
  \global
  <<
    \set DrumStaff.instrumentName = #"Drums"
    \new DrumVoice \up
    \new DrumVoice \down
  >>
}

%%%%%%%%%% It All Goes Together Here %%%%%%%%%%%

\score {
  <<
    \new StaffGroup = "horns" <<
      \new Staff = "trumpet" \trumpet
      \new Staff = "altosax" \altoSax
      \new ChordNames = "barichords" \bariHarmony
      \new Staff = "barisax" \bariSax
      \new Staff = "trombone" \trombone
    >>

    \new StaffGroup = "rhythm" <<
      \new ChordNames = "chords" \gtrHarmony
      \new Staff = "guitar" \guitar
      \new PianoStaff = "piano" \piano
      \new Staff = "bass" \bass
  >>
}

```

```
\new DrumStaff \drumContents
>>
>>
\layout {
  \context { \Staff \RemoveEmptyStaves }
  \context {
    \Score
    \override BarNumber #'padding = #3
    \override RehearsalMark #'padding = #2
    skipBars = ##t
  }
}
\midi { }
}
```

Song **(tune)**

moderato

Swing

Me

Trumpet

Alto Sax

Bari Sax

Trombone

Guitar

Piano

Bass

Drums

$B^{\Delta} C^{\#}m^7$ Solo

$Cm^{\Delta} D^{\Delta 9}$

Orchestra choir and piano template

This template demonstrates the use of nested `StaffGroup` and `GrandStaff` contexts to sub-group instruments of the same type together, and the use of `\transpose` for transposing instruments. All music in variables is stored in C. Music may be entered in C or, alternatively, entered in the instrument key and transposed to C before being assigned to a variable.

```
#(set-global-staff-size 17)
```

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

```
fluteMusic = \relative c { \key c \major c'1 d }
oboeMusic = \relative c { \key c \major c'1 d }
clarinetMusic = \relative c { \key c \major c'1 d }
bassoonMusic = \relative c { \clef bass \key c \major c1 d }
trumpetMusic = \relative c { \key c \major c''1 d }
tromboneMusic = \relative c { \key c \major c1 d }
hornIMusic = \relative c { c'1 d }
hornIIMusic = \relative c { c1 d }
percussionMusic = \relative c { \key c \major c1 d }
```

```

sopranoMusic = \relative c' { \key c \major c1 d }
sopranoLyrics = \lyricmode { Sop -- ra }
altoIMusic = \relative c' { \key c \major c1 d }
altoILyrics = \lyricmode { A -- one }
altoIIMusic = \relative c' { \key c \major c1 d }
altoIILyrics = \lyricmode { A -- two }
tenorMusic = \relative c' { \key c \major c1 d }
tenorLyrics = \lyricmode { Ten -- or }
pianoRHMus = \relative c { \key c \major c'1 d }
pianoLHMus = \relative c { \key c \major c1 d }
violinIMusic = \relative c { \key c \major c'1 d }
violinIIMusic = \relative c { \key c \major c'1 d }
violaMusic = \relative c { \clef alto \key c \major c'1 d }
celloMusic = \relative c { \clef bass \key c \major c1 d }
bassMusic = \relative c { \clef "bass_8" \key c \major c,1 d }

\score {
  \new GrandStaff = "GrandStaff_score" <<
    \new StaffGroup = "StaffGroup_woodwinds" <<
      \new Staff = "Staff_flute" {
        \set Staff.instrumentName = #"Flute"
        \fluteMusic
      }
      \new Staff = "Staff_oboe" {
        \set Staff.instrumentName = #"Oboe"
        \oboeMusic
      }
      \new Staff = "Staff_clarinet" {
        \set Staff.instrumentName = \markup \concat { "Clarinet in B" \flat }
        \transposition bes
        \transpose bes c' \clarinetMusic
      }
      \new Staff = "Staff_bassoon" {
        \set Staff.instrumentName = #"Bassoon"
        \bassoonMusic
      }
    >>
  \new StaffGroup = "StaffGroup_brass" <<
    \new GrandStaff <<
      \new Staff = "Staff_hornI" {
        \set Staff.instrumentName = #"Horn I"
        \transposition f
        \transpose f c' \hornIMusic
      }
      \new Staff = "Staff_hornII" {
        \set Staff.instrumentName = #"Horn II"
        \clef bass
        \transposition f'
        \transpose f c \hornIIMusic
      }
    >>
  \new Staff = "Staff_trumpet" {

```

```

        \set Staff.instrumentName = #"Trumpet in C"
        \trumpetMusic
    }
    \new Staff = "Staff_trombone" {
        \set Staff.instrumentName = #"Trombone"
        \clef bass
        \tromboneMusic
    }
>>
\new RhythmicStaff = "RhythmicStaff_percussion" <<
    \set RhythmicStaff.instrumentName = #"Percussion"
    \percussionMusic
>>
\new PianoStaff <<
    \set PianoStaff.instrumentName = #"Piano"
    \new Staff { \pianoRHMus }
    \new Staff {
        \clef bass
        \pianoLHMusic
    }
>>
\new ChoirStaff = "ChoirStaff_choir" <<
    \new Staff = "Staff_soprano" {
        \set Staff.instrumentName = #"Soprano"
        \new Voice = "soprano"
        \sopranoMusic
    }
    \new Lyrics \lyricsto "soprano" { \sopranoLyrics }
    \new GrandStaff = "GrandStaff_altos" \with { \accepts Lyrics } <<
        \new Staff = "Staff_altoI" {
            \set Staff.instrumentName = #"Alto I"
            \new Voice = "altoI"
            \altoIMusic
        }
        \new Lyrics \lyricsto "altoI" { \altoILyrics }
        \new Staff = "Staff_altoII" {
            \set Staff.instrumentName = #"Alto II"
            \new Voice = "altoII"
            \altoIIMusic
        }
        \new Lyrics \lyricsto "altoII" { \altoIILyrics }
    >>
    \new Staff = "Staff_tenor" {
        \set Staff.instrumentName = #"Tenor"
        \clef "treble_8"
        \new Voice = "tenor"
        \tenorMusic
    }
    \new Lyrics \lyricsto "tenor" { \tenorLyrics }
>>
\new StaffGroup = "StaffGroup_strings" <<
    \new GrandStaff = "GrandStaff_violins" <<

```

```
\new Staff = "Staff_violinI" {
  \set Staff.instrumentName = #"Violin I"
  \violinIMusic
}
\new Staff = "Staff_violinII" {
  \set Staff.instrumentName = #"Violin II"
  \violinIIMusic
}
>>
\new Staff = "Staff_viola" {
  \set Staff.instrumentName = #"Viola"
  \violaMusic
}
\new Staff = "Staff_cello" {
  \set Staff.instrumentName = #"Cello"
  \celloMusic
}
\new Staff = "Staff_bass" {
  \set Staff.instrumentName = #"Double Bass"
  \bassMusic
}
>>
>>
}
```

This musical score template is organized into systems for various instruments and voices. The instruments are listed on the left, and their corresponding staves are on the right. The staves are grouped with brackets to indicate sections: Woodwinds (Flute, Oboe, Clarinet in B \flat , Bassoon), Horns (Horn I, Horn II), Trumpets and Trombones (Trumpet in C, Trombone), Percussion, Piano, Strings (Violin I, Violin II, Viola, Cello, Double Bass), and Vocalists (Soprano, Alto I, Alto II, Tenor). Each staff begins with a common time signature 'C'. The vocal parts include lyrics: Soprano (Sop - ra), Alto I (A - one), Alto II (A - two), and Tenor (Ten - or). The score is divided into measures by vertical bar lines, with a double bar line indicating the end of a phrase. A small number '8' is visible at the bottom left of the page.

Flute

Oboe

Clarinet in B \flat

Bassoon

Horn I

Horn II

Trumpet in C

Trombone

Percussion

Piano

Soprano

Alto I

Alto II

Tenor

Violin I

Violin II

Viola

Cello

Double Bass

Sop - ra

A - one

A - two

Ten - or

8

Piano template (simple)

Here is a simple piano staff with some notes.

```
upper = \relative c'' {
  \clef treble
  \key c \major
  \time 4/4

  a4 b c d
}

lower = \relative c {
  \clef bass
  \key c \major
  \time 4/4

  a2 c
}

\score {
  \new PianoStaff <<
    \set PianoStaff.instrumentName = #"Piano  "
    \new Staff = "upper" \upper
    \new Staff = "lower" \lower
  >>
  \layout { }
  \midi { }
}
```



Piano template with centered dynamics

Many piano scores have the dynamics centered between the two staves. The `Dynamics` context, if placed between staves, will automatically position dynamics correctly.

```
global = {
  \key c \major
  \time 4/4
}

upper = \relative c'' {
  \clef treble
  a4 b c d
}

lower = \relative c {
```

```

\clef bass
a2 c
}

dynamics = {
  s2\fff\> s4 s\!\pp
}

pedal = {
  s2\sustainOn s\sustainOff
}

\score {
  \new PianoStaff = "PianoStaff_pf" <<
    \new Staff = "Staff_pfUpper" << \global \upper >>
    \new Dynamics = "Dynamics_pf" \dynamics
    \new Staff = "Staff_pfLower" << \global \lower >>
    \new Dynamics = "pedal" \pedal
  >>
  \layout { }
}

\score {
  \new PianoStaff = "PianoStaff_pf" <<
    \new Staff = "Staff_pfUpper" << \global \upper \dynamics \pedal >>
    \new Staff = "Staff_pfLower" << \global \lower \dynamics \pedal >>
  >>
  \midi { }
}

```



Piano template with centered lyrics

Instead of having a full staff for the melody and lyrics, lyrics can be centered between the staves of a piano staff.

```

upper = \relative c'' {
  \clef treble
  \key c \major
  \time 4/4

  a4 b c d
}

lower = \relative c {

```

```

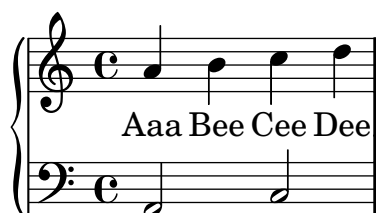
\clef bass
\key c \major
\time 4/4

a2 c
}

text = \lyricmode {
  Aaa Bee Cee Dee
}

\score {
  \new GrandStaff <<
    \new Staff = upper { \new Voice = "singer" \upper }
    \new Lyrics \lyricsto "singer" \text
    \new Staff = lower { \lower }
  >>
  \layout {
    \context {
      \GrandStaff
      \accepts "Lyrics"
    }
    \context {
      \Lyrics
      \consists "Bar_engraver"
    }
  }
  \midi { }
}

```



Piano template with melody and lyrics

Here is a typical song format: one staff with the melody and lyrics, with piano accompaniment underneath.

```

melody = \relative c'' {
  \clef treble
  \key c \major
  \time 4/4

  a b c d
}

text = \lyricmode {
  Aaa Bee Cee Dee
}

```

```

}

upper = \relative c'' {
  \clef treble
  \key c \major
  \time 4/4

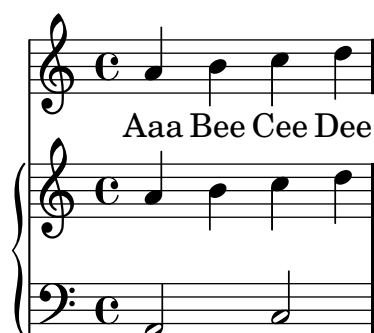
  a4 b c d
}

lower = \relative c {
  \clef bass
  \key c \major
  \time 4/4

  a2 c
}

\score {
  <<
    \new Voice = "mel" { \autoBeamOff \melody }
    \new Lyrics \lyricsto mel \text
    \new PianoStaff <<
      \new Staff = "upper" \upper
      \new Staff = "lower" \lower
    >>
  >>
  \layout {
    \context { \Staff \RemoveEmptyStaves }
  }
  \midi { }
}

```



Score for diatonic accordion

A template to write a score for a diatonic accordion.

- There is a horizontal staff indicating if the accordion must be pushed (thick line) or pulled (thin line)
- There is a small rhythmic staff with lyrics that describes the bass buttons to press. The bar lines are made from gridlines

- The tabulator staff for diatonic accordions shows the geographic position of the buttons and not (as for every other instrument) the pitch of the notes; the keys on the melody-side of the accordion are placed in three columns and about 12 rows

In the tabulator staff notation the outermost column is described with notes between lines, the innermost column is described with notes between lines and a cross as accidental, and the middle column is described with notes on a line, whereby the row in the middle is represented on the middle line in the staff.

Some words to transpose piano notes to the diatonic accordion:

1. Every diatonic accordion is built for some keys only (for example, for the keys of C major and F major), so it is important to transpose a piano melody to match one of these keys. Transpose the source code, not only the output because this code is required later on to translate it once more to the tabulator staff. This can be done with the command `displayLilyMusic`.

2. You have to alternate the push- and pull-direction of the accordion regularly. If the player has a too long part to pull the accordion gets broken. On the other hand, some harmonies are only available in one direction. Considering this, decide which parts of the melody are the push-parts and which the pull-parts.

3. For each pull- or push-part translate the piano notes to the according tabulature representation.

This snippet comes with a useful optional macro for the jEdit text editor.

```
verse = \lyricmode { Wie gross bist du! Wie gross bist du! }
```

```
harmonies = \new ChordNames \chordmode {
  \germanChords
  \set chordChanges = ##t
  bes8 bes8 bes8
  es2 f
  bes1
}
```

```
NoStem = \override Stem #'transparent = ##t
NoNoteHead = \override NoteHead #'transparent = ##t
ZeroBeam = \override Beam #'positions = #'(0 . 0)
```

```
staffTabLine = \new Staff \with {
  \remove "Time_signature_engraver"
  \remove "Clef_engraver"
} {
  \override Staff.StaffSymbol #'line-positions = #'(0)
  % Shows one horizontal line. The vertical line (simulating a bar-line) is simulated with a
  \set Staff.midiInstrument = #"choir aahs"
  \key c \major
  \relative c''
  {
    % disable the following line to see the the noteheads while writing the song
    \NoNoteHead
    \override NoteHead #'no-ledgers = ##t

    % The beam between 8th-notes is used to draw the push-line
    %How to fast write the push-lines:
    % 1. write repeatedly 'c c c c c c c c |' for the whole length of the song
```

```

% 2. uncomment the line \NoNoteHead
% 3. compile
% 4. Mark the positions on which push/pull changes.
%     In the score-picture click on the position the push- or pull-part starts
%     (on the noteHead, the cursor will change to a hand-icon).
%     The cursor in the source code will jump just at this position.
% a) If a push-part starts there, replace the 'c' by an 'e['
% b) If a pull-part starts there, replace the 'c' by an 's'
% 5. Switch into 'overwrite-mode' by pressing the 'ins' key.
% 6. For the pull-parts overwrite the 'c' with 's'
% 7. For every push-part replace the last 'c' with 'e['
%     8. Switch into 'insert-mode' again
% 9. At last it should look lik e.g. (s s e[ c | c c c c c c c c | c c c c c c e] s s)
% 10. re-enable the line \NoNoteHead
\autoBeamOff
\ZeroBeam
s8 s s e[ c c c c c c e] | s s s s s
}
}

%{
%}

% Accordion melody in tabulator score
% 1. Place a copy of the piano melody below
% 2. Separate piano melody into pull- and push-parts according to the staffTabLine you've a
% 3. For each line: Double the line. Remark the 1st one (Keeps unchanged as reference) and t
%     or the macros 'conv2diaton push.bsh' and 'conv2diaton pull.bsh'
% Tips:
% - In jEdit Search & Replace mark the Option 'Keep Dialog'

AccordionTabTwoCBesDur = {
% pull 1
%<f' bes'>8 <f' a'>8 <d' bes'>8 |
<g' a'>8 <g' b'>8 <e' a'>8 |
% push 2
%<g' c'>4 <f' d'> <g' ees'> <f' a'> |
<g' a'>4 <d' eisis'> <g' bisis'> <d' f'> |
% pull 3
% <f' bes'>2 r8 }
<g' a'>2 r8 }

AccordionTab= { \dynamicUp
% 1. Place a copy of the piano melody above
% 2. Separate piano melody into pull- and push-parts according to the staffTabLine you've
% 3. For each line: Double the line. Remark the 1st one (Keeps unchanged as reference) and
%     change the second line using the transformation paper
% Tips:
% - In jEdit Search & Replace mark the Option 'Keep Dialog'
% -
\AccordionTabTwoCBesDur
}

```

```

\layout {
  \context {
    \Score
    % The vertical line (simulating a bar-line) in
    % the staffBassRhythm is a gridline
    \consists "Grid_line_span_engraver"
  }
  \context {
    \Staff
    \consists "Grid_point_engraver"
    gridInterval = #(ly:make-moment 4 4) % 4/4 - tact. How many beats per bar
    % The following line has to be adjusted O-F-T-E-N.
    \override GridPoint #'Y-extent = #'(-2 . -21)
  }
  \context {
    \ChoirStaff
    \remove "System_start_delimiter_engraver"
  }
}

staffVoice = \new Staff = astaffvoice {
  \time 4/4
  \set Staff.instrumentName = "Voice"
  \set Staff.midiInstrument = "voice oohs"
  \key bes \major
  \partial 8*3
  \clef treble
  {
    \context Voice = "melodyVoi"
    { <f' bes'>8 <f' a'>8 <d' bes'>8 | <g' c''>4 <f' d''> <g' es''> <f' a'> | <f' bes'>2 r8
    \bar "|."
  }
}

staffAccordionMel =
\new Staff \with { \remove "Clef_engraver" } {
  #(set-accidental-style 'forget) %Set the accidentals (Vorzeichen) for each note,
  %do not remember them for the rest of the measure.
  \time 4/4
  \set Staff.instrumentName="Accordion"
  \set Staff.midiInstrument="voice oohs"
  \key c \major
  \clef treble
  { \AccordionTab \bar "|." }
}

AltOn =
#(define-music-function (parser location mag) (number?)
  #{ \override Stem #'length = #$( * 7.0 mag)

```

```

\override NoteHead #'font-size =
#$(inexact->exact (* (/ 6.0 (log 2.0)) (log mag))) #})

AltOff = {
  \revert Stem #'length
  \revert NoteHead #'font-size
}

BassRhytm = {s4 s8 | c2 c2 | c2 s8 }
LyricBassRhythmI= \lyricmode { c b | c }

staffBassRhytm =
\new Staff = staffbass \with { \remove "Clef_engraver" } {
  % This is not a RhythmicStaff because it must be possible to append lyrics.

  \override Score.GridLine #'extra-offset = #'( 13.0 . 0.0 ) % x.y
  \override Staff.StaffSymbol #'line-positions = #'( 0 )
  % Shows one horizontal line. The vertical line (simulating a bar-line) is simulated by a g
  % Search for 'grid' in this page to find all related functions
  \time 4/4
  {
    \context Voice = "VoiceBassRhytm"
    \stemDown \AltOn #0.6
    \relative c''
    {
      \BassRhytm
    }
    \AltOff
    \bar "|."
  }
}

\score {
  \new ChoirStaff <<
    \harmonies
    \staffVoice
    \context Lyrics = "lmelodyVoi"
    \with { alignBelowContext = astaffvoice }
    \lyricsto melodyVoi \verse
    \staffAccordionMel
    \staffTabLine
    \staffBassRhytm
    \context Lyrics = "lBassRhytmAboveI"
    \with { alignAboveContext = staffbass }
    \lyricsto VoiceBassRhytm \LyricBassRhythmI
  >>
}
%}

```

Voice

Wie gross bist du! Wie gross bist du!

Accordion

B E \flat F B

Single staff template with notes lyrics and chords

This template allows the preparation of a song with melody, words, and chords.

```
melody = \relative c' {
  \clef treble
  \key c \major
  \time 4/4

  a4 b c d
}

text = \lyricmode {
  Aaa Bee Cee Dee
}

harmonies = \chordmode {
  a2 c
}

\score {
  <<
    \new ChordNames {
      \set chordChanges = ##t
      \harmonies
    }
    \new Voice = "one" { \autoBeamOff \melody }
    \new Lyrics \lyricsto "one" \text
  >>
  \layout { }
  \midi { }
}
```

A C

Aaa Bee Cee Dee

Single staff template with notes lyrics chords and frets

Here is a simple lead sheet template with melody, lyrics, chords and fret diagrams.

```

verseI = \lyricmode {
  \set stanza = #"1."
  This is the first verse
}

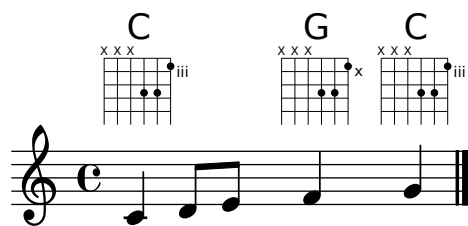
verseII = \lyricmode {
  \set stanza = #"2."
  This is the second verse.
}

theChords = \chordmode {
  % insert chords for chordnames and fretboards here
  c2 g4 c
}

staffMelody = \relative c' {
  \key c \major
  \clef treble
  % Type notes for melody here
  c4 d8 e f4 g
  \bar "|"
}

\score {
  <<
    \context ChordNames { \theChords }
    \context FretBoards { \theChords }
    \new Staff {
      \context Voice = "voiceMelody" { \staffMelody }
    }
    \new Lyrics = "lyricsI" {
      \lyricsto "voiceMelody" \verseI
    }
    \new Lyrics = "lyricsII" {
      \lyricsto "voiceMelody" \verseII
    }
  >>
  \layout { }
  \midi { }
}

```



1. This is the first verse
2. This is the second verse.

Single staff template with notes and chords

Want to prepare a lead sheet with a melody and chords? Look no further!

```
melody = \relative c' {
  \clef treble
  \key c \major
  \time 4/4

  f4 e8[ c] d4 g
  a2 ~ a
}

harmonies = \chordmode {
  c4:m f:min7 g:maj c:aug
  d2:dim b:sus
}

\score {
  <<
    \new ChordNames {
      \set chordChanges = ##t
      \harmonies
    }
    \new Staff \melody
  >>
  \layout{ }
  \midi { }
}
```



Single staff template with notes and lyrics

This small template demonstrates a simple melody with lyrics. Cut and paste, add notes, then words for the lyrics. This example turns off automatic beaming, which is common for vocal parts. To use automatic beaming, change or comment out the relevant line.

```
melody = \relative c' {
  \clef treble
  \key c \major
  \time 4/4

  a4 b c d
}
```

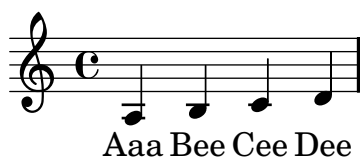
```

}

text = \lyricmode {
  Aaa Bee Cee Dee
}

\score{
  <<
    \new Voice = "one" {
      \autoBeamOff
      \melody
    }
    \new Lyrics \lyricsto "one" \text
  >>
  \layout { }
  \midi { }
}

```



Single staff template with only notes

This very simple template gives you a staff with notes, suitable for a solo instrument or a melodic fragment. Cut and paste this into a file, add notes, and you're finished!

```

melody = \relative c' {
  \clef treble
  \key c \major
  \time 4/4

  a4 b c d
}

```

```

\score {
  \new Staff \melody
  \layout { }
  \midi { }
}

```



String quartet template (simple)

This template demonstrates a simple string quartet. It also uses a `\global` section for time and key signatures

```
global= {
  \time 4/4
  \key c \major
}

violinOne = \new Voice \relative c' {
  \set Staff.instrumentName = #"Violin 1 "

  c2 d
  e1

  \bar "|."
}

violinTwo = \new Voice \relative c' {
  \set Staff.instrumentName = #"Violin 2 "

  g2 f
  e1

  \bar "|."
}

viola = \new Voice \relative c' {
  \set Staff.instrumentName = #"Viola "
  \clef alto

  e2 d
  c1

  \bar "|."
}

cello = \new Voice \relative c' {
  \set Staff.instrumentName = #"Cello "
  \clef bass

  c2 b
  a1

  \bar "|."
}

\score {
  \new StaffGroup <<
    \new Staff << \global \violinOne >>
    \new Staff << \global \violinTwo >>
    \new Staff << \global \viola >>
    \new Staff << \global \cello >>
  >>
  \layout { }
  \midi { }
```

}

String quartet template with separate parts

The “String quartet template” snippet produces a nice string quartet, but what if you needed to print parts? This new template demonstrates how to use the `\tag` feature to easily split a piece into individual parts.

You need to split this template into separate files; the filenames are contained in comments at the beginning of each file. `piece.ly` contains all the music definitions. The other files – `score.ly`, `vn1.ly`, `vn2.ly`, `vla.ly`, and `vlc.ly` – produce the appropriate part.

Do not forget to remove specified comments when using separate files!

```

%% piece.ly
%% (This is the global definitions file)

global= {
  \time 4/4
  \key c \major
}

Violinone = \new Voice { \relative c' {
  \set Staff.instrumentName = #"Violin 1 "

  c2 d e1

  \bar "|" } } %*****
Violintwo = \new Voice { \relative c' {
  \set Staff.instrumentName = #"Violin 2 "

  g2 f e1

  \bar "|" } } %*****
Viola = \new Voice { \relative c' {
  \set Staff.instrumentName = #"Viola "
  \clef alto

  e2 d c1

```

```
\bar "|" ."} } %*****
```

```
Cello = \new Voice { \relative c' {
  \set Staff.instrumentName = #"Cello "
  \clef bass
```

```

c2 b a1
```

```
\bar "|" ."} } %*****
```

```
music = {
  <<
    \tag #'score \tag #'vn1 \new Staff { << \global \Violinone >> }
    \tag #'score \tag #'vn2 \new Staff { << \global \Violintwo>> }
    \tag #'score \tag #'vla \new Staff { << \global \Viola>> }
    \tag #'score \tag #'vlc \new Staff { << \global \Cello>> }
  >>
}
```

```
%%% These are the other files you need to save on your computer
```

```
%%%%% score.ly
```

```
%%%%% (This is the main file)
```

```
%\include "piece.ly" %%% uncomment this line when using a separate file
```

```
#{set-global-staff-size 14)
```

```
\score {
  \new StaffGroup \keepWithTag #'score \music
  \layout { }
  \midi { }
}
```

```
%{ Uncomment this block when using separate files
```

```
%%%%% vn1.ly
```

```
%%%%% (This is the Violin 1 part file)
```

```
\include "piece.ly"
```

```
\score {
  \keepWithTag #'vn1 \music
  \layout { }
}
```

```
%%%%% vn2.ly
```

```
%%%%% (This is the Violin 2 part file)
```

```
\include "piece.ly"
```

```
\score {
  \keepWithTag #'vn2 \music
  \layout { }
}
```

```
}
```

```
%%%%% vla.ly
%%%%% (This is the Viola part file)
```

```
\include "piece.ly"
\score {
  \keepWithTag #'vla \music
  \layout { }
}
```

```
%%%%% vlc.ly
%%%%% (This is the Cello part file)
```

```
\include "piece.ly"
\score {
  \keepWithTag #'vlc \music
  \layout { }
}
```

```
%}
```

Vocal ensemble template with automatic piano reduction

This template adds an automatic piano reduction to the standard SATB vocal score demonstrated in “Vocal ensemble template”. This demonstrates one of the strengths of LilyPond – you can use a music definition more than once. If any changes are made to the vocal notes (say, `tenorMusic`), then the changes will also apply to the piano reduction.

```
global = {
  \key c \major
  \time 4/4
}

sopMusic = \relative c'' {
  c4 c c8[( b)] c4
}
sopWords = \lyricmode {
  hi hi hi hi
}
```

```

altoMusic = \relative c' {
  e4 f d e
}
altoWords = \lyricmode {
  ha ha ha ha
}

tenorMusic = \relative c' {
  g4 a f g
}
tenorWords = \lyricmode {
  hu hu hu hu
}

bassMusic = \relative c {
  c4 c g c
}
bassWords = \lyricmode {
  ho ho ho ho
}

\score {
  <<
    \new ChoirStaff <<
      \new Lyrics = sopranos { s1 }
      \new Staff = women <<
        \new Voice = sopranos { \voiceOne << \global \sopMusic >> }
        \new Voice = altos { \voiceTwo << \global \altoMusic >> }
      >>
      \new Lyrics = altos { s1 }
      \new Lyrics = tenors { s1 }
      \new Staff = men <<
        \clef bass
        \new Voice = tenors { \voiceOne << \global \tenorMusic >> }
        \new Voice = basses { \voiceTwo << \global \bassMusic >> }
      >>
      \new Lyrics = basses { s1 }
      \context Lyrics = sopranos \lyricsto sopranos \sopWords
      \context Lyrics = altos \lyricsto altos \altoWords
      \context Lyrics = tenors \lyricsto tenors \tenorWords
      \context Lyrics = basses \lyricsto basses \bassWords
    >>
    \new PianoStaff <<
      \new Staff <<
        \set Staff.printPartCombineTexts = ##f
        \partcombine
        << \global \sopMusic >>
        << \global \altoMusic >>
      >>
      \new Staff <<
        \clef bass

```

```

\set Staff.printPartCombineTexts = ##f
\partcombine
<< \global \tenorMusic >>
<< \global \bassMusic >>
>>
>>
>>
\layout {
  \context {
    % a little smaller so lyrics
    % can be closer to the staff
    \Staff
    \override VerticalAxisGroup #'minimum-Y-extent = #'(-3 . 3)
  }
}
}

```

The image shows a musical score for a vocal ensemble. It consists of three systems of staves. The first system has two staves: a soprano staff (treble clef) and a bass staff (bass clef). The second system has two staves: a tenor staff (treble clef) and a bass staff (bass clef). The third system has two staves: a soprano staff (treble clef) and a bass staff (bass clef). The lyrics are aligned above and below the staves. The lyrics are: "hi hi hi hi" above the first soprano staff, "ha ha ha ha" above the first tenor staff, "hu hu hu hu" below the first tenor staff, "ho ho ho ho" above the second soprano staff, and "ho ho ho ho" below the second soprano staff.

Vocal ensemble template with lyrics aligned below and above the staves

This template is basically the same as the simple “Vocal ensemble” template, with the exception that here all the lyrics lines are placed using `alignAboveContext` and `alignBelowContext`.

```

global = {
  \key c \major
  \time 4/4
}

sopMusic = \relative c'' {
  c4 c c8[( b)] c4
}

sopWords = \lyricmode {
  hi hi hi hi
}

```

```

}

altoMusic = \relative c' {
  e4 f d e
}
altoWords = \lyricmode {
  ha ha ha ha
}

tenorMusic = \relative c' {
  g4 a f g
}
tenorWords = \lyricmode {
  hu hu hu hu
}

bassMusic = \relative c {
  c4 c g c
}
bassWords = \lyricmode {
  ho ho ho ho
}

\score {
  \new ChoirStaff <<
    \new Staff = women <<
      \new Voice = "sopranos" { \voiceOne << \global \sopMusic >> }
      \new Voice = "altos" { \voiceTwo << \global \altoMusic >> }
    >>
    \new Lyrics \with { alignAboveContext = women } \lyricsto sopranos \sopWords
    \new Lyrics \with { alignBelowContext = women } \lyricsto altos \altoWords
    % we could remove the line about this with the line below, since we want
    % the alto lyrics to be below the alto Voice anyway.
    % \new Lyrics \lyricsto altos \altoWords

    \new Staff = men <<
      \clef bass
      \new Voice = "tenors" { \voiceOne << \global \tenorMusic >> }
      \new Voice = "basses" { \voiceTwo << \global \bassMusic >> }
    >>
    \new Lyrics \with { alignAboveContext = men } \lyricsto tenors \tenorWords
    \new Lyrics \with { alignBelowContext = men } \lyricsto basses \bassWords
    % again, we could replace the line above this with the line below.
    % \new Lyrics \lyricsto basses \bassWords
  >>
  \layout {
    \context {
      % a little smaller so lyrics
      % can be closer to the staff
      \Staff
      \override VerticalAxisGroup #'minimum-Y-extent = #'(-3 . 3)
    }
  }
}

```



Vocal ensemble template with verse and refrain

This template creates a score which starts with a solo verse and continues into a refrain for two voices. It also demonstrates the use of spacer rests within the `\global` variable to define meter changes (and other elements common to all parts) throughout the entire score.

```
global = {
  \key g \major

  % verse
  \time 3/4
  s2.*2
  \break

  % refrain
  \time 2/4
  s2*2
  \bar "|."
}

SoloNotes = \relative g' {
  \clef "treble"

  % verse
  g4 g g |
  b4 b b |

  % refrain
  R2*2 |
}

SoloLyrics = \lyricmode {
  One two three |
  four five six |
}

SopranoNotes = \relative c'' {
```

```

\clef "treble"

% verse
R2.*2 |

% refrain
c4 c |
g4 g |
}

SopranoLyrics = \lyricmode {
  la la |
  la la |
}

BassNotes = \relative c {
  \clef "bass"

  % verse
  R2.*2 |

  % refrain
  c4 e |
  d4 d |
}

BassLyrics = \lyricmode {
  dum dum |
  dum dum |
}

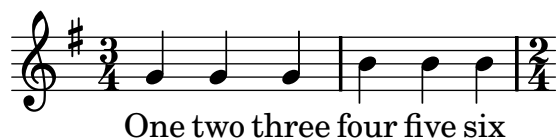
\score {
  <<
    \new Voice = "SoloVoice" << \global \SoloNotes >>
    \new Lyrics \lyricsto "SoloVoice" \SoloLyrics

    \new ChoirStaff <<
      \new Voice = "SopranoVoice" << \global \SopranoNotes >>
      \new Lyrics \lyricsto "SopranoVoice" \SopranoLyrics

      \new Voice = "BassVoice" << \global \BassNotes >>
      \new Lyrics \lyricsto "BassVoice" \BassLyrics
    >>
  >>
  \layout {
    ragged-right = ##t
    \context { \Staff
      % these lines prevent empty staves from being printed
      \RemoveEmptyStaves
      \override VerticalAxisGroup #'remove-first = ##t
    }
  }
}

```

}



Vocal ensemble template

Here is a standard four-part SATB vocal score. With larger ensembles, it is often useful to include a section which is included in all parts. For example, the time signature and key signature are almost always the same for all parts. Like in the “Hymn” template, the four voices are regrouped on only two staves.

```

global = {
  \key c \major
  \time 4/4
}

sopMusic = \relative c'' {
  c4 c c8[( b)] c4
}
sopWords = \lyricmode {
  hi hi hi hi
}

altoMusic = \relative c' {
  e4 f d e
}
altoWords = \lyricmode {
  ha ha ha ha
}

tenorMusic = \relative c' {
  g4 a f g
}
tenorWords = \lyricmode {
  hu hu hu hu
}

bassMusic = \relative c {
  c4 c g c
}

```

```

bassWords = \lyricmode {
  ho ho ho ho
}

\score {
  \new ChoirStaff <<
    \new Lyrics = sopranos { s1 }
    \new Staff = women <<
      \new Voice = "sopranos" {
        \voiceOne
        << \global \sopMusic >>
      }
      \new Voice = "altos" {
        \voiceTwo
        << \global \altoMusic >>
      }
    >>
    \new Lyrics = "altos" { s1 }
    \new Lyrics = "tenors" { s1 }
    \new Staff = men <<
      \clef bass
      \new Voice = "tenors" {
        \voiceOne
        << \global \tenorMusic >>
      }
      \new Voice = "basses" {
        \voiceTwo << \global \bassMusic >>
      }
    >>
    \new Lyrics = basses { s1 }
    \context Lyrics = sopranos \lyricsto sopranos \sopWords
    \context Lyrics = altos \lyricsto altos \altoWords
    \context Lyrics = tenors \lyricsto tenors \tenorWords
    \context Lyrics = basses \lyricsto basses \bassWords
  >>
  \layout {
    \context {
      % a little smaller so lyrics
      % can be closer to the staff
      \Staff
      \override VerticalAxisGroup #'minimum-Y-extent = #'(-3 . 3)
    }
  }
}

```

hi hi hi hi

ha ha ha ha
hu hu hu hu

ho ho ho ho