

'AAA-intro-regression.ly'

Introduction

This document presents proofs for LilyPond 2.11.28. When the text corresponds with the shown notation, we consider LilyPond Officially BugFree (tm). This document is intended for finding bugs and for documenting bugfixes.

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

TODO: order of tests (file names!), test only one feature per test. Smaller and neater tests.

(left blank intentionally)

`‘accidental-ancient.ly’`

Accidentals are available in different ancient styles, which all are collected here.



`‘accidental-cautionary.ly’`

Cautionary accidentals are indicated using either parentheses (default) or smaller accidentals.



`‘accidental-clef-change.ly’`

Accidentals are reset for clef changes.



`‘accidental-collision.ly’`

accidentals avoid stems of other notes too.



`‘accidental-double.ly’`

If two forced accidentals happen at the same time, only one sharp sign is printed.



‘accidental-forced-tie.ly’

Accidentals can be forced with ! and ? even if the notes are tied.



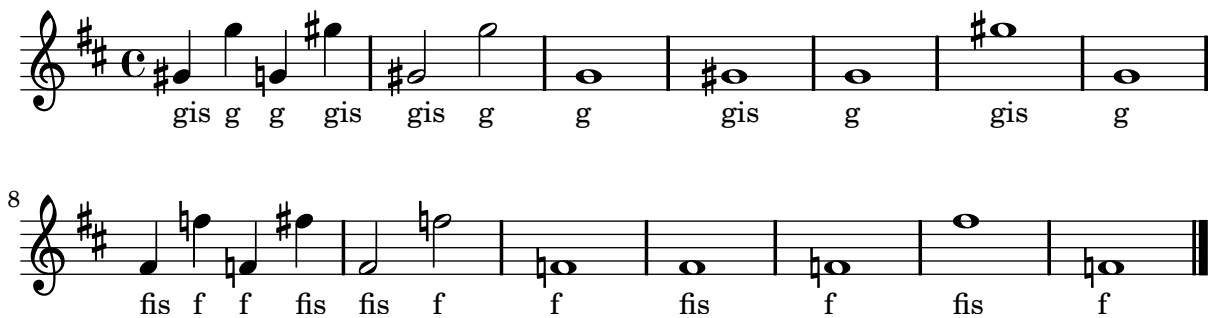
‘accidental-ledger.ly’

Ledger lines are shortened when there are accidentals. This happens only for the single ledger line close to the note head, and only if the accidental is horizontally close to the head.



‘accidental-octave.ly’

This shows how accidentals in different octaves are handled. The note names are also automatically printed but the octavation has been dropped out.



‘accidental-piano.ly’

In piano accidental style, notes in both staves influence each other. In this example, each note should have an accidental.



‘accidental-placement.ly’

Accidentals are placed as closely as possible. Accidentals in corresponding octaves are aligned. The top accidental should be nearest to the chord. The flats in a sixth should be staggered.





‘accidental-quarter.ly’

Quarter tone notation is supported, including threequarters flat.



‘accidental-single-double.ly’

A sharp sign after a double sharp sign, as well as a flat sign after a double flat sign is automatically prepended with a natural sign.



‘accidental-suggestions.ly’

setting the `suggestAccidentals` will print accidentals vertically relative to the note. This is useful for denoting Musica Ficta.



‘accidental-tie.ly’

The second and third notes should not get accidentals, 6 because they are tied to a note. However, an accidental is present if the line is broken at the tie, which happens for the G sharp.



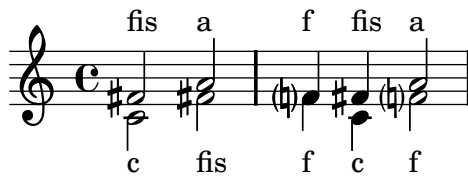
‘accidental-unbroken-tie-spacing.ly’

Tied accidentaled notes (which cause reminder accidentals) do not wreak havoc in the spacing when unbroken.



`‘accidental-voice.ly’`

This shows how modern cross voice auto cautionary accidentals are handled. The first two fisses get accidentals because they belong to different voices. The first f gets cautionary natural because of previous measure. The last f gets cautionary natural because fis was only in the other voice.



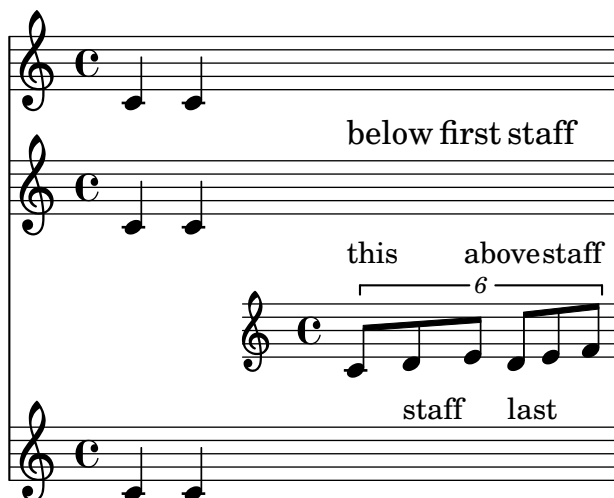
`‘accidental.ly’`

Accidentals work: the second note does not get a sharp. The third and fourth show forced and courtesy accidentals.



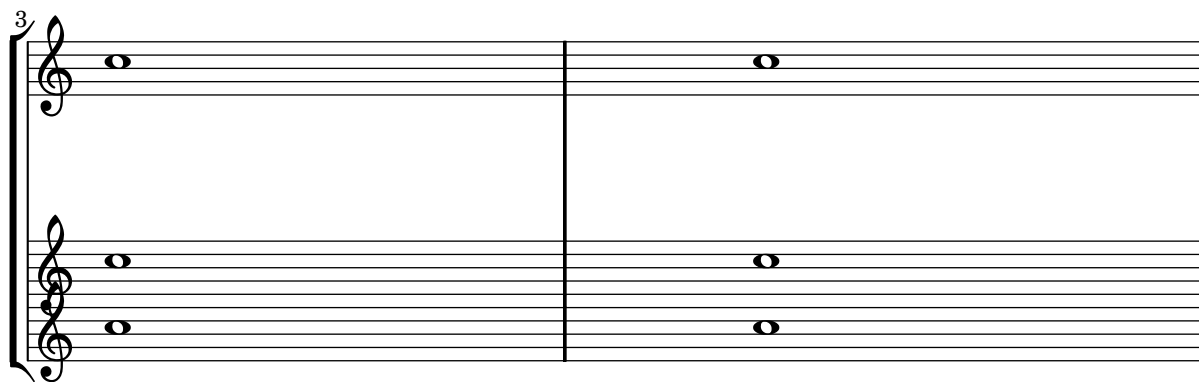
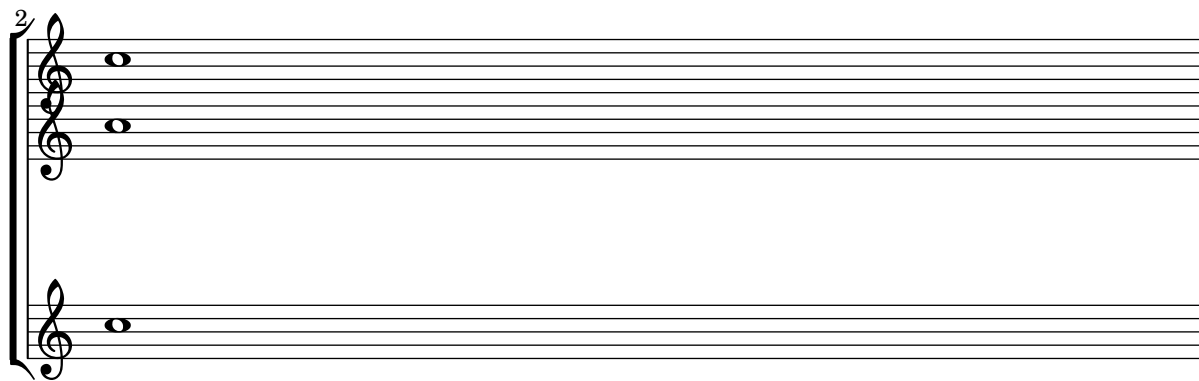
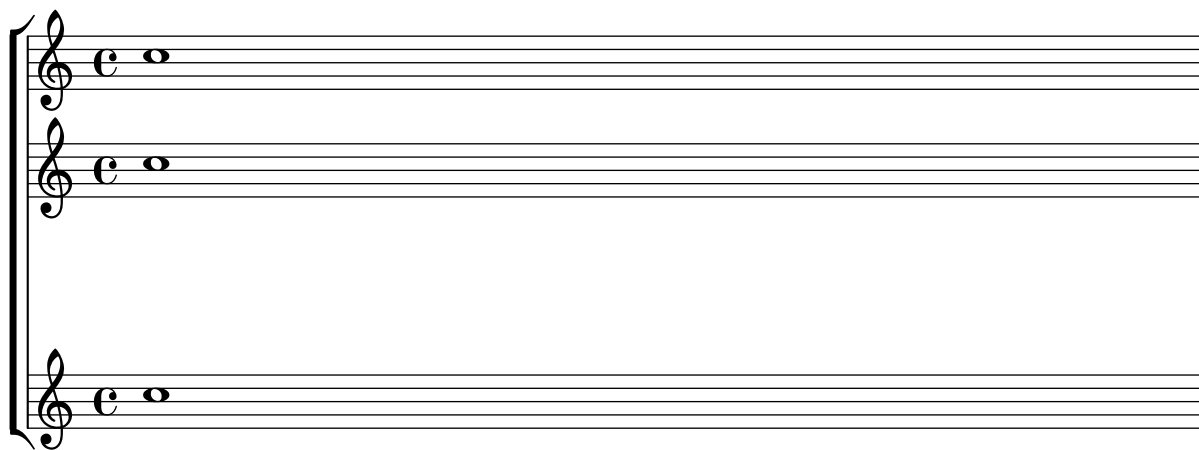
`‘alignment-order.ly’`

Newly created contexts can be inserted anywhere in the vertical alignment.



`‘alignment-vertical-manual-setting.ly’`

Alignments may be changed pre system by setting `alignment-offsets` in the `line-break-system-details` property

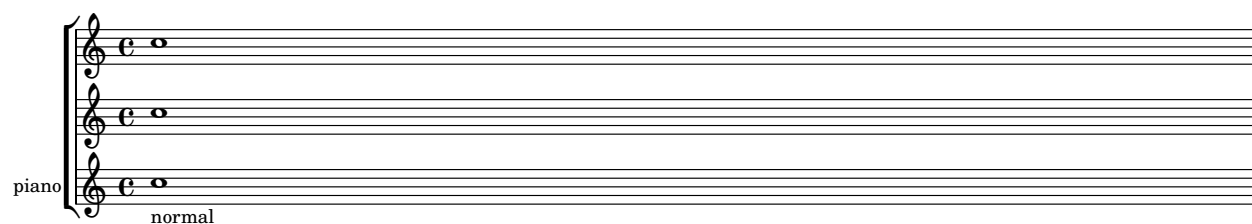


`'alignment-vertical-spacing.ly'`

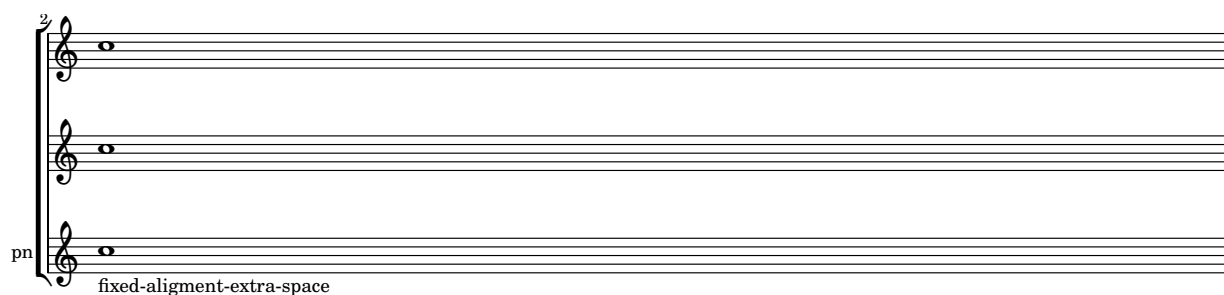
By setting properties in `NonMusicalPaperColumn`, vertical spacing of alignments can be adjusted per system.

By setting `alignment-extra-space` or `fixed-alignment-extra-space` an individual system may be stretched vertically.

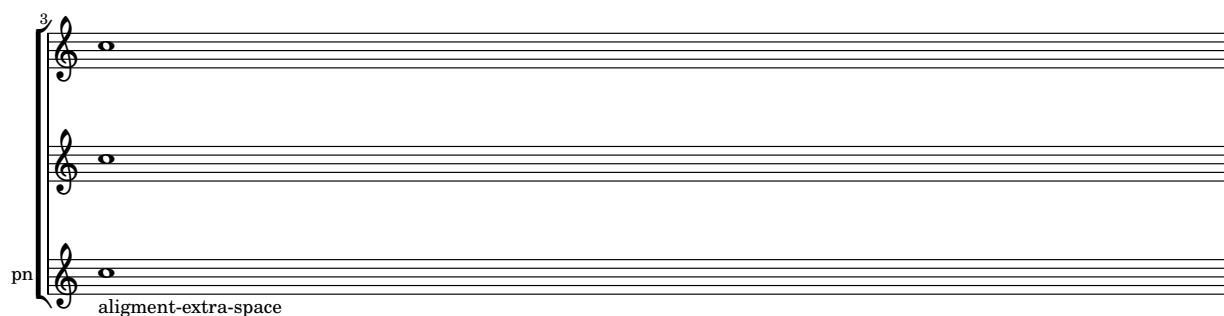
For technical reasons, `overrideProperty` has to be used for setting properties on individual object. `\override` in a `\context` block may still be used for global overrides.



A musical score system with three staves. The first staff is labeled 'piano' and the second 'normal'. The third staff is labeled 'normal'.



A musical score system with three staves. The first staff is labeled '2' and the second 'pn'. The third staff is labeled 'fixed-alignment-extra-space'.



A musical score system with three staves. The first staff is labeled '3' and the second 'pn'. The third staff is labeled 'alignment-extra-space'.

`'ambitus-pitch-ordering.ly'`

Ambituses use actual pitch not lexicographic ordering.



A musical score system with one staff. The staff is labeled 'c' and the second 'pn'.

`'ambitus.ly'`

Ambituses indicate pitch ranges for voices.

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



‘`apply-context.ly`’

With `\applyContext`, `\properties` can be modified procedurally. Applications include: checking bar numbers, smart octavation.

This example prints a bar-number during processing on stdout.



‘`apply-output.ly`’

The `\applyOutput` expression is the most flexible way to tune properties for individual grobs. Here, the layout of a note head is changed depending on its vertical position.



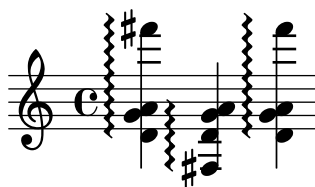
‘`arpeggio-bracket.ly`’

A square bracket on the left indicates that the player should not arpeggiate the chord.



‘`arpeggio-collision.ly`’

Arpeggio stays clear of accidentals and flipped note heads.



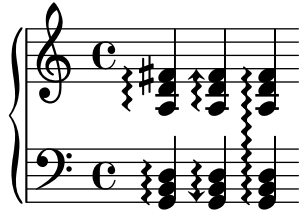
‘`arpeggio-span-one-staff.ly`’

Span arpeggios within one staff also work



`‘arpeggio.ly’`

Arpeggios are supported, both cross-staff and broken single staff.



`‘augmentum.ly’`

Augmentum dots are accounted for in horizontal spacing.



`‘auto-beam-bar.ly’`

No auto beams will be put over (manual) repeat bars.



`‘auto-beam-beaming-override.ly’`

Autobeamer remembers `subdivideBeams` and other beaming pattern related functions at the start of an autobeam.



`‘auto-beam-no-beam.ly’`

The autobeamer may be switched off for a single note with `\noBeam`.



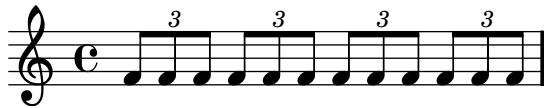
`‘auto-beam-triplet.ly’`

Automatic beaming is also done on triplets.



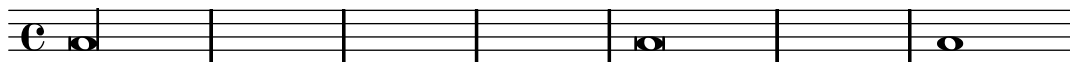
`'auto-beam-tuplets.ly'`

Tuplet-spanner should not put (visible) brackets on beams even if they're auto generated.



`'auto-beam.ly'`

Beams are place automatically; the last measure should have a single beam.



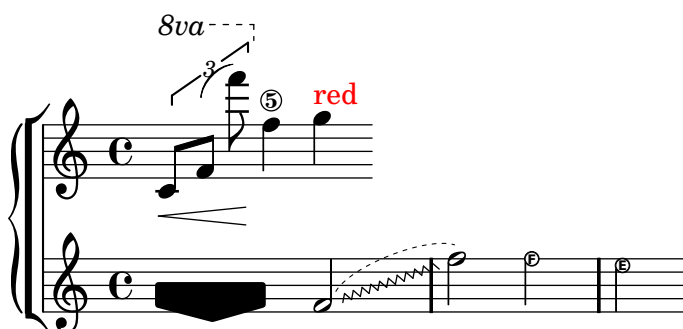
`'auto-change.ly'`

Auto change piano staff switches voices between up and down staves automatically rests are switched along with the coming note. When central C is reached, staff is not yet switched (by default).



`'backend-exercise.ly'`

Excercise all output functions



‘backend-svg.ly’

SVG output, rendered through inkscape.

SVG

LilyPond demo

Lieblih, etwas geschwind



1. Sü - ßes
2. Sü - ßes

Licht! Aus gol - denen Pfor - ten brichst du siegend durch die
та та ほへど ちり めるを Жьл дю лярн いろ はに 何

Nacht, Schöner Tag, du bist er - wacht.
та та ほへど ちり める Жьл дю лярн

cresc. f

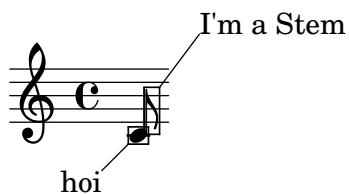
Music engraving by LilyPond 2.11.28—www.lilypond.org

bla bla bla

Music engraving by LilyPond 2.11.28—www.lilypond.org

‘balloon.ly’

With balloon texts, objects in the output can be marked, with lines and explanatory text added.



`'bar-check-redefine.ly'`

The meaning of | is stored in the identifier `pipeSymbol`.



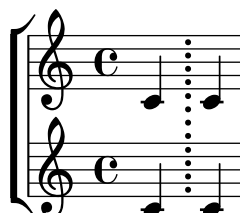
`'bar-line-dashed.ly'`

The dashes in a dashed bar line covers staff lines exactly. Dashed barlines between staves start and end on a half dash precisely.



`'bar-line-dotted.ly'`

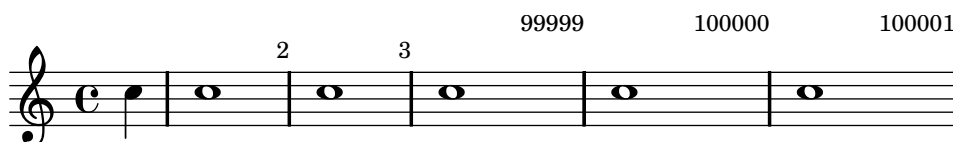
The dots in a dotted bar line are in spaces.



`'bar-number.ly'`

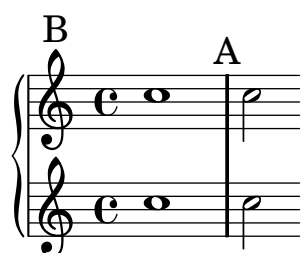
Bar number may be set and their padding adjusted individually. The counting of bar numbers is started after the anacrusis.

To prevent clashes at the beginning of a line, the padding may have to be increased.



`'bar-scripts.ly'`

Markings can be attached to (invisible) barlines.



`'beam-auto-knee.ly'`

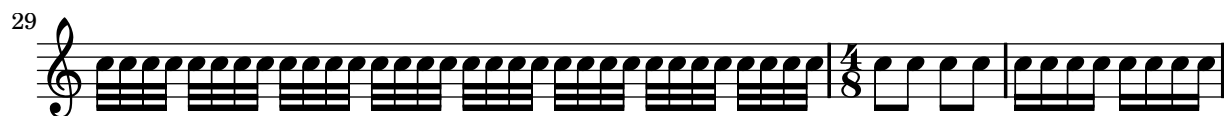
A knee is made automatically when a horizontal beam fits in a gap between note heads that is larger than a predefined threshold.



`'beam-auto.ly'`

There are presets for the `auto-beam` engraver in the case of common time signatures.

A series of musical staves in treble clef, each with a different time signature. The staves are numbered 4, 10, 12, 14, 18, 20, and 21. Each staff contains a sequence of notes, some of which are beamed together. The time signatures shown are 2/4, 4/4, 8/8, 1/4, 2/2, 3/4, 3/8, 3/2, 3/4, and 3/4. The examples demonstrate how the 'beam-auto' engraver applies different beaming rules based on the time signature.



`'beam-beamlet-grace.ly'`

Beamlets in grace notes remain readable.



`'beam-beat-grouping.ly'`

Beaming patterns obey the `beatGrouping` property.



`'beam-break-no-bar.ly'`

Broken beams have sane endings even if grobs are not present at the broken end.



2



`'beam-break.ly'`

Beams can be printed across line breaks, if forced.



`'beam-center-slope.ly'`

Simple beams on middle staffline are allowed to be slightly sloped, even if the notes have ledgers. Beams reaching beyond middle line can have bigger slope.



`'beam-concave-damped.ly'`

Beams that are not strictly concave are damped according to their concaveness.



`'beam-concave.ly'`

Fully concave beams should be horizontal. Informally spoken, concave refers to the shape of the notes that are opposite a beam. If an up-beam has high notes on its center stems, then we call it concave.

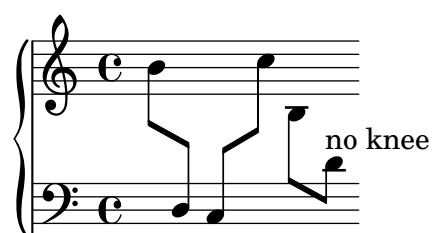
If a beam fails a test, the desired slope is printed next to it.





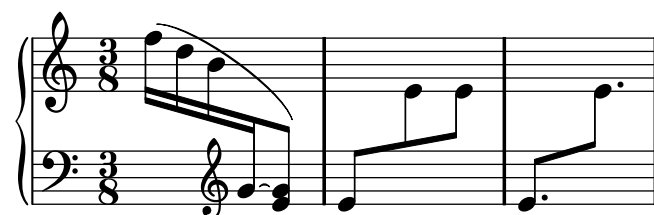
`'beam-cross-staff-auto-knee.ly'`

Automatic cross-staff knees work also (here they were produced with explicit staff switches).



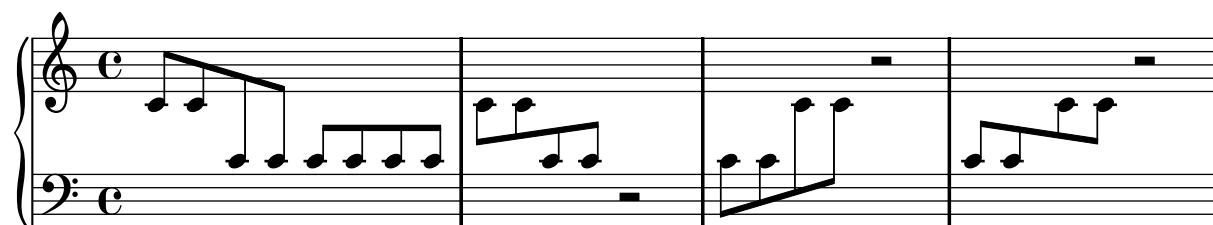
`'beam-cross-staff-slope.ly'`

Cross staff (kneed) beams do not cause extreme slopes.



`'beam-cross-staff.ly'`

Beams can be typeset over fixed distance aligned staves, beam beautification does not really work, but knees do. Beams should behave well, wherever the switching point is.



`'beam-damp.ly'`

Beams are less steep than the notes they encompass.



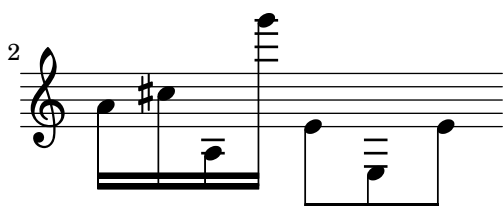
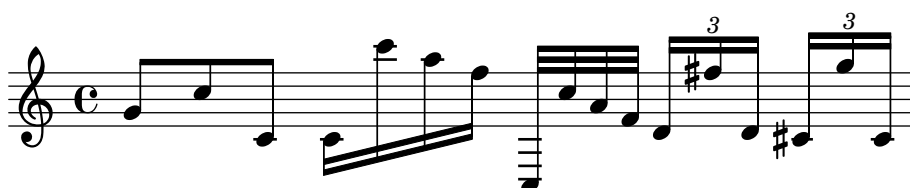
`'beam-default-lengths.ly'`

Beamed stems have standard lengths if possible. Quantization is switched off in this example.



`'beam-extreme.ly'`

Beams should behave reasonably well, even under extreme circumstances. Stems may be short, but noteheads should never touch the beam. Note that under normal circumstances, these beams would get knees here `Beam.auto-knee-gap` was set to false.



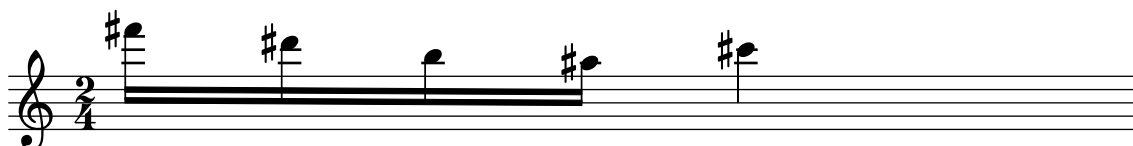
`'beam-feather.ly'`

Specifying `grow-direction` on a beam, will cause feathered beaming. The `\featherDurations` function can be used to adjust note durations.



`'beam-flat-retain-direction.ly'`

Even very flat but slanted patterns should give slanted beams.



`'beam-french.ly'`

In french style beaming, the stems do not go between beams.



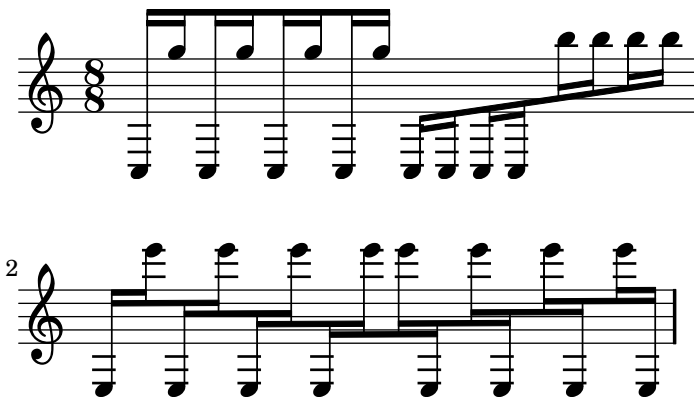
`'beam-funky-beamlet.ly'`

Funky kneed beams with beamlets also work. The beamlets should be pointing to the note head.



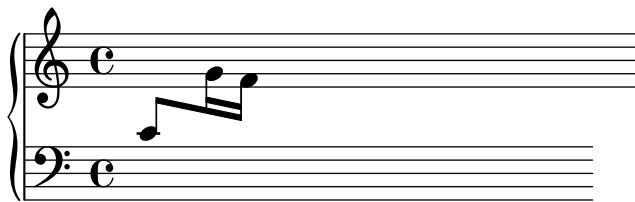
`'beam-funky.ly'`

In complex configurations of knee beaming, according to Paul Roberts, the first stem of a beam determines the direction of the beam, and as such the way that following (kneed) stems attach to the beam. This is in disagreement with the current algorithm.



`'beam-isknee.ly'`

Beams can be placed across a PianoStaff.



`'beam-knee-symmetry.ly'`

Point-symmetric beams should receive the same quanting. There is no up/down bias in the quanting code.



`'beam-length.ly'`

Beams should look the same.



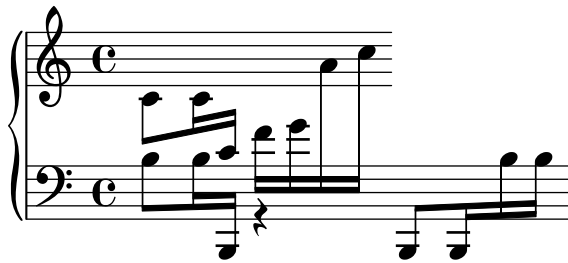
`'beam-manual-beaming.ly'`

Beaming can be overridden for individual stems.



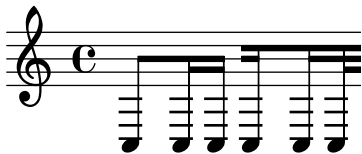
`'beam-multiple-cross-staff.ly'`

Kneaded beams (often happens with cross-staff beams) should look good when there are multiple beams: all the beams should go on continuously at the staff change. Stems in both staves reach up to the last beam.



`'beam-outside-beamlets.ly'`

Beams may overshoot stems. This is also controlled with `break-overshoot`.



`'beam-over-barline.ly'`

Explicit beams may cross barlines.



`'beam-position.ly'`

Beams on ledgered notes should always reach the middle staff line. The second beam counting from the note head side, should never be lower than the second staff line. This does not hold for grace note beams. Override with `no-stem-extend`.



‘beam-quant-standard.ly’

This file tests a few standard beam quant, taken from Ted Ross’ book. If LilyPond finds another quant, the correct quant is printed over the beam.

6

12

18

24

‘beam-quanting-32nd.ly’

Stem lengths take precedence over beam quant: ‘forbidden’ quants are only avoided for 32nd beams when they are outside of the staff. However, that leads to very long stems, which is even worse.

5

‘beam-quanting-horizontal.ly’

In this test for beam quant positions for horizontal beams, staff lines should be covered in all cases. For 32nd beams, the free stem lengths are between 2 and 1.5.



`'beam-quarter.ly'`

Quarter notes may be beamed: the beam is halted momentarily.



`'beam-rest.ly'`

The number of beams does not change on a rest.



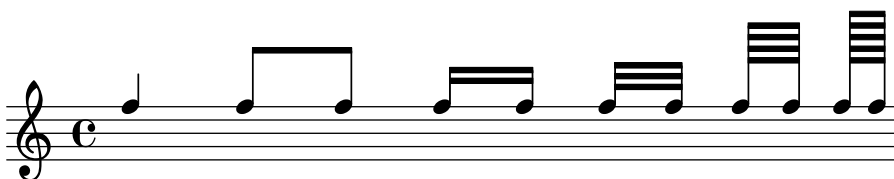
`'beam-second.ly'`

Engraving second intervals is tricky. We used to have problems with seconds being too steep, or getting too long stems. In a file like this, showing seconds, you'll spot something fishy very quickly.



`'beam-shortened-lengths.ly'`

Beams in unnatural direction, have shortened stems, but do not look too short.



`'beam-single-stem.ly'`

Single stem beams are also allowed. For such beams, clip-edges is switched off automatically.



`'beam-slope-stemlet.ly'`

For slope calculations, stemlets are treated as invisible stems.



`'beam-unconnected-beamlets.ly'`

By setting `max-beam-connect`, it is possible to create pairs of unconnected beamlets.



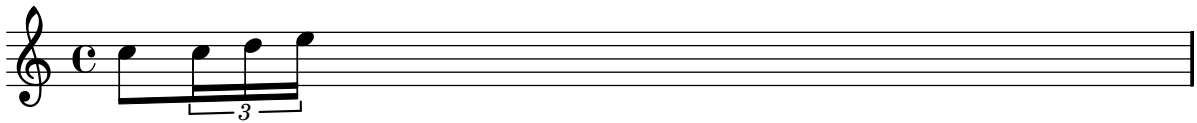
`'beaming-ternary-metrum.ly'`

Automatic beaming works also in ternary time sigs. In this case, the 8th is a beat, so the 16ths are split into two groups. This can be avoided by overriding `beatLength` to be 3 8th notes.



`'beaming.ly'`

Beaming is generated automatically. Beams may cross bar lines. In that case, line breaks are forbidden.



`'beams.ly'`

Beaming can be also given explicitly.



`'bend-after.ly'`

Falls and doits can be created with `bendAfter`. They run to the next note, or to the next barline.



`'bend-dot.ly'`

Bends avoid dots, but only if necessary.



`'break-alignment-anchor-alignment.ly'`

The default callback for break-align-anchor in clefs and time/key signatures reads the break-align-anchor-alignment property to align the anchor to the extent of the break-aligned grob.



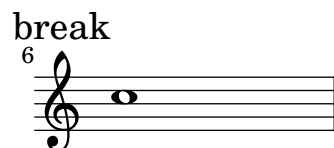
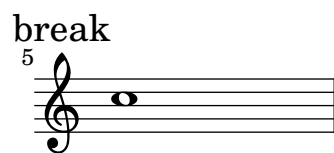
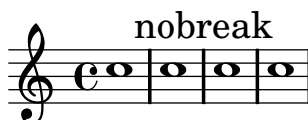
`'break-alignment-anchors.ly'`

The break-align-anchor property of a break-aligned grob gives the horizontal offset at which other grobs should attach.



`'break.ly'`

Breaks can be encouraged and discouraged using `\break` and `\noBreak`.



`'breathing-sign-ancient.ly'`

Gregorian chant notation sometimes also uses commas and ticks, but in smaller font size (we call it 'virgula' and 'caesura'). However, the most common breathing signs are divisio minima/maior/maxima and finalis, the latter three looking similar to bar glyphs.



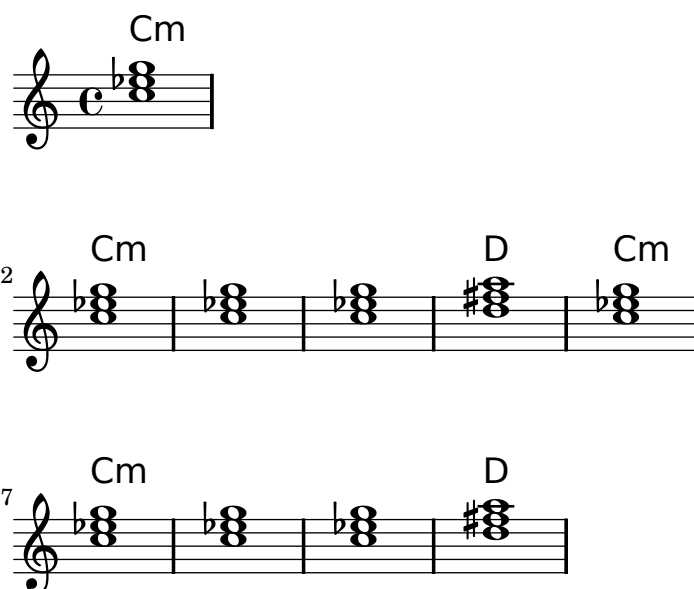
‘breathing-sign.ly’

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



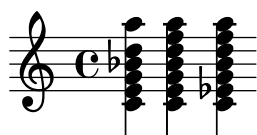
‘chord-changes.ly’

Property chordChanges: display chord names only when there’s a change in the chords scheme, but always display the chord name after a line break.



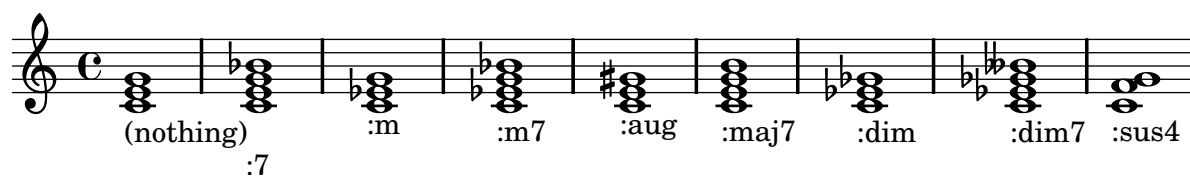
‘chord-name-entry-11.ly’

The 11 is only added to major-13 if it is mentioned explicitly.



‘chord-name-entry.ly’

Chords can be produced with the new chordname entry code (\chordmode mode), using a pitch and a suffix. Here, the suffixes are printed below pitches.



10

:sus2 :6 :m6 :7sus4 :3- :3+ :5+.3- :7 :9

19

:11 :13 :m13 :7\^{\sim}5 \^{\sim}3 /g \#/gis /a /+f /+g

`'chord-name-exceptions.ly'`

The property `chordNameExceptions` can be used to store a list of special notations for specific chords.

$C^{7/sus4}$ C^{o7}/F C^7wahh C^{o7}/F

`'chord-name-major7.ly'`

The layout of the major 7 can be tuned with `majorSevenSymbol`.

C^{\triangle}

C^{j7}

`'chord-names-bass.ly'`

In `ignatzek` inversions, a note is dropped down to act as the bass note of the chord. Bass note may be also added explicitly. Above the staff: computed chord names. Below staff: entered chord name.

F^{\triangle}/E F^{\triangle}/F F^{\triangle}/G F^{\triangle}/E F^{\triangle}/F F^{\triangle}/G
:maj7/e :maj7/f :maj7/g :maj7/+e :maj7/+f :maj7/+g

`'chord-names-languages.ly'`

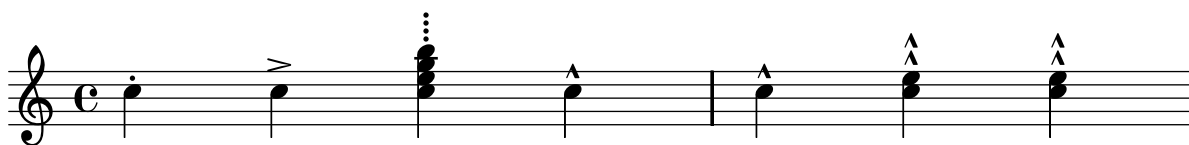
The english naming of chords (default) can be changed to german (`\germanChords` replaces B and Bes to H and B), semi-german (`\semiGermanChords` replaces B and Bes to H and Bb), italian (`\italianChords` uses Do Re Mi Fa Sol La Si), or french (`\frenchChords` replaces Re to R).

default	E/D	Cm	B/B	B [#] /B [#]	B ^b /B ^b
german	E/d	Cm	H/h	H [#] /his	B/b
semi-german	E/d	Cm	H/h	H [#] /his	B ^b /b
italian	Mi/Re	Do m	Si/Si	Si [#] /Si [#]	Si ^b /Si ^b
french	Mi/Ré	Do m	Si/Si	Si [#] /Si [#]	Si ^b /Si ^b



‘chord-scripts.ly’

Scripts can also be attached to chord elements.



‘chord-tremolo-short.ly’

Tremolo repeats can be constructed for short tremolos (total duration smaller than 1/4) too. Only some of the beams are connected to the stems.



‘chord-tremolo-single.ly’

Chord tremolos on a single note.



‘chord-tremolo-whole.ly’

chord tremolos don’t collide with whole notes.



‘chord-tremolo.ly’

Chord tremolos look like beams, but are a kind of repeat symbol. To avoid confusion, chord tremolo beams do not reach the stems, but leave a gap. Chord tremolo beams on half notes are not ambiguous, as half notes cannot appear in a regular beam, and should reach the stems.

In this example, each tremolo lasts exactly one measure.

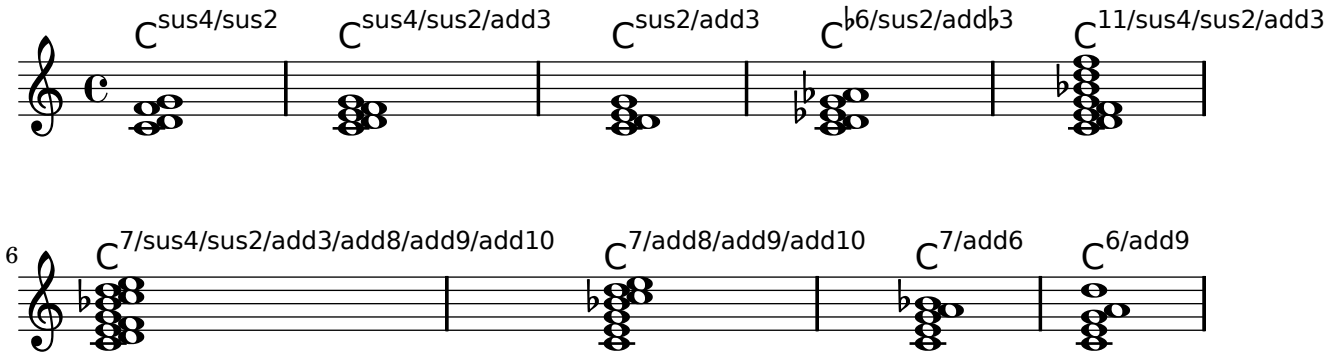
(To ensure that the spacing engine is not confused we add some regular notes as well.)





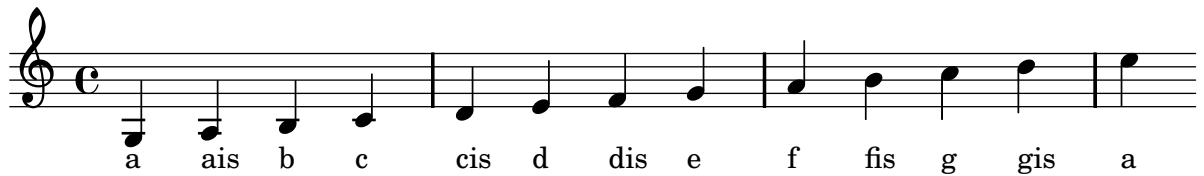
`'chords-funky-ignatzek.ly'`

Jazz chords may have unusual combinations.



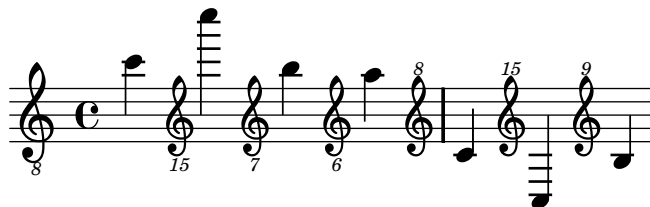
`'chromatic-scales.ly'`

`staffLineLayoutFunction` is used to change the position of the notes. This sets `staffLineLayoutFunction` to `ly:pitch-semitones` to produce a chromatic scale with the distance between a consecutive space and line equal to one semitone.



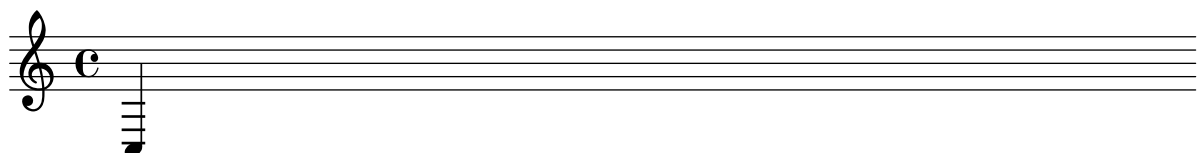
`'clef-oct.ly'`

Octavation signs may be added to clefs. These octavation signs may be placed below or above (meaning an octave higher or lower), and can take any value, including 15 for two octaves.



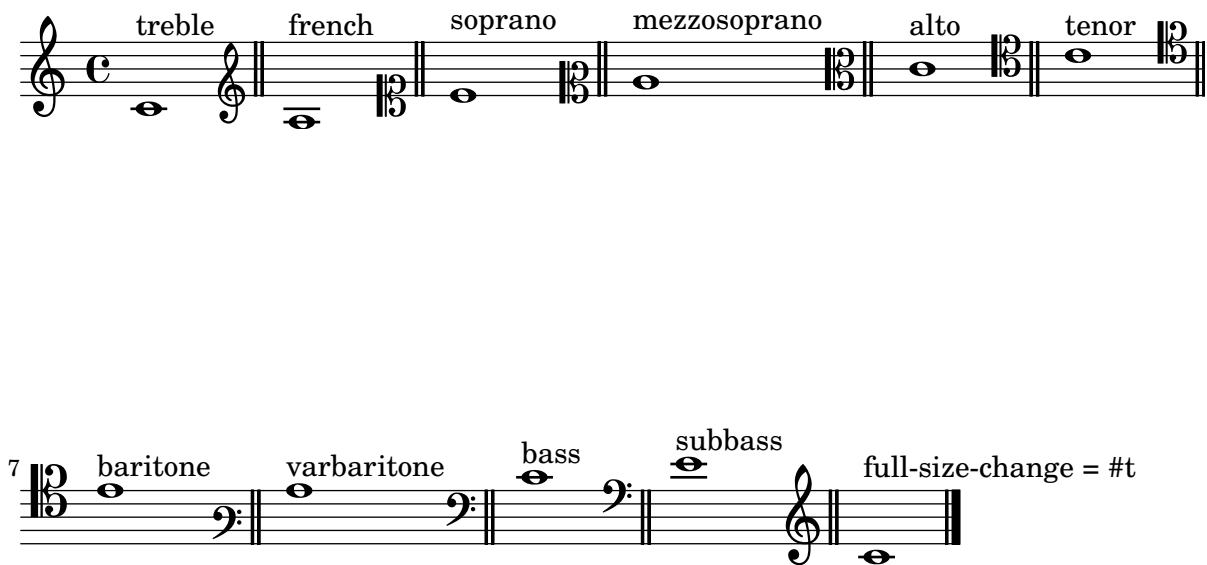
`'clef-warn.ly'`

Unknown clef name warning displays available clefs



`'clefs.ly'`

Clefs with `full-size-change` should be typeset in full size. For octaviated clefs, the “8” should appear closely above or below the clef respectively.



`'clip-systems.ly'`

Clipping snippets from a finished score

Notes:

- If system starts and ends are included, they include extents of the System grob, eg. 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.

This file needs to be run separately with `-dclip-systems`; the collated-files.html of the regression test does not adequately show the results.

The result will be files named `'base-from-start-to-end[-count].eps'`.



`'cluster-break.ly'`

Clusters behave well across line breaks.



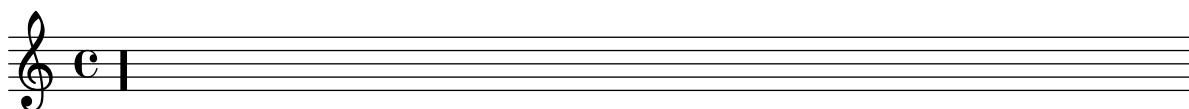
`'cluster-cross-staff.ly'`

Clusters can be written across staves.



`'cluster-single-note.ly'`

don't crash on single chord clusters.



`'cluster-style.ly'`

Clusters behave well across line breaks.



`'cluster.ly'`

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



`'collision-2.ly'`

Single head notes may collide.



`'collision-alignment.ly'`

Notes in different staves should be aligned to the left-most note, in case of collisions.



`'collision-dots-invert.ly'`

When notes are colliding, the resolution depends on the dots: notes with dots should go to the right, if there could be confusion to which notes the dots belong.



`'collision-dots-move.ly'`

If collision resolution finds dotted note head must remain on left hand side, move dots to the right.



`'collision-dots.ly'`

Collision resolution tries to put notes with dots on the right side.



`'collision-head-chords.ly'`

Note heads in collisions should be merged if they have the same positions in the extreme note heads.



`'collision-heads.ly'`

Open and black note heads are not merged by default.



`'collision-manual.ly'`

Collision resolution may be forced manually with `force-hshift`.



`'collision-merge-differently-dotted.ly'`

If `NoteCollision` has `merge-differently-dotted = ##t` note heads that have differing dot counts may be merged anyway. Dots should not disappear when merging similar note heads.



`'collision-merge-differently-headed.ly'`

If `merge-differently-headed` is enabled, then open note heads may be merged with black note heads, but only if the black note heads are from 8th or shorter notes.



`'collision-merge-dots.ly'`

When merging heads, the dots are merged too.



`'collision-mesh.ly'`

Oppositely stemmed chords, meshing into each other, are resolved.



`'collision-whole.ly'`

Mixed collisions with whole notes require asymmetric shifts.



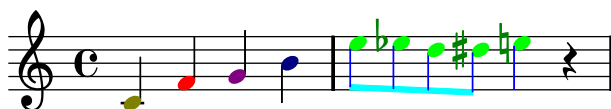
`'collisions.ly'`

In addition to normal collision rules, there is support for polyphony, where the collision are avoided by shifting middle voices horizontally.



`'color.ly'`

Each grob can have a color assigned to it. Use the `\override` and `\revert` expressions to set the `color` property.



`'completion-heads-multiple-ties.ly'`

The `Completion_heads_engraver` correctly handles notes that need to be split into more than 2 parts.



`'completion-heads-polyphony.ly'`

Completion heads are broken across bar lines. This was intended as a debugging tool, but it can be used to ease music entry. Completion heads are not fooled by polyphony with a different rhythm.



`'completion-heads.ly'`

If the `Note_heads_engraver` is replaced by the `Completion_heads_engraver`, notes that cross bar lines are split into tied notes.





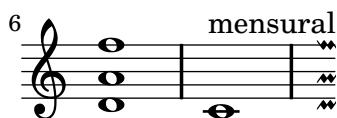
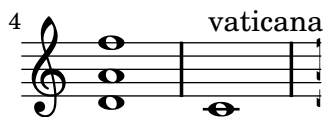
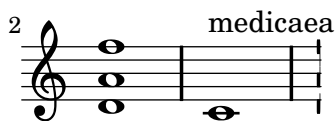
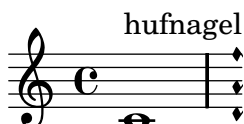
`'context-die-staff.ly'`

a staff should die if there is reference to it.



`'custos.ly'`

Custodes may be engraved in various styles.



`'display-lily-tests.ly'`

This is a test of the display-lily-music unit. Problems are reported on the stderr of this run.

`'dot-column-rest-collision.ly'`

Dot columns do not trigger beam slanting too early.



`'dot-flag-collision.ly'`

Dots move to the right when a collision with the (up)flag happens.



`'dot-up-voice-collision.ly'`

in collisions, the stems of outer voice are added to the dot support of the inner voices.

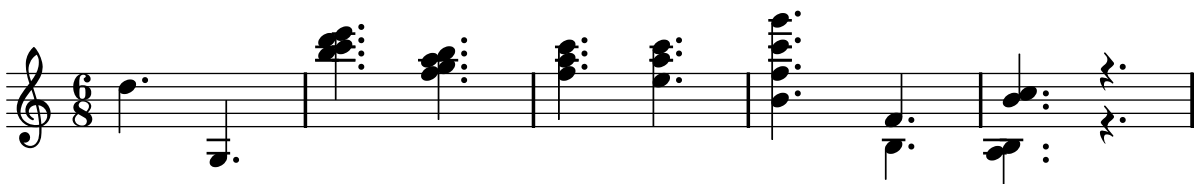


`'dots.ly'`

Noteheads can have dots, and rests too. Augmentation dots should never be printed on a staff line, but rather be shifted vertically. They should go up, but in case of multiple parts, the down stems have down shifted dots. In case of chords, all dots should be in a column. The dots follow the shift of rests when avoiding collisions.

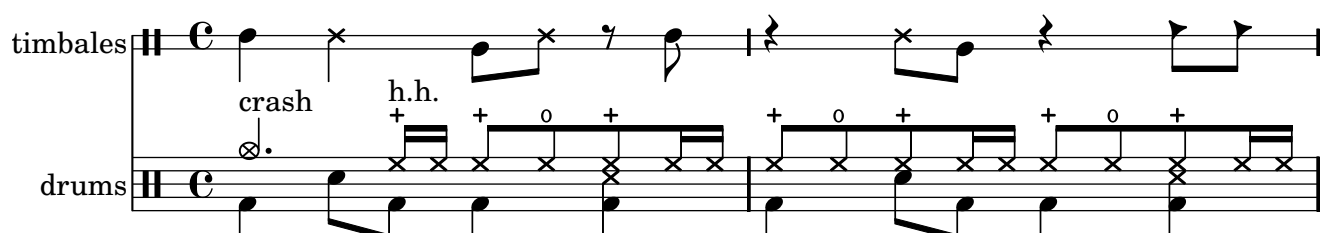
The priorities to print the dots are (ranked in importance):

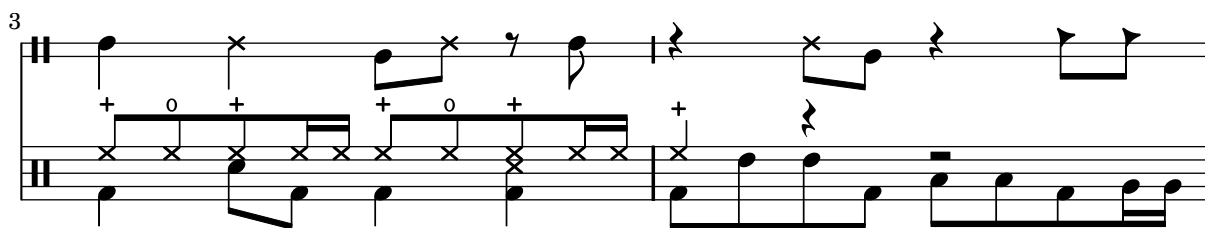
- keeping dots off staff lines,
- keeping dots close to their note heads,
- moving dots in the direction specified by the voice,
- moving dots up.



`'drums.ly'`

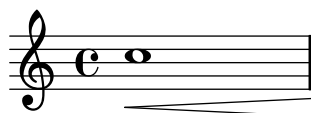
In drum notation, there is a special clef symbol, drums are placed to their own staff positions and have note heads according to the drum, an extra symbol may be attached to the drum, and the number of lines may be restricted.





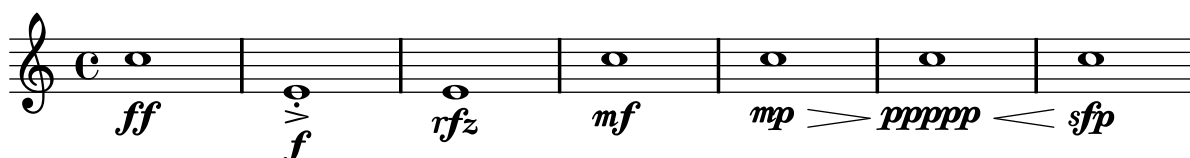
`'dynamics-broken-hairpin.ly'`

Broken crescendi should be open on one side.



`'dynamics-glyphs.ly'`

Dynamic letters are kerned, and their weight matches that of the hairpin signs. The dynamic scripts should be horizontally centered on the note head. Scripts that should appear closer to the note head (staccato, accent) are reckoned with.



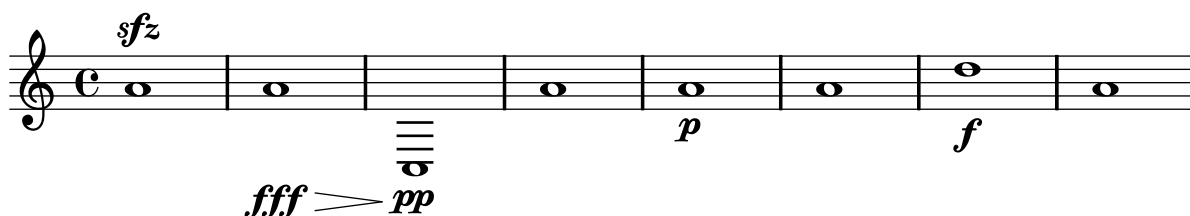
`'dynamics-hairpin-length.ly'`

Hairpins extend to the extremes of the bound if there is no adjacent hairpin or dynamic-text. If there is, the hairpin extends to the center of the column or the bound of the text respectively.



`'dynamics-line.ly'`

Dynamics appear below or above the staff. If multiple dynamics are linked with (de)crescendi, they should be on the same line. Isolated dynamics may be forced up or down.



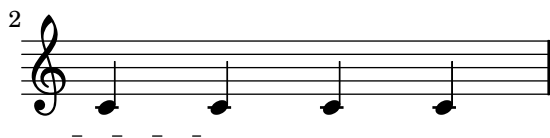
`'dynamics-text-spanner-abs-dynamic.ly'`

left attach dir for text *crescendi* starting on an absolute dynamic is changed, so *cresc.* and the absolute dynamic don't overstrike.



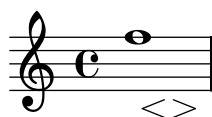
`'dynamics-text-spanner-padding.ly'`

The 2nd half of the *cresc.* stays at a reasonable distance from the notes.



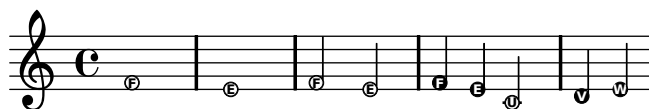
`'dynamics-unbound-hairpin.ly'`

Crescendi may start off-notes, however, they should not collapse into flat lines.



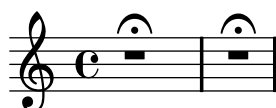
`'easy-notation.ly'`

Easy-notation (or Ez-notation) prints names in note heads. You also get ledger lines, of course.



`'fermata-rest-position.ly'`

Fermatas over multimeasure rests are positioned as over normal rests.



`'figured-bass-alteration.ly'`

Bass figures can carry alterations.



`'figured-bass-continuation-center.ly'`

Pairs of congruent figured bass extender lines are vertically centered if `figuredBassCenterContinuations` is set to true.



`'figured-bass-continuation-forbid.ly'`

By adorning a bass figure with `\!`, an extender may be forbidden.



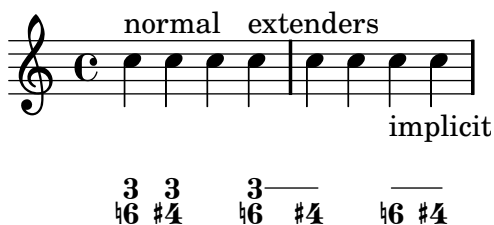
`'figured-bass-continuation.ly'`

Figured bass extender lines run between repeated bass figures. They are switched on with `useBassFigureExtenders`



`'figured-bass-implicit.ly'`

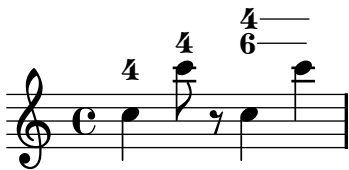
Implicit bass figures are not printed, but they do get extenders.



'figured-bass-staff.ly'

Figured bass can also be added to Staff context directly. In that case, the figures must be entered with `\figuremode` and be directed to an existing **Staff** context.

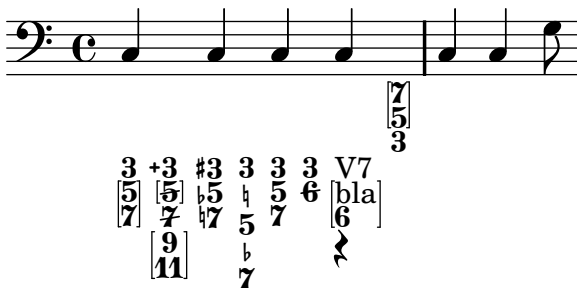
Since these engravers are on **Staff** level, properties controlling figured bass should be set in **Staff** context.



'figured-bass.ly'

Figured bass is created by the FiguredBass context which responds to figured bass events and rest events. You must enter these using the special `\figuremode { }` mode, which allows you to type numbers, like `<4 6+>` and add slashes and pluses.

You can also enter markup strings. The vertical alignment may also be tuned.



```
'fill-line-test.ly'
```

The fill-line markup command should align texts in columns. For example, the characters in the center should form one column.

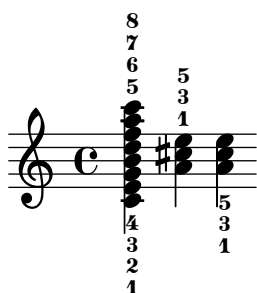
`‘finger-chords-accidental.ly’`

Scripts left of a chord avoid accidentals.



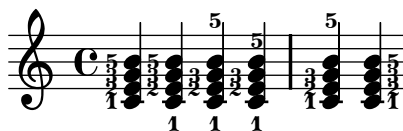
`‘finger-chords-order.ly’`

Ordering of the fingerings depends on vertical ordering of the notes, and is independent of up/down direction.



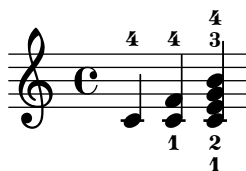
`‘finger-chords.ly’`

It is possible to associate fingerings uniquely with notes. This makes it possible to add horizontal fingerings to notes.



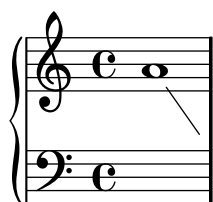
`‘fingering.ly’`

Automatic fingering tries to put fingering instructions next to noteheads.



`‘follow-voice-break.ly’`

The line-spanners connects to the Y position of the note on the next line. When put across line breaks, only the part before the line break is printed.

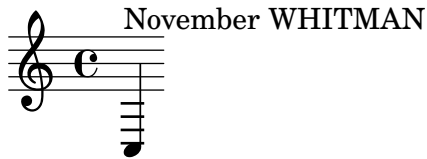


2



`'font-bogus-ligature.ly'`

TM and No should not be changed into trademark/number symbols. This may happen with incorrect font versions.



`'font-family-override.ly'`

The default font families for text can be overridden with `make-pango-font-tree`

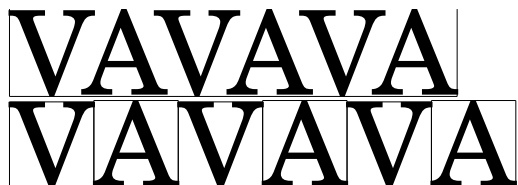


`'font-kern.ly'`

Text set in TrueType Fonts that contain kerning tables, are kerned.

With kerning:

Without kerning:



`'font-name.ly'`

Other fonts can be used by setting `font-name` for the appropriate object. The string should be a Pango font description without size specification.

Rest in LuxiMono



This text is in large Vera Bold

`'font-postscript.ly'`

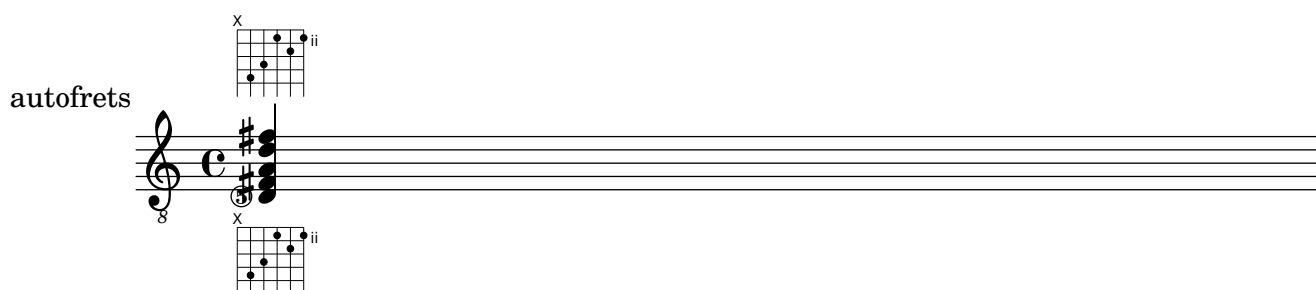
This file demonstrates how to load different (postscript) fonts. The file `'font.scm'` shows how to define the scheme-function `make-century-schoolbook-tree`.

This file should be run with the TeX and extra options should be passed to LaTeX and dvips to help it find the uncb font.

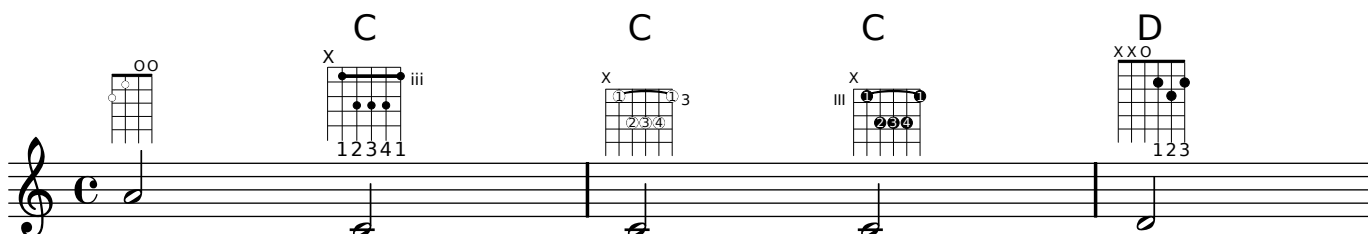


`'fret-boards.ly'`

Frets can be assigned automatically. The results will be best when one string number is indicated in advance



`'fret-diagrams.ly'`

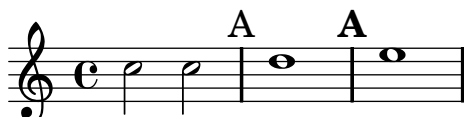


`'general-scheme-bindings.ly'`

This file tests various Scheme utility functions.

`'generic-output-property.ly'`

As a last resort, the placement of grobs can be adjusted manually, by setting the `extra-offset` of a grob.



`'glissando-broken.ly'`

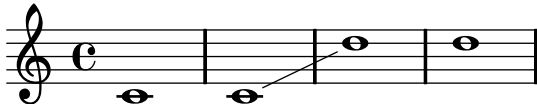
If broken, Glissandi anticipate on the pitch of the next line.





`'glissando-no-break.ly'`

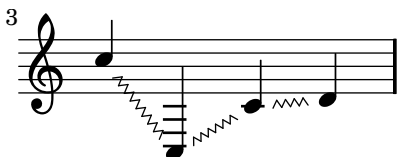
Glissandi are not broken. Here a `\break` is ineffective. Use `breakable` grob property to override.



`'glissando.ly'`

Between notes, there may be simple glissando lines. Here, the first two glissandi are not consecutive.

The engraver does no time-keeping, so it involves some trickery to get `<< { s8 s8 s4 } { c4 \gliss d4 } >>` working correctly.



`'grace-auto-beam.ly'`

The autobeamer is not confused by grace notes.



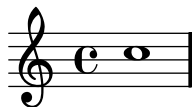
`'grace-bar-line.ly'`

Bar line should come before the grace note.



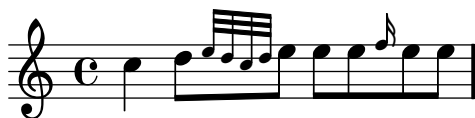
`'grace-bar-number.ly'`

Grace notes do tricky things with timing. If a measure starts with a grace note, the measure does not start at 0, but earlier. Nevertheless, lily should not get confused. For example, line breaks should be possible at grace notes, and the bar number should be printed correctly.



`'grace-beam.ly'`

Grace beams and normal beams may occur simultaneously. Unbeamed grace notes are not put into normal beams.



`'grace-direction-polyphony.ly'`

The `\voiceOne` setting is retained after finishing the grace section.



`'grace-end.ly'`

Grace notes after the last note do not confuse the timing code.



`'grace-nest.ly'`

Grace code should not be confused by nested sequential musics, containing grace notes; practically speaking, this means that the end-bar and measure bar coincide in this example.



`‘grace-nest1.ly’`

Grace code should not be confused by nested sequential musics, containing grace notes; practically speaking, this means that the end-bar and measure bar coincide in this example.



`‘grace-nest2.ly’`

Grace code should not be confused by nested sequential musics, containing grace notes; practically speaking, this means that the end-bar and measure bar coincide in this example.



`‘grace-nest3.ly’`

In nested syntax, graces are still properly handled.



`‘grace-nest4.ly’`

Also in the nested syntax here, grace notes appear rightly.



`‘grace-nest5.ly’`

Graces notes may have the same duration as the main note.



`‘grace-part-combine.ly’`

Grace notes may be put in a `partcombiner`.



`'grace-partial.ly'`

A `\partial` may be combined with a `\grace`.



`'grace-staff-length.ly'`

Stripped version of `trip.ly`. Staves should be of correct length.



`'grace-start.ly'`

Pieces may begin with grace notes.



`'grace-stem-length.ly'`

Stem lengths for grace notes should be shorter than normal notes, if possible. They should never be longer, even if that would lead to beam quanting program.



`'grace-stems.ly'`

Here `startGraceMusic` should set `no-stem-extend` to true; the two grace beams should be the same here.



`'grace-sync.ly'`

Grace notes in different voices/staves are synchronized.



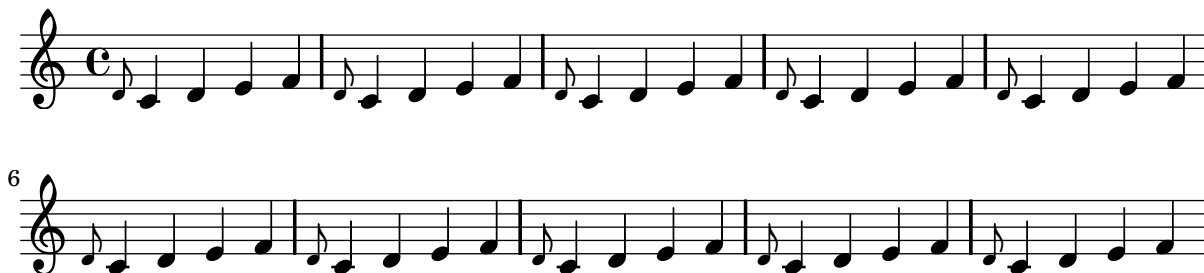
`'grace-types.ly'`

There are three different kinds of grace types: the base grace switches to smaller type, the appoggiatura inserts also a slur, and the acciaccatura inserts a slur and slashes the stem.



`'grace-unfold-repeat.ly'`

When grace notes are entered with unfolded repeats, line breaks take place before grace notes.



`'grace-volta-repeat-2.ly'`

A volta repeat may begin with a grace. Consecutive ending and starting repeat bars are merged into one `:||:.`



`'grace-volta-repeat.ly'`

Repeated music can start with grace notes. Bar checks preceding the grace notes do not cause synchronization effects.



'grace.ly'

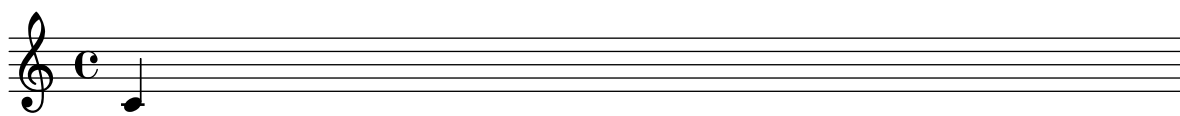
You can have beams, notes, chords, stems etc. within a `\grace` section. If there are tuplets, the grace notes will not be under the brace.

Main note scripts do not end up on the grace note.



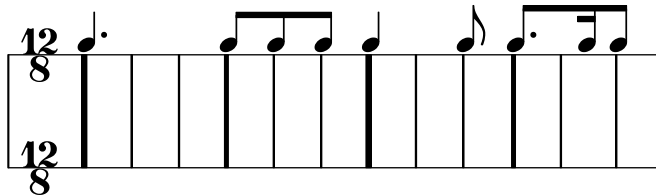
`'graphviz.ly'`

The graphviz feature draws dependency graphs for grob properties.



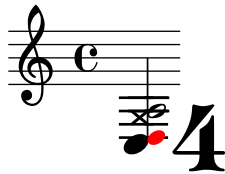
'grid-lines.ly'

With grid lines, vertical lines can be drawn between staves synchronized with the notes.



'grob-tweak.ly'

With the `\tweak` function, individual grobs that are directly caused by events may be tuned directly.



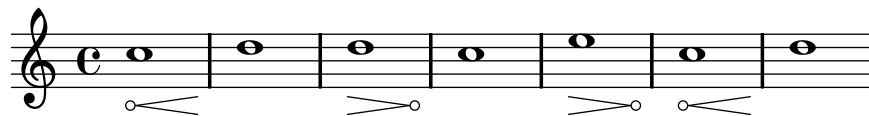
`'hairpin-barline-break.ly'`

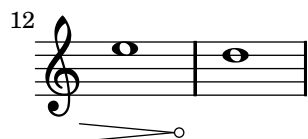
If a hairpin ends on the first note of a new stave, we don't print that ending. But on the previous line, this hairpin should not be left open, and should end at the barline.



'hairpin-circled.ly'

Hairpins can have circled tips. A decrescendo del niente followed by a crescendo al niente should only print one circle.





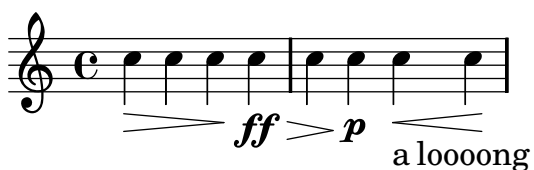
`'hairpin-dashed.ly'`

Hairpin crescendi may be dashed.



`'hairpin-ending.ly'`

Hairpin dynamics start under notes if there are no text-dynamics. If there are text dynamics, the hairpin does not run into them.



`'hairpin-to-barline-mark.ly'`

hairpinToBarline is not confused by very long marks.



`'hairpin-to-barline.ly'`

By setting `hairpinToBarline`, hairpins will stop at the barline preceding the ending note.



`'hairpin-to-rest.ly'`

Hairpins end at the left edge of a rest.



`'hara-kiri-pianostaff.ly'`

Hara-kiri staves kill themselves if they are empty. This example really contains three staves, but as they progress, empty ones are removed: this example has three staves, but some of them disappear: note how the 2nd line only has the bar number 2. (That the bar number is printed might be considered a bug, however, the scenario of all staves disappearing does not happen in practice.)

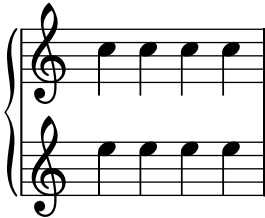
Any staff brackets and braces are removed, both in the single staff and no staff case.

This example was done with a pianostaff, which has fixed distance alignment; this should not confuse the mechanism.



2

3



4



`'hara-kiri-stanza-number.ly'`

stanza numbers remain, even on otherwise empty lyrics lines.

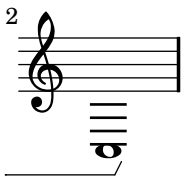
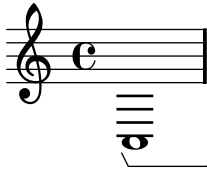


2



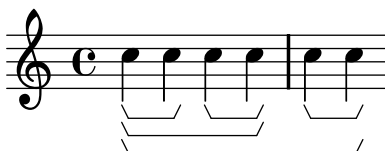
`‘horizontal-bracket-break.ly’`

Horizontal brackets connect over line breaks.



`‘horizontal-bracket.ly’`

Note grouping events are used to indicate where analysis brackets start and end.



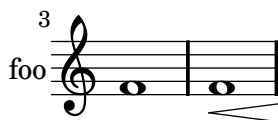
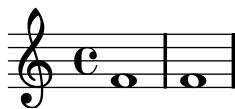
`‘identifiers.ly’`

test identifiers.



`'instrument-name-dynamic.ly'`

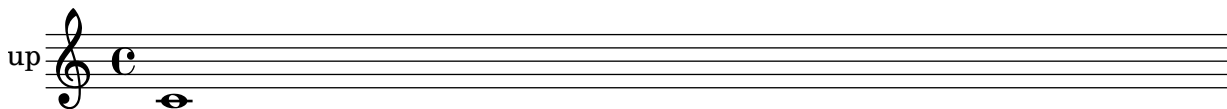
Instrument names (aligned on axis group spanners) ignore dynamic and pedal line spanners.



`'instrument-name-hara-kiri.ly'`

`PianoStaff.instrument` and `PianoStaff.instr` are removed when the staves are killed off.

In this example, the 2nd staff (marked by the barnumber 2) disappears as does the instrument name.



2

`'instrument-name-markup.ly'`

Instrument names are set with `Staff.instrument` and `Staff.instr`. You can enter markup texts to create more funky names, including alterations.



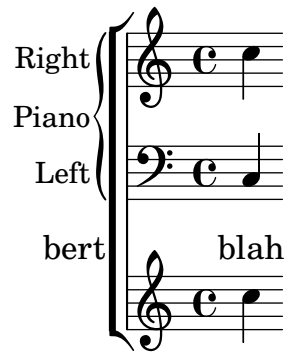
`'instrument-name-partial.ly'`

Instrument names are also printed on partial starting measures.



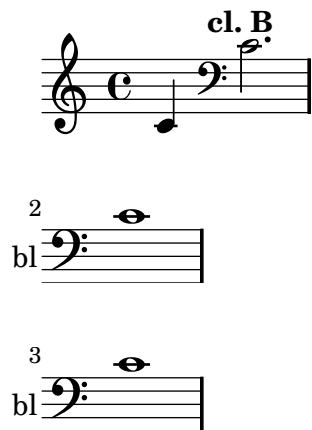
`'instrument-name.ly'`

Staff margins are also markings attached to barlines. They should be left of the staff, and be centered vertically with respect to the staff. They may be on normal staves, but also on compound staves, like the PianoStaff.



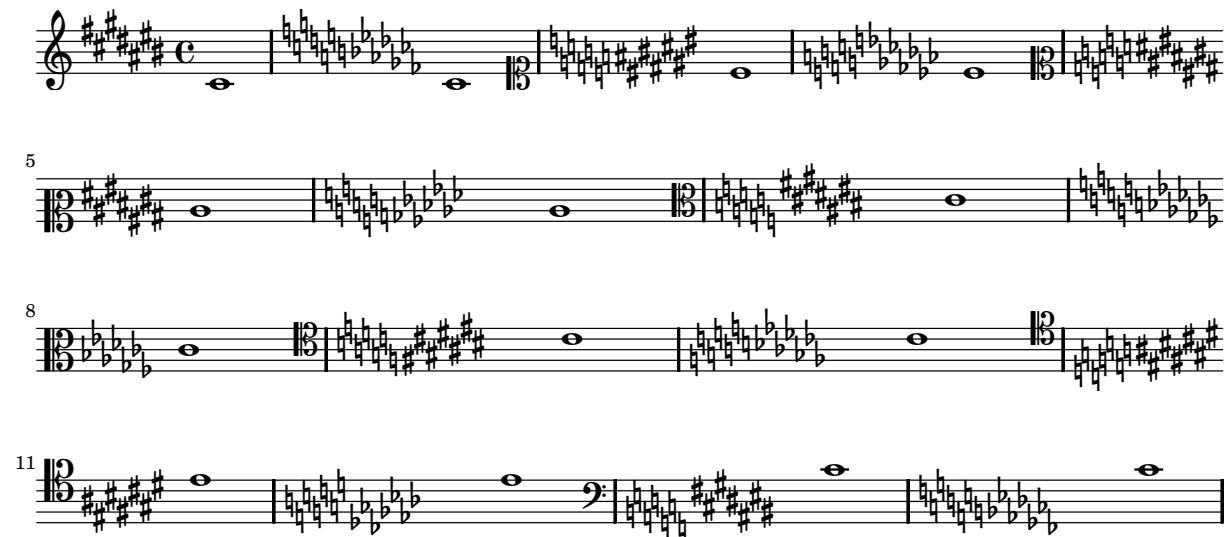
`'instrument-switch.ly'`

The `switchInstrument` music function modifies properties for an in staff instrument switch.



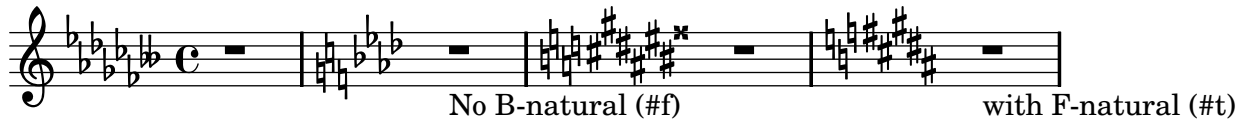
`'key-clefs.ly'`

Each clef have own accidental placing rules.



`'key-signature-cancellation-extra-natural.ly'`

If `extraNatural` is set then keys that are not altered farther away (eg from sharp to double sharp) are cancelled. Otherwise only keys that does not occur in the new key signature are cancelled.

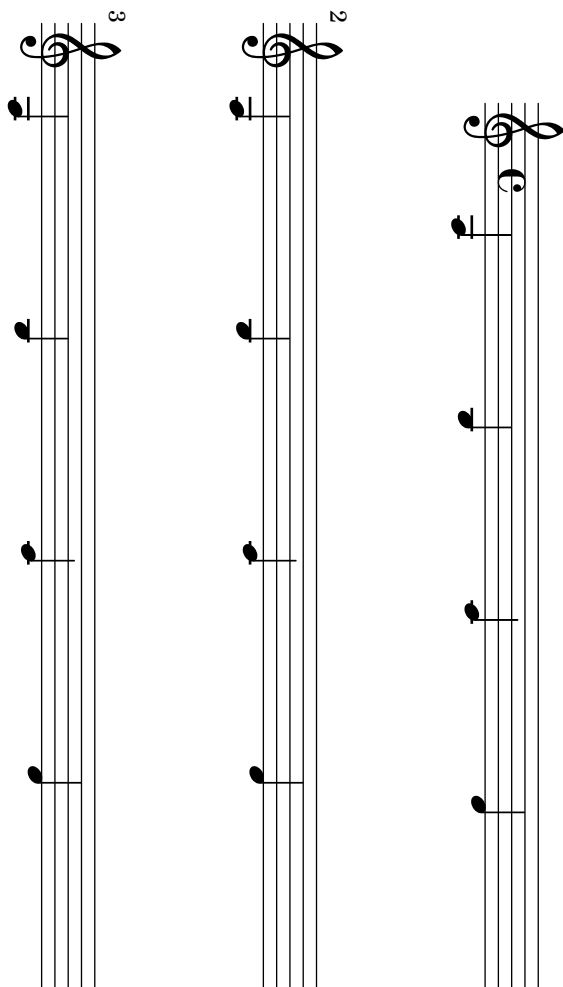


`'laissez-vibrer-ties.ly'`

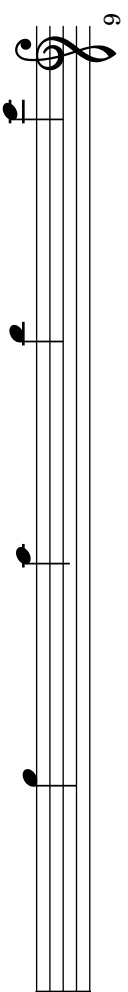
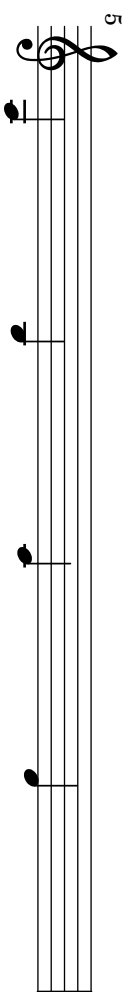
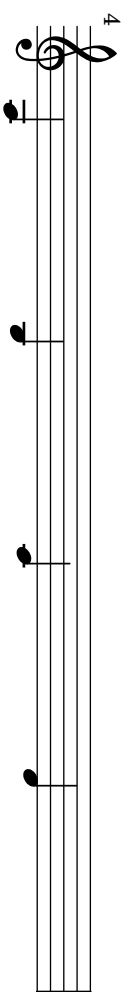
l.v. ties should avoid dots and staff lines, similar to normal ties. They have fixed size. Their formatting can be tuned with `tie-configuration`.

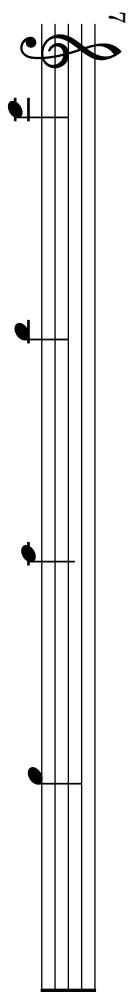
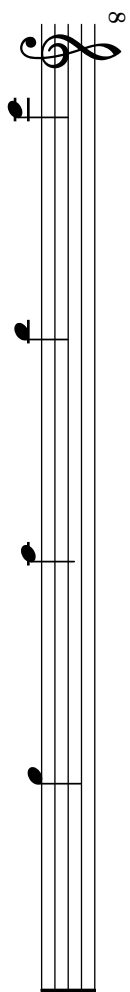


`'landscape.ly'`

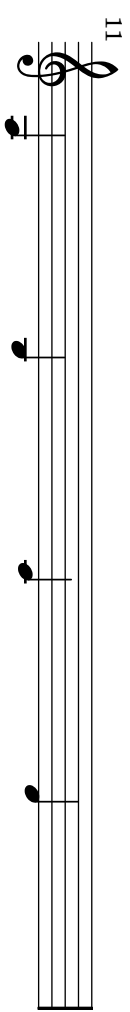
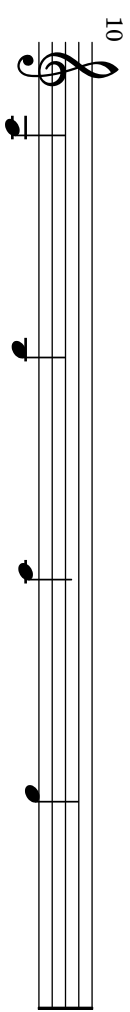
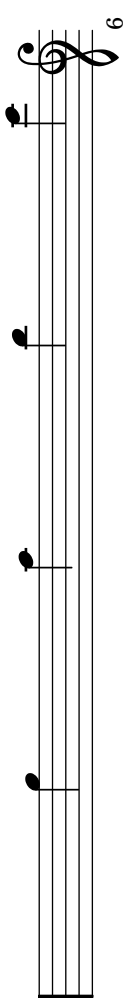


2

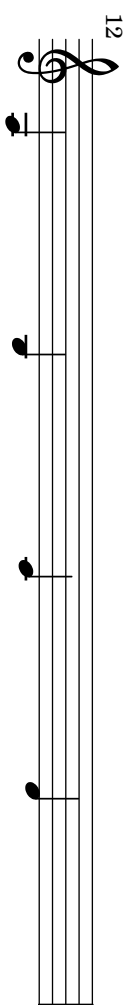




4

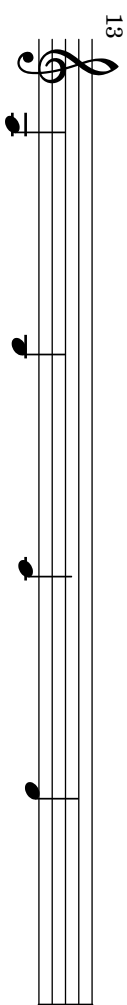


12



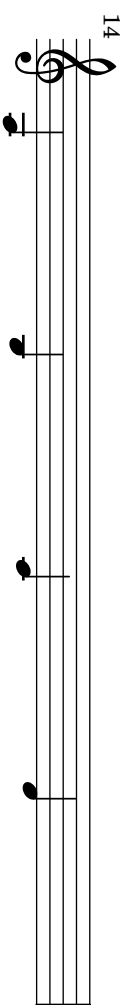
A musical staff with a treble clef and a key signature of one flat (B-flat). The staff contains four quarter notes: B-flat4, A4, G4, and F4.

13

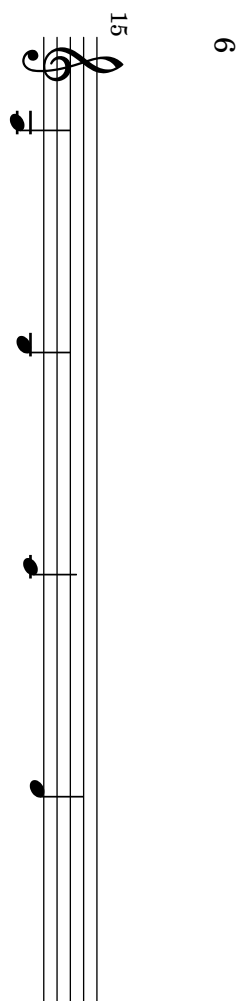


A musical staff with a treble clef and a key signature of one flat (B-flat). The staff contains four quarter notes: E4, D4, C4, and B3.

14



A musical staff with a treble clef and a key signature of one flat (B-flat). The staff contains four quarter notes: A3, G3, F3, and E3.



`'ledger-line-minimum.ly'`

When ledgered notes are very close, for example, in grace notes, they are kept at a minimum distance to prevent the ledgers from disappearing.



`'ledger-line-shorten.ly'`

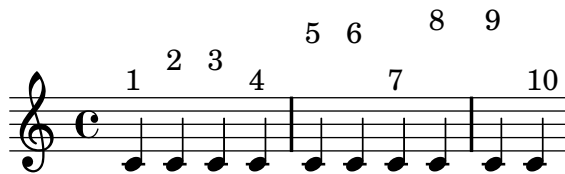
Ledger lines are shortened when they are very close. This ensures that ledger lines stay separate.



`'lily-in-scheme.ly'`

LilyPond syntax can be used inside scheme to build music expressions, with the `#{ ... #}` syntax. Scheme forms can be introduced inside these blocks by escaping them with a `$`, both in a LilyPond context or in a Scheme context.

In this example, the `\withpaddingA`, `\withpaddingB` and `\withpaddingC` music functions set different kinds of padding on the TextScript grob.



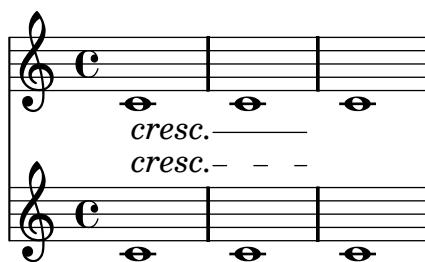
`'line-arrows.ly'`

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



`'line-dashed-period.ly'`

The period of a dashed line is adjusted such that it starts and ends on a full dash.



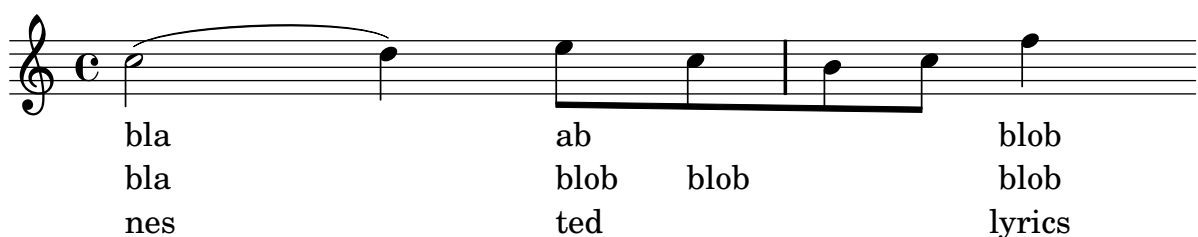
`'line-style.ly'`

Cover all line styles available



`'lyric-combine-new.ly'`

With the `\lyricsto` mechanism, individual lyric lines can be associated with one melody line. For each lyric line, can be tuned whether to follow melismata or not.



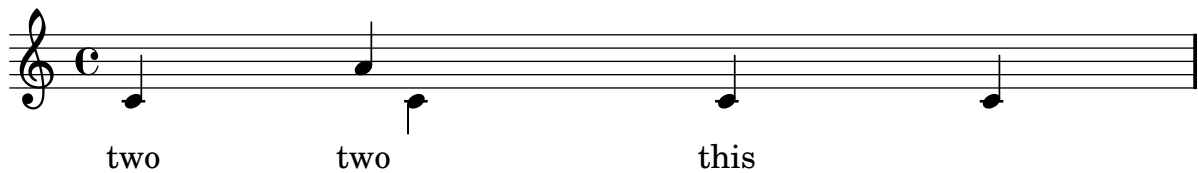
`'lyric-combine-polyphonic.ly'`

Polyphonic rhythms and rests do not disturb `\lyricsto`.



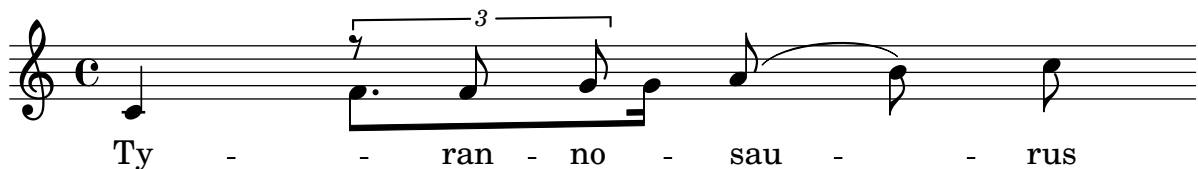
`'lyric-combine-switch-voice-2.ly'`

switching voices halfway a `lyricsto` is possible.



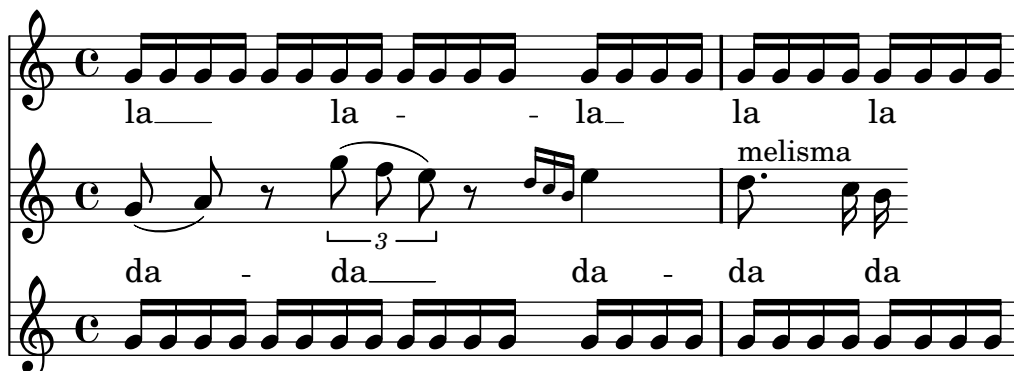
`'lyric-combine-switch-voice.ly'`

Switching melody to a different voice works even the switch occurs together with context instantiation.



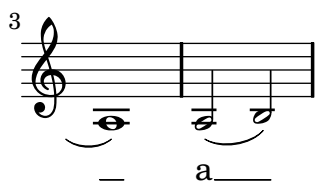
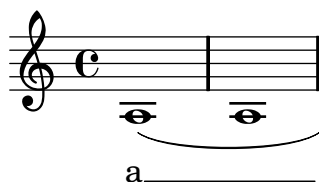
`'lyric-combine.ly'`

Lyrics can be set to a melody automatically. Excess lyrics will be discarded. Lyrics will not be set over rests. You can have melismata either by setting a property `melismaBusy`, or by setting `automaticMelismas` (which will set melismas during slurs and ties). If you want a different order than first Music, then Lyrics, you must precook a chord of staves/lyrics and label those. Of course, the lyrics ignores any other rhythms in the piece.



`'lyric-extender-broken.ly'`

Lyric extenders run to the end of the line if it continues the next line. Otherwise, it should run to the last note of the melisma.



`'lyric-extender-no-heads.ly'`

Extender engraver also notices the lack of note heads. Here the extender ends on the 2nd quarter note, despite the grace note without a lyric attached.



`'lyric-extender-rest.ly'`

If `extendersOverRests` is set, an extender is not terminated upon encountering a rest.



`'lyric-extender-right-margin.ly'`

extenders will not protude into the right margin





`'lyric-extender.ly'`

A LyricExtender may span several notes. A LyricExtender does not extend past a rest, or past the next lyric syllable.



`'lyric-hyphen-break.ly'`

Hyphens are print at the beginning of the line only when they go past the first note.



`'lyric-hyphen-retain.ly'`

The minimum distance between lyrics are determined by the `minimum-distance` of LyricHyphen and LyricSpace.

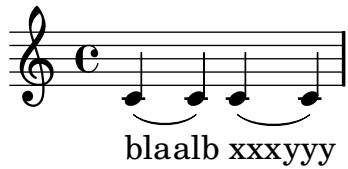
The ideal length of a hyphen is determined by its `length` property, but it may be shortened down to `minimum-length` in tight situations. If in this it still does not fit, the hyphen will be omitted.

Like all overrides within `\lyricsto` and `\addlyrics`, the effect of a setting is delayed is one syllable.



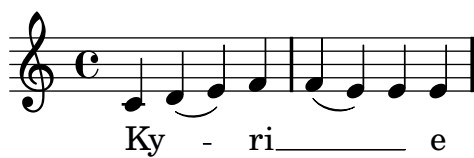
`'lyric-hyphen.ly'`

In lyrics, hyphens may be used.



`'lyric-melisma-manual.ly'`

Melisma's may be entered manually by substituting _ for lyrics on notes that are part of the melisma.



`'lyric-phrasing.ly'`

Normally, the lyric is centered on the note head. However, on melismata, the text is left aligned on the left-side of the note head.



`'lyric-tie.ly'`

Tildes in lyric syllables are converted to tie symbols.

wa o a

`'lyrics-after-grace.ly'`

Lyrics are ignored for aftergrace notes.



`'lyrics-bar.ly'`

Adding a `Bar_engraver` to the Lyrics context makes sure that lyrics do not collide with barlines.

‘markup-commands.ly’

test various markup commands.



foo **foo** LOWER **normal** normal Small-Caps SMALL-CAPS
 LOWER

justify:

This is a field containing text. Blah blah blah.

This is a field containing text. Blah blah blah.

This is a field containing text. Blah blah blah.

This is a field containing text. Blah blah blah.

This is a field containing text. Blah blah blah.

wordwrap:

This is a field containing text. Blah blah blah.

This is a field containing text. Blah blah blah.

This is a field containing text. Blah blah blah.

This is a field containing text. Blah blah blah.

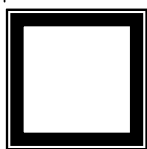
This is a field containing text. Blah blah blah.

draw-line:



‘markup-eps.ly’

The epsfile markup command reads an EPS file



‘markup-lines.ly’

Text that can spread over pages is entered with the \markuplines command.

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

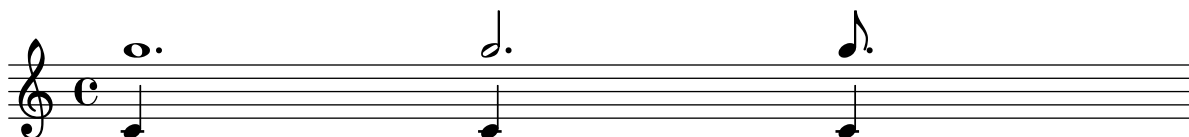
plus douces. Sa physionomie annonçait son

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.

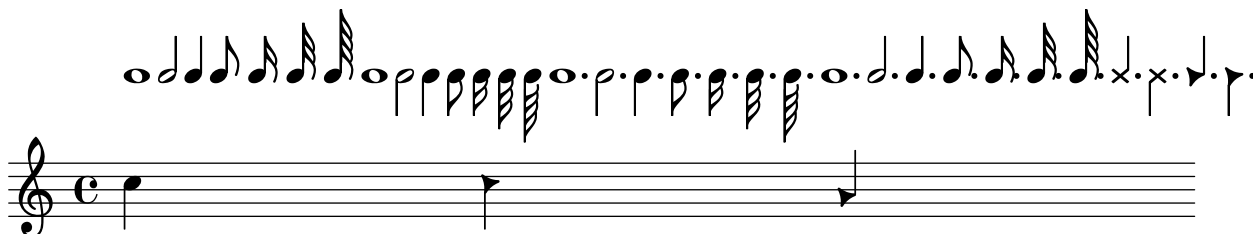
Monsieur le baron était un des plus puissants seigneurs de la Westphalie, car son château avait une porte et des fenêtres. Sa grande salle même était ornée d'une tapisserie. Tous les chiens de ses basses-cours composaient une meute dans le besoin ; ses palefreniers étaient

contes.

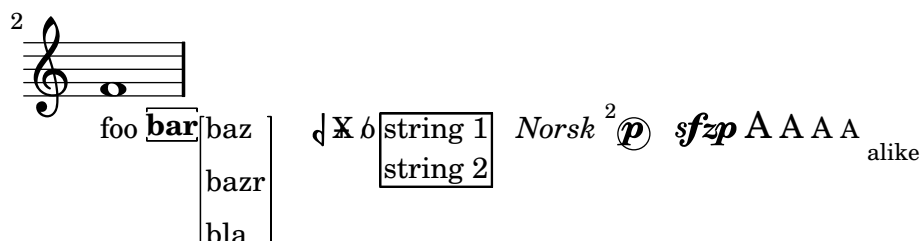
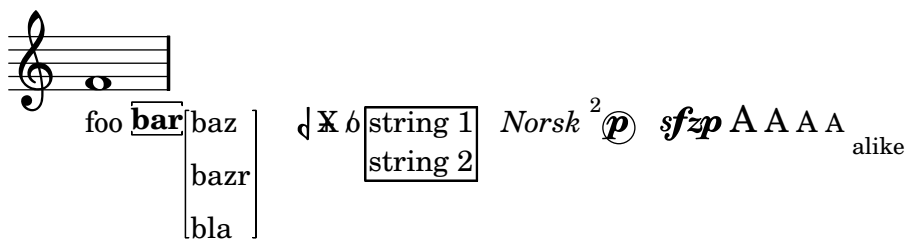
A dotted whole note displayed via the `\note` command must separate the note head and the dot. The dot avoids the upflag.



The note markup function may be used to make metronome markings. It works for a variety of flag, dot and duration settings.



There is a Scheme macro `markup` to produce markup texts using a similar syntax as `\markup`.


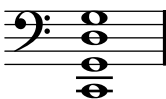


‘markup-score.ly’
Use \score block as markup command.

Solo Cello Suites


Suite IV

Originalstimmung:

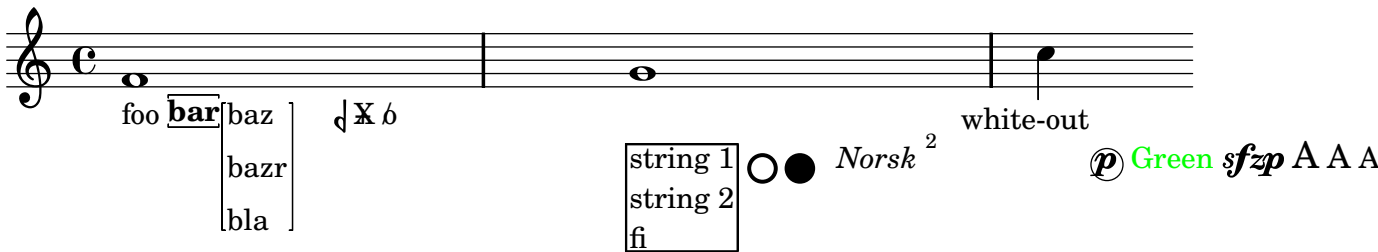


‘markup-stack.ly’
Markup scripts may be stacked.

a 1
2
3



‘markup-syntax.ly’
Demo of markup texts, using LilyPond syntax.



‘markup-user.ly’
Own markup commands may be defined by using the `define-markup-command` scheme macro.



HELLO WORLD

‘markup-word-wrap.ly’
The markup commands `\wordwrap` and `\justify` produce simple paragraph text.

this is normal text This is a test of the wordwrapping function. 1 This is a continuing
test of the wordwrapping function. 2 This is a test of the
wordwrapping function. 3 This is a test of the
wordwrapping function. 4 1a111 11111 **22222** 2222

this is normal text This is a test of the wordwrapping continuing function, but with justification. 1 This is a test of the wordwrapping function, but with justification. 2 This is a test of $\frac{a}{b}$ the wordwrapping function, but with justification. 3 This is a test of the wordwrapping function, but with justification. bla bla

Om mani padme hum Om mani	Om mani padme hum Om mani padme
padme hum Om mani padme hum Om	hum Om mani padme hum Om mani
mani padme hum Om mani padme	padme hum Om mani padme hum Om
hum Om mani padme hum Om mani	mani padme hum Om mani padme hum
padme hum Om mani padme hum.	Om mani padme hum.
Gate Gate paragate Gate Gate	Gate Gate paragate Gate Gate
paragate Gate Gate paragate Gate	paragate Gate Gate paragate Gate
Gate paragate Gate Gate paragate	Gate paragate Gate Gate paragate
Gate Gate paragate.	Gate Gate paragate.

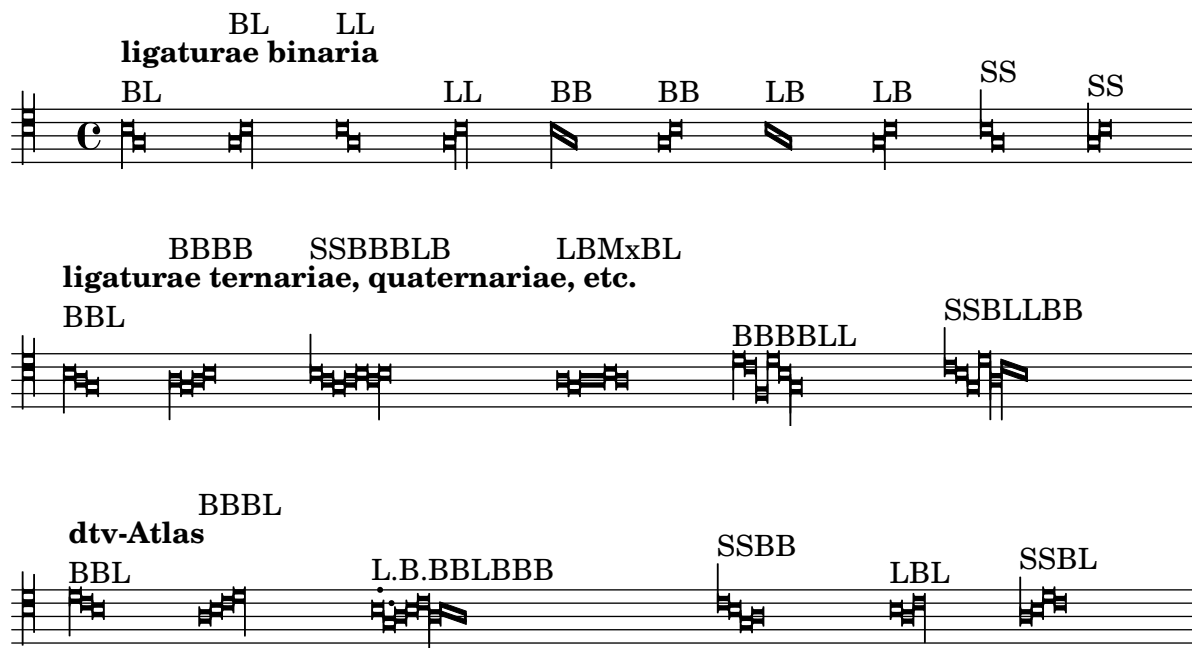
‘measure-grouping.ly’

The Measure_grouping_engraver adds triangles and brackets above beats when the beats of a time signature are grouped.



‘mensural-ligatures.ly’

Mensural ligatures show different shapes, depending on the rhythmical pattern and direction of the melody line.



LBBBB MxL BBB LBBBBB
Ockeghem: Missa De plus en plus
 MxMx BBBBL SSB LLLL

Ockeghem: Requiem
 SSBBBBBBL BBBBL

crazy ligatures
 BBBBB BB B.B.B.B.B.B.B.B. B.B.

BBB

‘mensural.ly’

There is limited support for mensural notation: note head shapes are available. Mensural stems are centered on the note heads, both for up and down stems.

9

‘metronome-marking.ly’

Here `\tempo` directives are printed as metronome markings.

The marking is left aligned with the time signature, if there is one.

$\text{♩} = 100$ $\text{♩}.. = 50$

‘midi-drums.ly’

`'midi-dynamics.ly'`

Midi also handles crescendo and decrescendo, either starting and ending from specified or unspecified sound level.

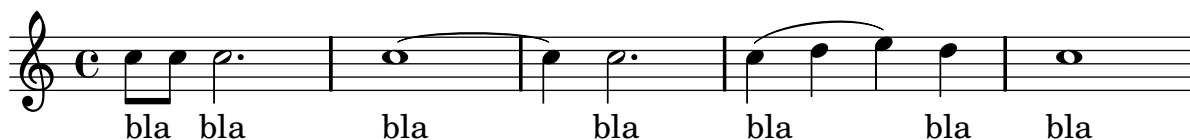


`'midi-grace.ly'`

Grace notes don't introduce syncing problems: the last note off will appear at tick 768 (2 * 384).

`'midi-lyric-barcheck.ly'`

Lyrics in MIDI are aligned to ties and beams: this examples causes no bar checks in MIDI.



`'midi-microtone-off.ly'`

Microtonal shifts should be corrected before the start of the next (possibly grace) note.

`'midi-microtone.ly'`

The pitch wheel is used for microtones.

`'midi-partial.ly'`

MIDI and partial measures work together.

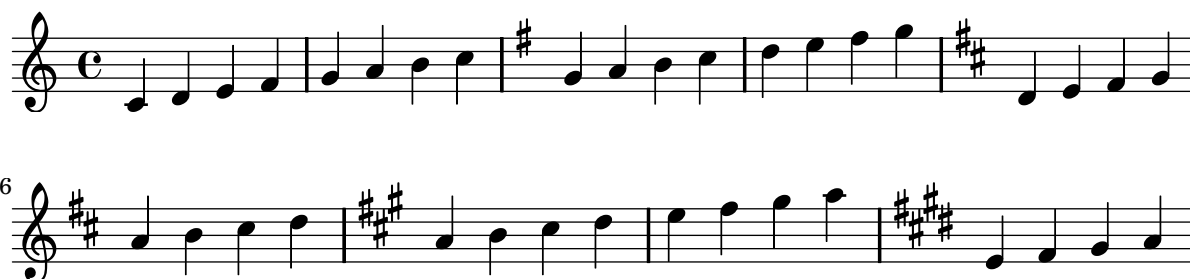
`'midi-pedal.ly'`

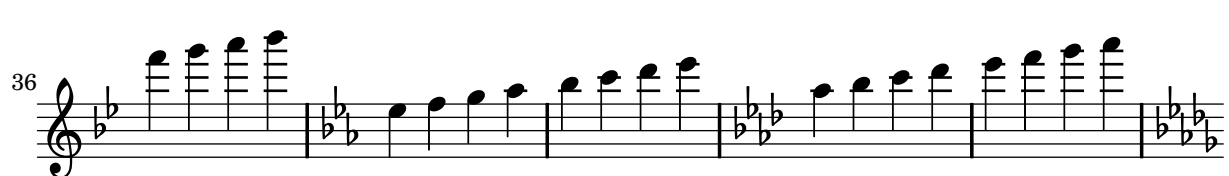
Pedals. Run `timidity -idv vv file.midi |grep Midi` to see midi events.



`'midi-scales.ly'`

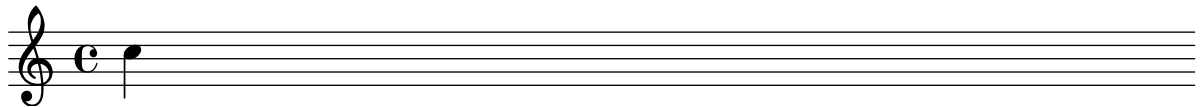
Converting LilyPond input to MIDI and then again back with `midi2ly.py` is a reversible procedure in some simple cases, which mean that the original `.ly` -file and the one converted back from the generated `.midi` -file do not differ. Here are produced some scales.





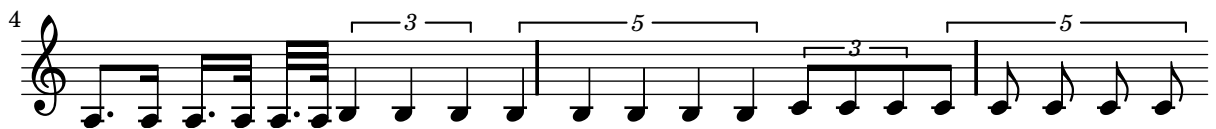
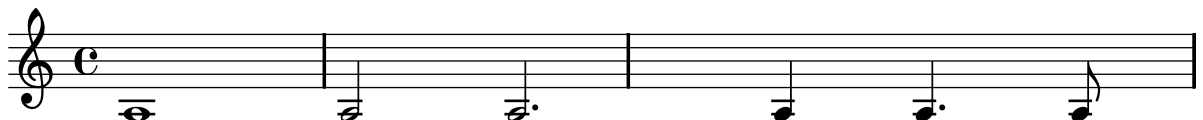


‘midi-transposition.ly’
should deliver f’ in MIDI



‘midi-tuplets.ly’
Midi2ly tuplet test.

```
python scripts/midi2ly.py --duration-quant=32 \  
  --allow-tuplet=4*2/3 \  
  --allow-tuplet=8*2/3 \  
  --allow-tuplet=4*3/5 \  
  --allow-tuplet=8*3/5 \  
tu.midi
```



‘midi-volume-equaliser.ly’

The full orchestra plays a notes, where groups stop one after another. Use this to tune equalizer settings.

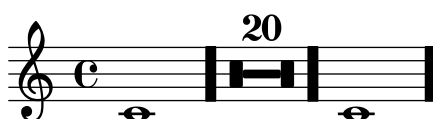
‘mm-rests2.ly’

If `Score.skipBars` is set, the signs for four, two, and one measure rest are combined to produce the graphical representation of rests for up to 10 bars. The number of bars will be written above the sign.



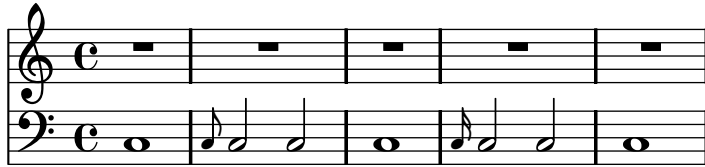
‘multi-measure-rest-center.ly’

The multimeasure rest is centered exactly between bar lines.



`'multi-measure-rest-grace.ly'`

Multi-measure rests are centered also in the case of grace notes.



`'multi-measure-rest-instr-name.ly'`

There are both long and short instrument names. Engraving instrument names should not be confused by the multimeasure rests.



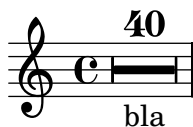
`'multi-measure-rest-multi-staff-center.ly'`

The centering of multi-measure rests is independent on prefatory matter in other staves.



`'multi-measure-rest-spacing.ly'`

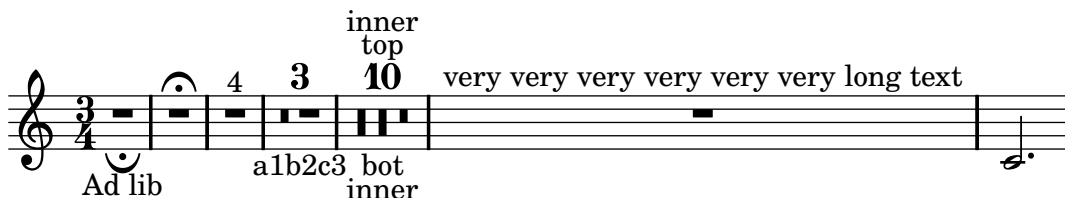
By setting texts starting with a multi-measure rest, an extra spacing column is created. This should not cause problems.



`'multi-measure-rest-text.ly'`

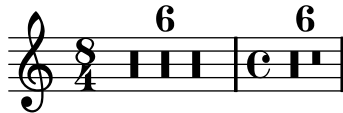
Texts may be added to the multi-measure rests.

By setting the appropriate `spacing-procedure`, we can make measures stretch to accomodate wide texts.



`'multi-measure-rest-usebreve.ly'`

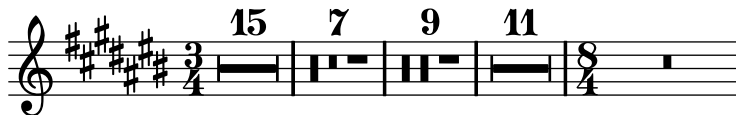
For longer measure lengths, the breve rest is used.



`'multi-measure-rest.ly'`

Multi-measure rests do not collide with barlines and clefs. They are not expanded when you set `Score.skipBars`. Although the multi-measure-rest is a `Spanner`, minimum distances are set to keep it colliding from barlines.

Rests over measures during longer than 2 wholes use breve rests. When more than 10 or more measures (tunable through `expand-limit`) are used then a different symbol is used.



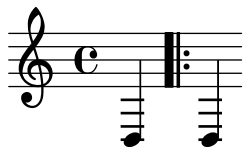
`'music-function-end-spanners.ly'`

the `endSpanners` music function inserts end span events at the end of a note.



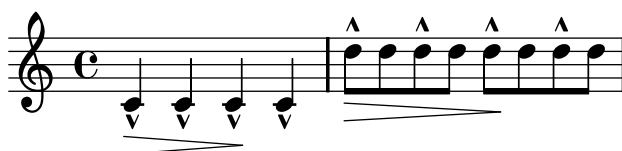
`'music-function.ly'`

Music function are generic music transformation functions, which can be used to extend music syntax seamlessly. Here we demonstrate a `\myBar` function, which works similar to `\bar`, but is implemented completely in Scheme.



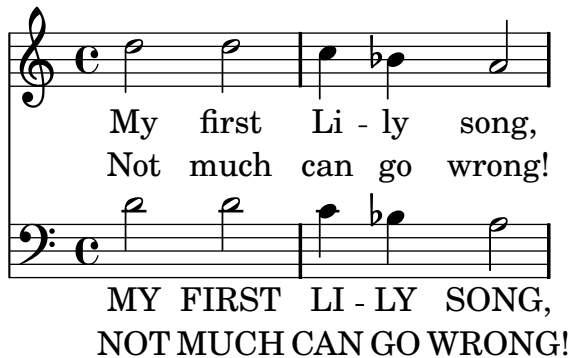
`'music-map.ly'`

With `music-map`, you can apply functions operating on a single piece of music to an entire music expression. In this example, the the function `notes-to-skip` changes a note to a skip. When applied to an entire music expression in the 1st measure, the scripts and dynamics are left over. These are put onto the 2nd measure.



`'newaddlyrics.ly'`

newlyrics, multiple stanzas, multiple lyric voices.



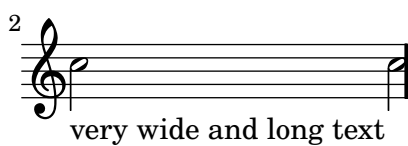
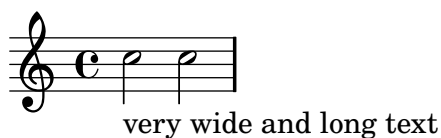
`'no-staff.ly'`

The printing of the staff lines may be suppressed by removing the corresponding engraver.



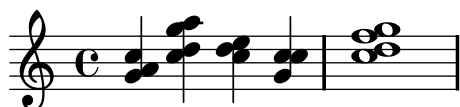
`'non-empty-text.ly'`

By default, text is set with empty horizontal dimensions. The boolean property `no-spacing-rods` in `TextScript` is used to control the horizontal size of text.



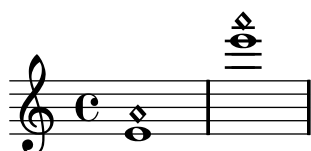
`'note-head-chord.ly'`

Note heads are flipped on the stem to prevent collisions. It also works for whole heads that have invisible stems.



`'note-head-harmonic-whole.ly'`

A harmonic note head must be centered if the base note is a whole note.



`'note-head-harmonic.ly'`

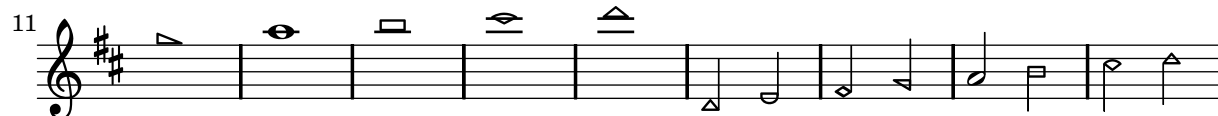
The handling of stems for harmonic notes must be completely identical to normal note heads.

Harmonic heads do not get dots. If `harmonicAccidentals` is unset, they also don't get accidentals.



`'note-head-solfa.ly'`

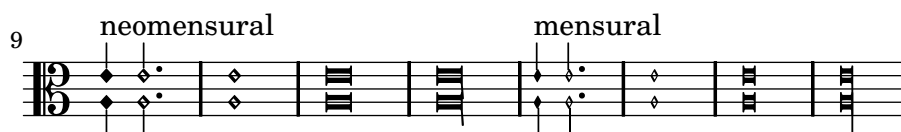
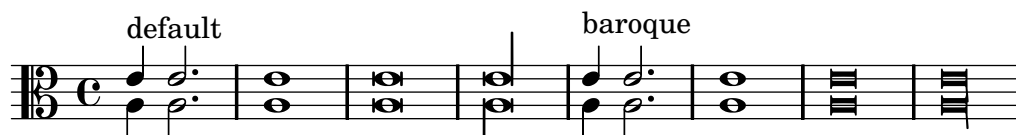
With `shapeNoteStyles`, the style of the note head is adjusted according to the step of the scale, as measured relative to the tonic property.

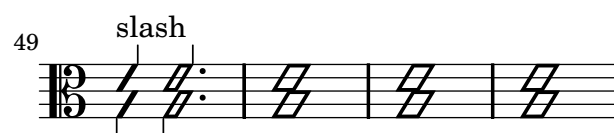
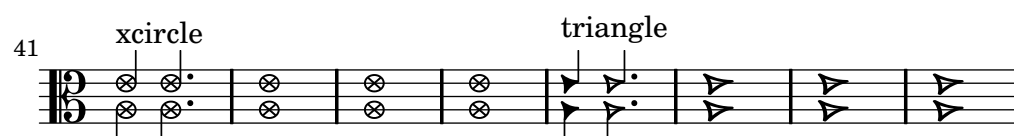
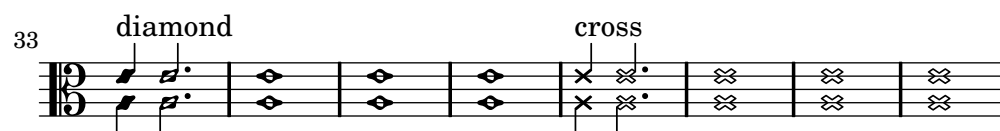
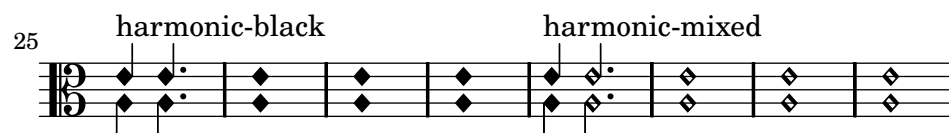


`'note-head-style.ly'`

Note head shapes may be set from several choices. The stem endings should be adjusted according to the note head. If you want different note head styles on one stem, you must create a special context.

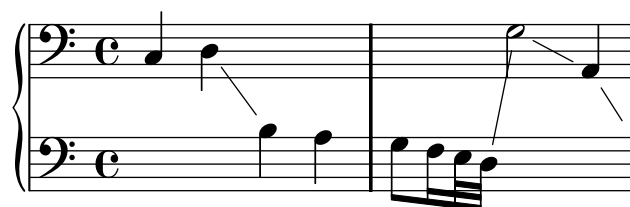
Harmonic notes have a different shape and different dimensions.



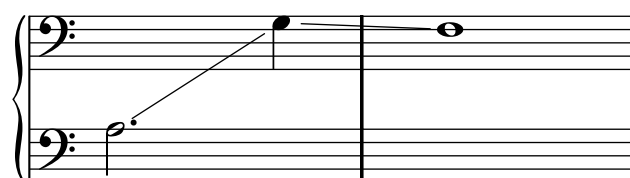


`'note-line.ly'`

Note head lines (e.g. glissando) run between centers of the note heads.



3



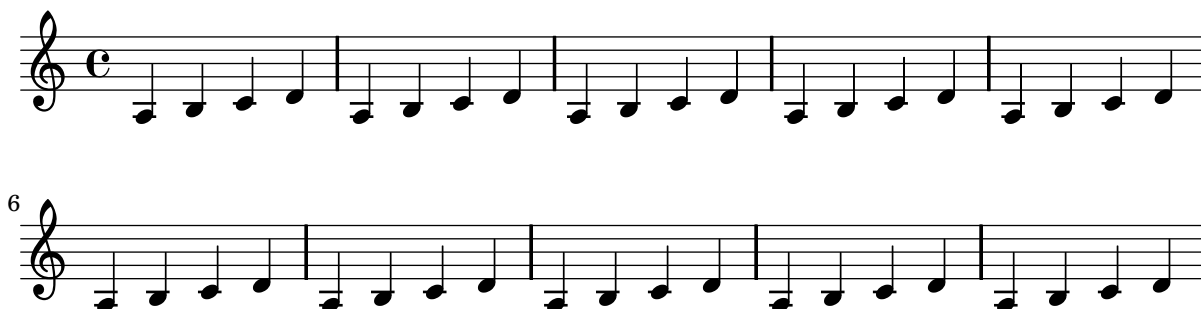
`'number-staff-lines.ly'`

The number of stafflines of a staff can be set. Ledger lines both on note heads and rests, as well as barlines, are adjusted accordingly.



`'optimal-page-breaking-hstretch.ly'`

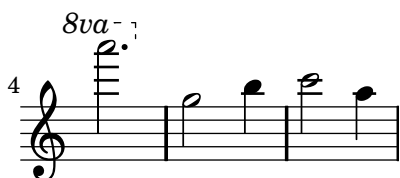
The optimal page breaker will stretch the systems horizontally so that the vertical spacing will be more acceptable. The `page-spacing-weight` parameter controls the relative importance of vertical/horizontal spacing. Because `ragged-last-bottom` is on, only the first page should be horizontally stretched.



`'option-help.ly'`

`'ottava-broken.ly'`

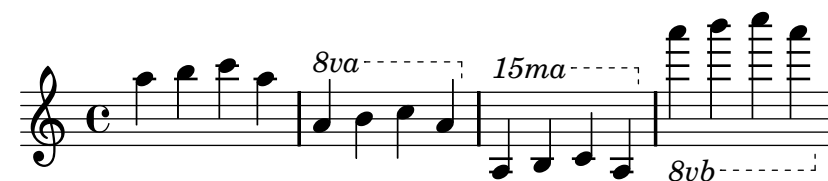
At line breaks, ottava brackets have no vertical line and their horizontal line does not stick out. The dashed line runs until the end of the line (regardless of prefatory matter).

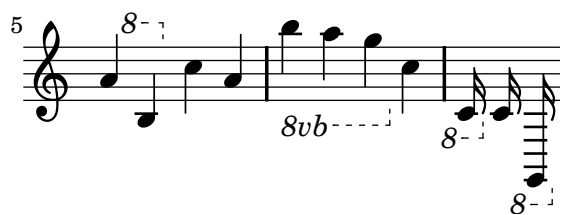


`'ottava.ly'`

Ottava brackets are supported, through the use of the scheme function `set-octavation`.

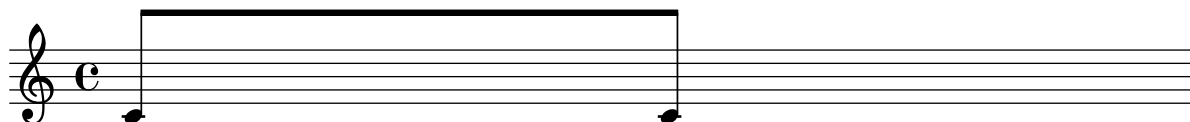
The spanner should go below a staff for `8va` bassa, and the ottavation string can be tuned with `Staff.ottavation`.





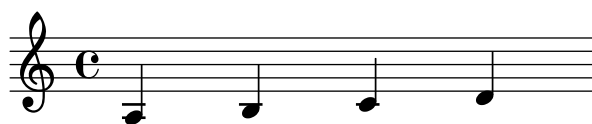
`'override-nest.ly'`

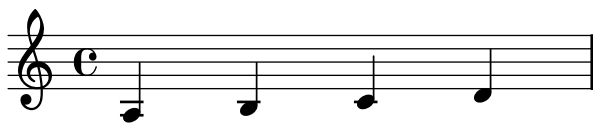
Sublist of grob property lists may be also tuned. In the next example, the `beamed-lengths` property of the `Stem` grob is tweaked.



`'page-break-between-scores.ly'`

Page breaks work when they are placed at the end of a score, or between scores.

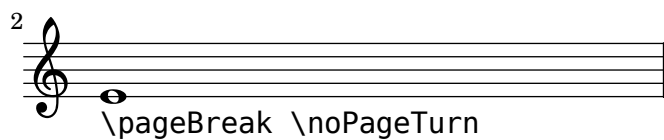
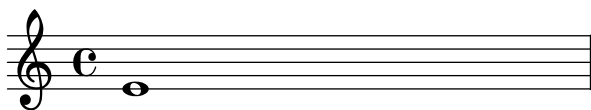
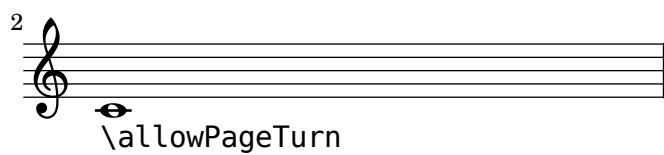
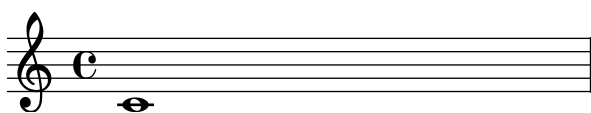


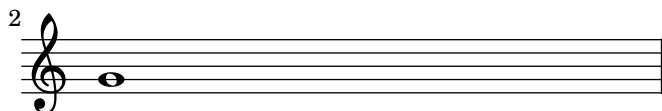
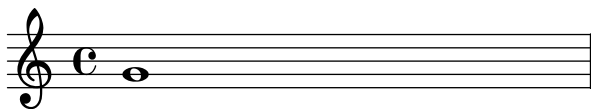


Music engraving by LilyPond 2.11.28—www.lilypond.org

‘page-break-turn-toplevel.ly’

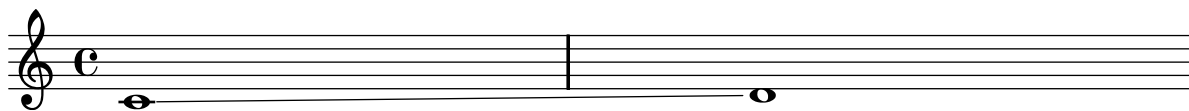
Page breaking and page turning commands (`\pageBreak`, `\noPageBreak`, etc), can be used at top level.





`'page-break-warn-forbidden.ly'`

If a page break is forced where it is forbidden, a warning is printed.



`'page-breaks.ly'`

Stress optimal page breaking. This should look nice and even on 4 a6 pages.



Title

(and (the) subtitle)

Sub sub title

Poet

Instrument

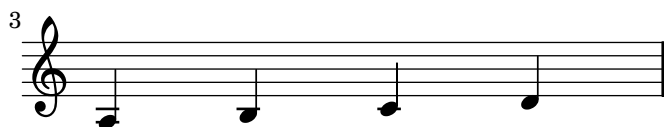
Composer

Meter (huh?)

Arranger

Piece

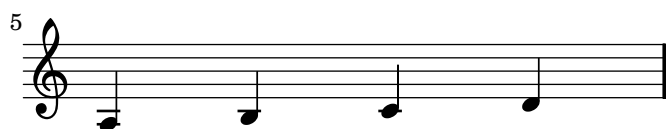
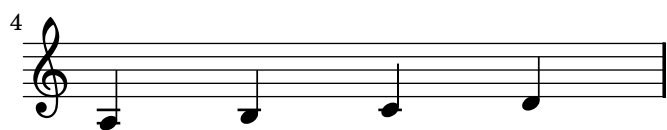
opus 0



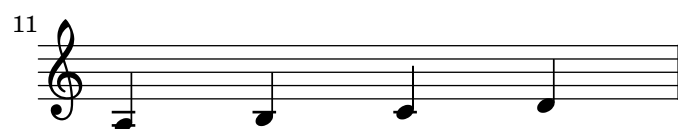
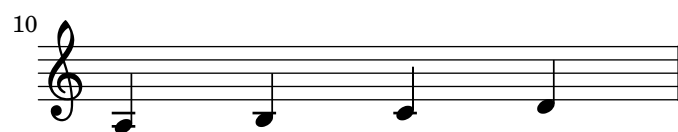
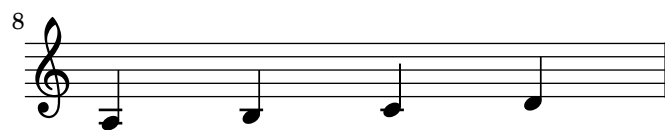
Copyright by /me



2 Instrument



Instrument 3



4Instrument

12

13

14

15

Music engraving by LilyPond 2.11.28 4

www.lilypond.org

‘page-label.ly’

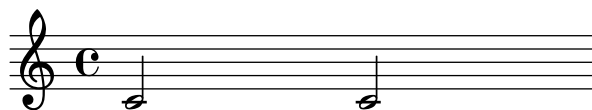
Page labels may be placed inside music or at top-level, and refered to in markups.

Title Page

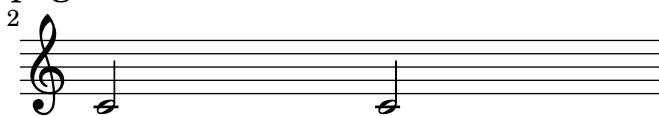
2

Table of contents	
Table of contents	2
First Score	3
Mark A	3
Mark B	4
Mark C	4
Unknown label	?

First score



A (page 3)



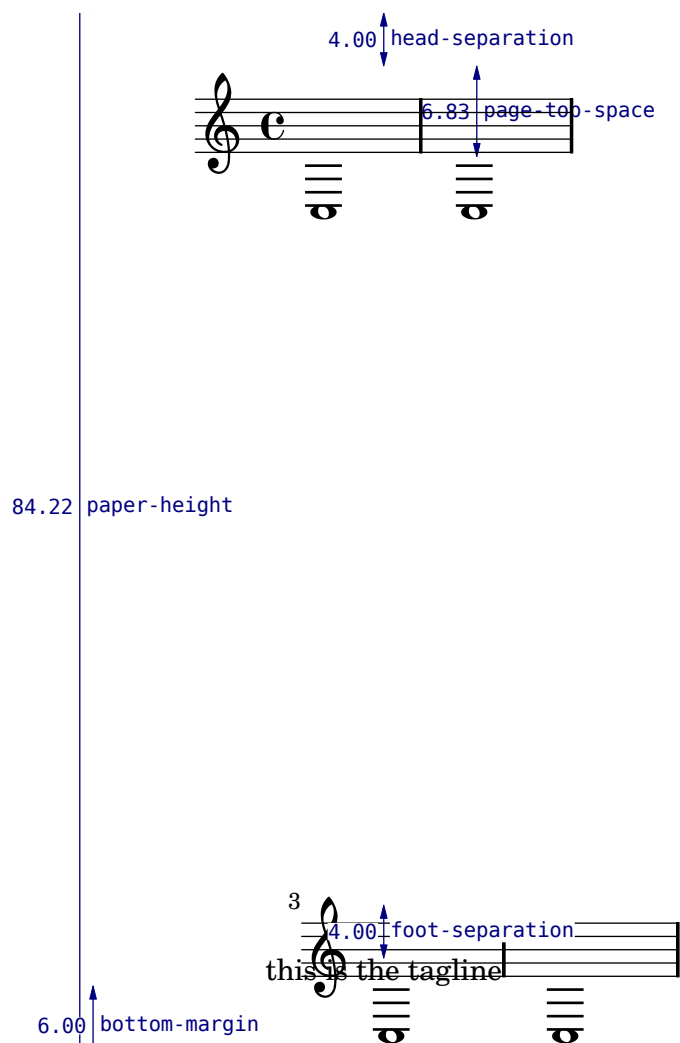
4



Music engraving by LilyPond 2.11.28—www.lilypond.org

`'page-layout-manual-position.ly'`

By setting `Y-offset` and `X-offset` for the `line-break-system-details` of `NonMusicalPaperColumn`, systems may be placed absolutely on the printable area of the page.



‘page-layout-twopass.ly’

Page breaking details can be stored for later reference.

Handwritten musical notation on three staves:

- Staff 1:** Treble clef, common time signature (C). It contains two measures, each with a whole note chord consisting of a G4 (one line), a B4 (two lines), and a D5 (three lines).
- Staff 2:** Treble clef. It begins with a triplet of eighth notes (G4, A4, B4) followed by a whole note chord of G4, B4, and D5.
- Staff 3:** Treble clef. It contains a sequence of 14 eighth notes, all marked with a fermion (a circle with a dot) below the staff.

‘page-layout.ly’

This shows how different settings on \paper modify the general page layout. Basically \paper will set the values for the whole paper while \layout for each \score block.

This file is best viewed outside the collated files document.

Diagram illustrating the layout of a music score page, showing the arrangement of staves and the associated spacing and padding parameters.

Page Information:

- Page Height: 169.01
- Page Title: Title (and the subtitle)
- Subtitle: Sub sub title
- Composer: Composer
- Arranger: Arranger

Staff Layout and Spacing Parameters:

- Staff 1:**
 - Y-extent: (-17.01, 0.00)
 - next-padding: 2.28
 - space after next-padding: 1.62
 - next-space+padding: 7.28
 - extent-estimate: (-8.10, -0.50)
- Staff 2:**
 - Y-extent: (-8.10, -0.15)
 - next-padding: 2.28
 - space after next-padding: 6.40
 - next-space+padding: 12.28
 - extent-estimate: (-8.10, -0.15)
- Staff 3:**
 - Y-extent: (-8.10, -0.50)
 - next-padding: 2.28
 - space after next-padding: 3.50
 - next-space+padding: 13.66
 - extent-estimate: (-8.10, -0.50)
- Staff 4:**
 - Y-extent: (-8.10, -0.22)
 - next-padding: 2.28
 - space after next-padding: 3.44
 - next-space+padding: 13.66
 - extent-estimate: (-8.10, -0.22)
- Staff 5:**
 - Y-extent: (-8.10, -0.15)
 - next-padding: 2.28
 - space after next-padding: 3.44
 - next-space+padding: 13.66
 - extent-estimate: (-8.10, -0.15)

Additional Parameters:

- head-separation: 4.00
- page-top-space: 6.83
- space left: 23.80
- foot-separation: 4.00

Music engraving by LilyPond 2.11.28—www.lilypond.org

‘page-limited-space.ly’

The space between systems can be limited when there is too much space left on the page by setting `page-limit-inter-system-space`.







`'page-minimal-page-breaking-last-page.ly'`

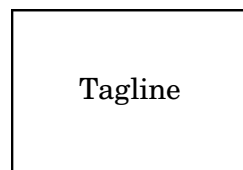
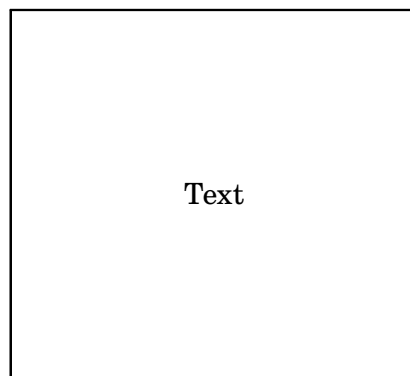
Minimal page breaker: special case when the last system is moved to an other page when there is not enough space because of the tagline.

Text

Text

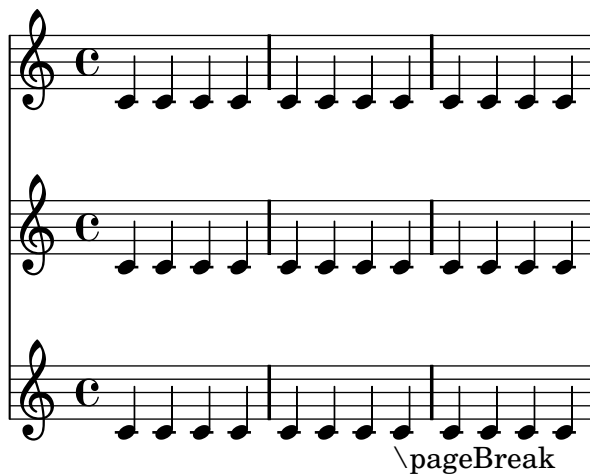
2

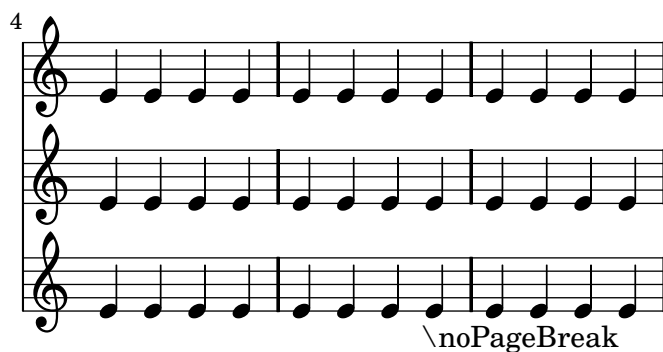
Text



`'page-minimal-page-breaking.ly'`

The minimal page breaker stacks as many lines on pages, only accounting for manual page break commands.





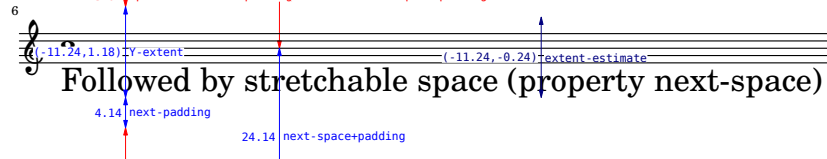
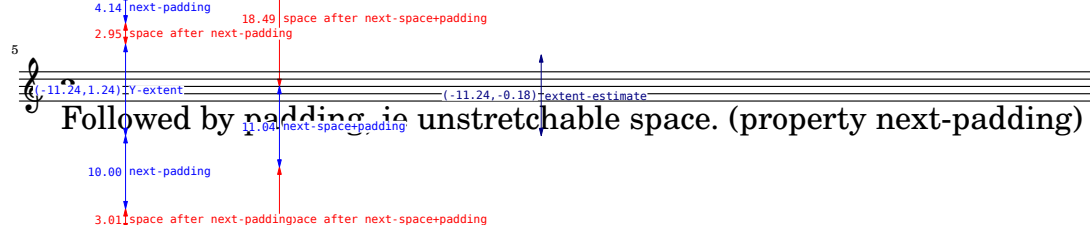
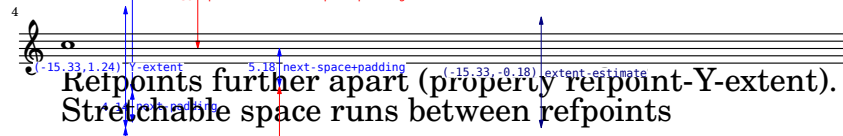
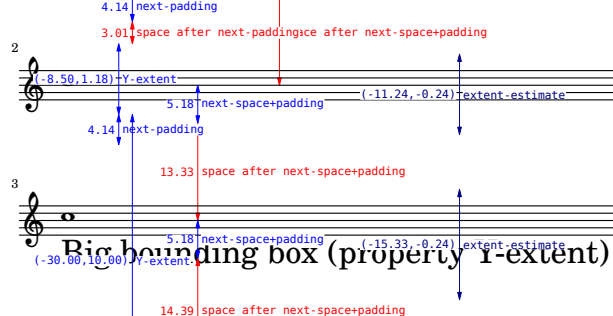
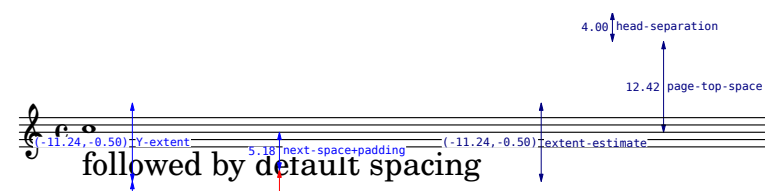
Music engraving by LilyPond 2.11.28—www.lilypond.org

‘page-spacing.ly’

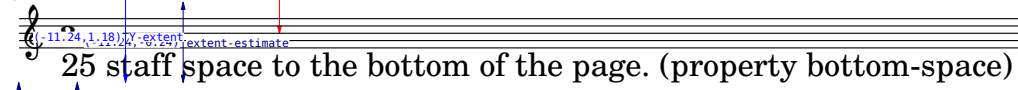
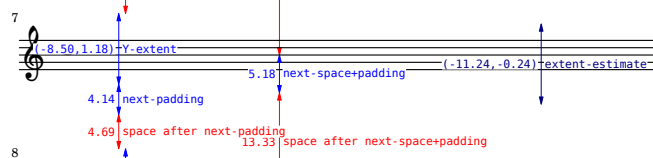
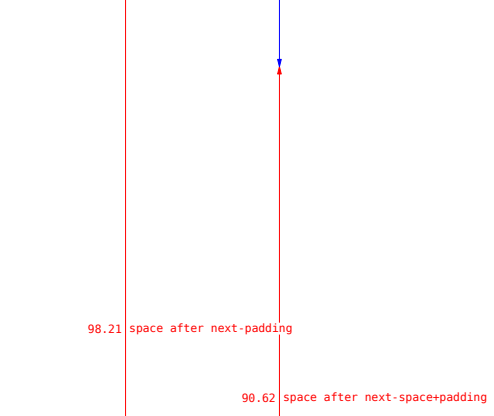
By setting properties in `NonMusicalPaperColumn`, vertical spacing of page layout can be adjusted.

For technical reasons, `overrideProperty` has to be used for setting properties on individual object. `\override` may still be used for global overrides.

By setting `annotate-spacing`, we can see the effect of each property.

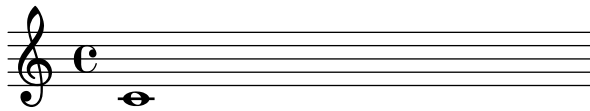


307.29 paper-height

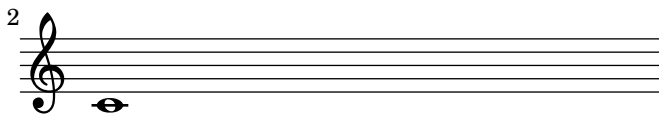


`'page-top-space.ly'`

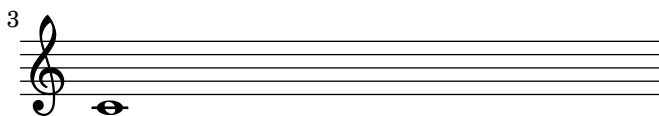
By setting `page-top-space`, the Y position of the first system can be forced to be uniform.



2

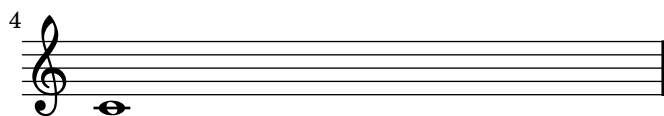


3



4

bla



Music engraving by LilyPond 2.11.28—www.lilypond.org

`‘page-turn-page-breaking-auto-first-page.ly’`

By default, we start with page 1, which is on the right hand side of a double page. In this example, `auto-first-page-number` is set to `##t` and the music won’t fit on a single page, so we should automatically set the first page number to 2 in order to avoid a bad page turn.



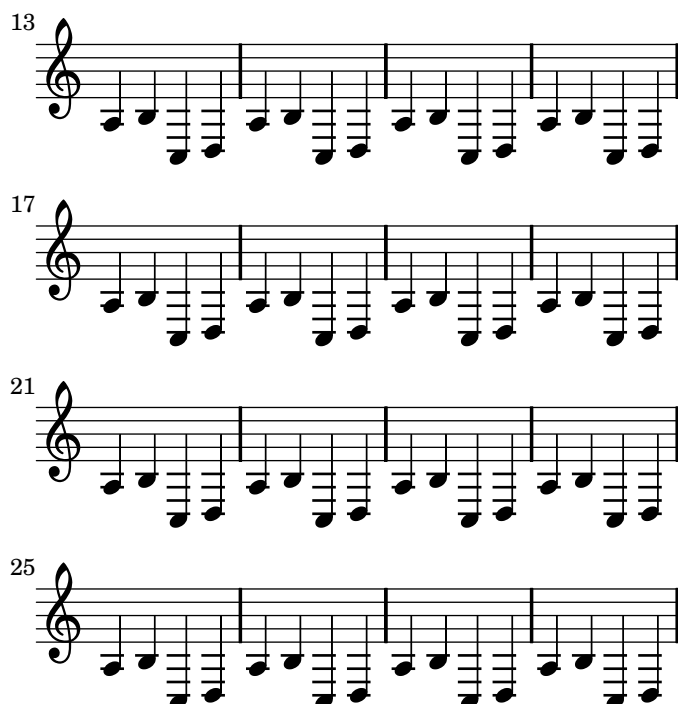
Music engraving by LilyPond 2.11.28—www.lilypond.org

`'page-turn-page-breaking-auto-first-page2.ly'`

By default, we start with page 1, which is on the right hand side of a double page. In this example, `auto-first-page-number` is set to `##t`. ALthough the music will fit on a single page, it would require stretching the first page badly, so we should automatically set the first page number to 2 in order to avoid a bad page turn.

2

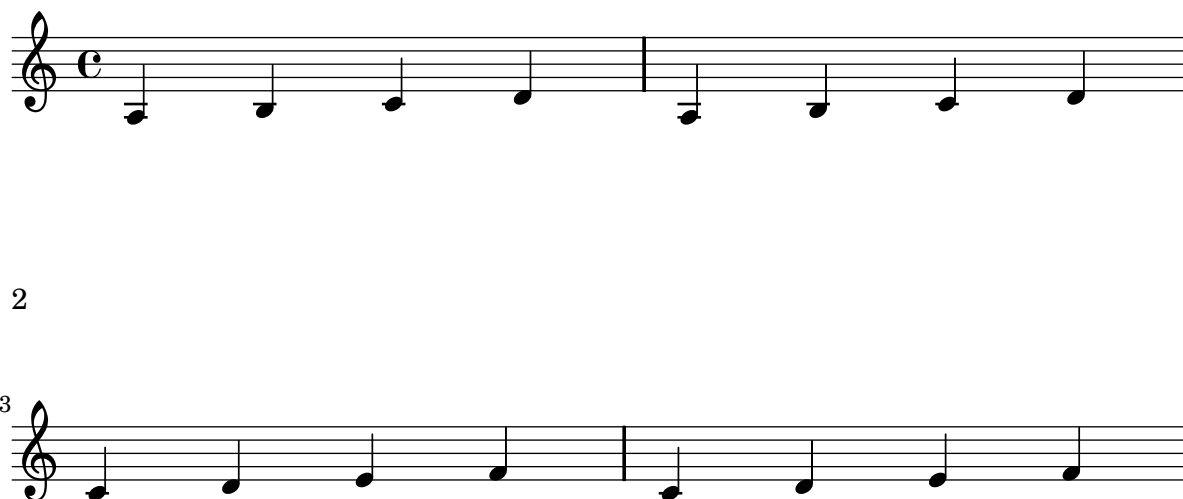


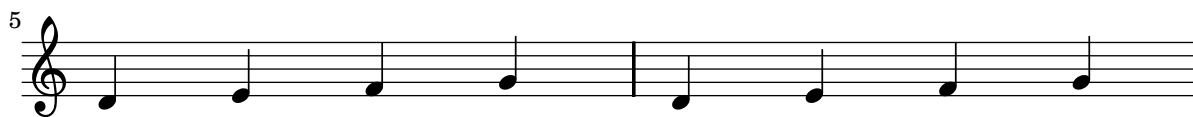


Music engraving by LilyPond 2.11.28—www.lilypond.org

‘page-turn-page-breaking-badturns.ly’

If there are no good places to have a page turn, the optimal-breaker will just have to recover gracefully. This should appear on 3 pages.





Music engraving by LilyPond 2.11.28—www.lilypond.org

‘page-turn-page-breaking-repeats.ly’

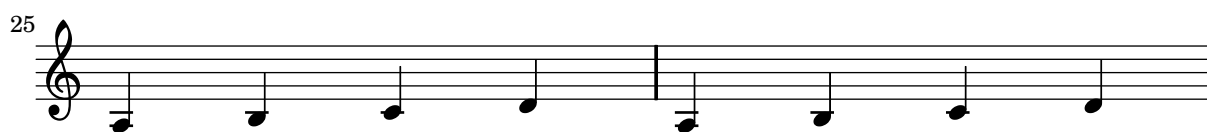
The page-turn engraver will not count potential page turns if they occur in the middle of a repeat unless there is a long gap at the beginning or at the end of the repeat.



2



4



6

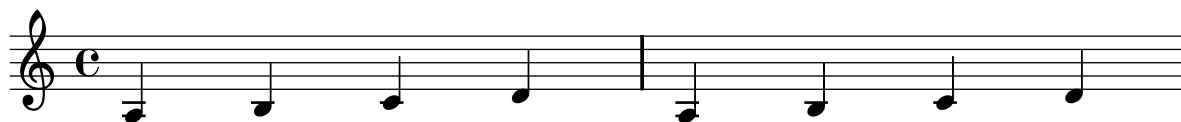


Music engraving by LilyPond 2.11.28—www.lilypond.org

‘page-turn-page-breaking.ly’

The page-turn breaker will put a page turn after a rest unless there is a ‘special’ barline within the rest, in which case the turn will go after the special barline.

2





4



6



8



`'parenthesize.ly'`

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



`'part-combine-a2.ly'`

The `a2` string is printed only on notes (i.e. not on rests), and only after chords, solo or polyphony.



`'part-combine-cross.ly'`

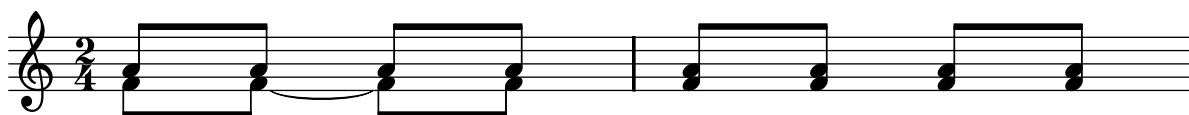
The part combiner stays apart for crossing voices.



`'part-combine-global.ly'`

The analysis of the part combiner is non-local: in the following example, the decision for using separate voices in the 1st measure is made on the 2nd note, but influences the 1st note.

In the 2nd measure, the pattern without the tie, leads to combined voices.



`'part-combine-mmrest-after-solo.ly'`

Multimeasure rests are printed after solos, both for `solo1` and for `solo2`.



`'part-combine-solo-end.ly'`

SOLO is printed even if the solo voice ends before the other one. Unfortunately, the multi-rest of the 1st voice (which is 2 bars longer than the 2nd voice) does not get printed.



`'part-combine-solo-global.ly'`

In this example, solo1 should not be printed over the 1st note, because of the slur which is present from the one-voice to the two-voice situation.



`'part-combine-solo.ly'`

A solo string can only be printed when a note starts. Hence, in this example, there is no Solo-2 although the 2nd voice has a dotted quarter, while the first voice has a rest.

A Solo indication is only printed once; (shared) rests do not require reprinting a solo indication.

Solo 1/2 can not be used when a spanner is active, so there is no solo over any of the tied notes.



`'part-combine-text.ly'`

The new part combiner detects a2, solo1 and solo2, and prints i texts accordingly.



`'part-combine-tuplet-end.ly'`

End tuplets events are sent to the starting context, so even after a switch, a tuplet ends correctly.



`'part-combine-tuplet-single.ly'`

Tuplets in combined parts only print one bracket.



`'part-combine.ly'`

The new part combiner stays apart from:

- different durations,
- different articulations (taking into account only slur/beam/tie), and
- wide pitch ranges.



`'pedal-bracket.ly'`

The brackets of a piano pedal should start and end at the left side of the note. If a note is shared between two brackets, these ends are flared.

At a line-break, there are no vertical endings.



`'pedal-end.ly'`

Unterminated piano pedal brackets run to the end of the piece.



`'pedal-ped.ly'`

The standard piano pedals style comes with Ped symbols. The pedal string can be also tuned, for example, to a shorter tilde/P variant at the end of the melody.



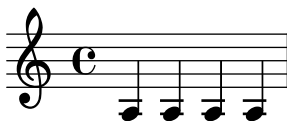
`'phrasing-slur-slur-avoid.ly'`

PhrasingSlurs go over normal slurs.



`'prefatory-empty-spacing.ly'`

The A is atop an invisible barline. The barline, although invisible, is also translated because it is the last one of the break alignment.



`'prefatory-spacing-matter.ly'`

Distances between prefatory items (e.g. clef, bar, etc.) are determined by engraving standards. These distances depend on which items are combined. Mid-line, the order for clef and bar-line is different from the start of line.



`'profile-property-access.ly'`

heavily mutilated Edition Peters Morgenlied by Schubert

LilyPond demo

Lieblich, etwas geschwind

1. Sü - ßes
2. いろはに ㄱㅇ

2.

3
Licht! Aus gol - denen Pfor - ten brichst du sie - gend durch die
ほへど ちり めるを ㄱㅇ いろはに ㄱㅇ

6
Nacht. Schöner Tag, du bist er - wacht.
ほへ ちり める

cresc. - - - - - *f*

`‘property-grace-polyphony.ly’`

Property overrides and reverts from `\grace` do not interfere with the overrides and reverts from polyphony.



`‘property-nested-revert.ly’`

nested properties may also be reverted. This uses Scheme list syntax.



`‘property-once.ly’`

Once properties take effect during a single time step only.



`‘quote-cue-during.ly’`

The `cueDuring` form of quotation will set stem directions on both quoted and main voice, and deliver the quoted voice in the `cue Voice`. The music function `\killCues` can remove all cue notes.

Spanners run to the end of a cue section, and are not started on the last note.

quoteMe

orig (killCues)

orig+quote

A musical score with three staves. The top staff is labeled 'quoteMe' and contains a sequence of notes with a fermata and a *ff* dynamic marking. The middle staff is labeled 'orig (killCues)' and contains a sequence of notes with a *ff* dynamic marking. The bottom staff is labeled 'orig+quote' and contains a sequence of notes. The staves are aligned to show the relationship between the quoted and original music.

`‘quote-cyclic.ly’`

Two quoted voices may refer to each other. In this example, there are notes with each full-bar rest.



‘quote-during.ly’

With `\cueDuring` and `\quoteDuring`, fragments of previously entered music may be quoted. `quotedEventTypes` will determines what things are quoted. In this example, a 16th rests is not quoted, since `rest-event` is not in `quotedEventTypes`.

The image shows three staves of musical notation. The top staff, labeled 'quoteMe', contains a sequence of notes: a quarter note with a sharp, an eighth note, a sixteenth rest, an eighth note with an accent, and a quarter note. The middle staff, labeled 'orig', contains a quarter note, an eighth note, a sixteenth rest, an eighth note with a sharp, and a quarter note. The bottom staff, labeled 'orig+quote', contains a quarter note, an eighth note, a sixteenth rest, an eighth note with a sharp, and a quarter note. The 'orig' staff has a *ff* dynamic marking above the eighth note with a sharp. The 'orig+quote' staff has an accent mark above the eighth note with a sharp.

‘quote-grace.ly’

Quotes may contain grace notes. The grace note leading up to an unquoted note is not quoted.

The image shows two staves of musical notation. The top staff, labeled 'quoted', contains a quarter rest, followed by a sequence of notes: an eighth note, a quarter note, an eighth note with a flat, a quarter note with a flat, and a quarter note. The bottom staff, labeled 'original', contains a quarter rest, followed by a sequence of notes: an eighth note, a quarter note, an eighth note with a flat, a quarter note with a flat, and a quarter note. The 'original' staff has a grace note (an eighth note with a flat) leading up to the first quarter note of the sequence.

‘quote-tie.ly’

Voices from different cues must not be tied together. In this example, the first note has a tie. This note should not be tied to the 2nd note.

The image shows a single staff of musical notation. It contains a quarter note with a tie, followed by a quarter rest, and then another quarter note with a tie.

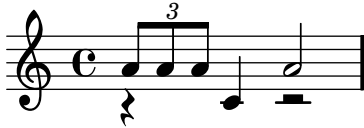
‘quote-transposition.ly’

Quotations take into account the transposition of both source and target. In this example, all instruments play sounding central C, the target is a instrument in F. The target part may be `\transposed`. In this case, all the pitches (including the quoted ones) will transposed as well.

The image shows two staves of musical notation. The top staff has two parts: 'clar' (clarinet) and 'sax' (saxophone). The bottom staff also has two parts: 'clar' and 'sax'. The 'clar' part on both staves is identical, consisting of a quarter note, an eighth note, and a quarter note. The 'sax' part on both staves is identical, consisting of a quarter note, an eighth note, and a quarter note. The bottom staff has a label 'up 1 tone' below it, indicating that the pitches are transposed up one tone.

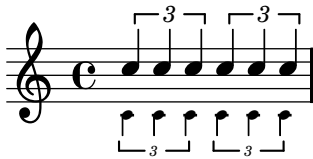
‘quote-tuplet-end.ly’

Tuplet bracket ends properly when quoting.



‘quote-tuplet.ly’

In cue notes, Tuplet stops are handled before new triplets start.



‘quote.ly’

With `\quote`, fragments of previously entered music may be quoted. `quotedEventTypes` will determines what things are quoted. In this example, a 16th rests is not quoted, since `rest-event` is not in `quotedEventTypes`.

quoteMe

orig

orig+quote

‘rehearsal-mark-align-priority.ly’

When the `break-align-symbols` property is given as a list, the alignment depends on which symbols are visible.



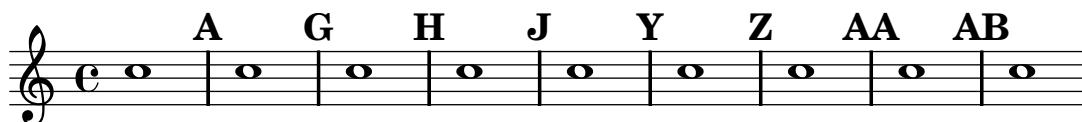
‘rehearsal-mark-align.ly’

The rehearsal mark is put on top a breakable symbol, according to the value of `break-align-symbols` value of the `RehearsalMark`. The same holds for `BarNumber` grobs.



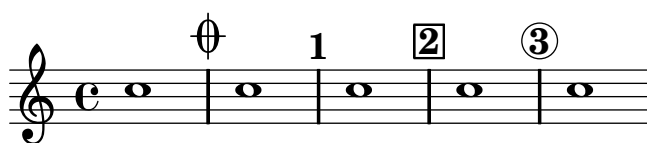
`'rehearsal-mark-letter.ly'`

Rehearsal marks in letter style: the I is skipped, and after Z, double letters are used. The mark may be set with `\mark NUMBER`, or with `Score.rehearsalMark`.



`'rehearsal-mark-number.ly'`

Marks can be printed as numbers. By setting `markFormatter` we may choose a different style of mark printing. Also, marks can be specified manually, with a markup argument.



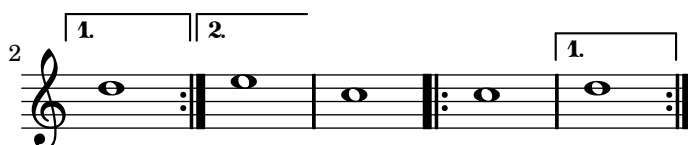
`'relative-repeat.ly'`

Relative mode for repeats uses order of entry.



`'repeat-line-break.ly'`

Across linebreaks, the left edge of a first and second alternative bracket should be equal.



`'repeat-percent-count.ly'`

Percent repeats get incremental numbers when `countPercentRepeats` is set, to indicate the repeat counts, but only if there are more than two repeats.



`'repeat-percent-grace.ly'`

Percent repeats are also centered when there is a grace note in a parallel staff.



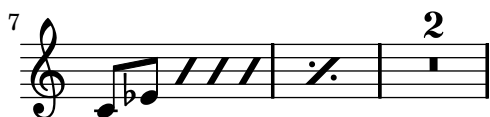
`'repeat-percent-skipbars.ly'`

Percent repeats are not skipped, even when `skipBars` is set.



`'repeat-percent.ly'`

Measure repeats may be nested with beat repeats.



`'repeat-slash.ly'`

Within a bar, beat repeats denote that a music snippet should be played again.



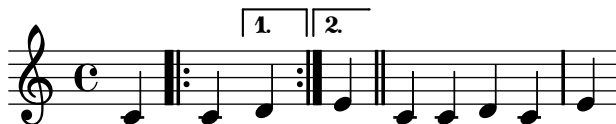
`'repeat-tie.ly'`

Repeat ties are only connected on the right side to a note head.



`'repeat-unfold-all.ly'`

Volta repeats may be unfolded through the music function `\unfoldRepeats`.



`'repeat-unfold-tremolo.ly'`

Unfolding tremolo repeats. All fragments fill one measure with 16th notes exactly.



`'repeat-unfold.ly'`

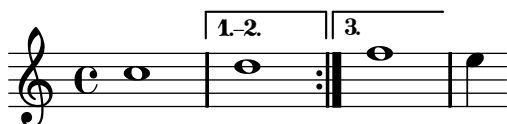
LilyPond has two modes for repeats: unfolded and semi-unfolded. Unfolded repeats are fully written out. Semi unfolded repeats have the body written and all alternatives sequentially. If the number of alternatives is larger than the repeat count, the excess alternatives are ignored. If the number of alternatives is smaller, the first alternative is multiplied to get to the number of repeats.

Unfolded behavior:



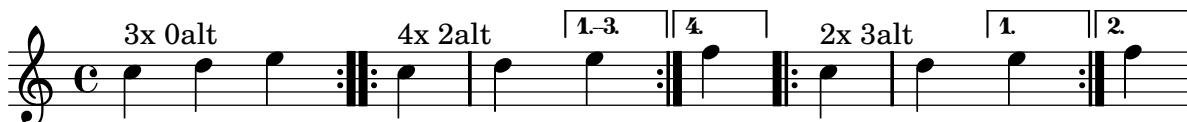
`'repeat-volta-skip-alternatives.ly'`

When too few alternatives are present, the first alternative is repeated, by printing a range for the 1st repeat.



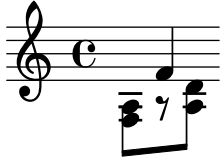
`'repeat-volta.ly'`

Volta (Semi folded) behavior. Voltas can start on non-barline moments. If they don't barlines should still be shown.



`'rest-collision-beam-note.ly'`

Beam/rest collision resolution and normal rest/note collisions can be combined.



`'rest-collision-beam-quantized.ly'`

Rests under beams moved by whole staff spaces.



`'rest-collision-beam-restdir.ly'`

Beam/rest collision takes offset due to Rest #'direction into account properly.



`'rest-collision-beam.ly'`

Rests under beams are shifted upon collision.



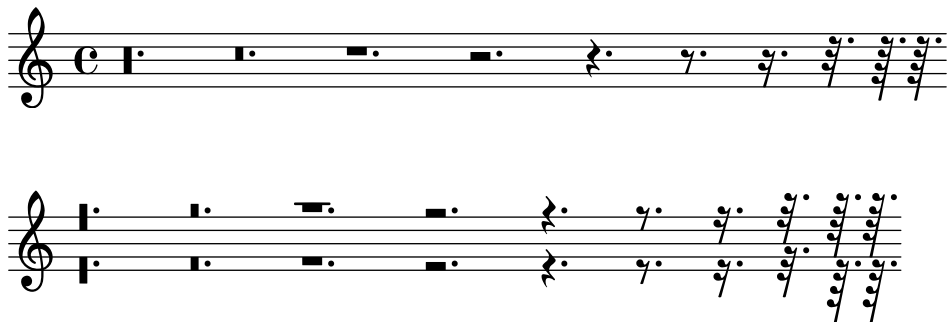
`'rest-collision.ly'`

Rests should not collide with beams, stems and noteheads. Rests may be under beams. Rests should be move by integral number of spaces inside the staff, and by half spaces outside. Notice that the half and whole rests just outside the staff get ledger lines in different cases.



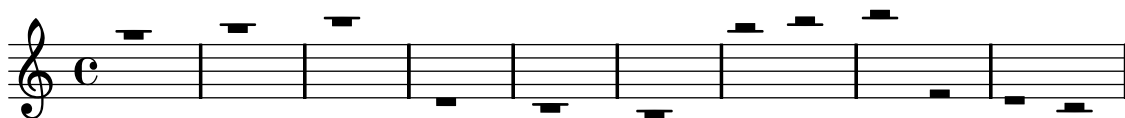
`'rest-dot-position.ly'`

Dots of rests should follow the rest positions.



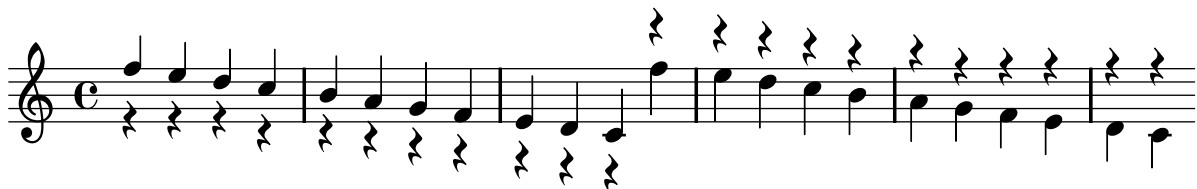
`'rest-ledger.ly'`

Whole and half rests moving outside the staff should get ledger lines.



`'rest-note-collision.ly'`

In rest-note collisions, the rest moves in discrete steps, and inside the staff, it moves in whole staff spaces.



`'rest-pitch.ly'`

Rests can have pitches—these will be affected by transposition and relativization. If a rest has a pitch, rest/rest and beam/rest collision resolving will leave it alone.



`'rest-pitched-beam.ly'`

Pitched rests under beams.



`'rest-polyphonic.ly'`

In polyphonic situations, rests are moved down even if there is no opposite note or rest. The amount is two `staff-spaces`.



`'rest.ly'`

There is a big variety of rests. Note that the dot of 8th, 16th and 32nd rests rest should be next to the top of the rest. All rests except the whole rest are centered on the middle staff line.



`'rhythmic-staff.ly'`

In rhythmic staves stems should go up, and bar lines have the size for a 5 line staff. The whole rest hangs from the rhythmic staff.



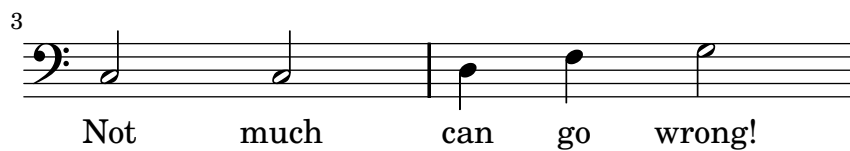
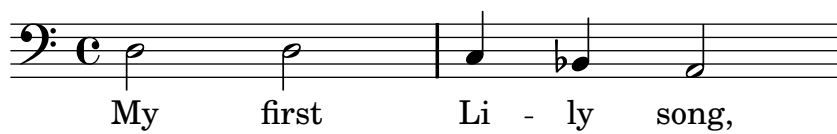
`'safe.ly'`

This should not survive lilypond `--safe-mode`

`'score-text.ly'`

Markup texts are rendered above or below a score.

High up above



2. My next Li-ly verse
Now it's getting worse!

3. My last Li-ly text
See what will be next!

`'script-center-seconds.ly'`

Scripts on chords with seconds remain centered on the extremal note head



`'script-collision.ly'`

Scripts are put on the utmost head, so they are positioned correctly when there are collisions.



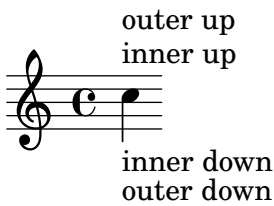
`'script-stack-horizontal.ly'`

horizontal scripts are ordered, so they do not overlap. The order may be set with script-priority.



`'script-stack-order.ly'`

Scripts can be stacked. The order is determined by a priority field, but when objects have the same priority, the input order determines the order. Objects specified first are closest to the note.



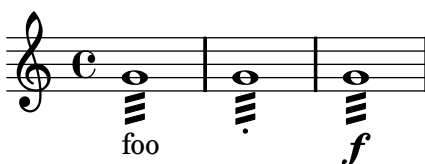
`'script-stacked.ly'`

Scripts may be stacked.



`'script-stem-tremolo.ly'`

Scripts avoid stem tremolos even if there is no visible stem.



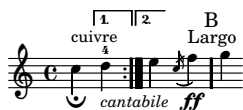
‘semi-tie-manual-direction.ly’

Semi tie directions may be forced from the input.



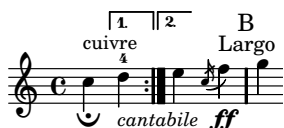
‘size11.ly’

Different text styles are used for various purposes.



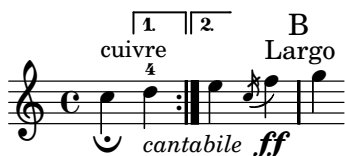
‘size13.ly’

Different text styles are used for various purposes.



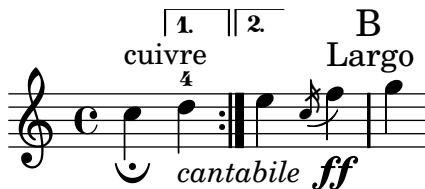
‘size16.ly’

Different text styles are used for various purposes.



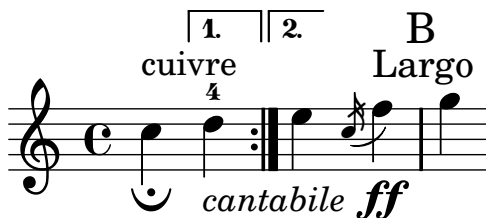
‘size20.ly’

Different text styles are used for various purposes.



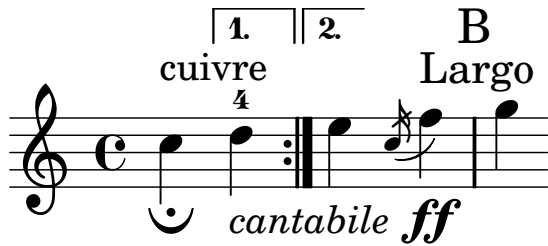
‘size23.ly’

Different text styles are used for various purposes.



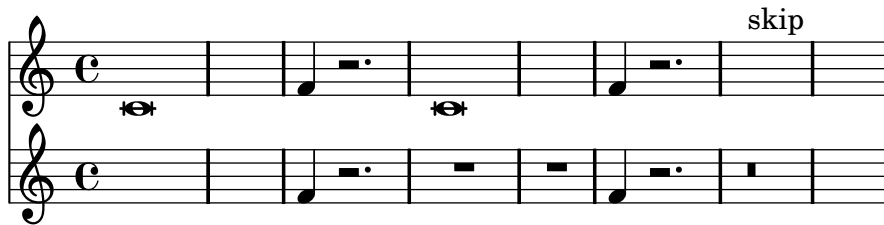
`‘size26.ly’`

Different text styles are used for various purposes.



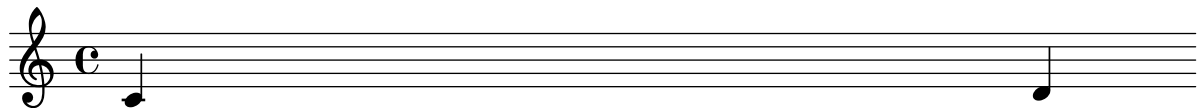
`‘skip-of-length.ly’`

`skip-of-length` and `mmrest-of-length` create skips and rests that last as long as their arguments.



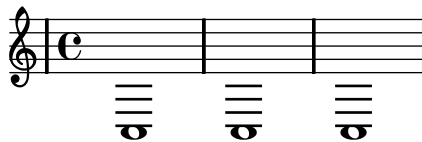
`‘skiptypesetting-bar-check.ly’`

`skipTypesetting` doesn’t affect bar checks.



`‘skiptypesetting-show-last.ly’`

`showLastLength` will only show the last bit of a score



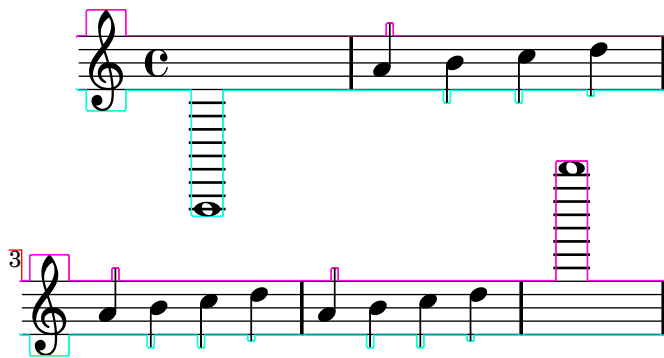
`‘skiptypesetting-tuplet.ly’`

Tuplet brackets are also skipped with `skipTypesetting`.



`'skyline-debug.ly'`

`-ddebug-skyline` draws the outline of the skyline used.



`'skyline-vertical-placement.ly'`

Grobs that have `outside-staff-priority` set are positioned using a skyline algorithm so that they don't collide with other objects.

this goes above the previous markup

this doesn't collide with the c

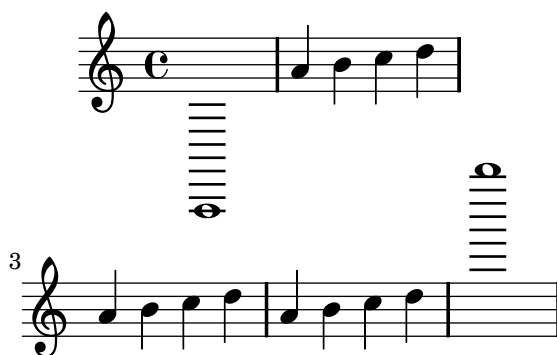
this goes below the dynamic

f

A musical score snippet on a single staff with a treble clef and a common time signature 'c'. It contains a whole note on G4, followed by a half note on A4, and two quarter notes on B4. Above the staff, the text "this goes above the previous markup" is positioned. Below the staff, the text "this goes below the dynamic" is positioned, with a dynamic marking *f* (forte) below it. The text "this doesn't collide with the c" is positioned above the first note. The skyline algorithm is visualized with magenta and cyan outlines around the notes and stems, showing how they are positioned relative to each other and the staff lines.

`'skyline-vertical-spacing.ly'`

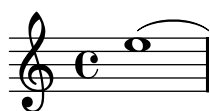
We use a skyline algorithm to determine the distance to the next system instead of relying only on bounding boxes. This keeps gaps between systems more uniform.

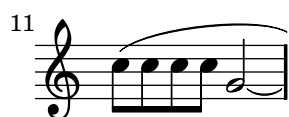


Music engraving by LilyPond 2.11.28—www.lilypond.org

‘`slur-broken-trend.ly`’

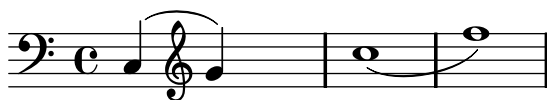
Across line breaks, slurs behave nicely. On the left, they extend to just after the preferatory matter, and on the right to the end of the staff. A slur should follow the same vertical direction it would have in unbroken state.





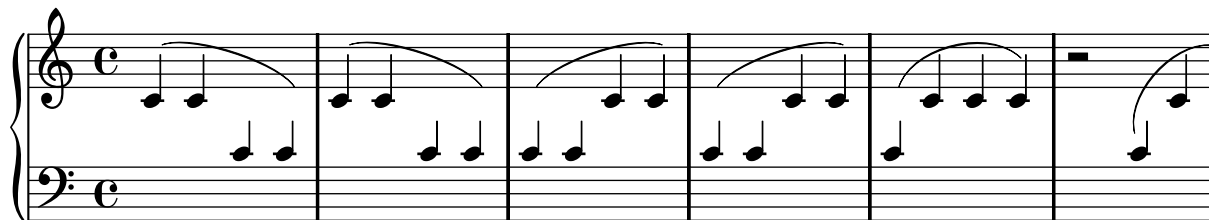
‘slur-clef.ly’

Slurs avoid clefs, but don't avoid barlines.

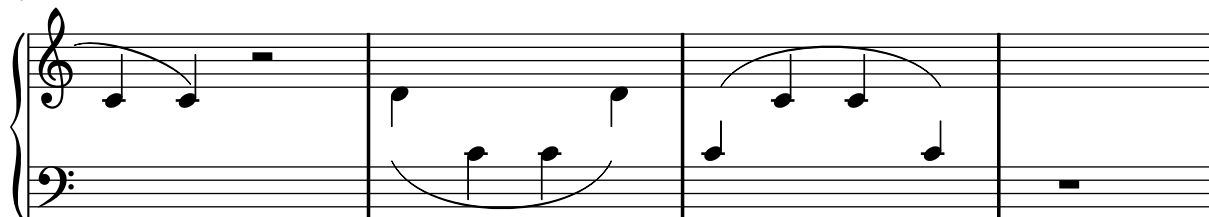


`'slur-cross-staff.ly'`

Slurs behave decently when broken across a linebreak.



7



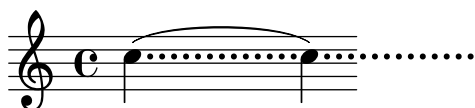
`'slur-dash.ly'`

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



`'slur-dots.ly'`

Slurs should not get confused by augmentation dots. With a lot of dots, the problems becomes more visible.



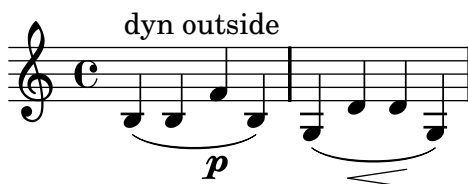
`'slur-double.ly'`

Some composers use slurs both above and below chords. This can be typeset by setting `doubleSlurs`



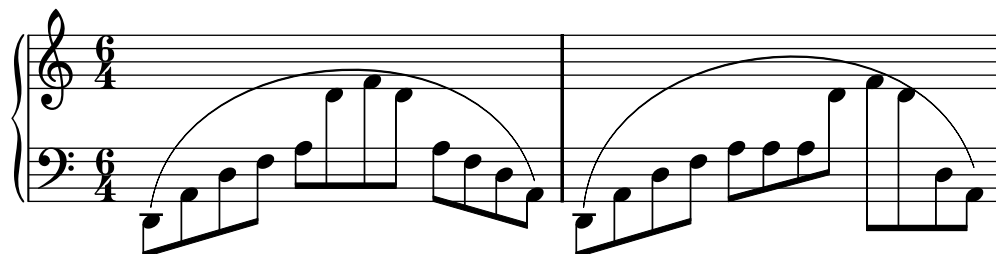
`'slur-dynamics.ly'`

Dynamics avoid collision with slur.

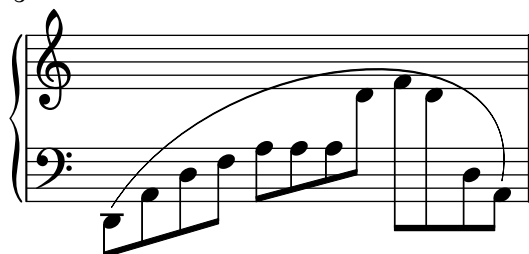


`'slur-extreme.ly'`

Extreme slurs are scaled to fit the pattern, but only symmetrically. Asymmetric slurs are created by setting `eccentricity`.



3



`'slur-manual.ly'`

Setting `positions` overrides the automatic positioning of the slur. It selects the slur configuration closest to the given pair.



`'slur-nice.ly'`

Slurs should look nice and symmetric. The curvature may increase only to avoid noteheads, and as little as possible. Slurs never run through noteheads or stems.



6



`'slur-rest.ly'`

Slurs may be placed over rest. The slur will avoid colliding with the rest.



‘slur-scoring.ly’

Slur formatting is based on scoring. A large number of slurs are generated. Each esthetic aspect gets demerits, the best configuration (with least demerits) wins. This must be tested in one big file, since changing one score parameter for one situation may affect several other situations.

Tunable parameters are in ‘scm/slur.scm’.

4

7

12

17

21 slurs forced down

27

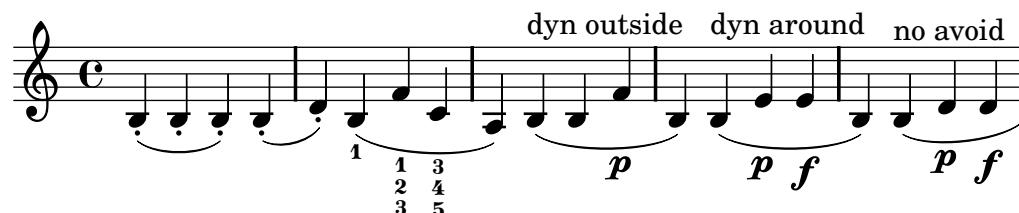
‘slur-script-inside.ly’

Slurs avoid scripts with `avoid-slur` set to `inside`, scripts avoid slurs with `avoid-slur` set to `around`. Slurs and scripts keep a distance of `slur-padding`.

‘slur-script.ly’

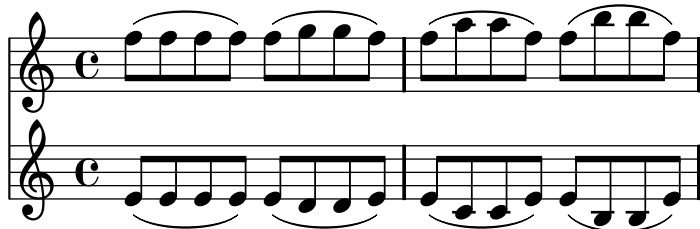
A slur avoids collisions with scripts. Articulations go inside the slur, dynamic markings go outside the slur. Fingerings and texts are placed either inside or outside.

For different configurations, the defaults can be changed, and scripts can be moved manually.



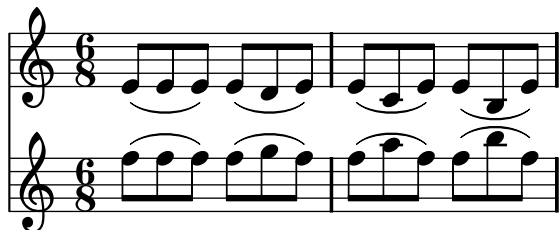
‘slur-symmetry-1.ly’

Symmetric figures should lead to symmetric slurs.



‘slur-symmetry.ly’

Symmetric figures should lead to symmetric slurs.



‘slur-tilt.ly’

The attachment point for strongly sloped slurs is shifted horizontally slightly. Without this correction, slurs will point into one note head, and point over another note head.

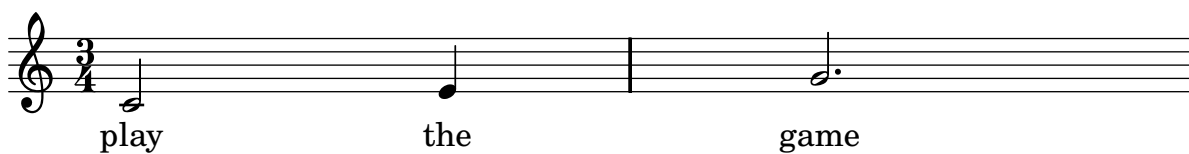


‘slur-tuplet.ly’

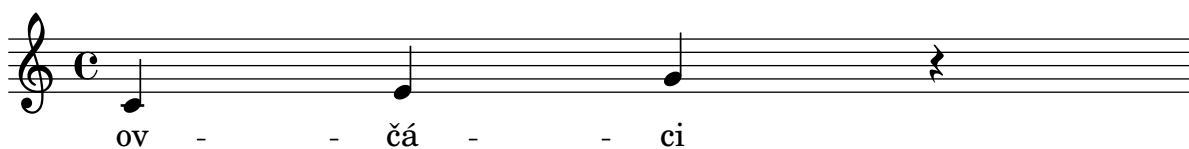
TupletNumber grobs are always inside slurs. This may not work if the slur starts after the tuplet.



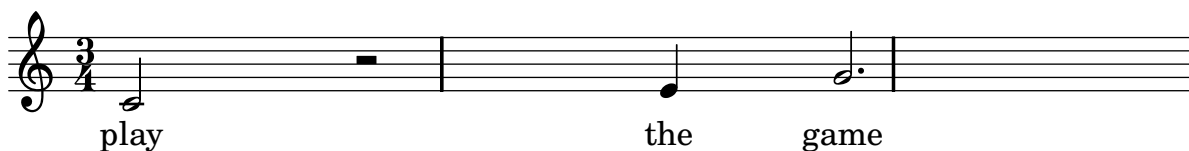
‘song-associated-voice.ly’



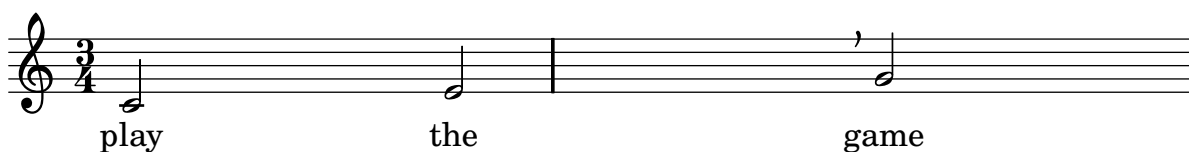
‘song-basic-nonenglish.ly’



‘song-basic.ly’



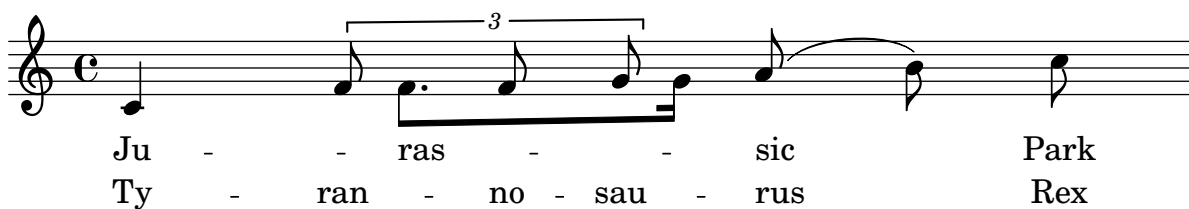
‘song-breathe.ly’



‘song-melisma.ly’



‘song-reordering.ly’



'song-reordering2.ly'

'song-repetition.ly'

do mi sol mi do do re mi fa sol la si do

dodo rere mimi fafa solsol

`'song-skip-noword.ly'`

'song-skip.ly'

twin - - - - - kle

'song-slurs.ly'

more
go

slow - - ly
fas - - ter still

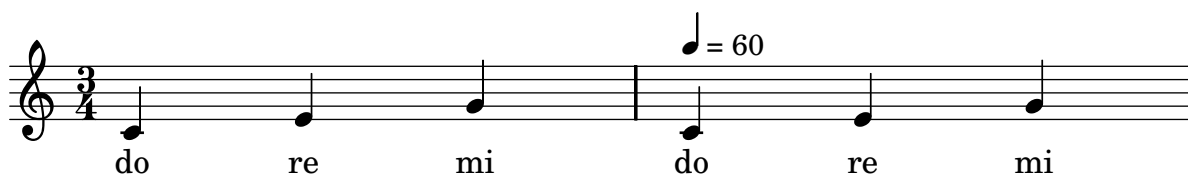
'song-splitpart.ly'

we shall not o- ver- come

‘song-stanzas.ly’



‘song-tempo.ly’



‘spacing-accidental-staffs.ly’

Accidentals in different staves do not affect the spacing of the eighth notes here.



‘spacing-accidental-stretch.ly’

Accidentals do not influence the amount of stretchable space. The accidental does add a little non-stretchable space.

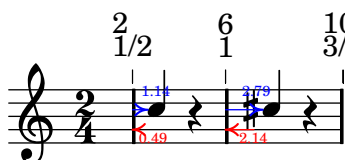
‘spacing-accidental.ly’

Accidentals sticking out to the left of a note will take a little more space, but only if the spacing is tight.



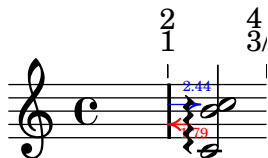
‘spacing-bar-accidental.ly’

An accidental following a bar gets space so the left edge of the acc is at 0.3 - 0.6 staff space of the bar line



`'spacing-bar-arpeggio.ly'`

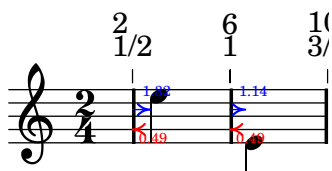
An arpeggio following a bar gets space



`'spacing-bar-stem.ly'`

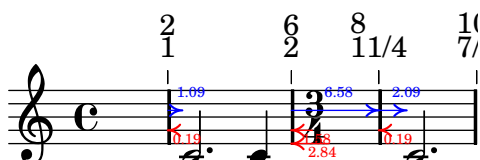
Downstem notes following a barline are printed with some extra space. This is an optical correction similar to juxtaposed stems.

The bar upstem should be approx 1.1 staff space, the bar downstem 1.3 to 1.5 staff space.



`'spacing-bar-whole-measure.ly'`

Notes that fill a whole measure are preceded by extra space.



`'spacing-clef-first-note.ly'`

Clef changes at the start of a line get much more space than clef changes halfway the line.



`'spacing-end-of-line.ly'`

Broken engraving of a bar at the end of a line does not upset the space following rests and notes.





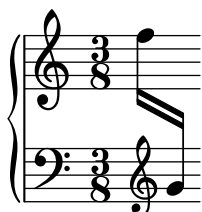
`'spacing-ended-voice.ly'`

A voicelet (a very short voice to get polyphonic chords correct) should not confuse the spacing engine.



`'spacing-folded-clef-cross-staff.ly'`

Clefs are also folded under cross staff constructs.



`'spacing-folded-clef.ly'`

A clef can be folded below notes in a different staff, if this does not disrupt the flow of the notes.



`'spacing-folded-clef2.ly'`

A clef can be folded below notes in a different staff, if there is space enough. With `Paper_column` stencil callbacks we can show where columns are in the score.



`'spacing-grace-duration.ly'`

Spacing uses the duration of the notes, but disregards grace notes for this. In this example, the 8ths around the grace are spaced exactly as the other 8th notes.



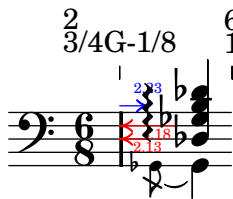
`‘spacing-grace.ly’`

Grace note runs have their own spacing variables in `Score.GraceSpacing`. So differing grace note lengths inside a run are spaced accordingly.



`‘spacing-horizontal-skyline-grace.ly’`

Skyline horizontal spacing may fold non-adjacent columns together, but they still do not collide. In this case, the arpeggio and the barline do not collide.



`‘spacing-horizontal-skyline.ly’`

accidentals may be folded under preceding notes.



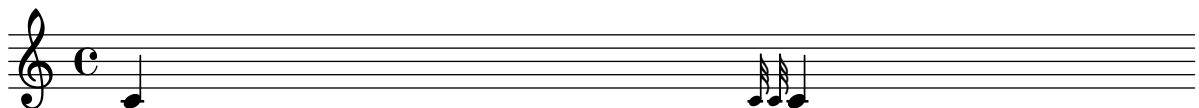
`‘spacing-knee.ly’`

For knees, the spacing correction is such that the stems are put at regular distances. This effect takes into account the width of the note heads and the thickness of the stem.



`‘spacing-loose-grace-error.ly’`

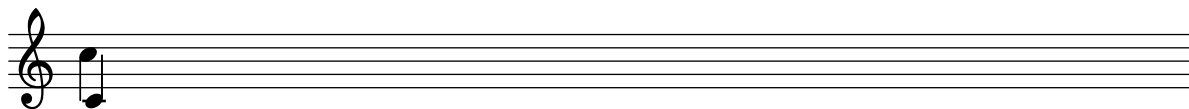
Even in case of incorrect contexts (eg. shortlived contexts) that break linking of columns through spacing wishes, `strict-note-spacing` defaults to a robust solution.



`‘spacing-loose-grace-linebreak.ly’`

If a floating grace spacing section attaches to a note across a line break, it gets attached to the end of line.





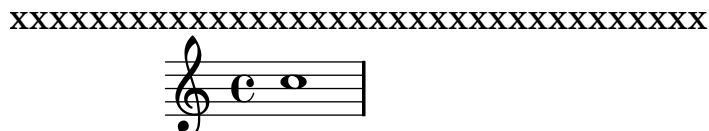
`'spacing-loose-grace.ly'`

With `strict-grace-spacing`, grace notes don't influence spacing.



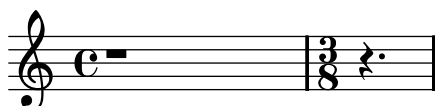
`'spacing-mark-width.ly'`

Width of marks does not affect spacing.



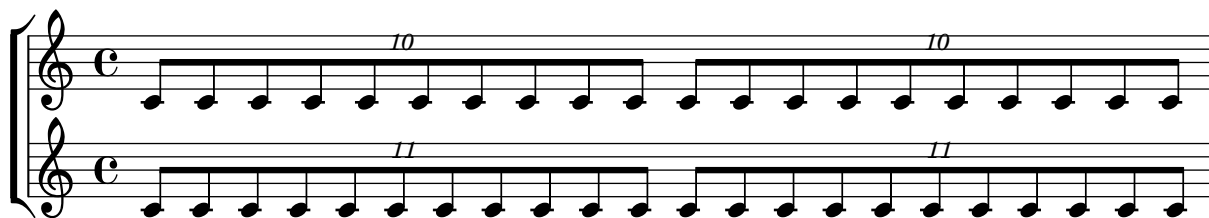
`'spacing-measure-length.ly'`

Horizontal spacing is bounded by of the current measure length. This means that the $\frac{3}{8}$ setting does not affect the whole rest spacing.



`'spacing-multi-tuplet.ly'`

Concurrent tuplets should be equidistant on all staves. Such equidistant spacing is at odds with elegant engraver spacing; hence it must be switched on explicitly with the `uniform-stretching` property of `SpacingSpanner`.



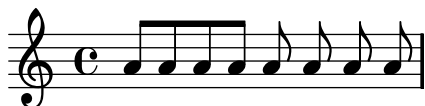
`'spacing-no-note.ly'`

In the absence of `NoteSpacings`, wide objects still get extra space. In this case, the slash before the barline gets a little more space.



`‘spacing-note-flags.ly’`

The flags of 8th notes take some space, but not too much: the space following a flag is less than the space following a beamed 8th head.



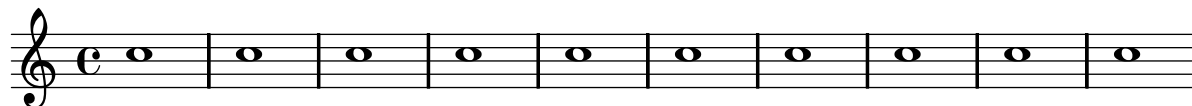
`‘spacing-proportional.ly’`

Proportional notation can be created by setting `proportionalNotationDuration`. Notes will be spaced proportional to the distance for the given duration.



`‘spacing-ragged-last.ly’`

If `raggedlast` is set, the systems are broken similar to paragraph formatting in text: the last line is unjustified.



`‘spacing-rest.ly’`

Rests get a little less space, since they are narrower. However, the quarter rest in feta font is relatively wide, causing this effect to be very small.



`‘spacing-section.ly’`

New sections for spacing can be started with `ewSpacingSection`. In this example, a section is started at the 4/16, and a 16th in the second section takes as much space as a 8th in first section.



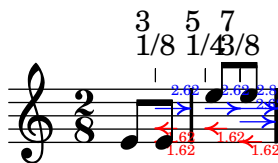
‘spacing-short-notes.ly’

Notes that are shorter than the common shortest note get a space (i.e. without the space needed for the note) proportional to their duration. So, the 16th notes get 1/2 of the space of an eighth note. The total distance for a 16th (which includes note head) is 3/4 of the eighth note.



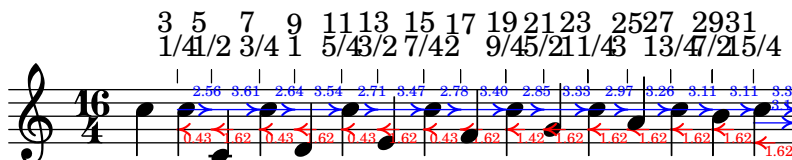
‘spacing-stem-bar.ly’

Upstem notes before a barline are printed with some extra space. This is an optical correction similar to juxtaposed stems.



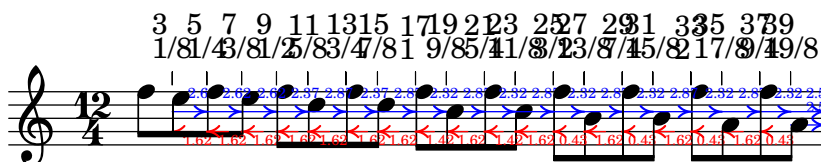
‘spacing-stem-direction.ly’

There are optical corrections to the spacing of stems. The overlap between two adjacent stems of different direction is used as a measure for how much to correct.



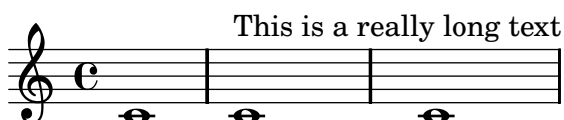
‘spacing-stem-same-direction.ly’

For juxtaposed chords with the same direction, a slight optical correction is used. It is constant, and works only if two chords have no common head-positions range.



‘spacing-stick-out.ly’

If keep-inside-line is set for the relevant PaperColumn, LilyPond will space a line to prevent text sticking out of the right margin.



`'spacing-strict-notespacing.ly'`

If `strict-note-spacing` is set, then spacing of notes is not influenced by bars and clefs half-way on the system. Rather, they are put just before the note that occurs at the same time. This may cause collisions.



`'spacing-strict-spacing-grace.ly'`

With `strict-note-spacing` spacing for grace notes (even multiple ones), is floating as well.



`'spacing-to-grace.ly'`

Space from a normal note (or barline) to a grace note is smaller than to a normal note.



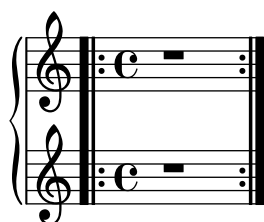
`'spacing-uniform-stretching.ly'`

Notes are spaced exactly according to durations, if `uniform-stretching` is set. Accidentals are ignored, and no optical-stem spacing is performed.

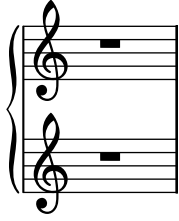


`'span-bar-break.ly'`

At the beginning of a system, the `|:` repeat barline is drawn between the staves, but the `:|` is not.



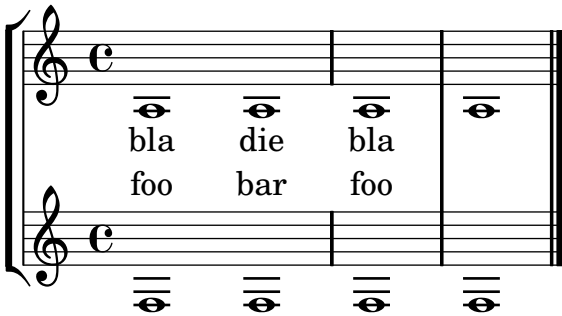
2



`'span-bar.ly'`

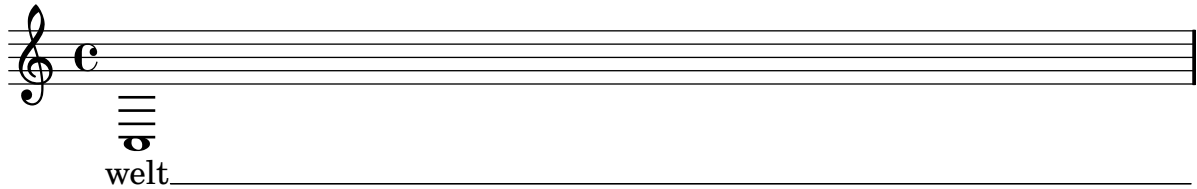
Span bars are drawn only between staff bar lines. By setting bar lines to transparent, they are shown only between systems.

Setting `SpanBar` transparent removes the barlines between systems.

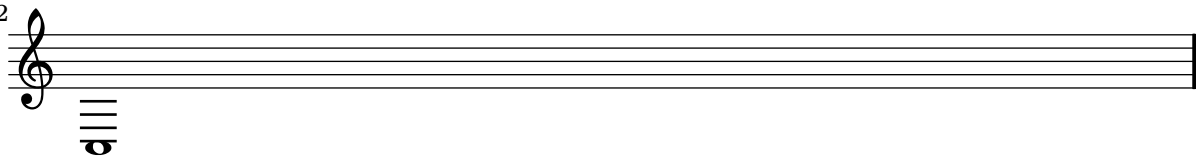


`'spanner-break-beyond-parent.ly'`

Spanners parts that extend beyond their parents are killed in case of line breaks.



2



`'spanner-break-overshoot.ly'`

The `break-overshoot` property sets the amount that a spanner (in this case: the beam) in case of a line break extends beyond the rightmost column and extends to the left beyond the prefatory matter.



2



`'staccato-pos.ly'`

Some scripts must have quantized positions. VERTICAL position descend monotonously for a descending scale. The staccato dot is close to the notehead. If the head is in a space, then the dot is in the space next to it.



`'staff-halfway.ly'`

Staves can be started and stopped at command.



`'staff-line-positions.ly'`

The vertical positions of staff lines may be specified individually, by setting the `line-positions` property of the `StaffSymbol`.



`'staff-mixed-size.ly'`

Staves may be present in several sizes within a score. This is achieved with an internal scaling factor. If the scaling factor is forgotten in some places, objects generally become too thick or too large on smaller staves.



`‘staff-tweak.ly’`

The staff is a grob (graphical object) which may be adjusted as well, for example, to have 6 thick lines and a slightly large `staff-space`. However, beams remain correctly quantized.



`‘stanza-number.ly’`

Stanza numbers are put left of their lyric. They are aligned in a column.



`‘stem-direction-context.ly’`

Stem directions for notes on the middle staff line are determined by the directions of their neighbors.



`‘stem-direction.ly’`

Stems, beams, ties and slurs should behave similarly, when placed on the middle staff line. Of course stem-direction is down for high notes, and up for low notes.



`‘stem-shorten.ly’`

If note head is ‘over’ the center line, the stem is shortened. This happens with forced stem directions, and with some chord configurations.



`'stem-stemlet.ly'`

Stemlets are small stems under beams over rests. Their length can be set with `stemlet-length`.



`'stem-tremolo-forced-dir.ly'`



`'stem-tremolo-position.ly'`

Tremolos are positioned a fixed distance from the end of the beam. Tremolo flags are shortened and made rectangular on beamed notes or on stem-up notes with a flag. Tremolo flags are tilted extra on stem-down notes with a flag.



`'stem-tremolo-staff-space.ly'`

stem tremolo vertical distance also obeys staff-space settings.



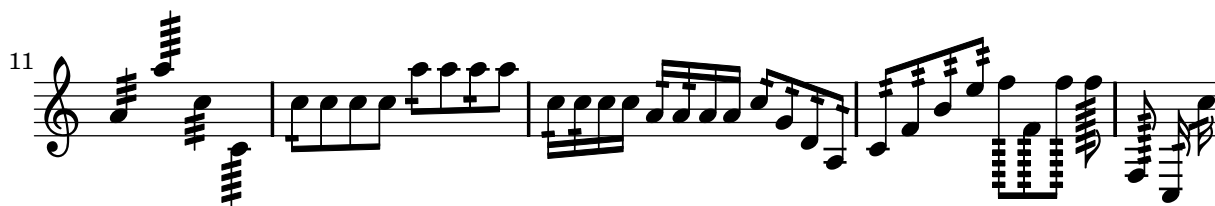
`'stem-tremolo.ly'`

Stem tremolos or rolls are tremolo signs that look like beam segments crossing stems. If the stem is in a beam, the tremolo must be parallel to the beam. If the stem is invisible (e.g. on a whole note), the tremolo must be centered on the note. If the note has a flag (eg. an unbeamed 8th note), the tremolo should be shortened if the stem is up and tilted extra if the stem is down.

The tremolos should be positioned a fixed distance from the end of the stems unless there is no stem, in which case they should be positioned a fixed distance from the note head.

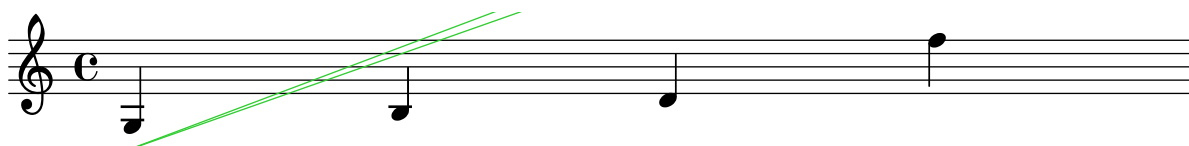
:4 :8 :16 :32 x :





`'stencil-color-rotation.ly'`

Combinations of rotation and color do work.



`'stencil-hacking.ly'`

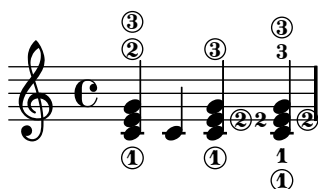
You can write stencil callbacks in Scheme, thus providing custom glyphs for notation elements. A simple example is adding parentheses to existing stencil callbacks.

The parenthesized beam is less successful due to implementation of the Beam. The note head is also rather naive, since the extent of the parens are also not seen by accidentals.



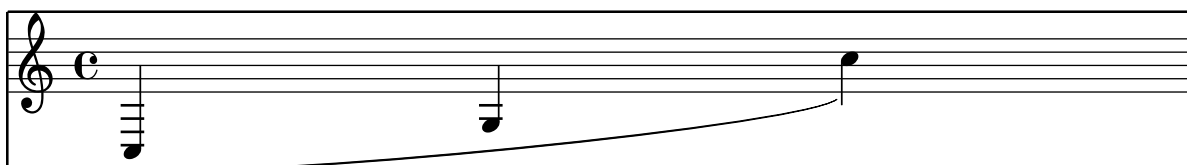
`'string-number.ly'`

String numbers can be added to chords. They use the same positioning mechanism as finger instructions.



`'system-extents.ly'`

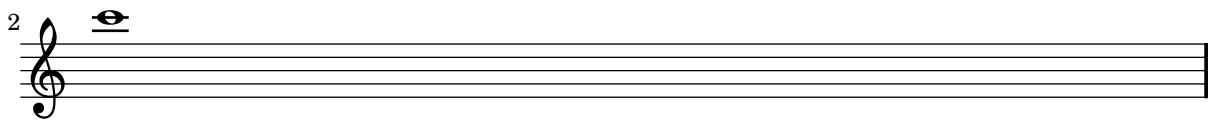
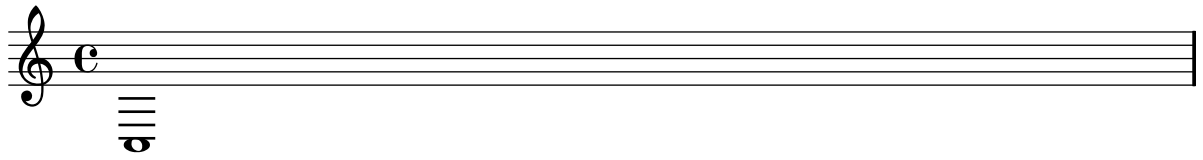
The size of every system is correctly determined; this includes postscript constructs such as slurs.



`'system-overstrike.ly'`

By setting `between-system-padding` to a negative value, it is possible to eliminate the anti-collision constraints. Then setting `between-system-space` to a low (nonzero) value, print systems in overstrike.

Unfortunately, this does not show in the collated texinfo document. Run this example stand-alone to see the effect.



`'system-separator.ly'`

System separators may be defined as markups in the `systemSeparator` field of the paper block. They are centered between the boundary staves of each system.

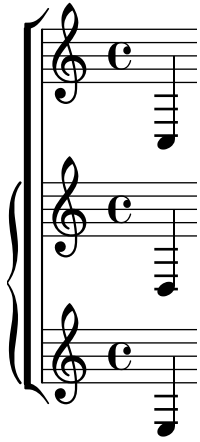
First system of a musical score. It consists of two staves joined by a brace on the left. Both staves are in common time (indicated by a 'C' time signature). The first measure of each staff contains a half note. The second measure of each staff is empty. A vertical bar line separates the two measures.

Second system of a musical score, marked with a double bar line and a fermata symbol. It consists of two staves joined by a brace on the left. The first measure of each staff contains a half note. The second measure of each staff is empty. A vertical bar line separates the two measures.

Third system of a musical score, marked with a double bar line and a fermata symbol. It consists of two staves joined by a brace on the left. The first measure of each staff contains a half note. The second measure of each staff is empty. A vertical bar line separates the two measures.

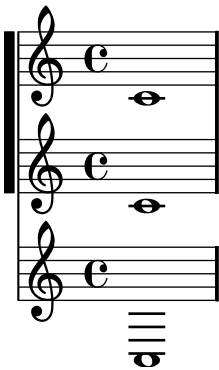
`'system-start-bracket.ly'`

The piano brace should be shifted horizontally if it is enclosed in a bracket.



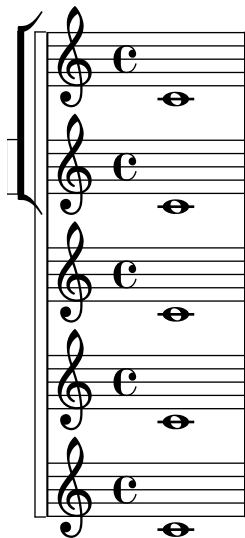
`'system-start-heavy-bar.ly'`

A heavy-bar system start delimiter may be created by tuning the `SystemStartBar` grob.



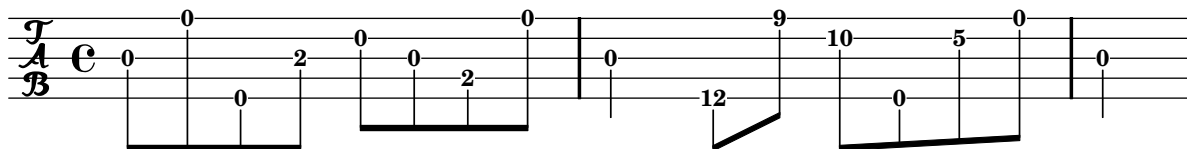
`'system-start-nesting.ly'`

Deeply nested system braces/brackets/etc. may be created with the `Nested_system_start_delimiter_engraver`



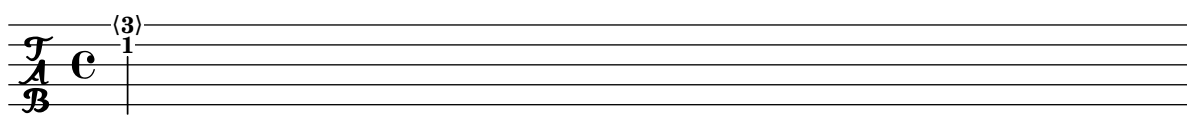
`'tablature-banjo.ly'`

Tablature may also be tuned for banjo.



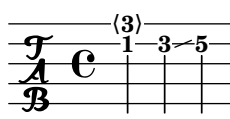
`'tablature-harmonic.ly'`

Harmonics get angled brackets in tablature



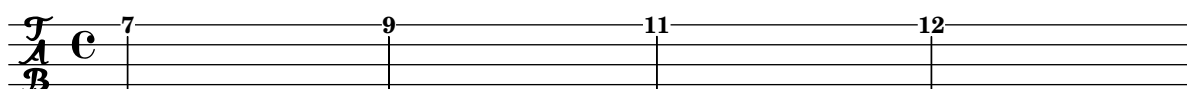
`'tablature-slide.ly'`

Tab supports slides.



`'tablature-string-tunings.ly'`

For other tunings, it is sufficient to set `stringTunings`. The number of staff lines is adjusted accordingly.

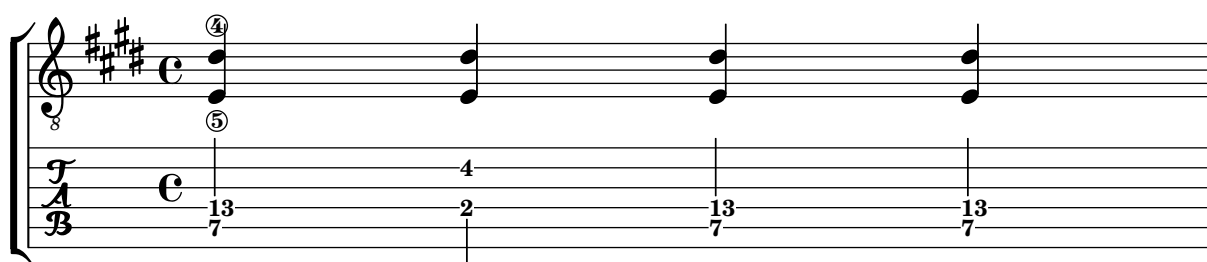


`'tablature.ly'`

A sample tablature, with both normal staff and tab.

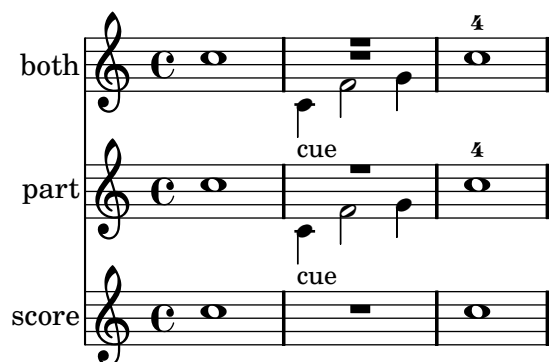
Tablature is done by overriding the note-head formatting function, and putting it on a 6-line staff. A special engraver takes care of going from string-number + pitch to number.

String numbers can be entered as note articulations (inside a chord) and chord articulations (outside a chord)



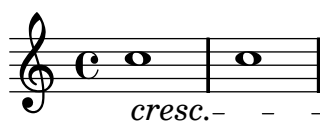
‘tag-filter.ly’

The `\tag` command marks music expressions with a name. These tagged expressions can be filtered out later. This mechanism can be used to make different versions of the same music. In this example, the top stave displays the music expression with all tags included. The bottom two staves are filtered: the part has cue notes and fingerings, but the score has not.



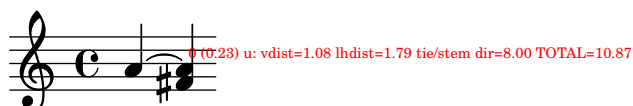
‘text-spanner.ly’

Text spanners should not repeat start text when broken.



‘tie-accidental.ly’

lilypond should flip the tie’s direction to avoid a collision with the sharp.



‘tie-arpeggio-collision.ly’

Advanced tie chord formatting also works with arpeggiated ties. Due to arpeggios, tie directions may be changed relative to the unarpeggiated case.



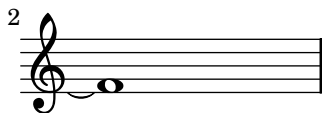
`'tie-arpeggio.ly'`

when `tieWaitForNote` is set, the right-tied note does not have to follow the left-tied note directly. When `tieWaitForNote` is set to false, any tie will erase all pending ties.



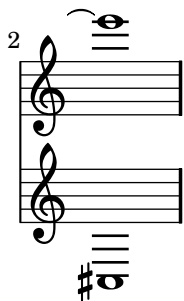
`'tie-broken-minimum-length.ly'`

Broken ties honor `minimum-length` also. This tie has a `minimum-length` of 5.



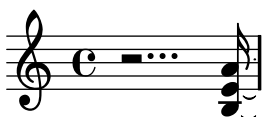
`'tie-broken-other-staff.ly'`

Broken tie lengths are not affected by clefs in other staves.



`'tie-broken.ly'`

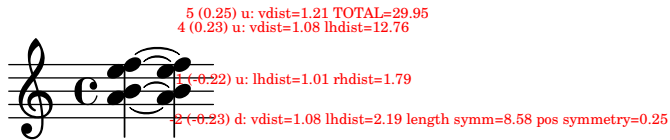
Ties behave properly at line breaks.





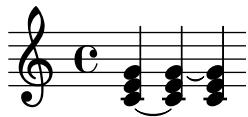
`'tie-chord-debug.ly'`

Switching on debug-tie-scoring annotates the tie scoring decisions made.



`'tie-chord-partial.ly'`

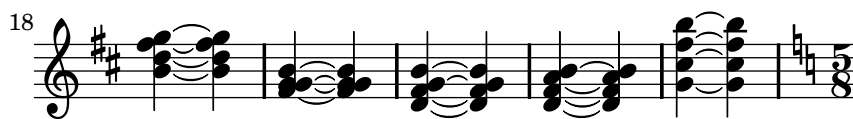
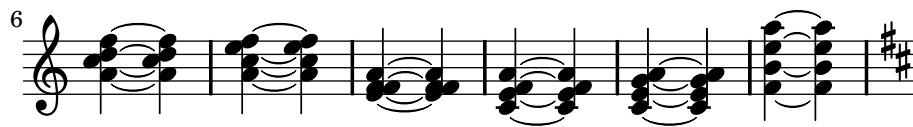
Individual chord notes can also be tied



`'tie-chord.ly'`

In chords, ties keep closer to the note head vertically, but never collide with heads or stems. Seconds are formatted up/down; the rest of the ties are positioned according to their vertical position.

The code does not handle all cases. Sometimes ties will printed on top of or very close to each other. This happens in the last chords of each system.





`'tie-direction-manual.ly'`

Tie directions can be set with `_` and `^`. This makes correction in complex chords easier.



`'tie-dot.ly'`

Ties avoid collisions with dots.



`'tie-grace.ly'`

Tieing a grace to the to a following grace or main note works.



`'tie-manual-vertical-tune.ly'`

If using integers, the tie will vertically tune for staff line avoidance. If using a floating point number, this is taken as the exact location.



`'tie-manual.ly'`

Tie formatting may be adjusted manually, by setting the `tie-configuration` property. The override should be placed at the second note of the chord.

You can leave a Tie alone by introducing a non-pair value (eg. `#t`) in the `tie-configuration` list.



`'tie-semi-single.ly'`

Like normal ties, single semities (`LaissezVibrerTie` or `RepeatTie`) get their direction from the stem direction, and may be tweaked with `#'direction`.



`'tie-single-chord.ly'`

Tie directions are also scored. In hairy configurations, the default rule for tie directions is overruled.



`'tie-single-manual.ly'`

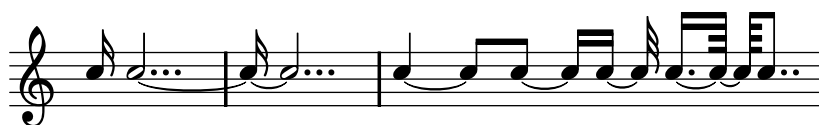
Individual ties may be formatted manually by specifying their `direction` and/or `staff-position`.



`'tie-single.ly'`

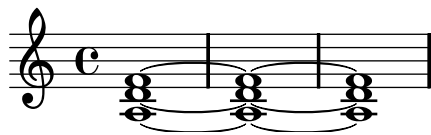
Formatting for isolated ties.

- short ties are in spaces
- long ties cross staff lines
- ties avoid flags of left stems.
- ties avoid dots of left notes.
- short ties are vertically centered in the space, as well those that otherwise don't fit in a space
- extremely short ties are put over the noteheads, instead of inbetween.



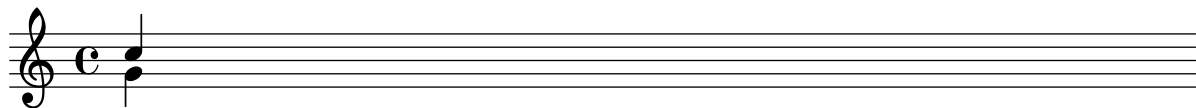
`'tie-whole.ly'`

For whole notes, the inside ties do not cross the center of the note head, horizontally.



`'to-xml.ly'`

The input representation is generic, and may be translated to XML.



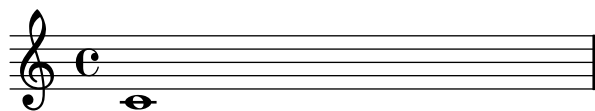
`'toc.ly'`

A table of contents is included using `\markuplines \table-of-contents`. The toc items are added with the `\tocItem` command.

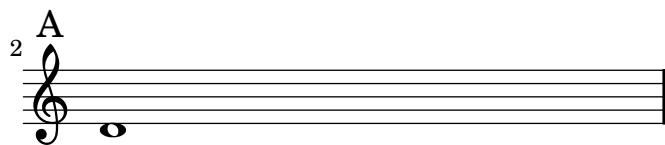
Table of Contents

The first score	2
Mark A	3
The second score	4

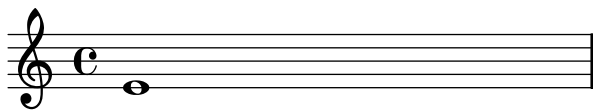
2



3



Second score



Music engraving by LilyPond 2.11.28—www.lilypond.org

`'trill-spanner-pitched.ly'`

Pitched trills are denoted by a small note head in parentheses following the main note. This note head is properly ledgered, and parentheses include the accidental.



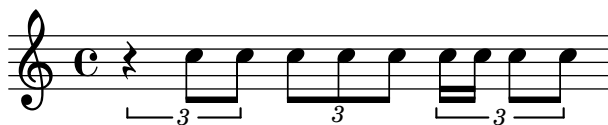
`'trill-spanner.ly'`

Trill spanner



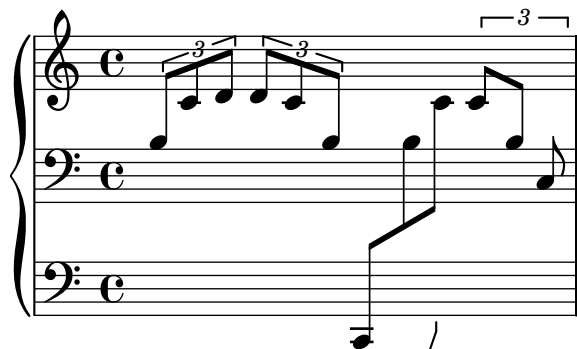
`'tuplet-beam.ly'`

In combination with a beam, the bracket of the tuplet bracket is removed. This only happens if there is one beam, as long as the bracket.



`'tuplet-bracket-cross-staff.ly'`

Cross-staff tuplets are drawn correctly, even across multiple staves.



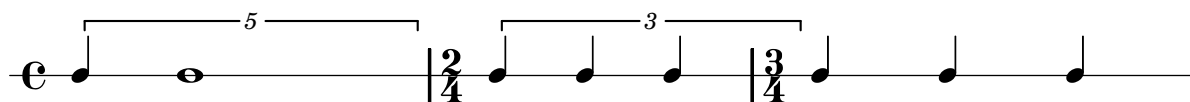
`'tuplet-broken.ly'`

Broken tuplets are adorned with little arrows. The arrows come from the `edge-text` property, and thus be replaced with larger glyphs or other text.



`'tuplet-full-length-note.ly'`

`tuplet` can be made to run to prefatory matter or the next note, by setting `tupletFullLengthNote`.



`'tuplet-full-length.ly'`

If `tupletFullLength` is set, tuplets end at the start of the next non-tuplet note.





`'tuplet-gap.ly'`

The size of the triplet bracket gap is adjusted to the width of the text.



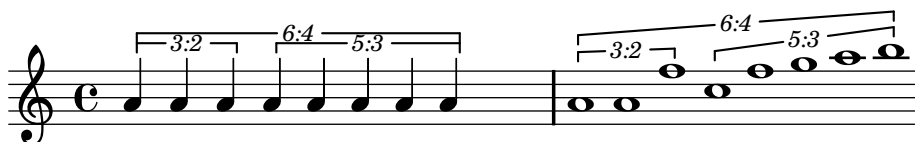
`'tuplet-nest-beam.ly'`

Nested triplets do collision resolution, also when they span beams.



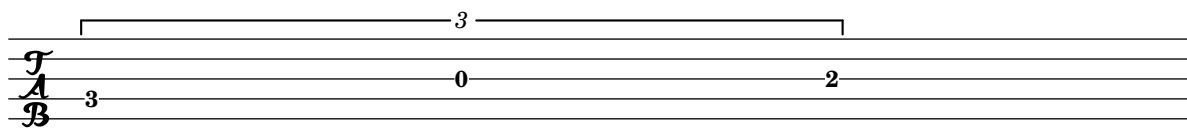
`'tuplet-nest.ly'`

Tuplets may be nested.



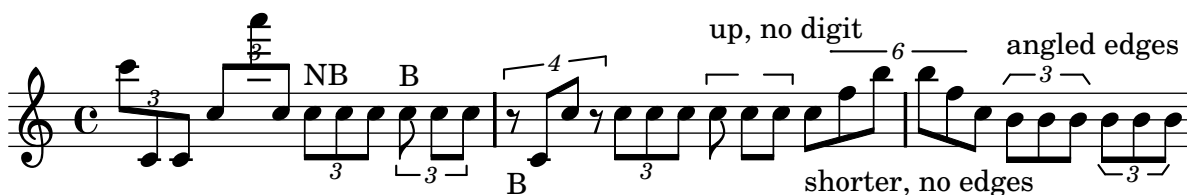
`'tuplet-no-stems.ly'`

Removing Stem_engraver doesn't cause crashes.



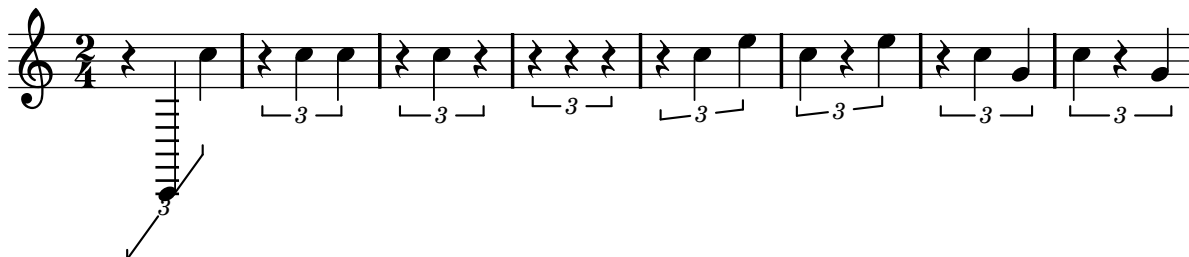
`'tuplet-properties.ly'`

Tuplet bracket formatting supports numerous options, for instance, bracketed (B) and non-bracketed (NB).



`'tuplet-rest.ly'`

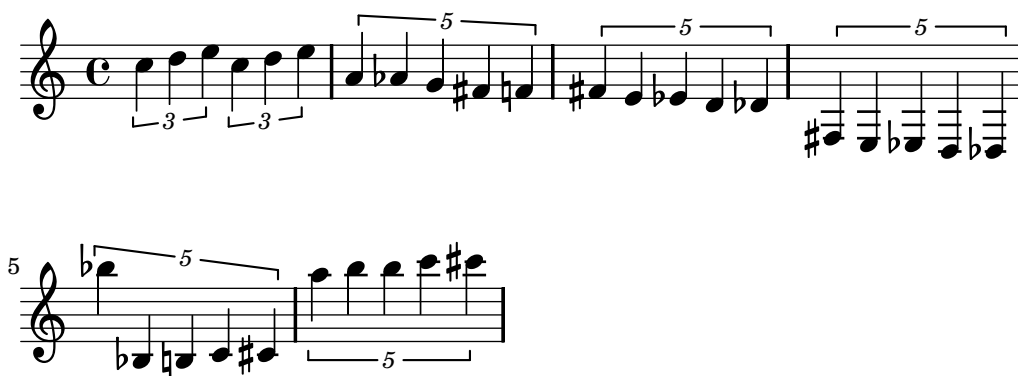
Tuplets may contain rests.



`'tuplet-slope.ly'`

Tuplet brackets stay clear of the staff. The slope is determined by the graphical characteristic of the notes, but if the musical pattern does not follow graphical slope, then the bracket is horizontal

The bracket direction is determined by the dominating stem direction.



`'tuplet-staffline-collision.ly'`

Horizontal tuplet brackets are shifted vertically to avoid staff line collisions.



`'tuplets.ly'`

Tuplets are indicated by a bracket with a number. There should be no bracket if there is a beam exactly matching the length of the tuplet. The bracket does not interfere with the stafflines, and the number is centered in the gap in the bracket.

The bracket stops at the end of the stems, if the stems have the same direction as the bracket. The endings can be adjusted with `bracket-flare`.



‘utf-8-mixed-text.ly’

words in mixed font in a single string are separated by spaces as in the input string. Here a Russian word followed by a roman word.

Hallo

‘utf-8.ly’

Various scripts may be used for texts (like titles and lyrics) introduced 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.



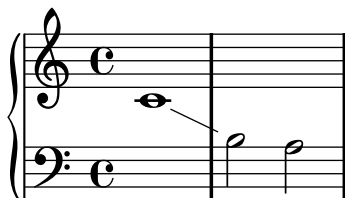
זֶה כִּי סתם לשמוע איך תנצח
いろはにほへど ちりぬるを わがよたれぞ つねならむ うみのおく やまけふこえて
à vo - cê uma can - ção



קֶרֶפֶד
あさきゆめみじ
legal

‘voice-follower.ly’

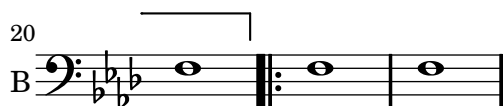
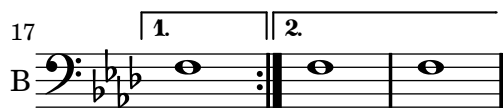
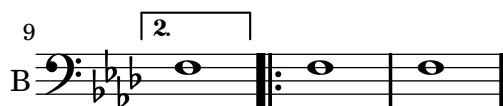
Whenever a voice switches to another staff a line connecting the notes can be printed automatically. This is enabled if the property `followVoice` is set to true.



‘volta-broken-left-edge.ly’

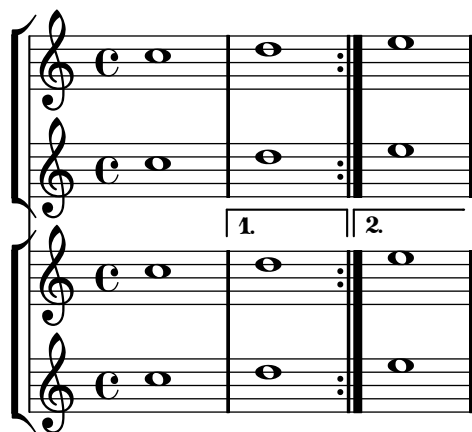
Broken volta spanners behave correctly at their left edge in all cases.





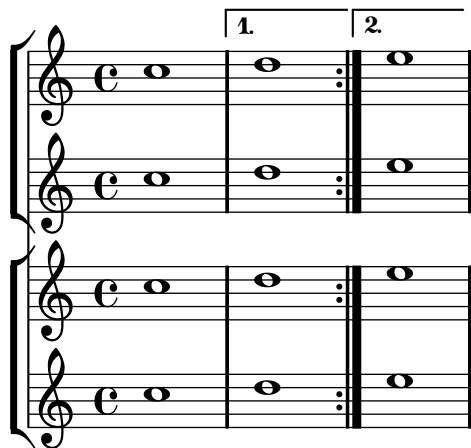
‘volta-multi-staff-inner-staff.ly’

By putting Volta_engraver in a staff context, one can get volta brackets on staves other than the topmost one.

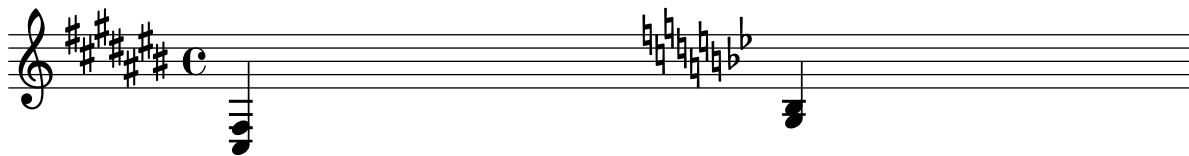


`‘volta-multi-staff.ly’`

By default, the volta brackets appear only in the topmost staff.



`‘warn-conflicting-key-signatures.ly’`



`‘whiteout.ly’`

The whiteout command underlays a white box under a markup. The whitening effect only is only guaranteed for staff lines, since staff lines are in a lower layer than most other grobs.

