LilyPond

The music typesetter

Changes

The LilyPond development team

This document lists changes and new features in LilyPond version 2.25.80 since 2.24.

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

If you are missing any manuals, the complete documentation can be found at `https://lilypond.org/`.

This document has been placed in the public domain.

For LilyPond version 2.25.80

Note: LilyPond releases can contain syntax changes, which may require modifications in your existing files written for older versions so that they work in the new version. To upgrade files, it is **strongly recommended** to use the `convert-ly` tool distributed with LilyPond, which is described in Section "Updating files with convert-ly" in *Application Usage*. `convert-ly` can perform almost all syntax updates automatically. Frescobaldi users can run `convert-ly` directly from Frescobaldi using "Tools > Update with convert-ly...". Other editing environments with LilyPond support may provide a way to run `convert-ly` graphically.

Table of Contents

Major changes in LilyPond **1**
Notes for source compilation and packagers 3

New for musical notation **4**
Pitches improvements 4
Rhythm improvements 4
Expressive mark improvements 8
Repeat improvements 9
Editorial annotation improvements 9
Text and font improvements 12

New for specialist notation **18**

Miscellaneous improvements **21**

Major changes in LilyPond

- Margins are now wider by default following the general layout of several publishers (and the recommendations of Elaine Gould).

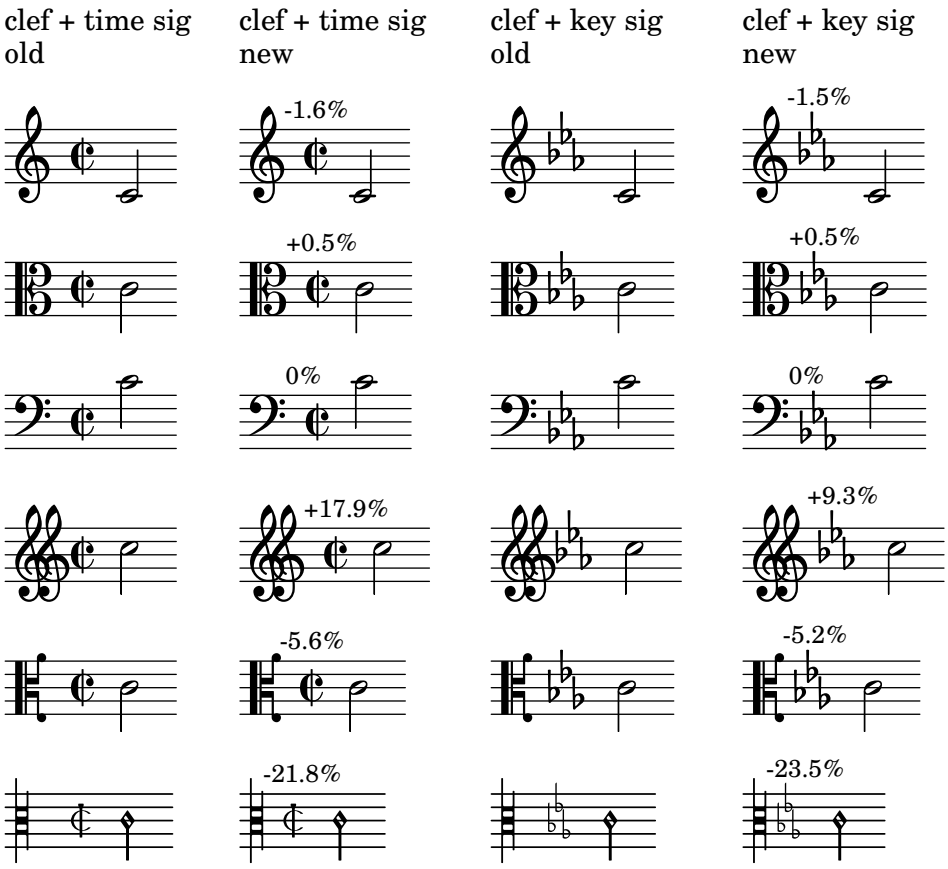
  In order to switch back to the previous settings (e.g., to keep the same layout when upgrading an existing score to version 2.25.80), add the following code:

```
\paper {
  top-margin = 5\mm
  bottom-margin = 6\mm
  top-system-spacing.basic-distance = 1
  top-markup-spacing.basic-distance = 0
  left-margin = 10\mm
  right-margin = 10\mm
  inner-margin = 10\mm
  outer-margin = 20\mm
  binding-offset = 0\mm
}
```

- Instead of generating PostScript or SVG output by itself, LilyPond can now use the Cairo library to produce its output. This is referred to as the 'Cairo backend', and can be turned on using the `-dbackend=cairo` command-line option. This works for all output formats (PDF, SVG, PNG, PostScript), and brings speed and rendering fidelity improvements in SVG output in particular. However, keep in mind that this backend does not yet implement all features of the default backends. Among the features not currently supported are PDF outlines, the `-dembed-source-code` option for PDF, and the `output-attributes` property for SVG.

- The distances between clefs and time signatures, together with the distances between clefs and key signatures, are now calculated differently. As a consequence, you will get better spacing for extra-wide clefs (like `\clef "GG"`) or extra-slim clefs (like `\clef "petrucci-c3"`).

  In the following image, both old and new positions are shown. The percentage gives the width difference of clef plus time signature and clef plus key signature, respectively.



Note that, as before, the widest clef in a staff group determines the horizontal position of all clefs in a system; this means, for example, that a piano score containing a treble and a bass clef doesn't change at all.

If you want to restore the previous default values for whatever reason, add

```
\override Staff.Clef.space-alist.time-signature =
          #'(minimum-space . 3.5)
\override Staff.Clef.space-alist.key-cancellation =
          #'(minimum-space . 3.5)
\override Staff.Clef.space-alist.key-signature =
          #'(minimum-space . 4.2)
```

to your score.

- The LilyPond major mode for the GNU Emacs text editor provided by the Emacs Lisp package `lilypond-mode.el` has been renamed from `LilyPond-mode` to `lilypond-mode`. The namespace prefix for the package is changed to all lowercase `lilypond-` from `LilyPond-`, meaning all its functions, variables, etc., now have the all-lowercase prefix. This change is made to comply with the idiomatic Emacs naming convention, most importantly, having the major mode name be the same as the package name that provides it. Therefore, this also makes it easier for new users to set up LilyPond in Emacs.

  As a result, this will break some existing Emacs configurations for end users. However, it only requires minimal changes to the Emacs initialization file. Here's an example of setting up `lilypond-mode` using the built-in `use-package` macro in Emacs:

```
(use-package lilypond-mode
  :ensure nil
  :mode "\\.\\(ly\\|ily\\)$")
```

Notes for source compilation and packagers

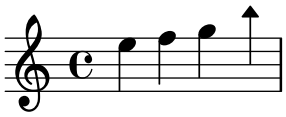
This section is aimed at enthusiasts compiling LilyPond from source and packagers preparing LilyPond for distribution. If you are not part of either group, you can skip over this section.

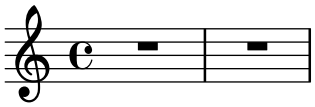
- LilyPond now requires Guile version 3.0.7 or newer. By default, `.scm` files are byte-compiled. If you don't want to do that for whatever reason, add `BYTECODE=no` as an argument to `make`.

New for musical notation

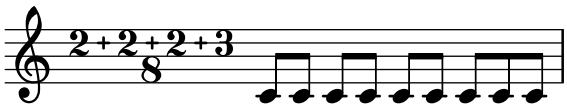
Pitches improvements

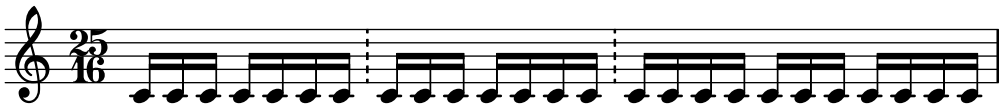
- The new \approximatePitch command creates notes of approximate pitch.

```
\relative c'' {
  e4 f g \approximatePitch c
}
```

- Certain spurious change clefs have been fixed.

```
{
  R1
  \clef treble
  R1
}
```

Rhythm improvements

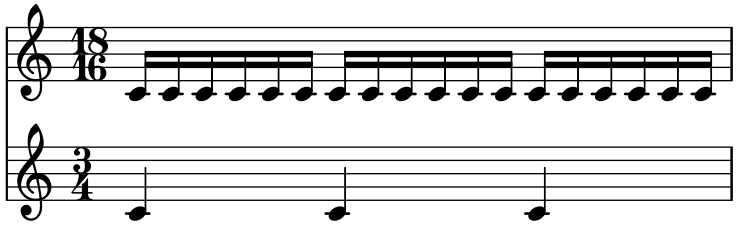
- The \time command now supports complex meters which previously required \compoundMeter, though it requires a slightly different syntax. The \compoundMeter command has been renamed to \timeAbbrev.

```
{
  \time #'((2 2 2 3) . 8)
  \repeat unfold 9 c'8
}
```

- The new \submeasureBarsOn command enables automatic bar lines between the components of strictly alternating time signatures (not shown), or where specified by setting the new submeasureStructure context property or grouping the beats in the optional argument to the \time command.

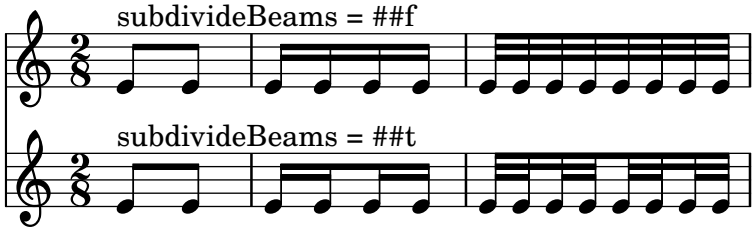
```
{
  \submeasureBarsOn
  \time #'((3 4) (3 4) (4 3 4)) 25/16
  \repeat unfold 25 c'16
}
```

- The new \polymetric command simplifies notating polymeter with aligned measures by setting scaled metric properties in contexts other than Timing.

```
<<
```

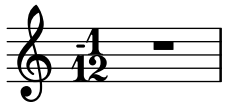
```
  \new Staff {
    \scaleDurations 4/6 {
      \context Staff \polymetric \time 6,6,6 18/16
      \repeat unfold 18 c'16
    }
  }
  \new Staff {
    \time 3/4
    c'4 4 4
  }
>>
```

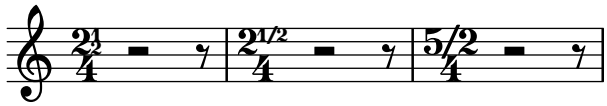


- The default auto-beaming settings for 2/8 time have been changed to allow beaming the full measure.



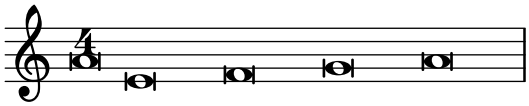
- `\enablePolymeter` has been renamed to `\enablePerStaffTiming` to avoid confusion because it is recommended in only a subset of polymetric use cases.
- The `TimeSignature.time-signature` property replaces the old `fraction` property. The new property is more general and can be overridden, which allows certain tricks without overriding the `stencil` property.

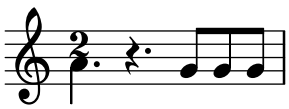
```
{
  \once \override Timing.TimeSignature.time-signature = #'(-1 . 12)
  R1
}
```

- The `\time` command now accepts a rational number in the numerator. The `TimeSignature` grob and `\compound-meter` markup function have new properties that control the style of fractional parts.

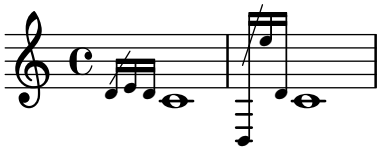


- The `\time` command now accepts a rational number in the denominator to support time signatures with a beat longer than a whole note.

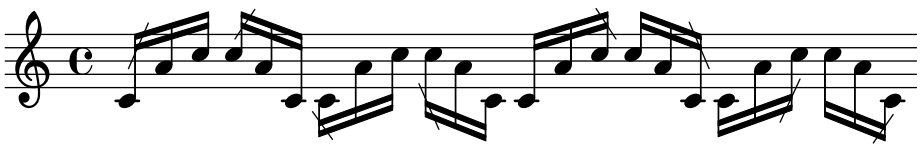
```
{
```

```
  \override Timing.TimeSignature.denominator-style = #'note
  \time #'(4 . 1/2)
  e'\breve f' g' a'
}
```

- The TimeSignature grob and \compound-meter markup function now support number-over-note time signatures via denominator-style and related properties.

```
{
  \override Timing.TimeSignature.style = #'numbered
  \override Timing.TimeSignature.denominator-style = #'note
  \once \override Timing.TimeSignature.time-signature = #'(2 . 8/3)
  \time 6/8
  r4. g'8 8 8
}
```

- \slashedGrace now prints slashed beams.

```
{
  \slashedGrace { d'16 e' d' } c'1
  \slashedGrace { d16 e'' d' } c'1
}
```

Use beam::slashed-stencil to slash arbitrary beams. Set details.slash-side to RIGHT to print the slash at the right side of the beam.

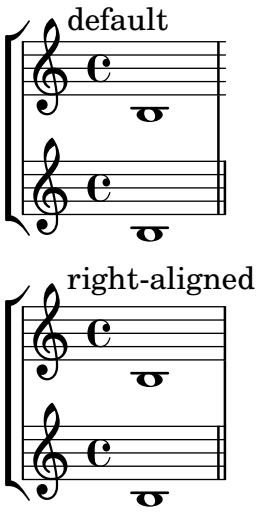
```
{
  \override Beam.stencil = #beam::slashed-stencil
  \cadenzaOn
  c'16^[ a' c'']
  c''^[ a' c']
  c'_[ a' c'']
  c''_[ a' c']
  \override Beam.details.slash-side = #RIGHT
  c'16^[ a' c'']
  c''^[ a' c']
  c'_[ a' c'']
  c''_[ a' c']
}
```

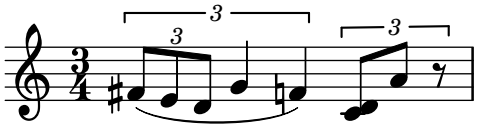


Use the over-beam-height, slash-slope, slash-side, slash-stem-fraction, slash-thickness, and slash-X-positions subproperties of the details property to customize the appearance of the slash.

- It is now possible to right-align different types of bar lines.

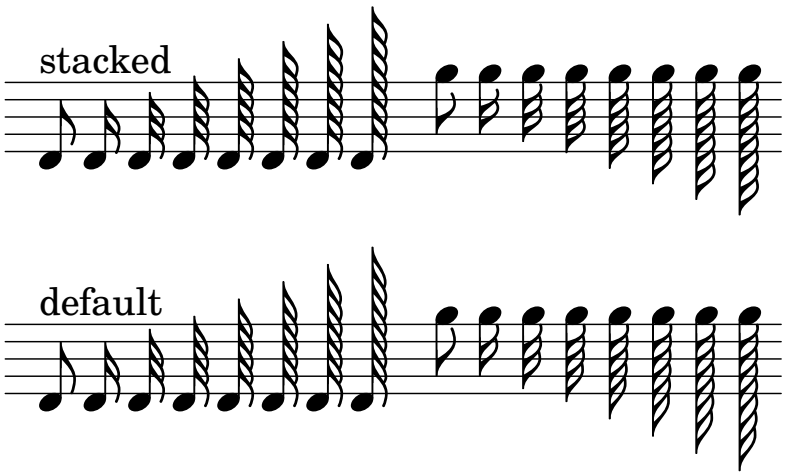
```
\new StaffGroup
  <<
    \new Staff { \textMark "default" b1 }
    \new Staff { b1 \section }
  >>

\new StaffGroup
  <<
    \new Staff
      { \textMark "right-aligned" b1 }
    \new Staff
      { b1
        \override StaffGroup.BarLine.right-justified = ##t
        \section }
  >>
```



- Bar checks (|) now implicitly create contexts. The developers deem this unlikely to impact real-world scores. Please report a bug if you find a problem without an obvious workaround.
- The new span-all-note-heads option may be used to make tuplet brackets span all note heads (not just the stems) as recommended by Gould and Ross.

- Automatic beam subdivision has been reworked. Previously, many imperfections could be found in the results of automatic subdivision of many complex beaming patterns due to overreliance of the value of beatBase. Now, LilyPond can correctly subdivide most beaming patterns and no longer uses the value of beatBase to limit beam subdivision. Simply setting subdivideBeams to #t now automatically subdivides all intervals by default. Three new properties have been introduced to tune automatic beam subdivision: beamMinimumSubdivision, beamMaximumSubdivision and respectIncompleteBeams. beamMinimumSubdivision limits subdivision intervals the same way as how beatBase

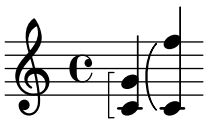
previously did (reducing frequency of subdivided beams). `beamMaximumSubdivision` limits the number of beamlets removed at subdivisions in general. `respectIncompleteBeams` limits the number of beamlets at subdivisions where the remaining length would not complete the metric value of the subdivision. Setting `beamMinimumSubdivision` to the value of `beatBase` at all times, even when `beatBase` implicitly changes, preserves old behavior.

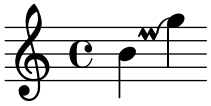
- New 'stacked' flag glyphs are available. All flag elements of a flag glyph have the same width but are vertically more compact.

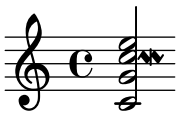
  Use `\flagStyleStacked` to access them; with `\flagStyleDefault` you can switch back to the standard flag style.

- The `TimeSignature` style `'single-digit` has been renamed to `'single-number`.

Expressive mark improvements

- The new commands `\chordSlur` and `\nonArpeggiato` support creating vertical slurs and brackets without abusing the `\arpeggio` command.

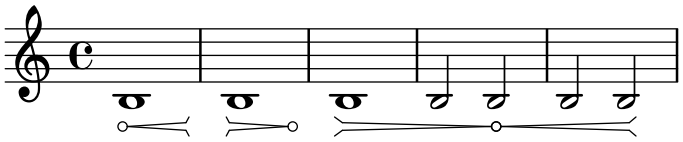
```
{ <c' g'>\nonArpeggiato <c' f''>\chordSlur }
```

- A new ornament, `bachschleifer`, is now available.

```
{ b' g''\bachschleifer }
```

- It is now possible to position `Script` to the left or right side of a `NoteHead`.

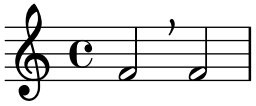
```
{ <c' g' c''\atRight \mordent e''>2 }
```

- Hairpins in the style of Ferneyhough now support *al niente* circles.

```
{
```

```
  \override Hairpin.circled-tip = ##t
  \override Hairpin.stencil = #flared-hairpin
  b1\< b\> b\> b2 b\< b2 b\!
}
```

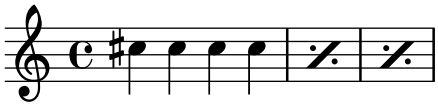
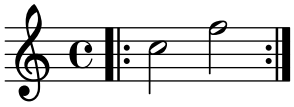


- Two new variant glyphs for breathing signs are available: 'laltcomma' and 'raltcomma'. These glyphs represent the old shapes of 'lcomma' and 'rcomma', respectively, before changing them to more common shapes.

```
{
  \override BreathingSign.text =
    \markup { \musicglyph "scripts.raltcomma" }
  f'2 \breathe f' |
}
```

- The wavy part of a `TrillSpanner` grob needs less vertical space now.

Repeat improvements

- The new music functions `\%` and `\*` serve as abbreviations for `\repeat percent` and `\repeat unfolded`.

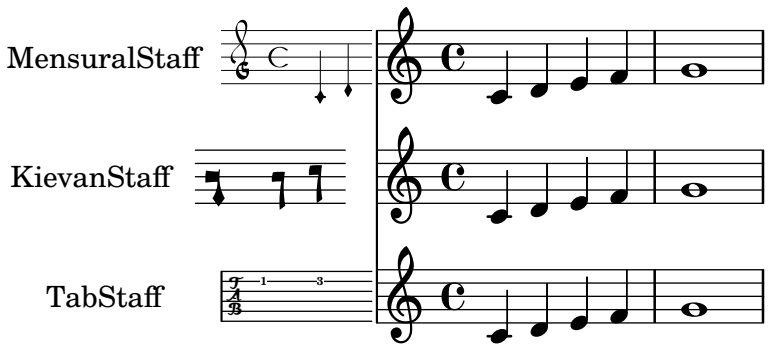
```
\%3 \*4 cis''4
```

- `\repeat volta` alternative endings no longer create invisible bar lines. This may affect line breaking, horizontal spacing, and `VoltaBracket` length where an alternative begins or ends without a bar line. In the case of an undesired change, try adding `\bar ""` or another command that creates a `BarLine` at that point.
- Using the new `printInitialRepeatBar` property, it is possible to print a start repeat bar line, automatically, at the beginning of the piece.

- The volta number position relative to the the volta bracket can now be adjusted with the `volta-number-offset` property of `VoltaBracket`.

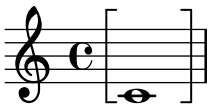
Editorial annotation improvements

- Incipits may now be printed with various context types. It also got an optional argument to shift the incipit to the left.

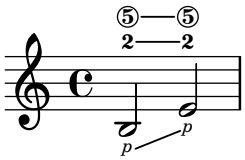
```
\score {
  <<
```

```
    \new Staff \with { instrumentName = "MensuralStaff" }
      {
        \incipit { c'4 d' }
        c'4 d' e' f' g'1
      }

    \new Staff \with { instrumentName = "KievanStaff" }
      {
        \incipit #2 \new KievanStaff { c'4 d' }
        c'4 d' e' f' g'1
      }
    \new Staff
      \with { instrumentName = "TabStaff" }
      {
        \incipit
          \new TabStaff
            \with { \magnifyStaff 0.5
                    \override InstrumentName.font-size = 6 }
            { c'4 d' }
        c'4 d' e' f' g'1
      }
  >>
  \layout {
    indent = 5\cm
    incipit-width = 2\cm
  }
}
```



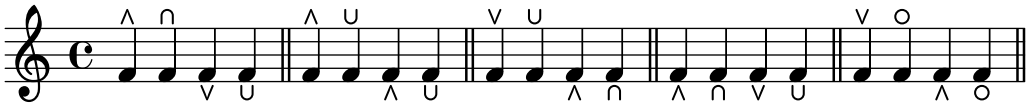
- Optional additional material can be enclosed in brackets that pass through the staff with `\startOptionalMaterial` and `\stopOptionalMaterial`.

- The `FingerGlideSpanner` may now connect `StringNumber` and `StrokeFinger` grobs.

```
{
  \set strokeFingerOrientations = #'(down)
  b2 \glide -\rightHandFinger #1 \glide -2 \glide \5
  e'-\rightHandFinger #1 -2 \5
}
```



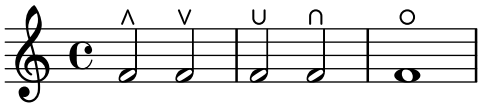
- A new engraver, `Toe_heel_engraver` (enabled by default) listens to the context variable `toeHeelStyle` to provide styling options for the commands `\rtoe`, `\ltoe`, `\rheel`, and `\lheel`.

```
music = { f'4\rtoe f'4\rheel f'4\ltoe f'4\lheel \bar "||" }

{
  \music            % #'default
  \set toeHeelStyle = #'standard    \music
  \set toeHeelStyle = #'reversed    \music
  \set toeHeelStyle = #'below       \music
  \set toeHeelStyle = #'circleheels \music
}
```



If the engraver is enabled, these commands no longer behave like normal `Script` grobs since they ignore direction changes (i.e., '^', '_', and `\tweak direction` are ignored). If the need arises to position pedal marks individually, the new, standard articulation commands `\toe`, `\vartoe`, `\heel`, `\varheel`, and `\heelcircle` (accessing the new glyph `scripts.pedalheelcircle`) should be used instead.

```
{
  f'2\toe f'2\vartoe |
  f'2\heel f'2\varheel |
  f'1\heelcircle
}
```

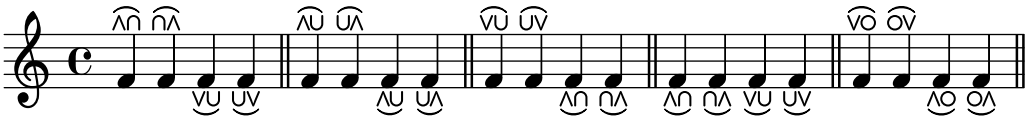


Removing the engraver makes `\rtoe` and siblings behave as in previous LilyPond versions.

- Directly related to the previous item are four new, articulation-like commands to indicate toe-heel and heel-toe substitutions, `\ltoeheel`, `\rtoeheel`, `\lheeltoe`, and `\rheeltoe`. They are influenced by the `Toe_heel_engraver` in the same way as `\rtoe` and siblings. Toe and heel glyphs have been harmonized in size to make these substitution marks look good.

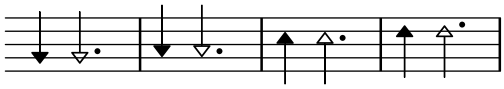
```
music = { f'4\rtoeheel f'4\rheeltoe
          f'4\ltoeheel f'4\lheeltoe \bar "||" }

{
  \music            % #'default
  \set toeHeelStyle = #'standard    \music
  \set toeHeelStyle = #'reversed    \music
  \set toeHeelStyle = #'below       \music
  \set toeHeelStyle = #'circleheels \music
```

```
}
```



- `NoteName` grobs are now horizontally centered by default.

Text and font improvements

- Triangular note heads have been added to the Emmentaler font. They can be used by overriding `NoteHead.style` to `arrow`.

- A new `\bar-line` markup command to print bar lines in text is now available.

```
\markup {
  \override #'(word-space . 2)
    \line {
      Examples
      \fontsize #-5 \translate-scaled #'(0 . 2) {
        \bar-line ":|."
        \bar-line ".|:"
        \bar-line ";!S!;"
        \bar-line "]{|}["
      }
    }
}
```

Examples

- The syntax for customizing text and music fonts has been changed. Instead of

```
\paper {
  #(define fonts
     (set-global-fonts
       #:music "Name of music font"
       #:brace "Name of music brace font"
       #:roman "Name of serif font"
       #:sans "Name of sans-serif font"
       #:typewriter "Name of typewriter font"))
}
```

or

```
\paper {
  #(define fonts
     (make-pango-font-tree
       "Name of serif font"
       "Name of sans-serif font"
       "Name of typewriter font"
       factor))
}
```

the new syntax is

```
\paper {
```

```
  property-defaults.fonts.music = "Name of music font"
  property-defaults.fonts.serif = "Name of serif font"
  property-defaults.fonts.sans = "Name of sans-serif font"
  property-defaults.fonts.typewriter = "Name of typewriter font"
}
```

Unlike the previous syntax, the new syntax does not interfere with font sizes, which should be set separately using `set-global-staff-size` or `layout-set-staff-size`.

There is no `brace` key in the fonts alist; braces glyphs now always default to the music font. However, it is still possible to override this by using an extra font family, as shown in this example (which requires the LilyJAZZ font):

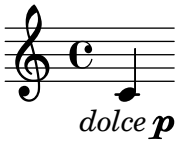
```
\layout {
  \context {
    \Score
    \override SystemStartBrace.fonts.music = "lilyjazz"
  }
}

\new PianoStaff <<
  \new Staff { c' }
  \new Staff { c' }
>>

\markup \override #'(fonts . ((music . "lilyjazz"))) \left-brace #20
```

Because `fonts` is simply a property, it is possible to override it on a per-grob basis, e.g.,

```
\layout {
  \override Score.SectionLabel.fonts.roman = "Custom font"
}
```

This is preferable over the already existing `font-name` property, since the latter makes commands such as `\bold` ineffective, instead requiring "Bold" to be included in the `font-name` string. Using `fonts` does not have such effects.

- The `\lookup` markup command can now only be used for braces; for other glyphs, use the `\musicglyph` command. Instead of `\lookup`, it is also generally recommended to use `\left-brace`.
- In markup, when a music font is used (such as for dynamic markings), a glyph absent from the music font was previously rendered in a normal text font. This is no longer the case; a warning about the missing glyph is output instead. In order to use a text font, use the `\serif`, `\sans`, or `\typewriter` markup commands. For example:

```
dolceP =
#(make-dynamic-script
  #{
    \markup {
      \serif \normal-weight dolce
      p
    }
  #})

{ c'\dolceP }
```

- Small caps are now achieved by overriding `font-variant` to `small-caps` instead of overriding `font-shape` to `caps`. Since `font-shape` is primarily for achieving italics, this change makes it possible to use small caps and italics at the same time.
- The `font-series` property is now more flexible and allows the specification of values such as `semibold` and `light` instead of only `normal` and `bold`.

  The `medium` value is now an intermediate value between `normal` and `bold` rather than an equivalent of `normal`. Accordingly, the `\medium` markup command has been renamed to `\normal-weight`.
- The new `font-stretch` property allows the selection of a condensed or expanded font.
- The text of a `VoltaBracket` grob, as set by `\override Score.VoltaBracket.text = ...` or `\set Score.repeatCommands = ...`, is no longer automatically typeset in a music font; use the `\volta-number` markup command on those parts that need to be. For example, convert

```
\set Score.repeatCommands = #'((volta "2, 5"))
```

  to

```
\set Score.repeatCommands =
  #`((volta ,#{ \markup {
                  \concat { \volta-number 2 , }
                  \volta-number 5 }
              #}))
```

- In markup, fingerings (`\markup \finger`) and bass figures (`\markup \figured-bass`) now get scaled along with normal text when using `\fontsize`.

```
myText = \markup {
  The fingering \finger { 5-4 } for a \figured-bass { 7 "6\\" } ...
}

\myText
\markup\fontsize #6 \myText
```

  The fingering 5-4 for a 7 6̸ …

  # The fingering 5-4 for a 7 6̸ …

  The previous behavior can be restored by setting the global variables `legacy-figured-bass-markup-fontsize` and `legacy-finger-markup-fontsize` to `#t`, respectively:

```
#(set! legacy-figured-bass-markup-fontsize #t)
#(set! legacy-finger-markup-fontsize #t)

myText = \markup {
  The fingering \finger { 4-5 } for a \figured-bass { 5+ 6 } ...
}

\myText
\markup\fontsize #6 \myText
```

The fingering 4-5 for a 5 6 …

The fingering 4-5 for a 5 6 …

- For best clarity, the `\roman` markup command has been renamed to `\serif`. Likewise, to cancel a setting of the `font-family` property to `sans` or `typewriter`, it should now be set to `serif`, not `roman`.
- The `\text` markup command has been removed. Instead, the `\serif`, `\sans`, or `\typewriter` markup commands should be used. These commands used to set the font style *only if a normal text font was used* (not a musical font, such as for dynamics), but now they *both* set the font style and make a normal text font used.
- The font size used by the `\volta-number` markup command has been reduced to make it better fit with surrounding text. At the same time, the size for the volta bracket number (which uses `\volta-number` for formatting by default) has been increased by the opposite amount, thus compensating for the change in the normal use case.
- Some glyphs like 'one' or 'accidentals.hufnagelM1' in the Emmentaler font had overly large glyph bounding boxes due to a bug in the font production chain. This is fixed now; be warned, however, that you might experience typesetting differences because tighter bounding boxes often lead to tighter typesetting.
- New markup commands to print the textual representation of accidentals are added: `\text-doubleflat`, `\text-flat`, `\text-natural`, `\text-sharp`, `\text-doublesharp` and the general `\text-accidental`.

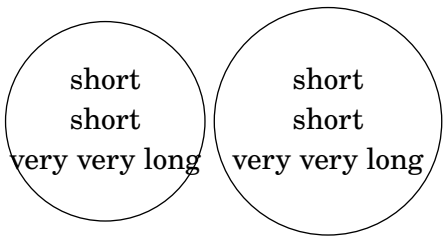
```
\markuplist \override #'(padding . 1) \table #'(-1 -1 -1 -1 -1 -1) {
  "Accidentals for text:"
   \text-doubleflat \text-flat \text-natural \text-sharp
   \text-doublesharp
  "Accidentals for music:"
  \doubleflat \flat \natural \sharp \doublesharp
}
```

Accidentals for text: 𝄫 ♭ ♮ ♯ 𝄪

Accidentals for music: 𝄫 ♭ ♮ ♯ 𝄪

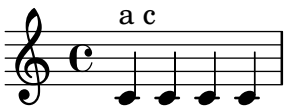
- The `\circle` markup command now listens to a `bbox` property. If set to `#t`, the bounding box of its argument is used for the circle's diameter.

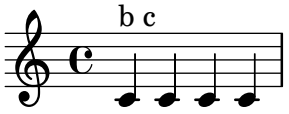
```
\markup {
  \circle {
    \center-column { "short" "short" "very very long" }
  }
  \override #'(bbox . #t) \circle {
    \center-column { "short" "short" "very very long" }
  }
}
```

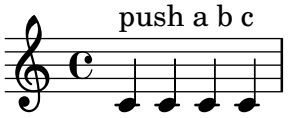


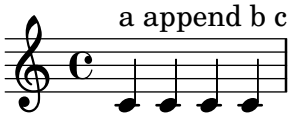
- The \tag command is now allowed in markup environments. New markup commands are added for filtering (\keep-with-tag and \remove-with-tag) and for inserting markups (\push-to-tag, \append-to-tag and \replace-with-tag) in markup context or from music context (\pushToTagMarkup, \appendToTagMarkup and \replaceWithMarkup).

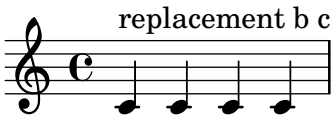
```
test = \markup {
  \tag #'a a
  \tag #'b b
  c
}

music = \relative {
  c'^\test c c c
}

\keepWithTag #'a \music
\removeWithTag #'a \music
\pushToTagMarkup #'a push \music
\appendToTagMarkup #'a append \music
\replaceWithTagMarkup #'a replacement \music

\markup { \keep-with-tag #'a \test }
\markup { \remove-with-tag #'a \test }
\markup { \push-to-tag #'a push \test }
\markup { \append-to-tag #'a append \test }
\markup { \replace-with-tag #'a replacement \test }
\markup { \keep-with-tag #'a \score { \music } }
```











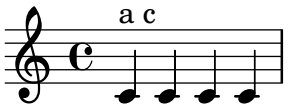
a c

b c

push a b c

a append b c

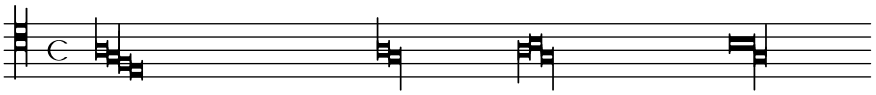
replacement b c



- The two new markup commands `\pad-x-left` and `\pad-x-right` behave similar to `\pad-x` but apply horizontal padding on one side of the markup only.

New for specialist notation

- For consistency with other ancient clefs, five new mensural clefs are available: "mensural-f2", "mensural-f3", "mensural-f4" (same as "mensural-f"), "mensural-f5", "mensural-g1", "mensural-g2" (same as "mensural-g").
- The default time signature and accidental style in a PetrucciStaff context is now the same as in MensuralStaff.
- White mensural ligatures now support some rare ligatures (semibreves alone or in the middle), and allow tweaks to show some non-necessary stems.

```
\score {
  \relative {
    \set Score.timing = ##f
    \set Score.measureBarType = #'()
    \override NoteHead.style = #'petrucci
    \override Staff.TimeSignature.style = #'mensural
    \clef "petrucci-c4"
    \[ a1 g f e \]
    \[ a1 g\longa \]
    \[ \tweak left-down-stem ##t a\breve b
       \tweak right-down-stem ##t g\longa \]
    \[ \tweak right-down-stem ##t b\maxima
       \tweak right-up-stem ##t g\longa \]
  }
  \layout {
    \context {
      \Voice
      \remove Ligature_bracket_engraver
      \consists Mensural_ligature_engraver
    }
  }
}
```

- The use of the file gregorian.ly is deprecated. While still distributed for backward compatibility, it should be replaced with a VaticanaScore context together with some manual \layout changes (if necessary): code like

```
\include "gregorian.ly"

\score {
  \new VaticanaStaff { ... }
}
```

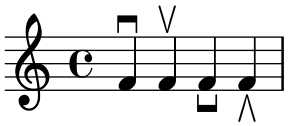
should become

```
\new VaticanaScore {
  \new VaticanaStaff { ... }
}

\layout {
  indent = 0
  ragged-last = ##t
```

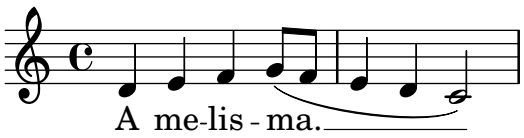
```
}
```

- LilyPond's 'arabic' note name language is deprecated. While still working for backward compatibility (if you load `hel-arabic.ly`), it is recommended to use 'english', 'italiano', or your preferred note name language instead.
- Defaults for fret labels in fret diagrams have changed.
  - The default value for `fret-label-vertical-offset` is set to -0.5, which centers the label in the fret space.
  - The default number format is now `'custom`, with a format string of `"~dfr"` (resulting in '3fr', for example), instead of `'roman-lower`.
- Two new options are available for `fret-diagram-details`:
  - `string-overhang` allows to define the extension of string lines beyond the last fret line, in multiples of `fret-distance`. Default value 1.
  - `barre-thickness` defines the thickness of the barre line, in multiples of `dot-radius`. Only defined for `barre-type=straight`. Default value 1.
- The command `\autoBeamOff` now stops auto-beaming immediately. Previously, its effect was delayed if a beam generated by the auto-beamer engraver was still active.
- It is no longer necessary to switch off auto-beaming while using `\crossStaff`.
- The `\upbow` and `\downbow` signs are now properly inverted if used below a staff. As a consequence, the glyphs `scripts.upbow` and `scripts.downbow` have been renamed to `scripts.uupbow` and `scripts.udownbow`, respectively, and the new inverted glyphs are named `scripts.dupbow` and `scripts.ddownbow`.

```
\relative c' {
  f4^\downbow f^\upbow f_\downbow f_\upbow
}
```



- In lyrics aligned to a melody, extender lines can now be added automatically for words ending in a melisma.

```
\relative {
  d'4 e f4 g8( f
  e4 d c2)
}
\addlyrics {
  \set autoExtenders = ##t
  A me -- lis -- ma.
}
```

- By default, chord names are now always shown with English pitch names, no longer dependent on `\language`. Similarly, commands like `\germanChords` are no longer dependent on `\language` either.

  This change might cause differences in displayed chord names.
- A new command `\englishChords` is now available, making it possible to switch back to English chord names.

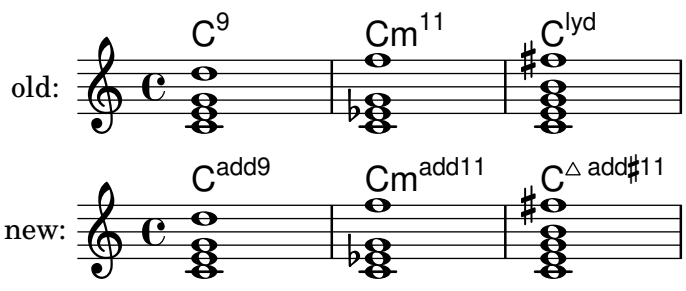
- Some Scheme functions related to chord names have been renamed.

old	new
note-name->string	pitch->name
alteration->text-accidental-markup	accidental->text-markup
note-name->markup	chord-name:markup
chord-name->german-markup	chord-name:german-markup
note-name->german-markup	chord-name:german-lowercase-name-markup
chord-name->italian-markup	chord-name:italian-markup

  `convert-ly` handles this automatically.

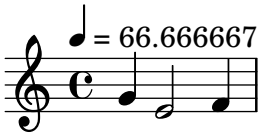
- Two new Scheme functions related to chord names are now available: `pitch->alteration` returns the accidental for a given pitch and language; `chord-name:name-markup` returns note name markup for a given pitch and language.
- LilyPond now follows the Ignatzek chord name scheme more closely. This reverts changes introduced in version 2.15.21 (in 2011), which are now considered as incorrect and/or ambiguous.

```
oldException = { <c e g b fis'>-\markup \super "lyd" }
chExceptions = #(append (sequential-music-to-chord-exceptions
                         oldException #t)
                        ignatzekExceptions)

music = { <c e g d'>1 | <c es g f'> | <c e g b fis'> }

\new StaffGroup \with { instrumentName = "old:" }
<<
  \new ChordNames \with {
    chordNameExceptions = #chExceptions
    additionalPitchPrefix = ""
  } \music
  \new Voice \transpose c c' \music
>>

\new StaffGroup \with { instrumentName = "new:" }
<<
  \new ChordNames \music
  \new Voice \transpose c c' \music
>>

\layout {
  indent = 1\cm
  line-width = 9\cm
  ragged-right = ##f
}
```

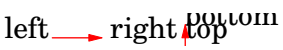


Miscellaneous improvements

- The `\tempo` command now accepts exact rational values. The default formatter rounds to the nearest 0.25; the context property `tempoCountPrecision` controls this.

```
\fixed c' {
  \once \set Timing.tempoCountPrecision = #1/1000000
  \tempo 4 = #200/3
  g4 e2 f4
}
```

- The new Scheme function `ly:parser-define-once!` defines a variable if it is not already defined.
- The `timeSignatureFraction` context property is deprecated in favor of `timeSignature`. The new property accepts subdivided and alternating time signatures in addition to the simple fractions that the old property accepted. Accessing the old property transparently derives a simple fraction from the new property and triggers a warning.
- `(make-music 'Music)` and `(make-music 'Event)` now create music expressions that have the `void` property set to `#t` by default. The function `empty-music` has been removed in favor of `(make-music 'Music)`. The new predicate `unspecified-music?` recognizes any music expression with the `void` flag set.
- The new Scheme function `ly:number->duration` converts a duration expressed in units of whole notes to a `ly:duration`. The log, number of dots, and scaling factor are chosen automatically.
- The markup commands `\hspace` and `\vspace` are used to insert space, which may be negative. Other objects are moved accordingly. This movement may be indicated by arrows printed by the new markup command `annotate-moving`. Note that the arrows do not reflect the actual extents of the objects created by `\vspace` and `\hspace`; you might use `\box` for that.

```
\markup {
  left \annotate-moving \hspace #4 right
  \column { top \annotate-moving \vspace #-4/3 bottom }
}
```



- The new `\contextPropertyCheck` command verifies that a property is set to an expected value or is unset in a specific context.
- Various context properties that previously held `ly:moment` values now hold rational numbers. To ease this transition, each number property is paired with a fallback `ly:moment` property. Using the fallback property transparently accesses the number property and triggers a warning.

Deprecated Context Property	**New Context Property**
`baseMoment`	`beatBase`
`completionUnitAsMoment`	`completionUnit`
`gridIntervalAsMoment`	`gridInterval`
`measureLengthAsMoment`	`measureLength`
`minimumPageTurnLength`	`pageTurnMinimumRestLength`

minimumRepeatLengthForPageTurn	pageTurnMinimumRepeatLength
proportionalNotationDurationAsMoment	proportionalNotationDuration
tempoWholesPerMinuteAsMoment	tempoWholesPerMinute
tupletSpannerDurationAsMoment	tupletSpannerDuration
voltaSpannerDurationAsMoment	voltaSpannerDuration

- `\tempo` *dur* = *min* - *max* used to set the context property `tempoWholesPerMinute` after rounding the center of the range to the nearest integer. It no longer rounds. This is expected to change the MIDI tempo in certain cases.
- The Scheme functions `duration-length` and `ly:duration-length` are renamed to `ly:duration->number` and `ly:duration->moment`, respectively.
- The new Scheme function `ly:parser-append-to-include-path` appends its argument to the current parser's include path.
- The value of the `timeSignatureSettings` context property uses the symbol `beatBase` where it formerly used `baseMoment`. `convert-ly` does not address this change.
- Embedding PNG images is now supported using the new `\image` markup command. This supplements the existing `\epsfile` command for EPS images.

  `\image` works for both PNG and EPS images. For EPS images, the difference between using `\image` and `\epsfile` is that `\image` adds a white background by default, while `\epsfile` does not.
- The new `\qr-code` markup command inserts a QR code of the specified size for the specified URL. This can be used to link to, e.g., the website of a composer or publisher, the LilyPond source files for the score, recordings, etc.

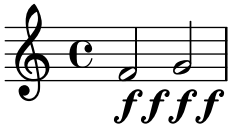
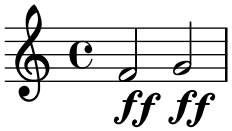
  ```
  \markup \qr-code #10 "https://lilypond.org"
  ```

- Parentheses for text markup have been added to the Emmentaler fonts, which harmonize with digits (in contrast to the already available parentheses for accidentals).
- Glyphs for a figure-dash (U+2012), an en-dash (U+2013), and a slash (U+002F) have been added to the Emmentaler fonts.
- A figure space (U+2007), a thin space (U+2009), and a hair space (U+200A) have been added to the Emmentaler fonts.
- The `-dinclude-settings` option can now be given multiple times to include several stylesheets.
- In the LaTeX backend of `lilypond-book`, all inline images are now vertically shifted. The amount can be controlled globally with command-line option `--inline-vshift` and locally with an argument to the snippet option `inline`.
- It is possible to use various command-line options of `musicxml2ly` as options for `lilypond-book`'s `musicxmlfile` command. This was actually introduced in version 2.15.9 (in 2011) but stayed undocumented until now.
- Two new command-line options `-dfirst` and `-dlast` have been introduced; they are equivalent to setting `showFirstLength` and `showLastLength`, respectively, in a LilyPond input file. For example, saying

  ```
  lilypond -dlast=R1*5 ...
  ```

  makes LilyPond render only the last five measures (assuming a 4/4 time signature).

- A visual index of all LilyPond graphical objects (grobs) is now available as a manual. This is based on Joram Berger's work for LilyPond 2.19 (`https://github.com/joram-berger/visualindex`).
- LilyPond provides support for in-notes, i.e., footnote-like annotations between music systems. This isn't new (it has actually been available since version 2.15.17, published in 2011) but it had some flaws and wasn't documented until now.
- The `lilysong` script has been removed. Besides lacking any documentation, it hasn't been maintained for a long time. Additionally, it has been using an external speech synthesis program called `festival`, which is no longer maintained either.
- Two new spacing styles are available for the `space-alist` grob property: `shrink-space` and `semi-shrink-space`; these spaces only shrink and don't stretch. They are also used directly in LilyPond, improving the formatting of tightly spaced staves.
- The `lilypond` binary has a new command-line option `-dstaff-size` to set the global staff size, equivalent to setting `set-global-staff-size` in a LilyPond input file.
- Instead of the functions `\bookOutputName` and `\bookOutputSuffix` we now recommend using the paper variables `output-filename` and `output-suffix` (which are not new but stayed undocumented until now). While the former will work unchanged, the latter is more coherent and easier to understand, especially if combined with predefined paper variables.
- The `Stem.details.lengths` property now also accepts pairs as list elements, allowing to set the length for up and down stems separately.
- The `ly:self-alignment-interface::aligned-on-x-parent` function (used by many grobs to compute the x-offset) now listens to a new `PaperColumn` property called `X-alignment-extent`. Set by default, it provides a fallback width for the `PaperColumn` grob in case it doesn't contain note heads. This helps align dynamic scripts that are attached to spacer rests, for example.

```
music =
  \new Staff <<
    { f'2 g'2 }
    { s4\f s\f s\f s\f }
  >>

\score {
  \music
}

\score {
  \music

  \layout {
    \context {
      \Score
      \override PaperColumn.X-alignment-extent = ##f
    }
  }
}
```

- BassFigureContinuation grobs now support horizontal-line-spanner-interface; the padding property has been replaced with the corresponding subproperties in bound-details.
- The \align-on-other markup command now accepts #f as a value for the alignment, indicating a markup's reference point.
- A new function \withRelativeDir is now available for markup commands that include files, and where such files should be found relative to the input file. Example:

```
\markup { \image #X #3 \withRelativeDir "test.png" }
```

- The positioning of horizontal (analysis) brackets has been improved; in particular, the HorizontalBracket grob now has an outside-staff-priority value of 800. As a consequence, however, nested horizontal brackets might be positioned differently than before. You can fix this by adjusting outside-staff-priority values with \tweak (where the outermost bracket should get the highest priority value).
- A new Scheme function to-staff-space is provided to convert absolute dimensions (in various units) to staff-space units. Examples:

```
top-markup-spacing.basic-distance = #(to-staff-space 2 'cm)

% default unit is pt
\markup
  \override #`(baseline-skip . ,(to-staff-space 20))
  \column {
    foo
    bar
  }
```

- Two new markup functions \abs-hspace and \abs-vspace are available to provide absolute dimensions that stay the same regardless of the current staff size.
- The data emitted by the command-line option -dshow-available-fonts is now sent to standard output.
- The function ly:font-config-display-fonts now takes an optional argument for output port selection.
- Scheme command-line option handling is now more robust. In the course of the new implementation, some minor changes were necessary.
  - On the command line, the argument for the -dpaper-size option no longer needs to be extra-quoted, i.e., a call like -dpaper-size=a3 works just fine.
  - The pixmap-format option now expects a string as a value, not a symbol. No change on the command line, but a call like

    ```
    #(ly:set-option 'pixmap-format 'pngalpha)
    ```

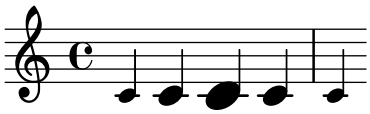
    must be changed to

    ```
    #(ly:set-option 'pixmap-format "pngalpha")
    ```

    The same holds for options separate-page-formats and tall-page-formats. Note that convert-ly can handle this automatically.
- The side-position-interface now offers two new properties, X-padding and minimum-X-space, to control the horizontal padding and minimum distance to a grob's parent object, independently of the vertical padding and minimum distance. This is useful

for grobs like `Fingering` that can be attached either horizontally or vertically to note heads, and which need different padding values for the X- and Y-axis, respectively.

- `\pushContextProperty` and `\popContextProperty` are two new commands for manipulating context properties. The first one pushes the current value to a stack, while the second one pops off the value from the stack and uses it to restore the property.

```
{
  c'
  \pushContextProperty Staff.fontSize
  \set Staff.fontSize = 3
  c'
  \pushContextProperty Staff.fontSize
  \set Staff.fontSize = 6
  c'
  \popContextProperty Staff.fontSize
  c'
  \popContextProperty Staff.fontSize
  c'
}
```

- A new grob property `whiteout-color` is available to set the color used for white-out operations. Similarly, the `\whiteout` markup command now listens to a `color` property to do the same.
- The `musicxml2ly` script has been completely revised to better convert MusicXML to LilyPond. It now supports more MusicXML features and elements and tries to be more faithful in the conversion, retaining more of the original appearance if possible (and sensible).

  Here is a list of some notable changes (not taking bug fixes into account).

  – The script is now much faster.
  – Overlapping and nested slurs work as expected.
  – Better automatic support of circumventing LilyPond's infamous issue #34 (i.e., synchronization of grace note timing between staves at the beginning of a piece).
  – Support of two-stem tremolos and nested tuplets.
  – Color and font size support for almost all MusicXML elements. A new command-line option `--dynamics-scale` helps compensate glyph size differences of dynamics signs in various music fonts.
  – Correct handling of ottava endings for MusicXML files created by Finale. The script looks at the `<software>` element to detect the used notation software. If necessary, this can be overridden with the new command-line option `--ottavas-end-early`.
  – Option `--shift-meter` has been replaced with option `--shift-durations`, using a simpler syntax.
  – New command-line option `--no-tagline` to suppress the output of a LilyPond tagline.
  – New command-line option `--book` to wrap the top-level score with `\book`.
  – If `<credit>` elements are present, they now fill the fields in LilyPond's `\header` block instead of metadata elements like `<work-title>` or `<creator>`.
  – New command-line option `--credit-page` to specify the page from which LilyPond should take `<credit>` data to fill the `\header` block.

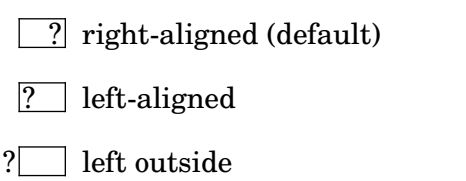
  - All metadata elements that are not mapped to standard \header fields (and which are not ignored like the 'page number' type) are output with more consistent field names: the prefix 'id: ' is used for <identification> children, 'credit: ' for <credit> elements, and the MusicXML element name (without a prefix) in all other cases. As a consequence, some (unused) fields got new names.

old	new
movementnumber	movement-number
encodingsoftware	id: software
encodingdate	id: encoding-date
encoder	id: encoder
encodingdescription	id: encoding-description
source	id: source

  - The bar numbers shown as comments in the created LilyPond input file are now as expected, i.e., the values shown refer to the previous bar.

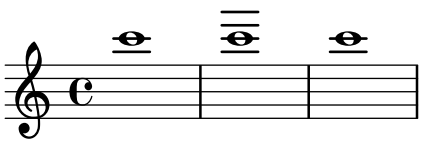
- The markup command \page-ref has a new property x-align to control the X-alignment of the gauge replacement.

```
\markup {
  \box
    \page-ref #'foo "???" "?" " right-aligned (default)"
}
\markup {
  \box
    \override #`(x-align . ,LEFT)
    \page-ref #'foo "???" "?" " left-aligned"
}
\markup {
  \box
    \override #'(x-align . -2.5)
    \page-ref #'foo "???" "?" " left outside"
}
```



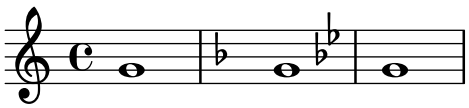
- It is now also possible to set the ledger-extra property via the NoteHead grob.

```
\score {
  {
    c'''1
    \tweak ledger-extra #4 c'''1
    c'''1
  }
}
```



- For grob NonMusicalPaperColumn, the subproperty bottom-space of the line-break-system-details property has been renamed to bottom-padding. While available since version 2.13.4 (released in 2009) it wasn't documented before.

- It is now possible to modify tag groups created with `\tagGroup` or `\tagGroupRef` by using the commands `\resetTagGroups`, `\resetTagGroup`, `\addToTagGroup`, and `\removeFromTagGroup`.
- The current input language (as selected with `\language`) can be accessed in the new Scheme variable `input-language` (as a symbol).
- The new function `\breakAlignInsert` makes it easy to change the order of notational elements like clefs, time signatures, bar lines etc. Previously, this required re-defining the `Score.BreakAlignment.break-align-orders` vector.

```
{
  g'1
  \key f \major
  g'
  \breakAlignInsert staff-bar after key-signature
  \key g \minor
  g'
}
```

- The output option `-dcrop` now takes all stencils into account while calculating the bounding box. This can change the size of cropped output.
- These functions, which have been deprecated since version 2.13.19, have been removed: `\deprecatedcresc`, `\deprecatedendcresc`, `\deprecateddim`, and `\deprecatedenddim`.