LilyPond

The music typesetter

Changes

The LilyPond development team

This document lists changes and new features in LilyPond version 2.25.15 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/`.

For LilyPond version 2.25.15

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.

Major changes in LilyPond

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

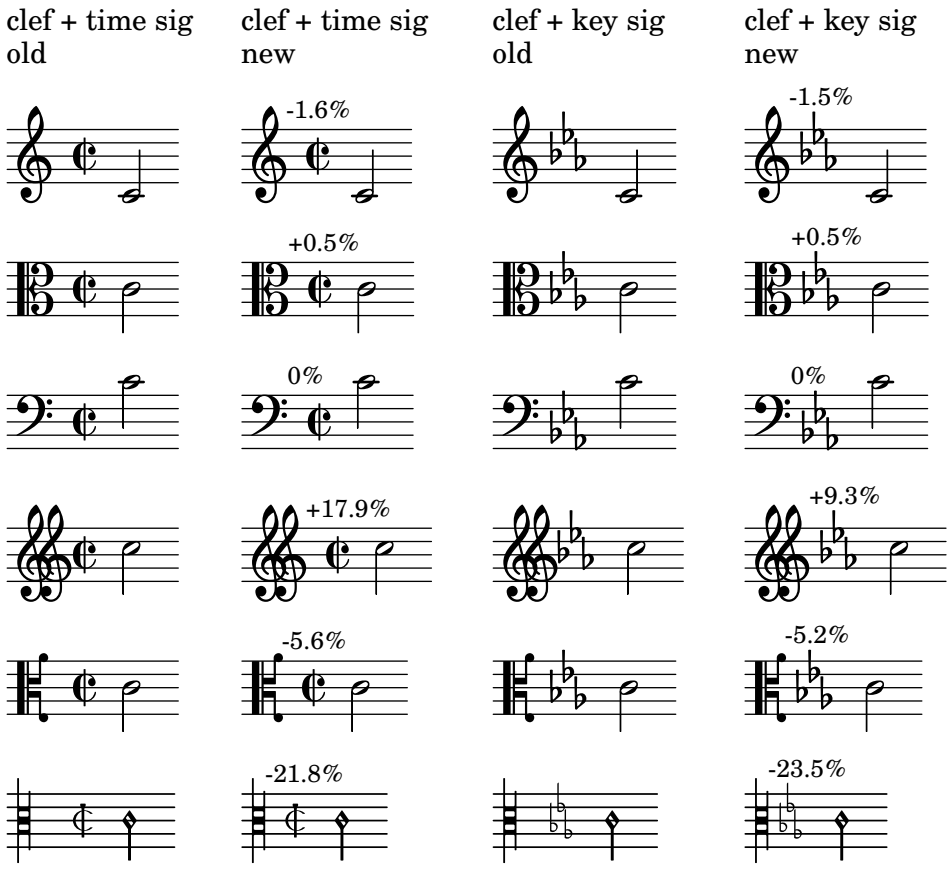
  In order to switch back to the previous settings (e.g., to keep the same layout when upgrading an existing score to version 2.25.15), add the following code:

```
\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 an alto 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.

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.

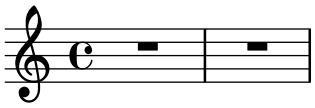
- LilyPond now requires Guile 3.0. As before, it is still strongly recommended to compile the `.scm` files into bytecode by first running `make bytecode` during compilation and then `make install-bytecode` in addition to `make install`.

New for musical notation

Pitches improvements

- Certain spurious change clefs have been fixed.

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

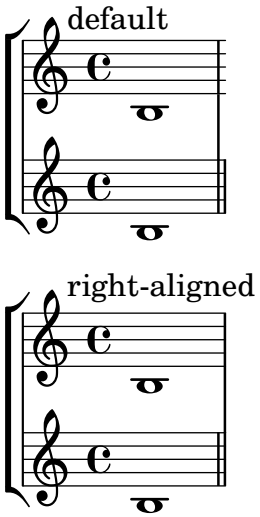


Rhythm improvements

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

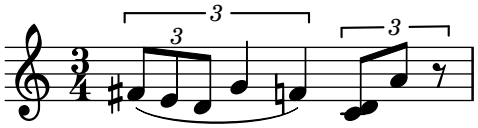
```
\new StaffGroup
  <<
    \new Staff { \textMark "default" b1 }
    \new Staff { b1 \section }
  >>

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



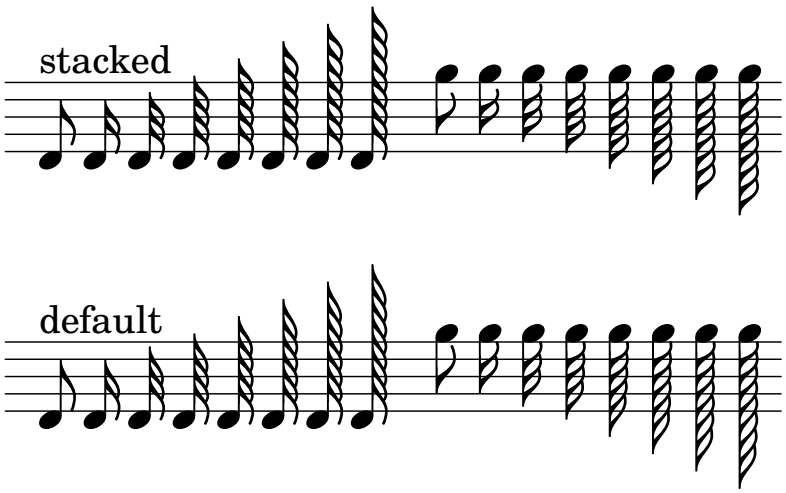
- Bar checks (|) now implicitly create contexts. The developers deem this unlikely to impact real-world scores. Please report a bug if you find a problem without an obvious workaround.

- The new option `span-all-note-heads` 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 `baseMoment`. Now, LilyPond can correctly subdivide most beaming patterns and no longer uses the value of `baseMoment` to limit beam subdivision. Simply setting `subdivideBeams` to true now automatically subdivides all intervals by default. 3 new properties have been introduced to tune automatic beam subdivision: `minimumBeamSubdivisionInterval`, `maximumBeamSubdivisionInterval` and `respectIncompleteBeams`. `minimumBeamSubdivisionInterval` limits subdivision intervals the same way as how `baseMoment` previously did (reducing frequency of subdivided beams). `maximumBeamSubdivisionInterval` 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 `minimumBeamSubdivisionInterval` to the value of `baseMoment` at all times, even when `baseMoment` 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.

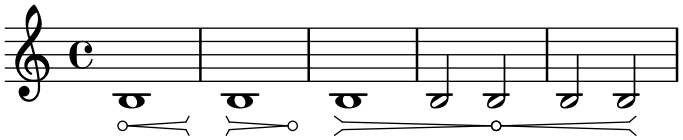


- The `TimeSignature` style `'single-digit` has been renamed to `'single-number`.

Expressive mark improvements

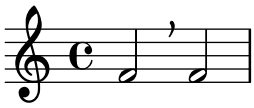
- Hairpins in the style of Ferneyhough now support al niente circles.

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



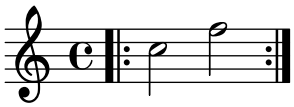
- Two new variant glyphs for breathing signs are available: ‘laltcomma’ and ‘raltcomma’. These glyphs represent the old shapes of ‘lcomma’ and ‘rcomma’, respectively, before changing them to more common shapes.

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



Repeat improvements

- `\repeat volta` alternative endings no longer create invisible bar lines. This may affect line breaking, horizontal spacing, and `VoltaBracket` extent 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 make a start repeat bar line automatically printed even 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

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

Text and font improvements

- 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"
```

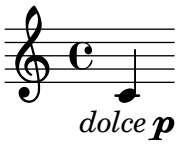
```
}
```

This is preferable over the already existing `font-name` property, since the latter makes commands such as `\bold` ineffective, instead requiring to include "Bold" 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 to specify 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 selecting 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 true, 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 …

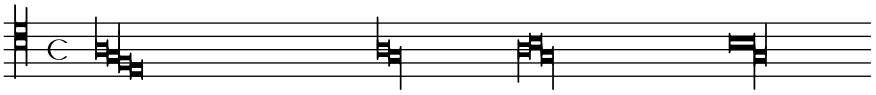
The fingering 4-5 for a 5+ 6 …

- 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.

New for specialist notation

- For orthogonality 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"
    \[ a1 g f e \]
    \[ a1 g\longa \]
    \[ \once \override NoteHead.left-down-stem = ##t
       a\breve b
       \once \override NoteHead.right-down-stem = ##t
       g\longa \]
    \[ \once \override NoteHead.right-down-stem = ##t
       b\maxima
       \once \override NoteHead.right-up-stem = ##t
       g\longa \]
  }
  \layout {
    \context {
      \Voice
      \remove Ligature_bracket_engraver
      \consists Mensural_ligature_engraver
    }
  }
}
```



- The use of the gregorian.ly is deprecated. While still working 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 diagram fret labels 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`.
- 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`.

Miscellaneous improvements

- 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"
  ```

- A figure-dash glyph (U+2012) and an en-dash glyph (U+2013) 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`.
- 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`).
- The printing of arpeggios has been improved, using new, different default values for the `Arpeggio.positions` property. Adjustments of this property must probably be updated.
- LilyPond provides support for in-notes, i.e., footnote-like annotations between music systems. This isn't new (it was actually 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.

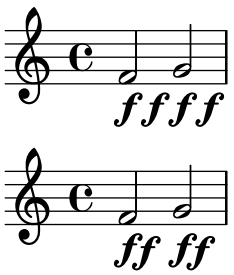
- Instead of the functions `\bookOutputName` and `\bookOutputSuffix` we now recommend to use the paper variables `output-filename` and `output-suffix` (which are not new but stayed undocumented until now). While the former will work unchanged, the latter is more coherent and easier to understand, especially 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 sub-properties 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.
- Function `ly:font-config-display-fonts` got an optional argument to select the output port.
- The Scheme command-line option handling is now more robust. In 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` got 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 and sets a new value, while the second one pops off the value from the stack and uses it to restore the previous value.

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

```
}
```