

LilyPond

The music typesetter

Changes

The LilyPond development team

This document lists changes and new features in LilyPond version 2.27.3 since 2.26.

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 2.27.3

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
New for musical notation	2
Pitches improvements	2
Rhythm improvements	2
Expressive mark improvements	3
Repeat improvements	3
Editorial annotation improvements	3
Text and font improvements.....	3
New for specialist notation	5
Miscellaneous improvements	7

Major changes in LilyPond

- Book and bookpart expressions now have different and separate types that can be tested using `ly:book?` and `ly:bookpart?`. A predicate testing for either (like `ly:book?` did before) is available as `ly:book-or-bookpart?`.

Book and bookpart identifiers as well as calls to syntactic scheme functions defined using `define-scheme-function` and returning book or bookpart expressions can now be directly used instead of book or bookpart blocks.

- `\book` and `\bookpart` blocks now contain their own local scopes for variables. This scope is active across the block and contains all variables defined within the block. As one consequence, assignments to variables are now permitted within `\book` and `\bookpart` blocks, making it easier to restructure standalone LilyPond files by wrapping them into books or bookparts. Outside of the containing block, the variables are not visible. The public interface of the module containing a book or bookpart's variables can be accessed programmatically using the new Scheme function `ly:book-scope`; accessors `ly:book-lookup` and `ly:book-set-variable!` are also provided.
- Books and bookparts may now contain output definition blocks `\layout` and/or `\midi` in addition to `\paper` (which was always possible). Those take their initial settings from the current active scope. If a bookpart has its own `\layout` block, that provides a default for typesetting all scores it includes that don't have an output definition of their own. The same holds for a book. Since a score produces only MIDI when it explicitly includes a `\midi` block, there is no similar default behavior for MIDI at processing time.

New for musical notation

Pitches improvements

None so far.

Rhythm improvements

- The new `\senzaMisura` command institutes “free time” and optionally prints an ‘X’ time signature. The sign is enabled with `\senzaMisuraTimeSignatureX` and disabled with `\senzaMisuraTimeSignatureOff`.

```
\fixed c' {
  \senzaMisuraTimeSignatureX
  \senzaMisura
  c8 d e f g f e f g1
  g8 a g f e d c e d1
}
```



- The new `\measure` command changes the measure length to match the duration of the given music and restores it afterward.

```
\fixed c'' {
  d2( c)
  \measure { d\breve }
}
```



- The new `\measureRemainder` command is similar to `\measure`, but it can be used to complete a measure that has already begun.
- The new `\setMeasureLengthFromHere` command simplifies creating irregular measures by setting the measure length relative to the current measure position. This complements `\partial`, which changes the measure position relative to the measure length. The new `\setDefaultMeasureLength` command restores the measure length to the value dictated by the time signature.
- The new `\premeasure music` command is an abbreviation for `\partial duration music` that computes the duration automatically.
- By default, strictly alternating meters consisting of 4/4 and 2/2 are printed in a historic style using multiple C or cut-C symbols.

```
{
  \time #'((2 . 2) (2 . 2))
  b'1 2 2
}
```



- The automatic beaming algorithm now can be told to interrupt an automatically created beam at some point using `\beamBreak`. Similarly, using `\noBeamBreak`, two adjoining notes can be forced to get beamed together:

```
\relative {
  c''8 c c c d d \beamBreak d d
  \time 2/4
  c8 8 8 8
  8 8 \noBeamBreak 8 8
  4. 8 \noBeamBreak
  8 4 8
}
```



Expressive mark improvements

None so far.

Repeat improvements

None so far.

Editorial annotation improvements

- The Scheme function `parenthesize-stencil` got an optional argument to lengthen the parentheses.
- The Scheme function `bracketify-stencil` got an optional argument to lengthen the brackets.

Text and font improvements

- An equal sign was added to the Emmentaler font.
- An ‘X’ time signature was added to the Emmentaler font. It can optionally be printed when meter is canceled with the new `\senzaMisura` command. Use of the sign is enabled with `\senzaMisuraTimeSignatureX` and disabled with `\senzaMisuraTimeSignatureOff`.



- Text variant glyphs for common time and cut time symbols were added to the Emmentaler font. They can be either accessed directly as Unicode characters or using the `\text-common-time` and `\text-cut-time` markup commands.

Common time (C), cut time (C).

- Fonts may be changed in a single markup using `\font-select`. The font names for the serif, sans and typewriter families are chosen independently.

```
\markup \font-select #'((sans . "DejaVu Sans")
                        (typewriter . "DejaVu Sans Mono"))
{
  \sans \line { Sans-serif font. }
  \typewriter \line { Typewriter font. }
  \line { Default serif font. }
  \font-select #'(serif . "DejaVu Serif")
  \line { Changed serif font. }
}
```

Sans-serif font. Typewriter font. Default serif font. Changed serif font.

New for specialist notation

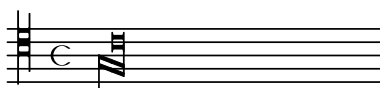
- The chord name input `x:1/x` gets now rendered as ‘N.C./x’.

```
\chords { d:1/d }
```

N.C./D

- White mensural ligatures now support final ascending longae facing backwards, obey tweaks in more cases, and handle errors more gracefully.

```
\score {
  \new PetrucciStaff {
    \clef "petrucci-c4"
    \[ g\breve f \tweak ligature-pes ##t c'\longa \]
  }
}
```



- Stanza numbers in lyrics are now created with a dedicated `\stanza ...` command. This replaces the former syntax `\set stanza = ....`

```
\relative {
  c'4 c g' g
  a a g2
}
\addlyrics {
  \stanza "1."
  Twin -- kle, twin -- kle, lit -- tle star
}
\addlyrics {
  \stanza "2."
  When the bla -- zing sun is gone
}
```



1. Twin-kle, twinkle, lit-tle star
2. When the bla-zing sun is gone

- Stanza numbers in lyrics can now be auto-repeated at the beginning of each line. To activate this feature, set the `stanzaReminders` property to `#t`:

```
\new Voice \relative {
  \key e \minor
  b'4 b a g
  fis2 e \break
  b'4 cis d b
  e2 dis \break
  e4 g fis fis
  e1
}
\addlyrics {
  \stanza "1."
}
```



```

\set stanzaReminders = ##t
Je -- su, mei -- ne Freu -- de,
mei -- nes Her -- zens Wei -- de,
\set stanzaReminderText = "(1st)"
Je -- su, mei -- ne Zier.
}
\addlyrics {
\stanza "2."
Un -- ter dei -- nem Schir -- men
bin ich vor den Stür -- men
\set stanzaReminders = ##t
\set stanzaReminderText = #make-tiny-markup
al -- ler Fein -- de frei.
}

```



1. Je-su, mei-ne Freu-de,
2. Unter deinem Schirmen



1. meines Herzens Wei-de,
- bin ich vor den Stürmen



- (1st) Jesu, mei-ne Zier.
2. al-ler Feinde frei.

- An alternative to using `\pes` and `\flexa` (which internally do the same) is the command `\~`. This isn't new (being introduced in 2003); however, it stayed undocumented and was broken for some time.

```

\new VaticanaScore {
  \new VaticanaVoice \relative {
    \[ a \pes b \flexa g \]
  }
}

```



```

\new VaticanaScore {
  \new VaticanaVoice \relative {
    \[ a \~ b \~ g \]
  }
}

```



Miscellaneous improvements

- Producing SVG output with the legacy `-dbackend=svg` backend is now much faster for large scores, due to caching extracted glyphs from the SVG font file.
- The new `\verboseBarNumbers` command simplifies documentation examples where bar numbers are a central concern. It might also be useful for debugging.
- The new commands `\startGradualTempoChange` and `\stopGradualTempoChange` describe a temporary departure from the prevailing tempo. Currently, these affect only MIDI output.
- These functions, which have been deprecated since version 2.13.19, have been removed: `\deprecatedcresc`, `\deprecateddendcresc`, `\deprecateddim`, and `\deprecatedenddim`.
- The `glyph-name` property of a `NoteHead` grob now contains a fully qualified glyph name, as documented. It is no longer necessary to replace the stencil just to draw a different note head glyph; setting this property suffices.

```
{
  c'4
  \once\override NoteHead.glyph-name = "noteheads.s2triangle"
  c'4
}
```



As a side effect, the Scheme function `note-head::calc-glyph-name` is not needed anymore and will be removed after a deprecation cycle. Depending on the intended use case, its use may be replaced either by accessing the `glyph-name` property of a specific `NoteHead` grob, or by calling `select-head-glyph` for a specific notehead style and duration-log.