New features in 2.18 since 2.16

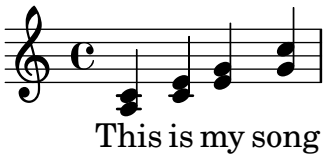
- The PostScript functionality of stroke adjustment is no longer applied automatically but left to the discretion of the PostScript device (by default, Ghostscript uses it for resolutions up to 150dpi when generating raster images). When it is enabled, a more complex drawing algorithm designed to benefit from stroke adjustment is employed mostly for stems and bar lines.

  Stroke adjustment can be forced by specifying the command line option '`-dstrokeadjust`' to LilyPond. When generating `PDF` files, this will usually result in markedly better looking `PDF` previews but significantly larger file size. Print quality at high resolutions will be unaffected.

- There is now a new context type called `NullVoice` which, while not appearing in the printed output, can be used to align lyrics. This can be particularly convenient when used in parallel with a `\partcombine` construct.

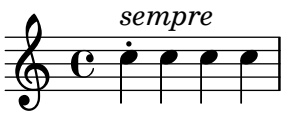
```
soprano = \relative c' { c e g c }
alto = \relative c' { a c e g }
verse = \lyricmode { This is my song }

\score {
  \new Staff <<
    \partcombine \soprano \alto
    \new NullVoice = "aligner" \soprano
    \new Lyrics \lyricsto "aligner" \verse
  >>
  \layout {}
}
```



- Several articulations can be put into a single variable or returned from an event function:

```
sempreStacc = -. ^\markup \italic sempre
\relative { c''4\sempreStacc c c c }
```



- The baseline of score markups is now taken from the reference point (usually the middle of the staff) of the first bottom system rather than the top of the bounding rectangle. The following

```
\markup {
  violin: \score { \new Staff { <g d' a' e''>1 }
                   \layout { indent=0 } } ,
  cello: \score { \new Staff { \clef "bass" <c, g, d a> }
                  \layout { indent=0 } }
}
```

violin: , cello:

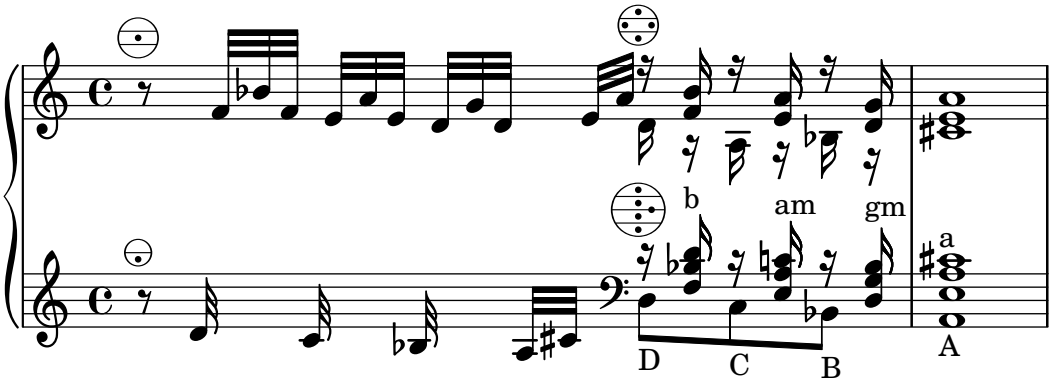
previously looked like

violin: , cello:

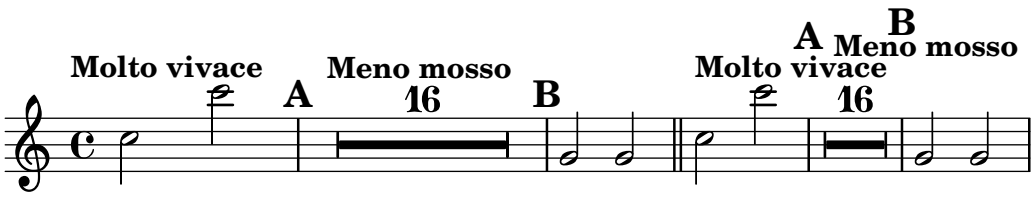
without a reliable way to get both scores to line up.

- LilyPond no longer automatically infers a '`\defaultchild`' context in a context definition with '`\accepts`' clauses. Any context definition without an explicit or inherited '`\defaultchild`' definition counts as a '`Bottom`' context and will be eligible for rhythmic events and overrides without causing the implicit creation of other contexts. Be sure to specify a '`\defaultchild`' for non-'`Bottom`' contexts when defining them from scratch.
- There is now extensive support for both discant and bass accordion register symbols in the '`scm accreg`' module, see Section "Accordion Registers" in *Notation Reference*.

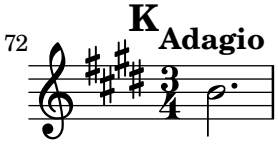
```
#(use-modules (scm accreg))
\new PianoStaff
<<
  \new Staff \relative
  { \clef "treble"  \discant "10"
    r8 s32 f'[ bes f] s e[ a e] s d[ g d] s16 e32[ a]
    \discant "121"
    << { r16 <f bes> r <e a> r <d g> } \\
       { d r a r bes r } >> |
    <cis e a>1
  }
  \new Staff \relative
  { \clef "treble"  \freeBass "1"
    r8 d'32 s16. c32 s16. bes32 s16. a32[ cis] s16
    \clef "bass"  \stdBass "Master"
    << { r16 <f, bes d>^"b" r <e a c>^"am" r <d g bes>^"gm" |
         <e a cis>1^"a" } \\
       { d8_"D" c_"C" bes_"B" | a1_"A" }
    >>
  }
>>
```



- New commands `markLengthOn` and `markLengthOff` control the allowance of horizontal space for tempo and rehearsal marks.



- Rehearsal marks at the beginning of a line are now placed to the right of the clef and key signature by default. As in previous versions, the `break-alignable-interface` controls the behavior.



- Decimal numbers can now be written directly in music, without a hash sign. Together with the previous change in the way object properties are specified, the code to change the length of stems has changed from this:

```
\override Stem #'length = #5.6
e' f' g' a'
```

  to this:

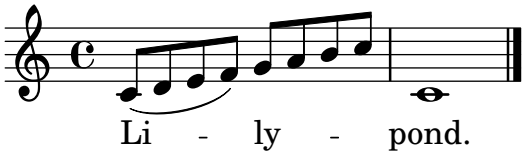
```
\override Stem.length = 5.6
e' f' g' a'
```

  One has to write a digit on both sides of the dot – values like `4.` or `-.3` are not allowed. Decimal fractions are also not accepted in `\chordmode`.

- A number of shorthands like `(`, `)`, `|`, `[`, `]`, `~`, `\(`, `\)` and others can now freely be redefined like normal commands. An example would be

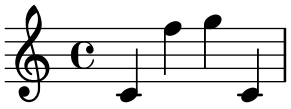
```
"\\{" = (
"\\}" = )
"(" = \melisma
")" = \melismaEnd

\new Staff <<
  \relative c' {
    c8 \{ d e f \} % slurred
    g ( a b c ) % no slur, but with melisma
    c,1 \bar "|."
   }
   \addlyrics { Li -- ly -- pond. }
>>
```



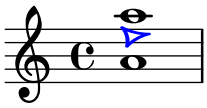
- The articulation shorthand for `\staccatissimo` has been renamed from `-|` to `-!`.
- Tempo change ranges are now written as `\tempo 4 = 60 - 68` rather than `\tempo 4 = 60 ~ 68`.
- Grob `OctavateEight` was renamed to `ClefModifier`. Related context properties were renamed from `xxxOctavationyyy` to `xxxTranspositionyyy`.
- There is a new `\absolute` command explicitly marking music as being entered in absolute pitch. While this has been the default previously, an explicit `\absolute` also prevents reinterpretation when the passage is placed inside of `\relative`:

```
\relative c { c'4 \absolute { f'' g'' } c }
```



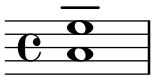
- When `\relative` is used without an explicit reference pitch, the reference pitch now is the middle of the first octave, making the first entered pitch indistinguishable from absolute pitch. Previously, omitting the reference pitch would have lead to a default of `c'`. Since that choice was somewhat arbitrary, recommended usage was to always specify the reference pitch.
- A new command `\single` can be used for converting a property override into a tweak to be applied on a single music expression:

```
<a \single\voiceTwoStyle e' a>1
```



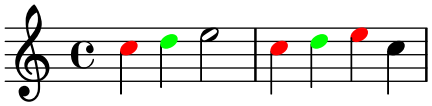
- Two ways of letting graphical objects not appear in the output are overriding its `transparent` property with `#t` (retaining the original spacing) or overriding its `stencil` property with `#f` (not using any space at all). Those two operations now have the shorthands `\hide` and `\omit`, respectively. They can either be given a music expression to tweak, or the name of a graphical object for which an override should be created (for specifying both, use `\single` on the override form):

```
\new Staff \with { \omit Clef }
\relative c'' <a e' \hide a>1
```



- A new command `\temporary` can be applied to overrides in order to not have them replace previous property settings. If a `\revert` is applied to the same property subsequently, the previous setting reappears:

```
\override NoteHead.color = #red c4
\override NoteHead.color = #green d
\revert NoteHead.color e2
\override NoteHead.color = #red c4
\temporary\override NoteHead.color = #green d
\revert NoteHead.color e
\revert NoteHead.color c
```



  This is mainly useful for writing music functions that need to have some property changed just for the duration of the function.
- `\tag`, `\removeWithTag`, and `\keepWithTag` can now accept a list of symbols rather than just a single symbol for marking, removing, and keeping music with any of multiple tags. This is particularly important for `\keepWithTag` since one cannot achieve the same effect by using multiple consecutive `\keepWithTag` commands.

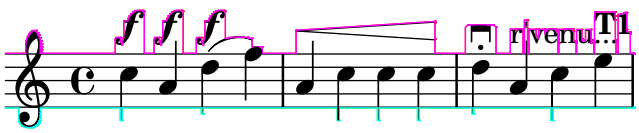
- The '`-d old-relative`' option has been removed. Not actually accessible from the command line any more, its remaining use was for interpretating `\relative` in LilyPond files converted automatically from version 1.8 or older. It is unclear how much of this was actually still operative.
- The meaning of `instrumentTransposition` has been reversed. After

  ```
  \set instrumentTransposition = #{ b #}
  ```

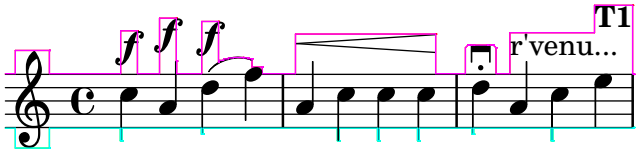
  a written `c'` now sounds like `b`. Previously, this would have been the other way round. This and the following change should make dealing with transposing instruments more straightforward.
- The music generated by `\set` and `\override` commands is no longer affected by `\transpose`. The main consequence is that `\transpose` will transpose audible/concert pitch and printed pitch by the same amount even when the transposed music contains `\transposition`. Previously,

  ```
  \transpose c' f' \transposition bes'
  ```

  was equivalent to `\transposition f'`. Now it stays equivalent to `\transposition bes'`.
- When checking for collisions, LilyPond no longer treats objects as rectangles. Instead, the actual shape of objects is approximated using an integral-like approach. This generally results in more even and snug positioning of objects and systems:

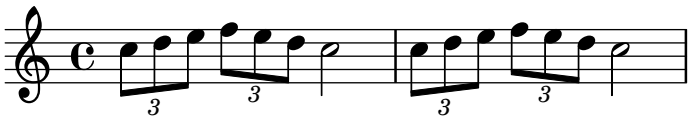


  Previously, the above snippet looked like this:



  Affected objects include `Accidentals`, `Beams`, `Clefs`, `Dynamics`, `FiguredBass`, `Flags`, `Glissandos`, `Lyrics`, `MetronomeMarks`, `OttavaBrackets`, `Pedals`, `RehearsalMarks`, `Rests`, `Scripts`, `TextScripts`, `Ties`, `Tuplets` and `VoltaBrackets`.
- Tuplets are now created with the `\tuplet` command, which takes a fraction `t/n` to specify that t notes are played in the time usually allowed for n. One `\tuplet` command can create several tuplet groups if their duration is typed after the fraction.

  ```
  \tuplet 3/2 { c8 d e } \tuplet 3/2 { f e d } c2
  \tuplet 3/2 4 { c8 d e f e d } c2
  ```



  The `\times` command with its inverted fraction order `n/t` is still available.
- Introducing two new markup-commands; `\draw-dashed-line` and `\draw-dotted-line`.

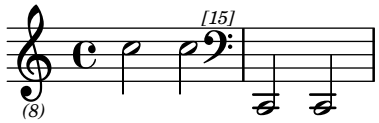
  The dashed-line extends to the whole length given by *dest*, if `full-length` is set to `#t` (this is the default) without any space at the beginning or end. `off` will then be altered to fit. To insist on the given (or default) values of `on`, `off` use `\override #'(full-length . #f)`. Manual settings for `on`, `off` and `phase` are possible.

The dotted-line always extends to the whole length given by *dest*, without any space at the beginning or end. Manual settings for `off` are possible to get larger or smaller space between the dots. The given (or default) value of `off` will be altered to fit the line-length.

```
\markup {
  \draw-dashed-line #'(5.1 . 2.3)
  \override #'(on . 0.3)
  \override #'(off . 0.5)
  \draw-dashed-line #'(5.1 . 2.3)
  \draw-dotted-line #'(5.1 . 2.3)
  \override #'(thickness . 2)
  \override #'(off . 0.2)
  \draw-dotted-line #'(5.1 . 2.3)
}
```

- Starting with version 2.17.10, error messages or the `textedit` URI used for point-and-click functionality specify column numbers starting with 1 rather than 0. The byte offset (also part of `textedit` URIs) still starts at 0.
- The `\clef` command supports optional transposition:

```
\clef "treble_(8)"
c2 c
\clef "bass^[15]"
c2 c
```



- The LilyPond syntax of dot-separated words `Voice.Accidental` has been made interchangeable with `#'(Voice Accidental)`, a Scheme list of symbols. As one result, code like

```
\override Voice.TextSpanner #'(bound-details left text) = "rit."
```

is now equivalent to

```
\override Voice.TextSpanner bound-details.left.text = "rit."
```

or even

```
\override #'(Voice TextSpanner) bound-details.left.text = "rit."
```

- Grob and grob property path no longer need to be specified as two separate arguments to commands like '`\override`' and `\revert`, allowing for the syntax

```
\override Voice.TextSpanner.bound-details.left.text = "rit."
```

Since complementary music functions like '`\overrideProperty`' cannot support forms with and without separating space at the same time, using a single dotted path is now the preferred form. Specifying grob path and grob property path separately, currently still supported with '`\override`' and '`\revert`' for compatibility reasons, is deprecated.

- Due to words now being accepted as symbol function arguments, the interfaces of '`\accidentalStyle`', '`\alterBroken`', '`\footnote`' and '`\tweak`' had to be redesigned where optional symbol arguments were involved. Please check the respective music function documentation for details.

- Several commands now accept symbol lists (conveniently entered as dot-separated words) for various kinds of arguments. These include '`\accidentalStyle`', '`\alterBroken`', '`\footnote`', '`\hide`', '`\omit`', '`\overrideProperty`', '`\shape`', and '`\tweak`'.
- The bar line user interface has changed. Bar glyphs now resemble the appearance of the bar line, so a left repeat sign has to be coded as `.|:`. The command `\defineBarLine` provides an easy way to define additional bar line styles.
- Accidentals in the key signature may be printed in octaves other than their traditional positions, or in multiple octaves.