

## Funcionalidades nuevas de la versión 2.20 desde la 2.18

- Ahora se pueden mover sistemas respecto a su posición actual usando la subpropiedad `extra-offset` de `NonMusicalPaperColumn.line-break-system-details`. Son posibles tanto desplazamientos horizontales como verticales. Esta funcionalidad es especialmente útil para hacer pequeños ajustes sobre la posición vertical predeterminada de los sistemas individuales. Véase Sección “Explicit staff and system positioning” en *Referencia de la Notación* para más información.
- Ahora se puede añadir texto sobre los corchetes de análisis mediante el objeto `HorizontalBracketText`.

```
\layout {
  \context {
    \Voice
    \consists "Horizontal_bracket_engraver"
  }
}

{
  \once \override HorizontalBracketText.text = "a"
  c''\startGroup d''\stopGroup
  e''-\tweak HorizontalBracketText.text "a'" \startGroup d''\stopGroup
}
```



- Se puede hacer ahora un ajuste fino de los extremos de los reguladores usando la propiedad de `grob shorten-pair`, que anteriormente afectaba solo a los objetos extensos de texto como `TupletBracket` y `OttavaBracket`. Los valores positivos producen un desplazamiento a la derecha, los negativos a la izquierda.

```
{ \once \override Hairpin.shorten-pair = #'(2 . 2)
  c'1~\<
  c'2~ c'\!
  \once \override Hairpin.shorten-pair = #'(-2 . -2)
  c'1~\<
  c'2~ c'\! }
```



- En los diagramas de posiciones de acorde, la distancia entre los trastes y entre las cuerdas se puede ajustar de forma independiente. Están disponibles `fret-distance` y `string-distance` como subpropiedades de `fret-diagram-details`.

```
fretMrkp = \markup { \fret-diagram-terse #"x;x;o;2;3;2;" }

\markuplist
\override #'(padding . 2)
\table #'(0 -1) {
```

```

"default"

\fretMrkp

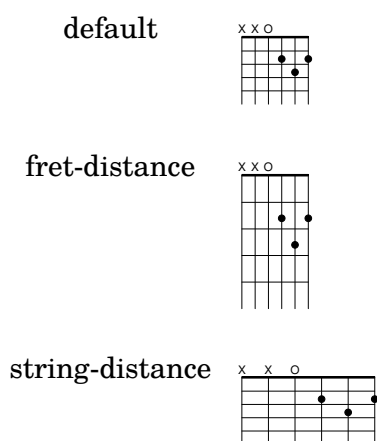
"fret-distance"

\override #'(fret-diagram-details . ((fret-distance . 2)))
\fretMrkp

"string-distance"

\override #'(fret-diagram-details . ((string-distance . 2)))
\fretMrkp
}

```



- Las reglas de alteraciones accidentales se pueden definir ahora para todo un contexto de **ChoirStaff**. Se encuentran disponibles dos nuevas reglas, **choral** y **choral-cautionary**, que combinan las características de **modern-voice** y de **piano** o sus equivalentes con alteraciones de precaución.
- La función musical **\unfoldRepeats** ahora puede tomar una lista de argumentos opcional que especifica qué tipo de música (o músicas) repetida(s) se debe(n) desplegar. Los valores posibles son **percent** (para repeticiones de compás o parte de ellos), **tremolo** y **volta** (para casillas de primera y segunda vez). Si no se especifica la lista opcional de argumentos, se usa **repeated-music**, que lo despliega todo.
- Ahora se usa una nueva propiedad de grob, **output-attributes**, para la salida de SVG en lugar de la propiedad de grob **id**. Permite que más de un atributo se pueda definir como una lista de asociación. Por ejemplo, `\#'(id . 123) (class . fulano) (data-loquesea . \mengano))` produce la siguiente etiqueta de grupo en el archivo SVG de salida: `<g id=\123" class=\fulano" data-loquesea=\mengano"> ... </g>`.
- Las ligaduras de expresión y de fraseo pueden comenzar ahora a partir de las notas individuales de un acorde. Las diferentes ligaduras simultáneas por cada contexto **Voice** se deben distinguir especificando su **spanner-id**.
- La propiedad musical y de grob **spanner-id** para distinguir ligaduras simultáneas de expresión y de fraseo ha cambiado de ser una cadena a ser una 'key', un entero no negativo o un símbolo.
- Existe una instrucción nueva **\=** para especificar el identificador de objeto extenso **spanner-id** para ligaduras simultáneas de expresión o de fraseo.

```

\fixed c' {

```

```
<c~ f\=1( g\=2( >2 <c e\=1) a\=2) >
}
```



- Los bloques `\header` se pueden almacenar dentro de variables y usados como argumentos para las funciones musicales y de Scheme, y como cuerpo de construcciones del tipo `#{...#}`. Se representan como un módulo de Guile.

Si bien los bloques `\book`, `\bookpart`, `\score`, `\with`, `\layout`, `\midi` y `\paper` se pueden pasar de manera similar, están representados por distintos tipos de datos.

- Las listas de elementos separados por puntos como `FretBoard.stencil` ya estaban contempladas a partir de la versión 2.18. Ahora también pueden contener enteros sin signo, y se pueden hacer separar opcionalmente mediante comas. Ello permite un uso semejante a

```
{ \time 2,2,1 5/8 g'8 8 8 8 8 }
```



y

```
\tagGroup violin,oboe,bassoon
```

- Tales listas se pueden usar también dentro de expresiones para asignaciones, conjuntos y sobreescrituras. Así, ahora puede usarse de la forma siguiente:

```
{ \unset Timing.beamExceptions
  \set Timing.beatStructure = 1,2,1
  g'8 8 8 8 8 8 8 }
```



- Anteriormente se podían asignar valores a los elementos de las listas de asociación individualmente (por ejemplo, variables de papel como `system-system-spacing.basic-distance`). Ahora también pueden ser referenciadas de esta manera, como en

```
\paper {
  \void \displayScheme \system-system-spacing.basic-distance
}
```

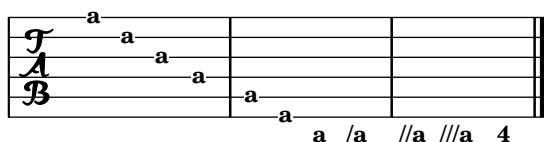
En combinación con los cambios mencionados anteriormente, esto permite el establecimiento de valores y la referenciación de pseudovariables como `violin.1`.

- Ahora se pueden incrustar los archivos de código fuente de LilyPond dentro de los archivos PDF generados. Esta funcionalidad experimental está inhabilitada de forma predeterminada y se puede calificar como de no segura, pues los documentos PDF con contenido oculto tienden a presentar un riesgo para la seguridad. Observe que no todos los visores de PDF son capaces de manejar documentos incrustados (en caso de que no sea así, la salida de PDF se verá normalmente y los archivos fuente serán invisibles). Esta funcionalidad solamente funciona con el backend PDF.
- Ahora se pueden definir específicamente los nombres de las notas en idioma francés en lugar de constituir un alias de los nombres italianos: además de la sintaxis genérica derivada del italiano, la altura de la nota *d* (*re*) se puede escribir ahora como *ré*. Los dobles sostenidos se pueden escribir también usando el sufijo *-x*.

- Están contempladas cuerdas bajas adicionales (para la tablatura de laúd).

```
m = { f'4 d' a f d a, g, fis, e, d, c, \bar "|" }

\score {
  \new TabStaff \m
  \layout {
    \context {
      \Score
      tablatureFormat = #fret-letter-tablature-format
    }
    \context {
      \TabStaff
      stringTunings = \stringTuning <a, d f a d' f'>
      additionalBassStrings = \stringTuning <c, d, e, fis, g,>
      fretLabels = #("a" "b" "r" "d" "e" "f" "g" "h" "i" "k")
    }
  }
}
```



- Ahora está disponible la instrucción de lista de marcados `\table`. Cada columna se puede alinear de distinta forma.

```
\markuplist {
  \override #'(padding . 2)
  \table
  #'(0 1 0 -1)
  {
    \underline { center-aligned right-aligned center-aligned left-aligned }
    one "1" thousandth "0.001"
    eleven "11" hundredth "0.01"
    twenty "20" tenth "0.1"
    thousand "1000" one "1.0"
  }
}
```

center-aligned   right-aligned   center-aligned   left-aligned

one	1	thousandth	0.001
eleven	11	hundredth	0.01
twenty	20	tenth	0.1
thousand	1000	one	1.0

- Una instrucción de marcado nueva, `\with-dimensions-from`, hace que sea más fácil de usar `\with-dimensions` obteniendo las dimensiones nuevas a partir de un objeto de marcado, dado como primer argumento.

```
\markup {
  \pattern #5 #Y #0 "x"
  \pattern #5 #Y #0 \with-dimensions-from "x" "f"
  \pattern #5 #Y #0 \with-dimensions-from "x" "g"
  \override #'(baseline-skip . 2)
  \column {
    \pattern #5 #X #0 "n"
    \pattern #5 #X #0 \with-dimensions-from "n" "m"
    \pattern #5 #X #0 \with-dimensions-from "n" "!"
  }
}
```

```
x f g nnnnn
x f g mmmmm
x f g !!!!!
```

- La nueva función de salto de página `ly:one-line-auto-height-breaking` sitúa una partitura completa en una sola línea y cambia la anchura del papel de forma correspondiente, de igual manera que `ly:one-line-breaking`. La diferencia es que también modifica automáticamente la altura de la página para que se ajuste a la altura de la música.
- Está disponible la instrucción de marcado `\draw-squiggle-line`. Es posible la personalización a través de la sobreescritura de `thickness`, `angularity`, `height` and `orientation`.

```
\markup
\overlay {
  \draw-squiggle-line #0.5 #'(3 . 3) ##t

  \translate #'(3 . 3)
  \override #'(thickness . 4)
  \draw-squiggle-line #0.5 #'(3 . -3) ##t

  \translate #'(6 . 0)
  \override #'(angularity . -5)
  \draw-squiggle-line #0.5 #'(-3 . -3) ##t

  \translate #'(3 . -3)
  \override #'(angularity . 2)
  \override #'(height . 0.3)
  \override #'(orientation . -1)
  \draw-squiggle-line #0.2 #'(-3 . 3) ##t
}
```



- Existe una instrucción nueva para eliminar los pentagramas vacíos, `\RemoveAllEmptyStaves`, que actúa de la misma forma que `\RemoveEmptyStaves`, excepto porque también elimina los pentagramas vacíos del primer sistema de una partitura.
- Las instrucciones de marcado `\undertie` y `\overtie` están disponibles, así como la instrucción de marcado genérica `\tie`.

```
\markup {
```


```

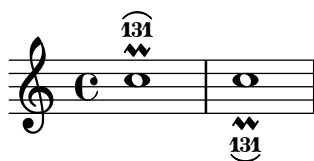
\undertie "undertied"
\overtie "overtied"
}

m = {
  c''1 \prall -\tweak text \markup \tie "131" -1
}

{ \voiceOne \m \voiceTwo \m }

```

undertied overtied



- TabStaff puede ahora imprimir microtonos para las curvaturas de tono y otros efectos.

```

\layout {
  \context {
    \Score
    supportNonIntegerFret = ##t
  }
}

mus = \relative { c'4 cih d dih }

<<
  \new Staff << \clef "G_8" \mus >>
  \new TabStaff \mus
>>

```



- Ahora están disponibles dos estilos nuevos de enmarcado en blanco para enmascarar el material de fondo. El estilo **outline** aproxima los contornos de la forma de un glifo, y la forma se produce a partir de un conjunto de copias desplazadas del glifo. El estilo **rounded-box** produce una forma de rectángulo redondeado. Para los tres estilos, incluido el estilo **box** predeterminado, se puede personalizar el **thickness** o grosor de la forma del enmarcado en blanco, como un múltiplo del grosor de una línea del pentagrama.

```

\markup {
  \combine
    \filled-box #'(-1 . 15) #'(-3 . 4) #1
    \override #'(thickness . 3)
    \whiteout whiteout-box
}

```

```

}
\markup {
  \combine
    \filled-box #'(-1 . 24) #'(-3 . 4) #1
    \override #'(style . rounded-box)
    \override #'(thickness . 3)
    \whiteout whiteout-rounded-box
}
\markup {
  \combine
    \filled-box #'(-1 . 18) #'(-3 . 4) #1
    \override #'(style . outline)
    \override #'(thickness . 3)
    \whiteout whiteout-outline
}
\relative {
  \override Staff.Clef.whiteout-style = #'outline
  \override Staff.Clef.whiteout = 3
  g'1
}

```

whiteout-box

whiteout-rounded-box

whiteout-outline

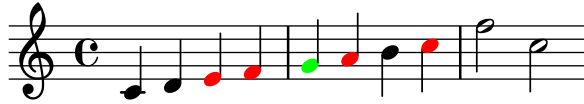


- Todas las instrucciones `\override`, `\revert`, `\set` y `\unset` ahora funcionan con el prefijo `\once` para hacer ajustes de una sola vez.

```

\relative {
  c'4 d
  \override NoteHead.color = #red
  e4 f |
  \once \override NoteHead.color = #green
  g4 a
  \once \revert NoteHead.color
  b c |
  \revert NoteHead.color
  f2 c |
}

```



- Al crear una salida MIDI, LilyPond ahora almacena el `title` que se ha definido en el bloque `\header` de la partitura (o, si no existe tal definición en el nivel de `\score`, la primera definición de ese tipo que aparezca en un bloque `\header` del `\bookpart`, `\book`, o ámbito de nivel superior que encierra a la partitura) como el nombre de la secuencia dentro del archivo MIDI. De forma opcional, el nombre de la secuencia MIDI se puede sobrescribir usando el nuevo campo `midititle` del bloque `\header` independientemente del `title` (por ejemplo, en caso de que `title` contenga código de elementos de marcado que no se convierta automáticamente a texto sencillo de manera satisfactoria).
- Las funciones musicales (además de las funciones vacías y de Scheme) e instrucciones de marcado que se limitan a aplicar los parámetros finales a una cadena de sobreescrituras, se pueden definir ahora de forma que solamente escriben la expresión interrumpida con `\etc.`

```
bold-red-markup = \markup \bold \with-color #red \etc
highlight = \tweak font-size 3 \tweak color #red \etc
```

```
\markup \bold-red "text"
\markuplist \column-lines \bold-red { One Two }
```

```
{ c' \highlight d' e'2-\highlight -! }
```

**text**

**One**

**Two**



- Las funciones de LilyPond definidas con `define-music-function`, `define-event-function`, `define-scheme-function` y `define-void-function` ahora se pueden llamar directamente desde Scheme como si fuesen procedimientos de Scheme reales. Se sigue efectuando una comprobación de los argumentos en su número y tipo de la misma forma que cuando se llama a la función a través de código de LilyPond. Esto incluye la inserción de valores predeterminados para los argumentos opcionales que no se corresponden con sus predicados. En lugar de usar `\default` en la propia lista de argumentos para saltar explícitamente una secuencia de argumentos opcionales, se puede emplear `*unspecified*`.
- La posición actual del cursor de entrada y del analizador sintáctico se almacenan ahora dentro de fluidos de GUILE y se pueden referenciar a través de las llamadas de función (`*location*`) y (`*parser*`). Como consecuencia, un gran número de funciones que anteriormente tomaban un argumento `parser` explícito, ya no lo hacen.

Las funciones definidas con `define-music-function`, `define-event-function`, `define-scheme-function` y `define-void-function` ya no usan los argumentos `parser` y `location`.

Con estas definiciones, LilyPond trata de reconocer el uso obsoleto de los argumentos `parser` y `location`, ofreciendo durante algún tiempo una semántica compatible hacia atrás.



- En el idioma de nombre de nota "english", los nombres largos de las alturas con alteración ahora contienen un guión para una mejor legibilidad. Ahora se debe escribir

```
\key a-flat \major
```

en lugar de

```
\key aflat \major
```

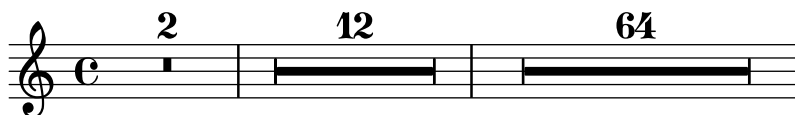
como se hacía antes.

Las alteraciones dobles no llevan otro guión, de forma que el holandés `cisis` lleva el nombre largo `c-sharpsharp` en inglés.

- El estilo visual de las barras de trémolo (forma, estilo e inclinación) se controla ahora con más precisión.



- Los silencios de varios compases tienen una longitud que depende de su duración, bajo el control de `MultiMeasureRest.space-increment`.



- Los números de página se pueden imprimir ahora en números romanos, fijando el valor de la variable `page-number-type` del bloque `\paper`.
- Ahora se pueden usar `\time` y `\partial` combinados para cambiar la indicación de compás en la mitad de un compás.

```
\override Score.BarNumber.break-visibility = #end-of-line-invisible
\partial 4 \time 3/4 f4 | 2 4 | 2 \bar "||"
\time 9/8 \partial 4. f8 8 8 | 2. 8 8 8 |
```



- Ahora es posible la sobreescritura de la propiedad `text` de los nombres de acorde.

```
<<
\new ChordNames \chordmode {
  a' b c:7
  \once \override ChordName.text = #"foo"
  d
}
>>
```

A B C<sup>7</sup> foo

- Se ha mejorado la alineación horizontal cuando se usa `TextScript`, con `DynamicText` o con `LyricText`.
- Se ha añadido una instrucción nueva `\magnifyStaff` que cambia la escala visual de los pentagramas, líneas, barras de compás, barras cortadas y el espaciado horizontal general en el nivel del contexto de `Staff`. Se evita que las líneas del pentagrama disminuyan a un tamaño menor que el predeterminado porque todos los grosores de las plicas, ligaduras y otros objetos gráficos están basados en el grosor de las líneas del pentagrama.

- `InstrumentName` contempla ahora el `text-interface`.
- Se contempla ahora el control del ‘nivel de expresión’ de los canales MIDI usando la propiedad de contexto `Staff.midiExpression`. Se puede usar para alterar incluso el volumen percibido de notas mantenidas (si bien a muy ‘bajo nivel’) y acepta un valor numérico entre 0.0 y 1.0.

```
\score {
  \new Staff \with {
    midiExpression = #0.6
    midiInstrument = #"clarinet"
  }
  <<
  { a'1~ a'1 }
  {
    \set Staff.midiExpression = #0.7 s4\f\<
    \set Staff.midiExpression = #0.8 s4
    \set Staff.midiExpression = #0.9 s4
    \set Staff.midiExpression = #1.0 s4

    \set Staff.midiExpression = #0.9 s4\>
    \set Staff.midiExpression = #0.8 s4
    \set Staff.midiExpression = #0.7 s4
    \set Staff.midiExpression = #0.6 s4\!
  }
  >>
  \midi { }
}
```

- Se ha añadido la posibilidad de hacer más fácil usar tipografías alternativas ‘de música’ diferentes de la predeterminada Emmentaler de LilyPond. Véase <http://fonts.openlilylib.org/> para más información.
- Los ‘grobs’ u objetos gráficos y sus ancestros se pueden ahora alinear separadamente permitiendo más flexibilidad para las posiciones de los grobs. Por ejemplo, el borde izquierdo de un grob se puede alinear sobre el centro de su ancestro.
- Se han hecho mejoras en la instrucción `\partial` para evitar problemas cuando se usa en varios contextos en paralelo.
- `\chordmode` puede usar ahora las construcciones `< >` y `<< >>`.
- Se ha añadido una instrucción `\tagGroup` que complementa a las instrucciones existentes `\keepWithTag` y `\removeWithTag`. Por ejemplo:
 

```
\tagGroup #'(violinI violinII viola cello)
```

 declara una lista de ‘tags’ o etiquetas que pertenecen a un solo ‘tag group’ o grupo de etiquetas.
 

```
\keepWithTag #'violinI
```

 se ocupa solamente de las ‘tags’ del grupo de etiquetas ‘violinI’.
 Cualquier elemento de la música incluida que lleve una o más etiquetas del grupo, pero *no* la etiqueta *violinI*, será eliminado.
- La función `\addlyrics` funciona ahora con contextos arbitrarios, entre ellos `Staff`.
- Ahora se pueden usar también los números de cuerda para imprimir números romanos (p. ej. para instrumentos de cuerda sin trastes).

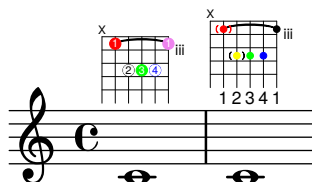
```
c2\2
\romanStringNumbers
```

```
c\2
\arabicStringNumbers
c1\3
```



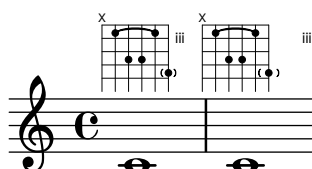
- El nombre de la propiedad `thin-kern` del objeto gráfico `BarLine` ha cambiado a `segno-kern`.
- Los objetos gráficos `KeyCancellation` ahora ignoran las claves de las notas guía (como hacen los objetos `KeySignature`).
- Se contempla ahora `\once \unset`
- Ahora es posible colorear independientemente tanto los puntos como los paréntesis en los diagramas de posición de acordes, si se usa la instrucción de marcado `\fret-diagram-verbose`.

```
\new Voice {
  c1^\markup {
    \override #'(fret-diagram-details . (
      (finger-code . in-dot))) {
      \fret-diagram-verbose #'((mute 6)
        (place-fret 5 3 1 red)
        (place-fret 4 5 2 inverted)
        (place-fret 3 5 3 green)
        (place-fret 2 5 4 blue inverted)
        (place-fret 1 3 1 violet)
        (barre 5 1 3 ))
      )
    }
  c1^\markup {
    \override #'(fret-diagram-details . (
      (finger-code . below-string))) {
      \fret-diagram-verbose #'((mute 6)
        (place-fret 5 3 1 red parenthesized)
        (place-fret 4 5 2 yellow
          default-paren-color
          parenthesized)
        (place-fret 3 5 3 green)
        (place-fret 2 5 4 blue )
        (place-fret 1 3 1)
        (barre 5 1 3))
      )
    }
  }
}
```



- Se han añadido dos propiedades nuevas para usarlas dentro de `fret-diagram-details` con la instrucción de marcado `\fret-diagram-verbose`; `fret-label-horizontal-offset`, que afecta a la `fret-label-indication`, y `paren-padding`, que controla la distancia entre el puntillo y los paréntesis que lo rodean.

```
\new Voice {
  c1^\markup {
    \fret-diagram-verbose #'((mute 6)
                        (place-fret 5 3 1)
                        (place-fret 4 5 2)
                        (place-fret 3 5 3)
                        (place-fret 1 6 4 parenthesized)
                        (place-fret 2 3 1)
                        (barre 5 2 3))
  }
  c1^\markup {
    \override #'(fret-diagram-details . (
      (fret-label-horizontal-offset . 2)
      (paren-padding . 0.25))) {
      \fret-diagram-verbose #'((mute 6)
                          (place-fret 5 3 1)
                          (place-fret 4 5 2)
                          (place-fret 3 5 3)
                          (place-fret 1 6 4 parenthesized)
                          (place-fret 2 3 1)
                          (barre 5 2 3))
    }
  }
}
```



- Se ha añadido una instrucción de marcado nueva, `\justify-line`. Similar a la instrucción de marcado `\fill-line` excepto que en lugar de disponer *palabras* en columnas, la instrucción `\justify-line` equilibra la distancia entre ellas, haciendo que cuando hay tres o más palabras en una instrucción de marcado, las distancias sean siempre consistentes.

```
\markup \fill-line {ooooooo oooooo oooooo oooooo}
\markup \fill-line {ooooooooo ooooooooo oo ooo}

ooooooo      oooooo      oooooo      oooooo

oooooooooooo  oooooooooo      oo      ooo

\markup \justify-line {ooooooo oooooo oooooo oooooo}
\markup \justify-line {ooooooooo ooooooooo oo ooo}

ooooooo      oooooo      oooooo      oooooo

oooooooooooo  oooooooooo      oo      ooo
```

- Se ha añadido la instrucción `\magnifyMusic`, que permite modificar el tamaño de la notación sin alterar el tamaño del pentagrama, mientras que se escalan automáticamente las plicas, barras y el espaciado horizontal.

```

\new Staff <<
  \new Voice \relative {
    \voiceOne
    <e' e'>4 <f f'>8. <g g'>16 <f f'>8 <e e'>4 r8
  }
  \new Voice \relative {
    \voiceTwo
    \magnifyMusic 0.63 {
      \override Score.SpacingSpanner.spacing-increment = #(* 1.2 0.63)
      r32 c'' a c a c a c r c a c a c a c
      r c a c a c a c a c a c a c a c
    }
  }
>>

```



- Se ha añadido una plantilla flexible adecuada para una amplia variedad de piezas de música coral. Se puede usar para crear música coral sencilla, con o sin acompañamiento de piano, en dos o en cuatro pentagramas. A diferencia de otras, esta plantilla está ‘incorporada’, lo que significa que no necesita ser copiada y editada: en lugar de ello sencillamente se incluye con la instrucción `\include` en el archivo de entrada. Para ver más detalles, consulte Sección “Plantillas incorporadas” en *Manual de Aprendizaje*.
- Se ha mejorado significativamente el posicionado de los números de grupos especiales para las barras en forma de codo. Anteriormente, los números de grupo especial se colocaban de acuerdo a la posición del corchete del grupo, incluso si éste no se imprimía. Ello podía dar lugar a números de tresillo descolocados. Ahora se sitúan más cerca de la barra cuando hay un segmento de barra adecuado para su colocación y cuando el corchete no se dibuja.

Se ha añadido también detección de colisiones, desplazando horizontalmente si está demasiado cerca de una columna de notas adyacente pero preservando la distancia vertical del número a la barra acodada. Si el propio número es demasiado grande como para caber en el espacio disponible, se usa en su lugar el sistema original de posicionamiento basado en el corchete; en cambio, en caso de colisión (p. ej. con una alteración accidental) el número de tresillo se aleja moviéndolo verticalmente.

```

\time 3/4
\override Beam.auto-knee-gap = 3
\tuplet 3/2 4 {
  g8 c'' e,
  c'8 g,, e''
  g,,8 e''' c,,
}

```



El comportamiento original del tresillo para las barras acodadas aún está disponible mediante una sobreescritura con la instrucción `\override` a través de la nueva propiedad `knee-to-beam`.

```
\time 3/4
\override Beam.auto-knee-gap = 3
\override TupletNumber.knee-to-beam = ##f
\tuplet 3/2 4 {
  g8 c'' e,
  c'8 g,, e''
  g,,8 e''' c,,
}
```



- Se han ‘armonizado’ las instrucciones `\lyricsto` y `\addLyrics`. Ambas aceptan ahora el mismo tipo de lista de argumentos delimitada que aceptan `\lyrics` y `\chords`. Se añade compatibilidad hacia atrás de manera que se permiten como argumentos identificadores musicales (p. ej. `\mus`). Se ha añadido una regla de `convert-ly` que elimina los usos redundantes de `\lyricmode` y reorganiza las combinaciones con iniciadores de contexto de forma que `\lyricsto` en general se aplica al final (es decir, como lo haría `\lyricmode`).
- Las funciones e identificadores de Scheme se pueden usar ahora como definiciones de salida.
- Las expresiones de Scheme se pueden usar ahora como constituyentes de acordes.
- Espaciado visual mejorado de las cabezas de nota con formas ‘MI’ Funk y Walker pequeñas y normales, de forma que tengan la misma anchura que otras notas con forma dentro de sus respectivos conjuntos. Las cabezas del tipo SOL también han mejorado visualmente cuando se usan tanto con las cabezas normales de tipo Aiken como con las de tipo Sacred Harp, así como con las variantes de línea delgada.
- `LeftEdge` tiene ahora unas dimensiones verticales `Y-extent` definibles. Véase Sección “Left-Edge” en *Referencia de Funcionamiento Interno*.
- Se ha añadido una función nueva `make-path-stencil` que contempla todas las instrucciones `path` tanto relativas como absolutas: `lineto`, `rlineto`, `curveto`, `rcurveto`, `moveto`, `rmoveto`, `closepath`.

La función también contempla la sintaxis de ‘letra única’ utilizada en las instrucciones de ruta estándares del SVG: L, l, C, c, M, m, Z y z.

Asimismo, la nueva instrucción es compatible hacia atrás con la función original `make-connected-path-stencil`. Véase también `scm/stencil.scm`.

- Las propiedades de contexto nombradas en la propiedad ‘`alternativeRestores`’ se restauran a su valor al comienzo de la *primera* alternativa en todas las alternativas siguientes.

Actualmente el conjunto predeterminado restaura el ‘compás actual’:

```
\time 3/4
\repeat volta 2 { c2 e4 | }
\alternative {
  { \time 4/4 f2 d | }
}
```

```

{ f2 d4 | }
}
g2. |

```

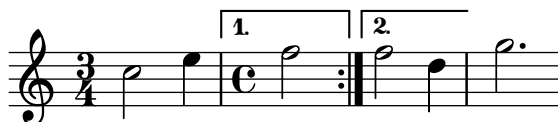


la ‘posición dentro del compás’:

```

\time 3/4
\repeat volta 2 { c2 e4 | }
\alternative {
{ \time 4/4
\set Timing.measurePosition = #(ly:make-moment -1/2)
f2 | }
{ f2 d4 | }
}
g2. |

```

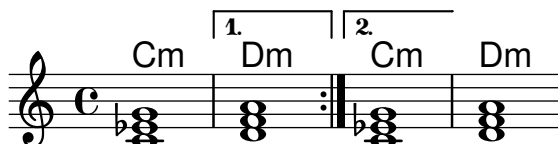


y los ‘cambios de acorde’;

```

<<
\new ChordNames {
\set chordChanges = ##t
\chordmode { c1:m d:m c:m d:m }
}
\new Staff {
\repeat volta 2 { \chordmode { c1:m } }
\alternative {
{ \chordmode { d:m } }
{ \chordmode { c:m } }
}
\chordmode { d:m }
}
>>

```



- Salida MIDI mejorada para las marcas de respiración. Después de las notas unidas mediante una ligadura, las respiraciones toman la duración *solo* de la última nota de la ligadura; p. ej. { c4~ c8 \breathe } se ejecuta como { c4~ c16 r } en lugar de { c4 r8 }. Esto es más consistente con las articulaciones y con la forma en que los humanos interpretan las respiraciones después de las ligaduras de unión. También hace que ahora sea más fácil alinear varias marcas de respiración simultáneas sobre más de una parte, aunque las notas tengan distintas duraciones.

- Se ha añadido un nuevo estilo de cabeza de nota para la tablatura: `TabNoteHead.style = #'slash`.
- Se ha añadido cuatro nuevos glifos de clave *Doble de Sol*, *Sol de tenor*, *Variante de percusión* y *Variante de Do* así como sus correspondientes formas reducidas de cambio de clave.

```
\override Staff.Clef.full-size-change = ##t
```

```
\clef "GG" c c c c
\clef "tenorG" c c c c
\clef "varC" c c c c
\clef "altovarC" c c c c
\clef "tenorvarC" c c c c
\clef "baritonevarC" c c c c
\clef "varpercussion" c c c c
```

```
\break
```

```
\override Staff.Clef.full-size-change = ##f
```

```
\clef "GG" c c c c
\clef "tenorG" c c c c
\clef "varC" c c c c
\clef "altovarC" c c c c
\clef "tenorvarC" c c c c
\clef "baritonevarC" c c c c
\clef "varpercussion" c c c c
```



- Las duraciones aisladas en las secuencias musicales ahora tienen el significado de notas sin altura. Esto puede ser de utilidad para especificar duraciones de música o de funciones de Scheme. Cuando se encuentran en la partitura final, las alturas vienen provistas por la nota o acorde anterior. He aquí dos ejemplos en los que se aprecia que produce una entrada más legible:

```
\new DrumStaff \with { \override StaffSymbol.line-count = 1 }
\drummode {
  \time 3/4
```



```
tambourine 8 \tuplet 3/2 { 16 16 16 }
      8 \tuplet 3/2 { 16 16 16 } 8 8 |
}
```



```
\new Staff { r16 c'16 ~ 8 ~ 4 ~ 2 | }
```



- `\displayLilyMusic` y sus funciones de Scheme subyacentes ya no omiten las duraciones de nota redundantes. Ello hace que sea más fácil reconocer correctamente y formatear las duraciones aisladas en expresiones como

```
{ c4 d4 8 }
```

- Las excepciones en el barrado se pueden construir ahora usando la función de Scheme `\beamExceptions`. Ahora se puede escribir

```
\time #'(2 1) 3/16
\set Timing.beamExceptions =
  \beamExceptions { 32[ 32] 32[ 32] 32[ 32] }
c16 c c |
\repeat unfold 6 { c32 } |
```



con las diversas excepciones separadas mediante comprobaciones de compás | (la escritura del patrón de la excepción la altura de las notas es conveniente pero no obligatorio). Anteriormente habría sido necesario escribir las excepciones del barrado como

```
\set Timing.beamExceptions =
#'(
  (end . ;start of alist
    ( ;entry for end of beams
      ( ;start of alist of end points
        ((1 . 32) . (2 2 2)) ;rule for 1/32 beams -- end each 1/16
      ))
    )
  )
)
```

- Las articulaciones más comunes se reflejan ahora en la salida MIDI. El acento y el marcato hacen a las notas sonar más fuerte; el picado, el staccato, el staccatissimo y el portato las hacen más cortas. Las marcas de respiración acortan la nota anterior.

Este comportamiento se puede personalizar a través de las propiedades `midiLength` `midiExtraVelocity` sobre `ArticulationEvent`. Para ver ejemplos, consúlre `script-init.ly`.

- La funcionalidad de PostScript del ajuste del trazo ya no se aplica automáticamente, sino que se deja a la discreción del dispositivo PostScript (de forma predeterminada, Ghostscript lo usa para las resoluciones de hasta 150 ppp al generar imágenes de matriz de puntos). Cuando se activa, se emplea (principalmente para las plicas y las líneas divisorias) un algoritmo de trazado más complejo diseñado para sacar provecho del ajuste del trazo.

El ajuste del trazo se puede forzar especificando la opción de línea de órdenes ‘`-dstrokeadjust`’ al llamar a LilyPond. CUando se generan archivos PDF, ello dará lugar por lo general a vistas previas de PDF con un aspecto marcadamente mejorado pero un tamaño de archivo significativamente mayor. La calidad de impresión en resoluciones altas no resulta afectada.