Esmuflily - SMuFL / Ekmelos for LilyPond

Esmuflily is an extension for LilyPond that supports SMuFL compliant fonts, in particular, to facilitate the use of glyphs from Ekmelos : clefs, time signatures, note heads, articulations, etc.
Esmuflily provides switches to turn the SMuFL support on or off for individual types of graphical objects (clefs, note heads, etc.) and it defines additional commands and styles for SMuFL glyphs which are not available in LilyPond (note head styles, function theory symbols, etc.) So scores can benefit from both SMuFL's comprehensive character set and LilyPond's awesome Emmentaler font.
This documentation uses the Ekmelos font for all SMuFL glyph.

Esmuflily requires LilyPond version 2.24.0 or higher.

See Ekmelily for accidentals and key signatures.

16 October 2025

Contents

Author and License

Esmuflily was written by Thomas Richter, thomas-richter@aon.at

Download, Installation, Usage

Download

The folder `ly` contains the include files.
Copy the following files into an appropriate folder:

```
esmufl.ily
ekmd.scm
ekmd-template.scm
```

`ekmd-template.scm` is required only if no metadata cache file exists for the desired font.
For some fonts, a cache file is already available, among others:

```
ekmd-bravura.scm
ekmd-ekmelos.scm
```

Usage

Add the following lines near the top of your LilyPond input file.

```
ekmFont = FONTNAME
\include "esmufl.ily"
```

Esmuflily + Ekmelily

To combine Esmuflily with Ekmelily add e.g. the following lines near the top of your LilyPond input file.

```
ekmFont = FONTNAME
ekmSystem = TUNING
\include "cosmufl.ily"
\language "LANGUAGE"
\ekmStyle STYLENAME
```

The default values correspond with LilyPond:

`FONTNAME`	Ekmelos
`TUNING`	24, i.e. it includes `ekmel-24.ily`
`LANGUAGE`	`nederlands` in most tunings
`STYLENAME`	`stc` (Stein/Couper) in tuning 24

Fonts

Esmuflily requires a SMuFL compliant font.
It uses Ekmelos by default. Another font can be selected, either with the variable

```
ekmFont = FONTNAME
```

preceding the include file,
or with the command line option

```
-dekmfont=FONTNAME
```

Note that this option produces a warning 'no such internal option', which can be ignored. Warnings can be suppressed with the command line option `--loglevel=ERROR` or `--loglevel=NONE`.

Drawing paths

Esmuflily supports drawing paths instead of font glyphs, which allows e.g. to produce stand-alone SVG output. This requires the Scheme procedure `ekm-path-stencil` as it is provided for Ekmelos by the include file `ly/ekmelos-paths.ily`.

A trailing (or solitary) `#` in FONTNAME draws paths globally, i.e. it effects all SMuFL output except for the markup commands `\ekm-charf` and `\ekm-str`.

Note that spaces and other glyphs without a contour, as well as side-bearing and font features like stylistic alternates or ligatures are not available with paths. See the second output below:

```
\ekm-chars #'(#xE262 #xE566 #xEAA6 #xEAA5)
```

To draw Ekmelos glyphs as paths, add the following lines near the top of your LilyPond input file. A single `"#"` is equivalent to `"Ekmelos#"`.

```
ekmFont = "#"
\include "ekmelos-paths.ily"
...
```

Font Metadata

Esmuflily can use font-specific metadata provided by a SMuFL compliant JSON file or by a cache file. If the latter doesn't exist, the JSON file is read and a cache file is created to be used subsequently.
Note: In the JSON file, glyphs must be given with their canonical glyph name, not with the Unicode code point.

JSON file path:

```
MD_LOC/MD_NAME
```

MD_NAME is one of the file names:

1. `FNAME_metadata.json`
2. `FNAME.json`
3. `metadata.json`

FNAME is the FONTNAME in all lowercase.

MD_LOC is one of the locations (see SMuFL ch. 3.11):

1. `PRIVATE_LOC`
2. `USER_LOC/SMuFL/Fonts/FONTNAME`
3. `SYSTEM_LOC/SMuFL/Fonts/FONTNAME`

PRIVATE_LOC can be specified either with the variable

```
ekmMetadata = PRIVATE_LOC
```

preceding the include file,
or with the command line option

```
-dekmmetadata=PRIVATE_LOC
```

A trailing (or solitary) `%` in PRIVATE_LOC forces creating a cache file even if it already exists.

USER_LOC and SYSTEM_LOC are platform-specific.

	USER_LOC	SYSTEM_LOC
Linux	`$XDG_DATA_HOME`	`$XDG_DATA_DIRS`
macOS	`~/Library/Application Support`	`/Library/Application Support`
Windows	`%LOCALAPPDATA%`	`%CommonProgramFiles%` `%CommonProgramFiles(x86)%`

Cache file path:

```
EKMD_LOC/ekmd-FNAME.scm
```

EKMD_LOC is one of the locations:

1. a LilyPond include directory (usually the location of `esmufl.ily`)
2. `MD_LOC` (as for the JSON file)

Font Symbols

The musical symbols supported by Esmuflily are SMuFL-compliant glyphs given with their Unicode code point.

All symbols are assembled in a single table arranged according to type (usage), where some types correspond to LilyPond's graphical objects, like note heads, flags, rests, and clefs.
The standard table contains SMuFL recommended characters. The cache file `ekmd-FNAME.scm` can provide a font-specific table (beside the metadata) that is merged into the standard table.

The styles, names, tokens, and size ranges for musical symbols listed in this documentation come from the standard table. The font-specific details for Ekmelos or other fonts are listed separately after that.

`\ekmMergeType TYPE TABLE`

Merge TABLE into the table for TYPE. New styles, names, and tokens are added. Already existing ones are replaced. See `internals.md` for all supported types and their required table structures.
Use this command with care.

Note: The style `ekm` is used for internal purposes, (tremolo marks, arrows, percussion beaters, etc.)

Commands

Most of the commands, in particular, all markup commands always produce SMuFL output, independent of any switches . Other commands behave differently when the corresponding switch is turned off:

[Ly] Produces normal LilyPond output.

[Err] Causes an error or produces useless output.

Some commands with a corresponding LilyPond command are simpler implemented, e.g. they ignore properties, while a few provide additional features.

Some commands and properties accept one of the following special values:

- EXTEXT : A code point, a list of code points, or markup.
- DEFINITION : A string of tokens.
- ORIENTATION : Sum of axis and direction.

SMuFL glyphs are always accessed by code point (EXTEXT). See the file `ly/ekmelos-map.ily` at Ekmelos with definitions to access glyphs by name.

All commands have the prefix `ekm` or `ekm-` .

SMuFL switches

```
\ekmSmuflOn #'TYPE
\ekmSmuflOn #'(TYPE ...)
\ekmSmuflOff #'TYPE
\ekmSmuflOff #'(TYPE ...)
```

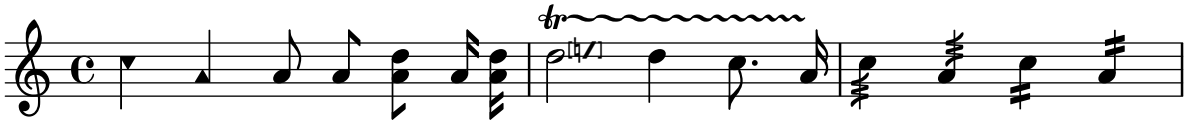
Turn the SMuFL support on and off, respectively, for one or more types of graphical objects. TYPE is one of the following symbols. Any other value is ignored.

These commands set/undo context and grob properties (usually the stencil) in the current bottom context, except for `colon` and `segno` which are set independently of a context and cannot be turned off.

`all`	All following types
`clef`	Clefs and clef modifiers
`time`	Time signatures
`notehead`	Note heads
`dot`	Augmentation dots
`flag`	Flags and grace note slashes
`rest`	Rests and multi-measure rests
`systemstart`	System start delimiters
`dynamic`	Absolute dynamic marks
`script`	Scripts
`textspan`	Text span
`trill`	Trill span and trill pitch
`lv`	Laissez vibrer
`colon`	Colon bar lines
`segno`	Segno bar lines
`percent`	Percent repeats
`tremolo`	Tremolos
`arpeggio`	Arpeggios
`tuplet`	Tuplet numbers
`fingering`	Fingering instructions
`stringnumber`	String number indications
`pedal`	Piano pedals
`fbass`	Figured bass
`lyric`	Lyric text

The following example demonstrates possible places for SMuFL switches : a `\with` block, a `\layout` block, and in the music stream. Note that `\ekmTremolo` works independent of the `tremolo` switch which is turned on after that.

```
\score {
  \new Staff \with {
    \ekmSmuflOn #'trill
  }
  \relative c'' {
    \ekmSmuflOn #'notehead
    \override NoteHead.style = #'triangle
    c4 a
    \ekmSmuflOff #'notehead
    \revert NoteHead.style

    \autoBeamOff
    a8
    \ekmFlag #'straight
    a <a d> a16 <a d>

    \ekmFlag #'default
    \ekmPitchedTrill #'slash #'bracket
    d2 \ekmStartTrillSpan #'(-4 . 0) e d4 c8. a16 \stopTrillSpan

    \ekmTremolo unmeasured { c4:16 a: }

    \ekmSmuflOn #'tremolo
    { c4:16 a: }
  }
  \layout {
    \context {
      \Score
      \ekmSmuflOn #'(flag dot)
    }
  }
}
```

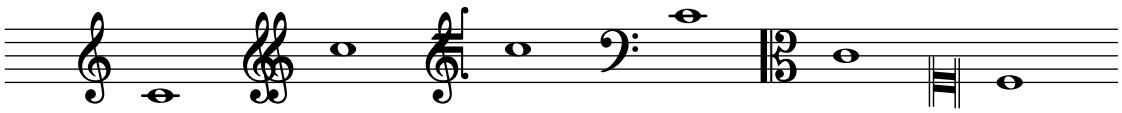


Clefs

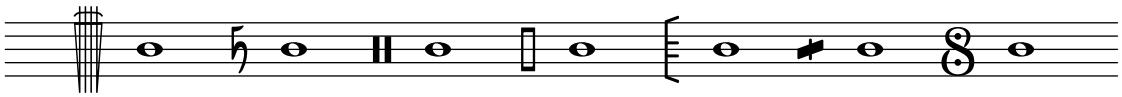
`\ekmSmuflOn #'clef`

Draw SMuFL clefs and clef modifiers.

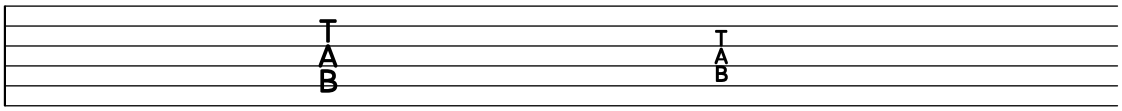
`"G" "treble" "french" …`	U+E050	gClef
`"GG"`	U+E055	gClef8vbOld
`"tenorG"`	U+E056	gClef8vbCClef
`"F" "bass" …`	U+E062	fClef
`"C" "alto" …`	U+E05C	cClef
`"neomensural-c1" …`	U+E060	cClefSquare



`"bridge"`	U+E078	bridgeClef
`"accordion"`	U+E079	accdnDiatonicClef
`"percussion"`	U+E069	unpitchedPercussionClef1
`"varpercussion"`	U+E06A	unpitchedPercussionClef2
`"semipitched"`	U+E06B	semipitchedPercussionClef1
`"varsemipitched"`	U+E06C	semipitchedPercussionClef2
`"indiandrum"`	U+ED70	indianDrumClef



`"tab"`	U+E06D	6stringTabClef
`"4stringtab"`	U+E06E	4stringTabClef

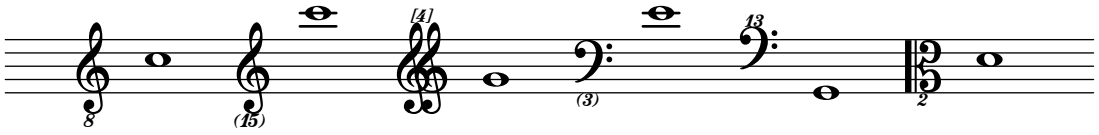


Clef modifiers

Transposition and style are drawn with the fingering italic symbols for digits, parentheses, and brackets, and with the following special symbols (i.e. not with precomposed clef glyphs).

8	*8*	U+E07D	clef8
15	*15*	U+E07E	clef15

```
"G_8"
"G_(15)"
"GG^[4]"
"F_(3)"
"subbass^13"
"C_2"
```



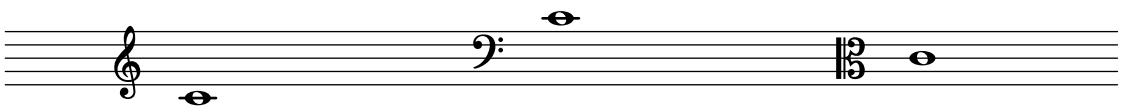
Change clefs

They are drawn either with a special glyph or with the normal glyph but smaller.
The relative font size for change clefs can be set with:

```
#(set! ekm:clef-change-font-size '(SPECIAL-SIZE . NORMAL-SIZE))
```

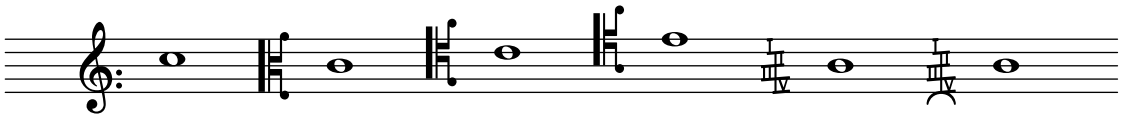
The standard value is `'(1.5 . -2)`.

"G"	U+E07A	gClefChange
"F"	U+E07C	fClefChange
"C"	U+E07B	cClefChange

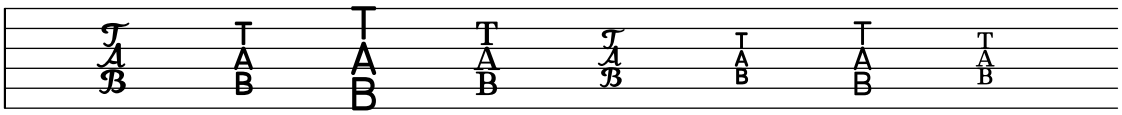


Ekmelos clefs

"frenchG"	U+F40E	gClef8vbFrench
"varC" "altovarC"	U+F633	cClefFrench20C
"tenorvarC"	:	
"baritonevarC"	:	
"string"	U+F71C	stringClef
"behindbridgestring"	U+F71D	behindBridgeStringClef

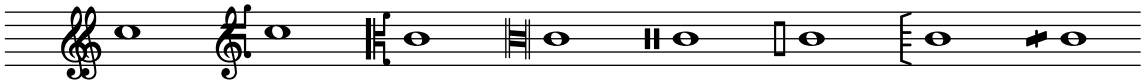


"tab"	U+F61E	6stringTabClefClassic
"moderntab"	U+E06D	6stringTabClef
"talltab"	U+F40A	6stringTabClefTall
"seriftab"	U+F40B	6stringTabClefSerif
"4stringtab"	U+F61F	4stringTabClefClassic
"4stringmoderntab"	U+E06E	4stringTabClef
"4stringtalltab"	U+F40C	4stringTabClefTall
"4stringseriftab"	U+F40D	4stringTabClefSerif



Ekmelos change clefs

"GG"	U+F630	gClef8vbOldChange
"tenorG"	U+F631	gClef8vbCClefChange
"varC"	U+F634	cClefFrench20CChange
"neomensural-c3"	U+F632	cClefSquareChange
"percussion"	U+F635	unpitchedPercussionClef1Change
"varpercussion"	U+F636	unpitchedPercussionClef2Change
"semipitched"	U+F6BE	semipitchedPercussionClef1Change
"varsemipitched"	U+F6BF	semipitchedPercussionClef2Change



Time signatures

`\ekmSmuflOn #'time`

Draw SMuFL time signatures.

`\ekmCompoundMeter TIME-SIGNATURE`

Set the numeric time signature.

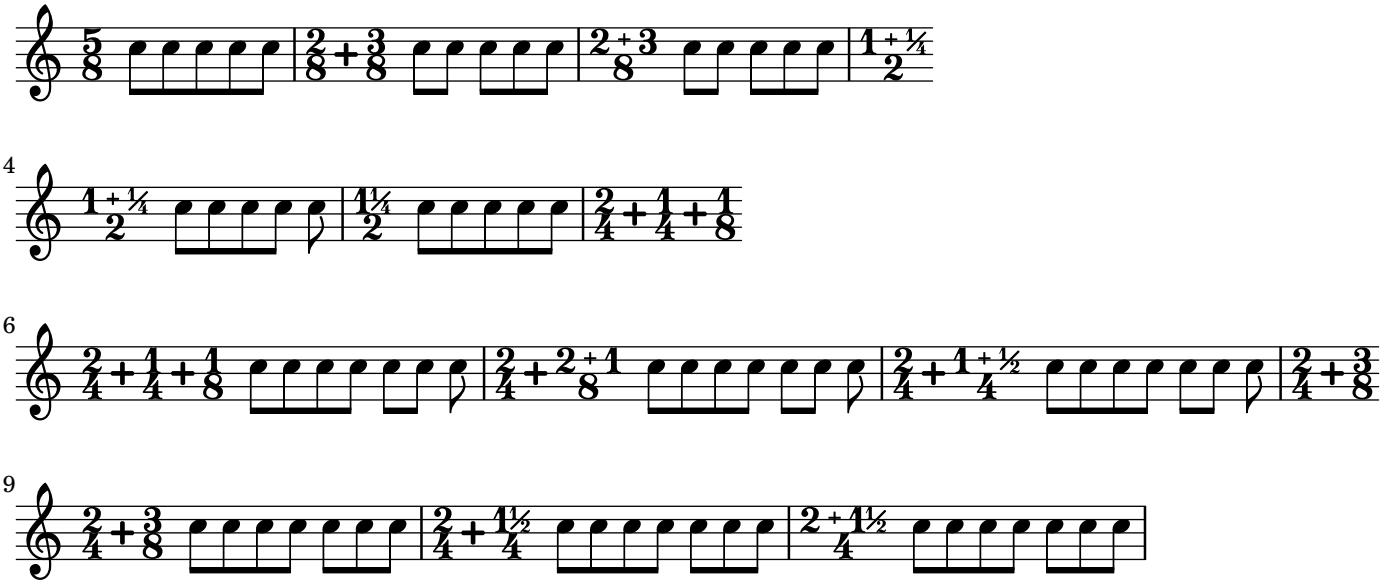
`\ekm-compound-meter TIME-SIGNATURE`

Draw the numeric time signature as markup.

Compound meters use the large plus sign between fractions and the small plus sign between the numbers in a numerator. Some rational numbers can be part of a numerator. If specified in a pair, e.g. `(1 . 1/2)` , this is treated as a single number without a plus sign in between.

4/4	𝄴	U+E08A	timeSigCommon
2/2	𝄵	U+E08B	timeSigCutCommon
0	𝟬	U+E080	timeSig0
	:		
9	𝟵	U+E089	timeSig9
+	+	U+E08C	timeSigPlus
	+	U+E08D	timeSigPlusSmall
1/4	¼	U+E097	timeSigFractionQuarter
1/2	½	U+E098	timeSigFractionHalf
3/4	¾	U+E099	timeSigFractionThreeQuarters
1/3	⅓	U+E09A	timeSigFractionOneThird
2/3	⅔	U+E09B	timeSigFractionTwoThirds

```
\relative c'' {
  \ekmCompoundMeter #'(5 8)
  c8 c c c c
  \ekmCompoundMeter #'((2 8) (3 8))
  c8 c c c c
  \ekmCompoundMeter #'(2 3 8)
  c8 c c c c
  \break

  \ekmCompoundMeter #'(1 1/4 2)
  c8 c c c c
  \ekmCompoundMeter #'(((1 . 1/4) 2))
  c8 c c c c
  \break

  \ekmCompoundMeter #'((2 4) (1 4) (1 8))
  c8 c c c c c c
  \ekmCompoundMeter #'((2 4) (2 1 8))
  c8 c c c c c c
  \ekmCompoundMeter #'((2 4) (1 1/2 4))
  c8 c c c c c c
  \break

  \ekmCompoundMeter #'((2 4) (3 8))
  c8 c c c c c c
  \ekmCompoundMeter #'((2 4) ((1 . 1/2) 4))
  c8 c c c c c c
  \ekmCompoundMeter #'(2 (1 . 1/2) 4)
  c8 c c c c c c
}
```



Cadenza signatures

`\ekmCadenzaOn STYLE`

Start a cadenza like `\cadenzaOn` and set a signature.

Styles

`"X"`		U+E09C	timeSigX
`"~"`		U+E09D	timeSigOpenPenderecki
`'time-x`	deprecated		
`'time-penderecki`	deprecated		

Staff dividers and Separators

`\ekmStaffDivider DIRECTION`

Draw the next barline with an indicator to split or recombine the staff and set a `\break` . The direction specifies the type of indicator (arrow).

`DOWN`	↘	U+E00B	staffDivideArrowDown
`UP`	↗	U+E00C	staffDivideArrowUp
`CENTER`	↕	U+E00D	staffDivideArrowUpDown

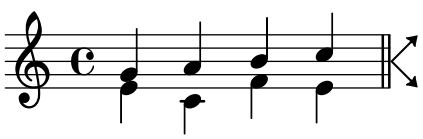
`system-separator-markup = \ekmSlashSeparator SIZE`

Draw a system separator mark, corresponding to SIZE (set within a `\paper` block).

Size ranges

`0`		U+E007	systemDivider
`≤ 1`		U+E008	systemDividerLong
`> 1`		U+E009	systemDividerExtraLong

```
\new Staff
<<
  \new Voice {
    \relative c'' {
      \voiceOne
      g a b c
      \bar "||" \ekmStaffDivider #CENTER
    }
  }
  \new Voice {
    \relative c' {
      \voiceTwo
      e c f e
    }
  }
>>
```



Note heads

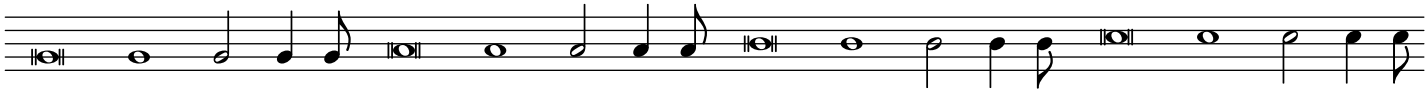
`\ekmSmuflOn #'notehead`

Draw SMuFL note heads. The `harmonic` and `cross` glyphs are also used with commands like `\harmonic` and `\xNote` .

Styles

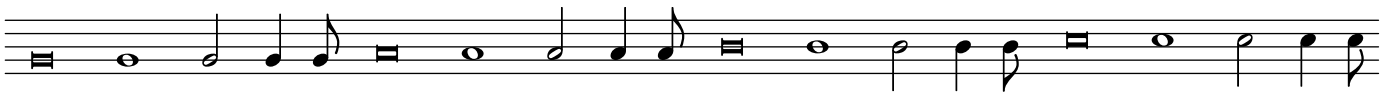
`'default`

U+E0A0 noteheadDoubleWhole
U+E0A2 noteheadWhole
U+E0A3 noteheadHalf
U+E0A4 noteheadBlack



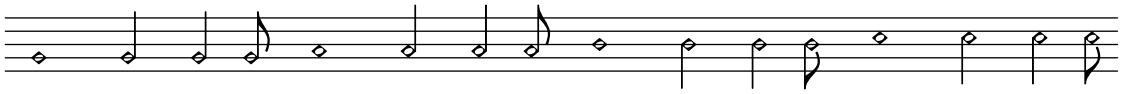
`'baroque`

U+E0A1 noteheadDoubleWholeSquare
:



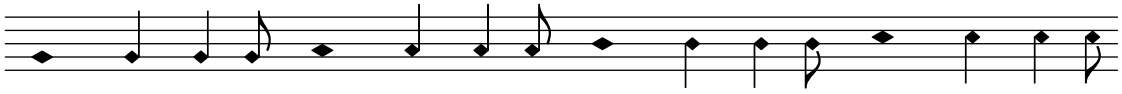
`'harmonic`

U+E0D9 noteheadDiamondHalf



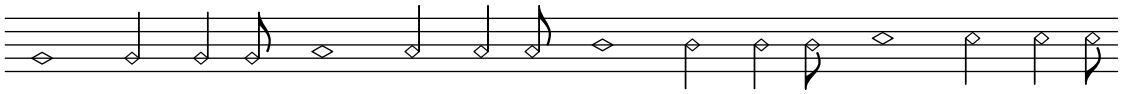
`'harmonic-black`

U+E0DC noteheadDiamondBlackWide
U+E0DB noteheadDiamondBlack



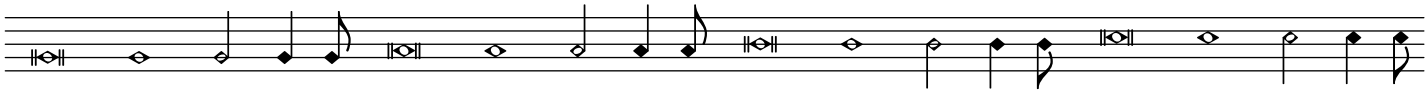
`'harmonic-white`

U+E0DE noteheadDiamondWhiteWide
U+E0DD noteheadDiamondWhite



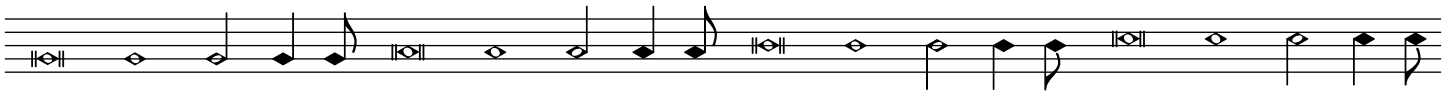
`'harmonic-mixed`

U+E0D7 noteheadDiamondDoubleWhole
U+E0D8 noteheadDiamondWhole
U+E0D9 noteheadDiamondHalf
U+E0DB noteheadDiamondBlack



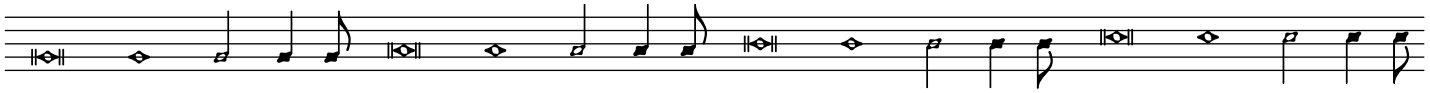
`'harmonic-wide`

U+E0D7 noteheadDiamondDoubleWhole
U+E0D8 noteheadDiamondWhole
U+E0DA noteheadDiamondHalfWide
U+E0DC noteheadDiamondBlackWide



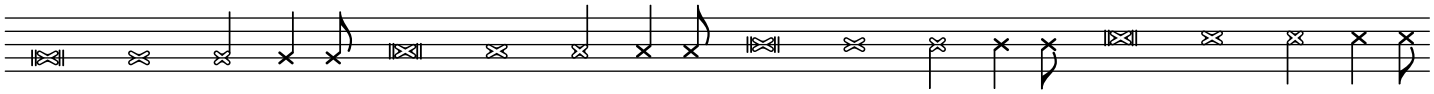
'diamond

U+E0DF	noteheadDiamondDoubleWholeOld
U+E0E0	noteheadDiamondWholeOld
U+E0E1	noteheadDiamondHalfOld
U+E0E2	noteheadDiamondBlackOld



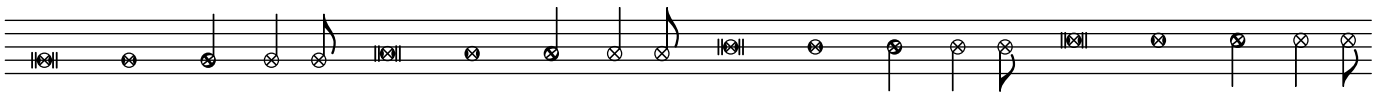
'cross

U+E0A6	noteheadXDoubleWhole
U+E0A7	noteheadXWhole
U+E0A8	noteheadXHalf
U+E0A9	noteheadXBlack



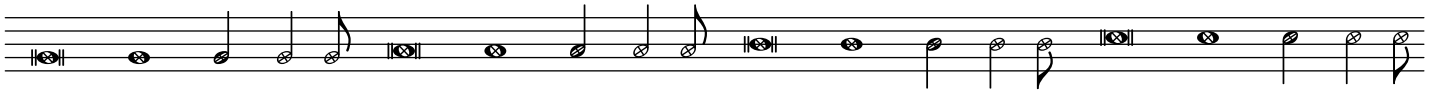
'xcircle

U+E0B0	noteheadCircleXDoubleWhole
U+E0B1	noteheadCircleXWhole
U+E0B2	noteheadCircleXHalf
U+E0B3	noteheadCircleX



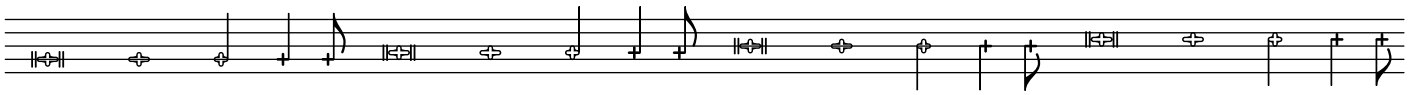
'withx

U+E0B4	noteheadDoubleWholeWithX
U+E0B5	noteheadWholeWithX
U+E0B6	noteheadHalfWithX
U+E0B7	noteheadVoidWithX



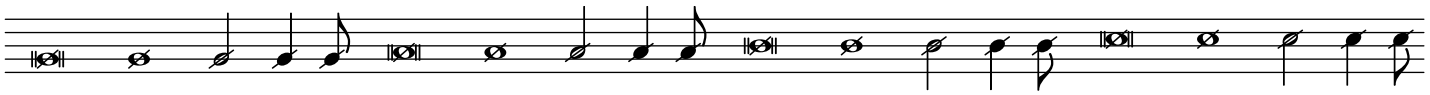
'plus

U+E0AC	noteheadPlusDoubleWhole
U+E0AD	noteheadPlusWhole
U+E0AE	noteheadPlusHalf
U+E0AF	noteheadPlusBlack



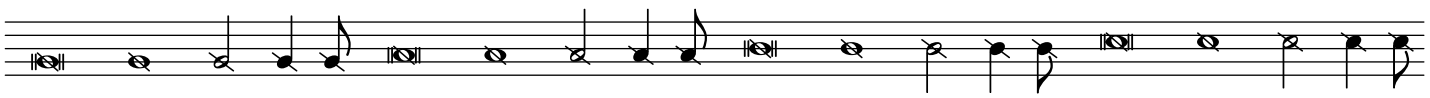
'slashed

U+E0D5	noteheadSlashedDoubleWhole1
U+E0D3	noteheadSlashedWhole1
U+E0D1	noteheadSlashedHalf1
U+E0CF	noteheadSlashedBlack1



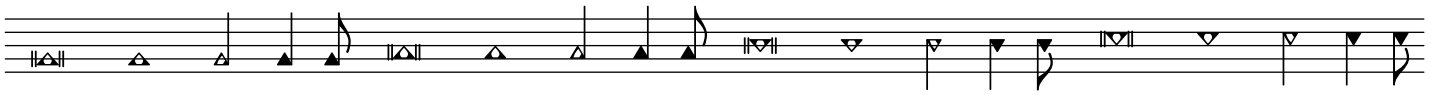
'backslashed

U+E0D6	noteheadSlashedDoubleWhole2
U+E0D4	noteheadSlashedWhole2
U+E0D2	noteheadSlashedHalf2
U+E0D0	noteheadSlashedBlack2

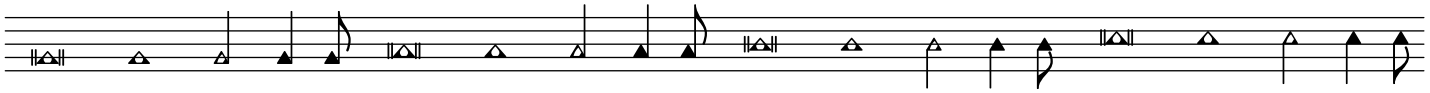


'triangle

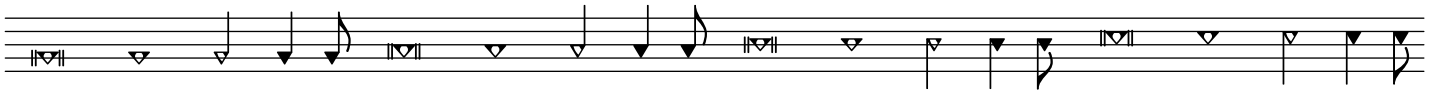
U+E0BA	noteheadTriangleUpDoubleWhole
U+E0BB	noteheadTriangleUpWhole
U+E0BC	noteheadTriangleUpHalf
U+E0BE	noteheadTriangleUpBlack
U+E0C3	noteheadTriangleDownDoubleWhole
U+E0C4	noteheadTriangleDownWhole
U+E0C5	noteheadTriangleDownHalf
U+E0C7	noteheadTriangleDownBlack



'triangle-up



'triangle-down

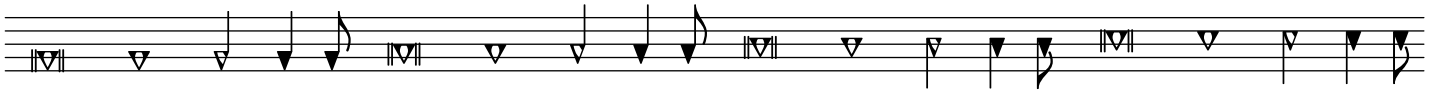


'arrow

U+E0ED	noteheadLargeArrowUpDoubleWhole
U+E0EE	noteheadLargeArrowUpWhole
U+E0EF	noteheadLargeArrowUpHalf
U+E0F0	noteheadLargeArrowUpBlack
U+E0F1	noteheadLargeArrowDownDoubleWhole
U+E0F2	noteheadLargeArrowDownWhole
U+E0F3	noteheadLargeArrowDownHalf
U+E0F4	noteheadLargeArrowDownBlack

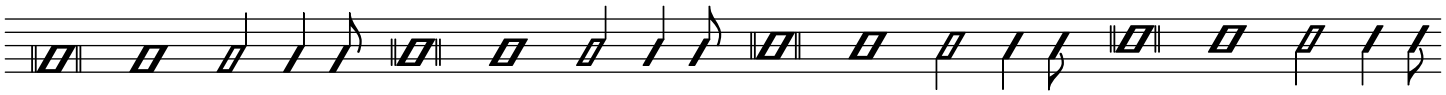
'arrow-up

'arrow-down



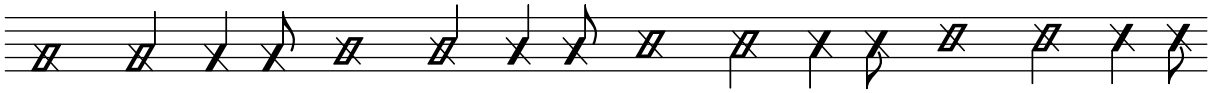
'slash

U+E10A noteheadSlashWhiteDoubleWhole
U+E102 noteheadSlashWhiteWhole
U+E103 noteheadSlashWhiteHalf
U+E101 noteheadSlashHorizontalEnds



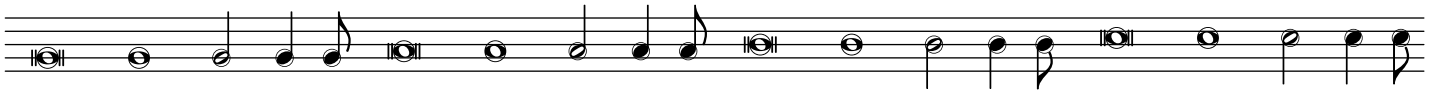
'slash-muted

U+E109 noteheadSlashWhiteMuted
U+E108 noteheadSlashHorizontalEndsMuted



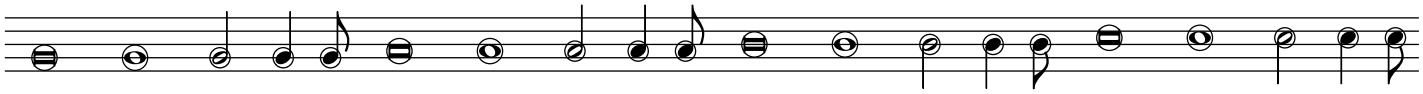
'circled

U+E0E7 noteheadCircledDoubleWhole
U+E0E6 noteheadCircledWhole
U+E0E5 noteheadCircledHalf
U+E0E4 noteheadCircledBlack



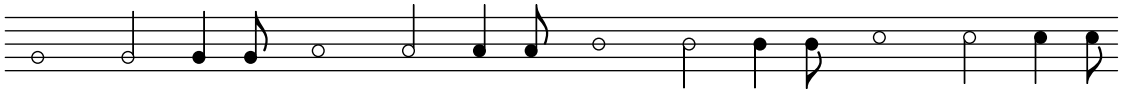
'circled-large

U+E0EB noteheadCircledDoubleWholeLarge
U+E0EA noteheadCircledWholeLarge
U+E0E9 noteheadCircledHalfLarge
U+E0E8 noteheadCircledBlackLarge



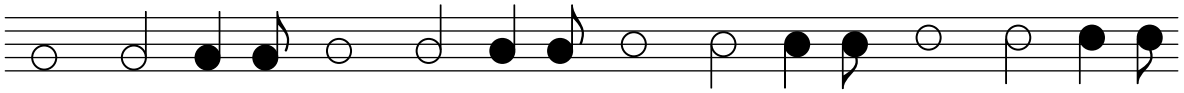
'round

U+E114 noteheadRoundWhite
U+E113 noteheadRoundBlack



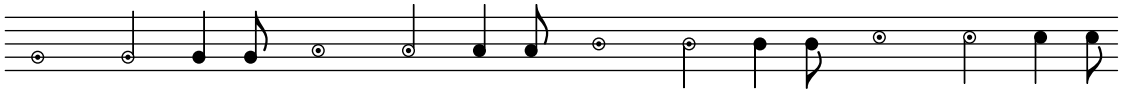
'round-large

U+E111 noteheadRoundWhiteLarge
U+E110 noteheadRoundBlackLarge



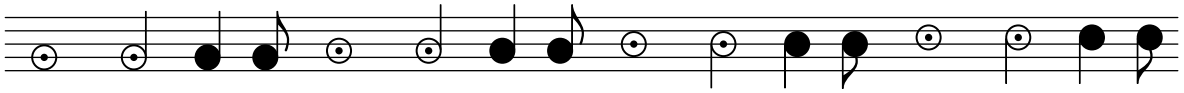
'round-dot

U+E115 noteheadRoundWhiteWithDot
U+E113 noteheadRoundBlack



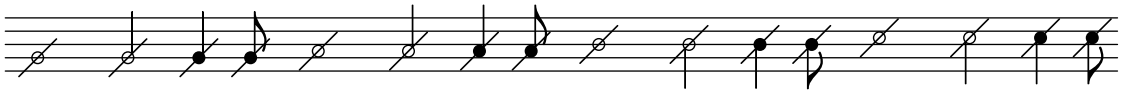
'round-dot-large

U+E112 noteheadRoundWhiteWithDotLarge
U+E110 noteheadRoundBlackLarge



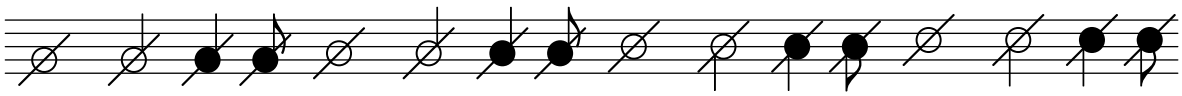
'round-slashed

U+E119 noteheadRoundWhiteSlashed
U+E118 noteheadRoundBlackSlashed



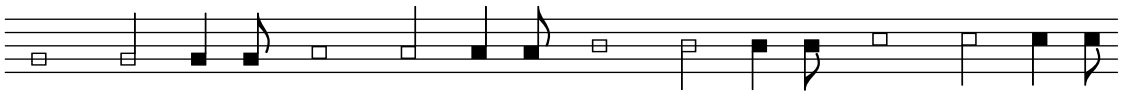
'round-slashed-large

U+E117 noteheadRoundWhiteSlashedLarge
U+E116 noteheadRoundBlackSlashedLarge



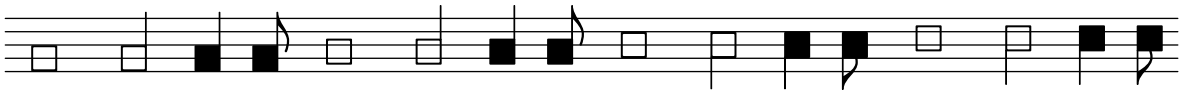
'square

U+E0B8 noteheadSquareWhite
U+E0B9 noteheadSquareBlack



'square-large

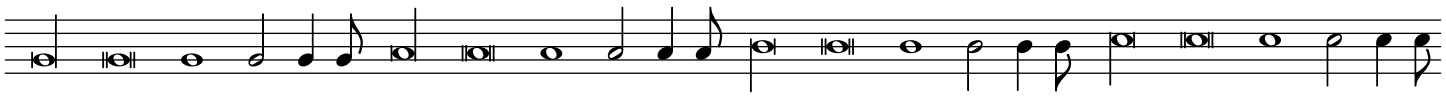
U+E11B noteheadSquareBlackWhite
U+E11A noteheadSquareBlackLarge



Ekmelos

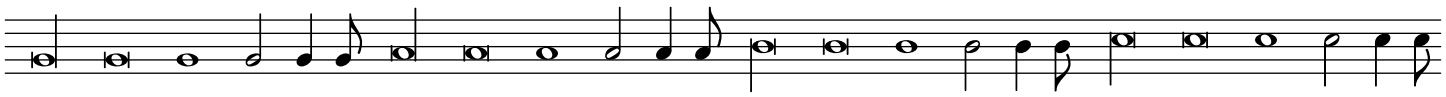
'default

U+F637 noteheadLongaUp
U+F638 noteheadLongaDown
:



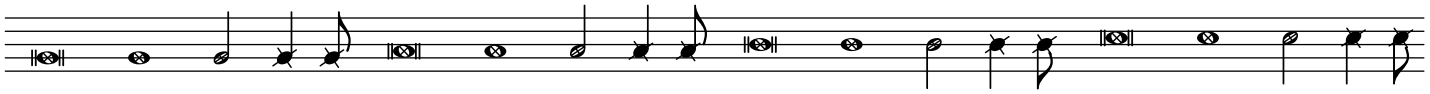
'altdefault

U+F637 noteheadLongaUp
U+F638 noteheadLongaDown
U+F639 noteheadDoubleWholeAlt
:



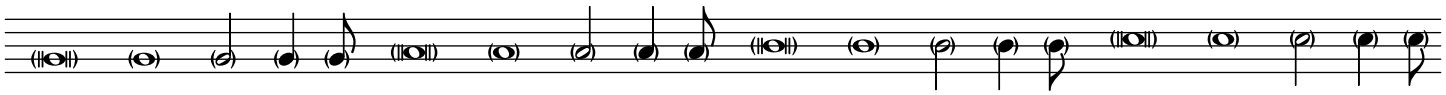
'withx-black

:
U+F680 noteheadBlackWithX



'parenthesised

U+F5DF noteheadDoubleWholeParens
U+F5DE noteheadWholeParens
U+F5DD noteheadHalfParens
U+F5DC noteheadBlackParens



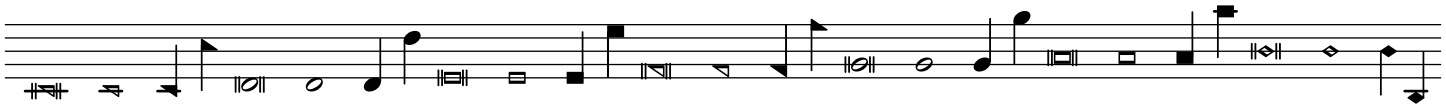
Shape note heads

All forms in LilyPond are supported, but some note heads of Feta don't have exact matches in SMuFL, e.g. the thin shapes of `\southernHarmonyHeads` and the reversed shapes for stem up of `\funkHeads` .

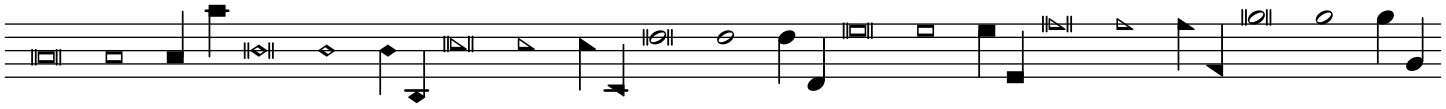
Sacred Harp

`\sacredHarpHeads`

fa	U+ECD3	noteShapeTriangleLeftDoubleWhole
	U+E1B6	noteShapeTriangleLeftWhite
	U+E1B7	noteShapeTriangleLeftBlack
	U+ECD2	noteShapeTriangleRightDoubleWhole
	U+E1B4	noteShapeTriangleRightWhite
	U+E1B5	noteShapeTriangleRightBlack
sol	U+ECD0	noteShapeRoundDoubleWhole
	U+E1B0	noteShapeRoundWhite
	U+E1B1	noteShapeRoundBlack
la	U+ECD1	noteShapeSquareDoubleWhole
	U+E1B2	noteShapeSquareWhite
	U+E1B3	noteShapeSquareBlack
mi	U+ECD4	noteShapeDiamondDoubleWhole
	U+E1B8	noteShapeDiamondWhite
	U+E1B9	noteShapeDiamondBlack



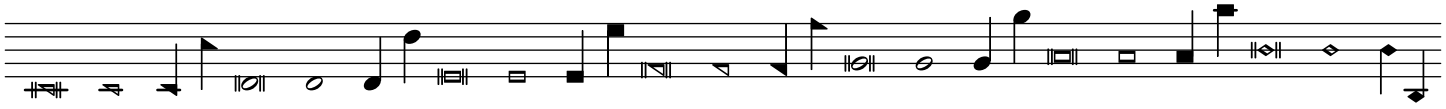
`\sacredHarpHeadsMinor`



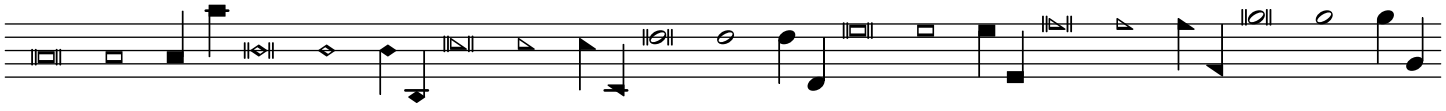
Southern Harmony

`\southernHarmonyHeads`

fa	U+ECD3	noteShapeTriangleLeftDoubleWhole
	U+E1B6	noteShapeTriangleLeftWhite
	U+E1B7	noteShapeTriangleLeftBlack
	U+ECD2	noteShapeTriangleRightDoubleWhole
	U+E1B4	noteShapeTriangleRightWhite
	U+E1B5	noteShapeTriangleRightBlack
sol	U+ECD0	noteShapeRoundDoubleWhole
	U+E1B0	noteShapeRoundWhite
	U+E1B1	noteShapeRoundBlack
la	U+ECD1	noteShapeSquareDoubleWhole
	U+E1B2	noteShapeSquareWhite
	U+E1B3	noteShapeSquareBlack
mi	U+ECD4	noteShapeDiamondDoubleWhole
	U+E1B8	noteShapeDiamondWhite
	U+E1B9	noteShapeDiamondBlack



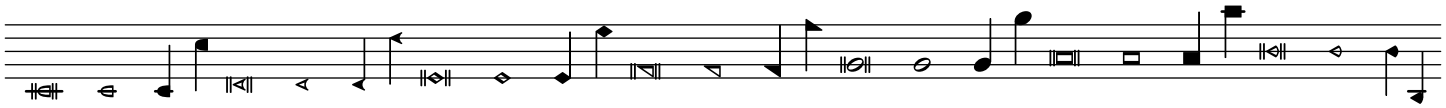
`\southernHarmonyHeadsMinor`



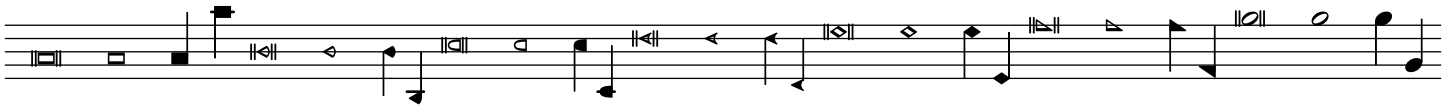
Funk (Harmonia Sacra)

`\funkHeads`

do	U+ECDB	noteShapeMoonLeftDoubleWhole
	U+E1C6	noteShapeMoonLeftWhite
	U+E1C7	noteShapeMoonLeftBlack
re	U+ECDC	noteShapeArrowheadLeftDoubleWhole
	U+E1C8	noteShapeArrowheadLeftWhite
	U+E1C9	noteShapeArrowheadLeftBlack
mi	U+ECD4	noteShapeDiamondDoubleWhole
	U+E1B8	noteShapeDiamondWhite
	U+E1B9	noteShapeDiamondBlack
fa	U+ECD3	noteShapeTriangleLeftDoubleWhole
	U+E1B6	noteShapeTriangleLeftWhite
	U+E1B7	noteShapeTriangleLeftBlack
	U+ECD2	noteShapeTriangleRightDoubleWhole
	U+E1B4	noteShapeTriangleRightWhite
	U+E1B5	noteShapeTriangleRightBlack
sol	U+ECD0	noteShapeRoundDoubleWhole
	U+E1B0	noteShapeRoundWhite
	U+E1B1	noteShapeRoundBlack
la	U+ECD1	noteShapeSquareDoubleWhole
	U+E1B2	noteShapeSquareWhite
	U+E1B3	noteShapeSquareBlack
ti	U+ECDD	noteShapeTriangleRoundLeftDoubleWhole
	U+E1CA	noteShapeTriangleRoundLeftWhite
	U+E1CB	noteShapeTriangleRoundLeftBlack



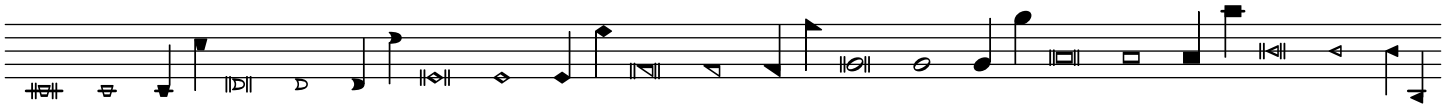
`\funkHeadsMinor`



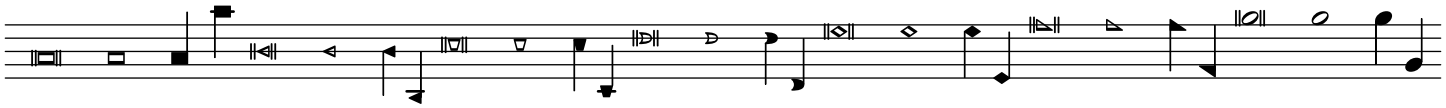
Walker

`\walkerHeads`

do	U+ECD8	noteShapeKeystoneDoubleWhole
	U+E1C0	noteShapeKeystoneWhite
	U+E1C1	noteShapeKeystoneBlack
re	U+ECD9	noteShapeQuarterMoonDoubleWhole
	U+E1C2	noteShapeQuarterMoonWhite
	U+E1C3	noteShapeQuarterMoonBlack
mi	U+ECD4	noteShapeDiamondDoubleWhole
	U+E1B8	noteShapeDiamondWhite
	U+E1B9	noteShapeDiamondBlack
fa	U+ECD3	noteShapeTriangleLeftDoubleWhole
	U+E1B6	noteShapeTriangleLeftWhite
	U+E1B7	noteShapeTriangleLeftBlack
	U+ECD2	noteShapeTriangleRightDoubleWhole
	U+E1B4	noteShapeTriangleRightWhite
	U+E1B5	noteShapeTriangleRightBlack
sol	U+ECD0	noteShapeRoundDoubleWhole
	U+E1B0	noteShapeRoundWhite
	U+E1B1	noteShapeRoundBlack
la	U+ECD1	noteShapeSquareDoubleWhole
	U+E1B2	noteShapeSquareWhite
	U+E1B3	noteShapeSquareBlack
ti	U+ECDA	noteShapeIsoscelesTriangleDoubleWhole
	U+E1C4	noteShapeIsoscelesTriangleWhite
	U+E1C5	noteShapeIsoscelesTriangleBlack



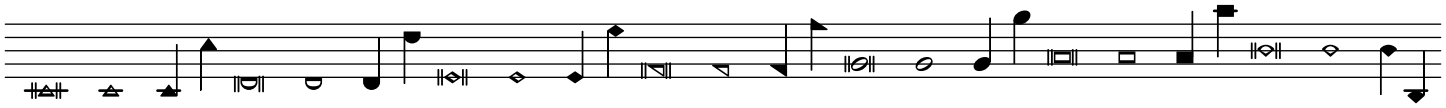
`\walkerHeadsMinor`



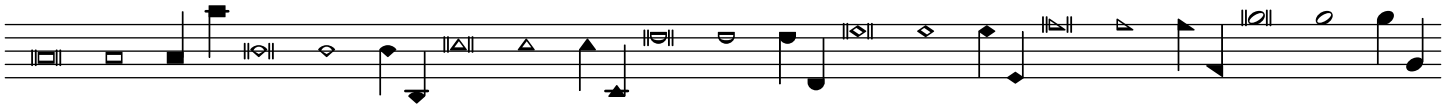
Aiken (Christian Harmony)

`\aikenHeads`

do	U+ECD5	noteShapeTriangleUpDoubleWhole
	U+E1BA	noteShapeTriangleUpWhite
	U+E1BB	noteShapeTriangleUpBlack
re	U+ECD6	noteShapeMoonDoubleWhole
	U+E1BC	noteShapeMoonWhite
	U+E1BD	noteShapeMoonBlack
mi	U+ECD4	noteShapeDiamondDoubleWhole
	U+E1B8	noteShapeDiamondWhite
	U+E1B9	noteShapeDiamondBlack
fa	U+ECD3	noteShapeTriangleLeftDoubleWhole
	U+E1B6	noteShapeTriangleLeftWhite
	U+E1B7	noteShapeTriangleLeftBlack
	U+ECD2	noteShapeTriangleRightDoubleWhole
	U+E1B4	noteShapeTriangleRightWhite
	U+E1B5	noteShapeTriangleRightBlack
sol	U+ECD0	noteShapeRoundDoubleWhole
	U+E1B0	noteShapeRoundWhite
	U+E1B1	noteShapeRoundBlack
la	U+ECD1	noteShapeSquareDoubleWhole
	U+E1B2	noteShapeSquareWhite
	U+E1B3	noteShapeSquareBlack
ti	U+ECD7	noteShapeTriangleRoundDoubleWhole
	U+E1BE	noteShapeTriangleRoundWhite
	U+E1BF	noteShapeTriangleRoundBlack



`\aikenHeadsMinor`



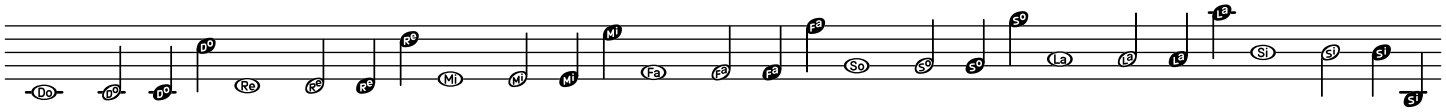
Note name note heads

`\ekmNameHeads...`

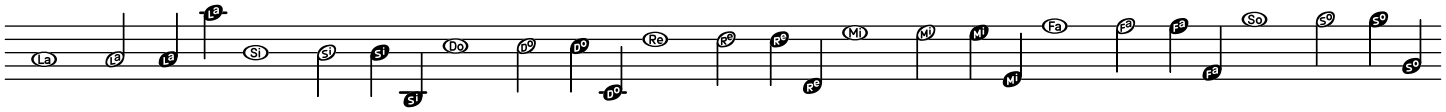
Draw note heads with solfège (easy play) note names. [Err]

`\ekmNameHeads`

do	U+E150	noteDoWhole
	U+E158	noteDoHalf
	U+E160	noteDoBlack
re	U+E151	noteReWhole
	U+E159	noteReHalf
	U+E161	noteReBlack
mi	U+E152	noteMiWhole
	U+E15A	noteMiHalf
	U+E162	noteMiBlack
fa	U+E153	noteFaWhole
	U+E15B	noteFaHalf
	U+E163	noteFaBlack
so	U+E154	noteSoWhole
	U+E15C	noteSoHalf
	U+E164	noteSoBlack
la	U+E155	noteLaWhole
	U+E15D	noteLaHalf
	U+E165	noteLaBlack
si	U+E157	noteSiWhole
	U+E15F	noteSiHalf
	U+E167	noteSiBlack

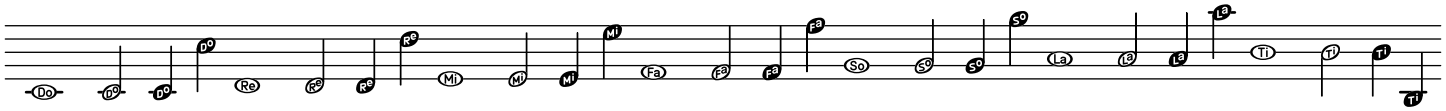


`\ekmNameHeadsMinor`

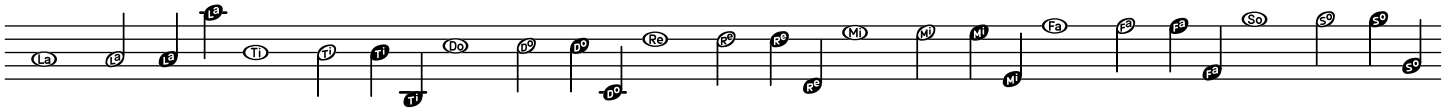


`\ekmNameHeadsTi`

do ... la	:	
ti	U+E156	noteTiWhole
	U+E15E	noteTiHalf
	U+E166	noteTiBlack



`\ekmNameHeadsTiMinor`



Note clusters

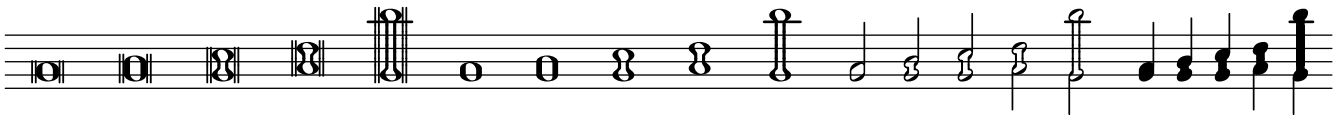
`\ekmMakeClusters MUSIC`

Draw clusters instead of chords in MUSIC, consisting of a bottom and a top note head, and ignoring inner notes of the chords ('Cowell clusters').

Styles

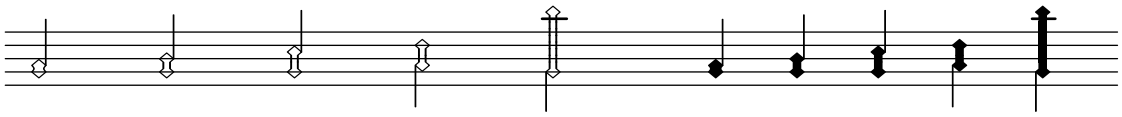
`'default`

U+E124	noteheadClusterDoubleWhole2nd
U+E128	noteheadClusterDoubleWhole3rd
U+E12C	noteheadClusterDoubleWholeTop
U+E12D	noteheadClusterDoubleWholeMiddle
U+E12E	noteheadClusterDoubleWholeBottom
U+E125	noteheadClusterWhole2nd
U+E129	noteheadClusterWhole3rd
U+E12F	noteheadClusterWholeTop
U+E130	noteheadClusterWholeMiddle
U+E131	noteheadClusterWholeBottom
U+E126	noteheadClusterHalf2nd
U+E12A	noteheadClusterHalf3rd
U+E132	noteheadClusterHalfTop
U+E133	noteheadClusterHalfMiddle
U+E134	noteheadClusterHalfBottom
U+E127	noteheadClusterQuarter2nd
U+E12B	noteheadClusterQuarter3rd
U+E135	noteheadClusterQuarterTop
U+E136	noteheadClusterQuarterMiddle
U+E137	noteheadClusterQuarterBottom



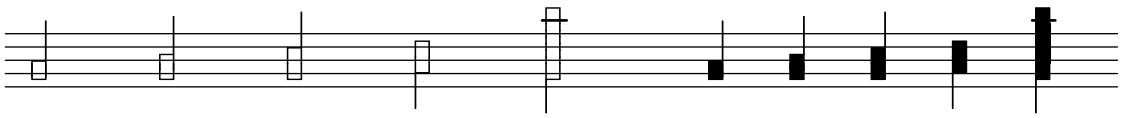
`'harmonic`

U+E138	noteheadDiamondClusterWhite2nd
U+E13A	noteheadDiamondClusterWhite3rd
U+E13C	noteheadDiamondClusterWhiteTop
U+E13D	noteheadDiamondClusterWhiteMiddle
U+E13E	noteheadDiamondClusterWhiteBottom
U+E139	noteheadDiamondClusterBlack2nd
U+E13B	noteheadDiamondClusterBlack3rd
U+E13F	noteheadDiamondClusterBlackTop
U+E140	noteheadDiamondClusterBlackMiddle
U+E141	noteheadDiamondClusterBlackBottom



`'square`

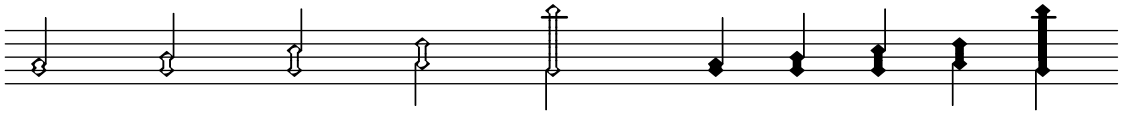
U+E145	noteheadRectangularClusterWhiteTop
U+E146	noteheadRectangularClusterWhiteMiddle
U+E147	noteheadRectangularClusterWhiteBottom
U+E142	noteheadRectangularClusterBlackTop
U+E143	noteheadRectangularClusterBlackMiddle
U+E144	noteheadRectangularClusterBlackBottom



Ekmelos

`'diamond`	U+F64B	noteheadDiamondClusterHalf2nd
	U+F64C	noteheadDiamondClusterHalf3rd
	U+F64D	noteheadDiamondClusterHalfTop
	U+F64E	noteheadDiamondClusterHalfMiddle
	U+F64F	noteheadDiamondClusterHalfBottom

:



Note:
For intervals larger than a third (except for `square`) the drawn cluster is a stack of one bottom segment, M middle segments, and one top segment. Mid and Top are the staff positions of the middle and top segments relative to the bottom segment.

Interval	M	Mid	Top
4th	0	-	3
5th	1	2	4
6th	2	2 3	5
7th	3	2 3 4	6
octave	4	2 3 4 5	7

...

The segment glyphs in Ekmelos are designed for these values.
However, in the implementation notes of SMuFL Note clusters, the octave cluster is said to have 3 middle segments, while the 6th cluster has 2 middle segments. The “appropriate number of middle segments” varies apparently depending on the font.

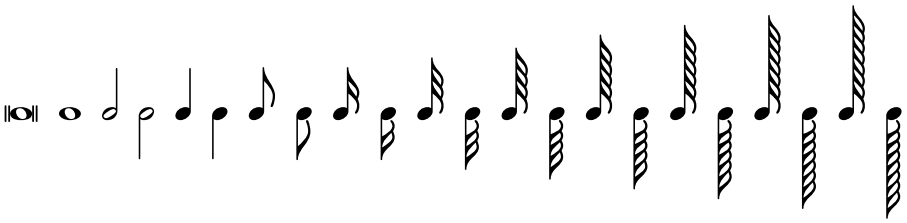
Note head markup

`\ekm-note-by-number STYLE LOG DOTS DIRECTION`

Draw a note with augmentation dots as markup. It does not support stem lengths. STYLE can also be one of the note head styles . LOG is usually in the range -1 to 10.

Styles

`'note`	U+E1D0	noteDoubleWhole
	U+E1D2	noteWhole
	U+E1D3	noteHalfUp
	U+E1D4	noteHalfDown
	:	
	U+E1E6	note1024thDown

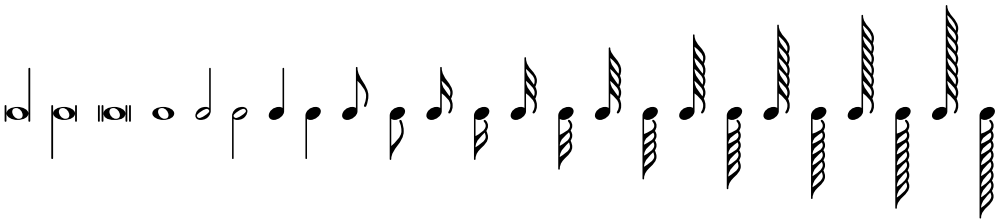


`'metronome`	U+ECA0	metNoteDoubleWhole
	U+ECA2	metNoteWhole
	U+ECA3	metNoteHalfUp
	U+ECA4	metNoteHalfDown
	:	
	U+ECB6	metNote1024thDown

Ekmelos

'note

U+F637 noteheadLongaUp
U+F638 noteheadLongaDown
:



'straight

U+F637 noteheadLongaUp
:
U+F683 note8thUpStraight
U+F686 note16thUpStraight
U+F689 note32ndUpStraight

'short

U+F637 noteheadLongaUp
:
U+F684 note8thUpShort
U+F687 note16thUpShort
U+F68A note32ndUpShort

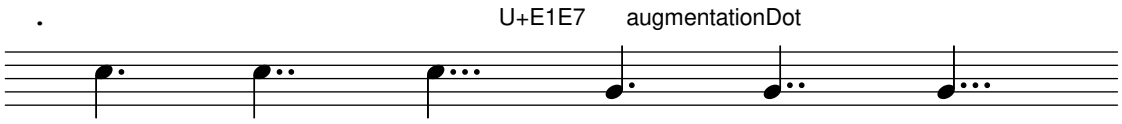
'beamed

U+F637 noteheadLongaUp
:
U+F685 note8thUpBeamed
U+F688 note16thUpBeamed
U+F68B note32ndUpBeamed

Augmentation dots

`\ekmSmuflOn #'dot`

Draw SMuFL augmentation dots.



Flags and Grace note slashes

`\ekmSmuflOn #'flag`

Draw SMuFL flags and grace note slashes with `\slashedGrace`.

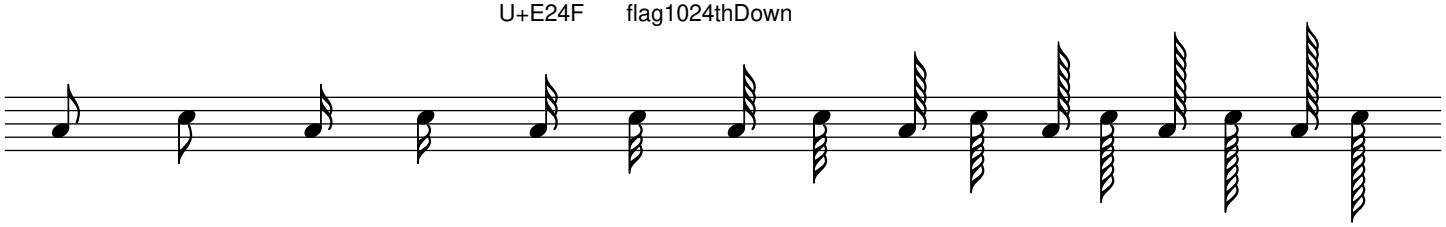
`\ekmFlag STYLE`

Set the specified flag style. It overrides the properties `Flag.style` and `Stem.details.lengths`. [Err]

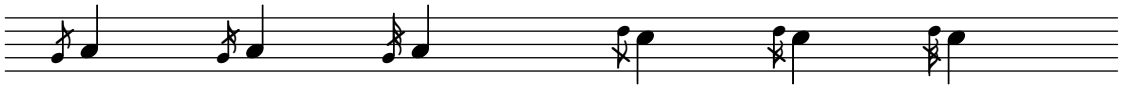
Styles

`'default`

U+E240	flag8thUp
U+E241	flag8thDown
U+E242	flag16thUp
U+E243	flag16thDown
U+E244	flag32ndUp
U+E245	flag32ndDown
U+E246	flag64thUp
U+E247	flag64thDown
U+E248	flag128thUp
U+E249	flag128thDown
U+E24A	flag256thUp
U+E24B	flag256thDown
U+E24C	flag512thUp
U+E24D	flag512thDown
U+E24E	flag1024thUp
U+E24F	flag1024thDown

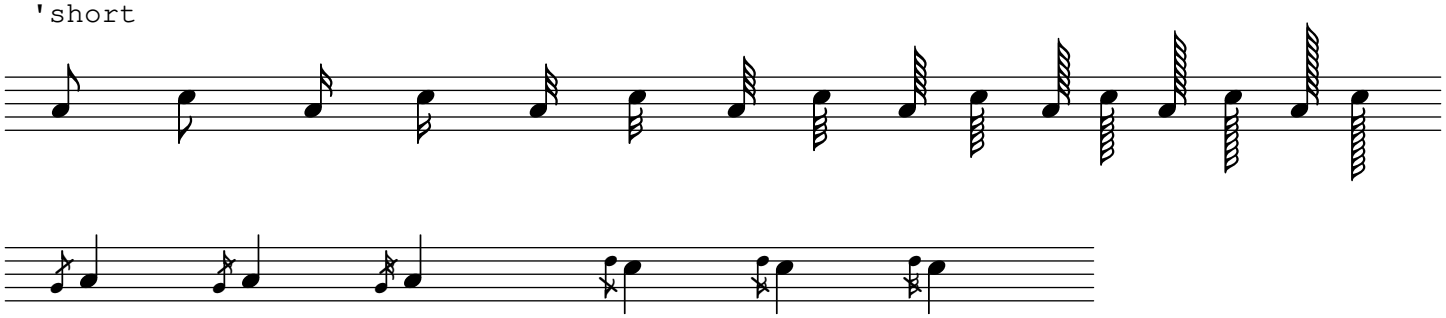


U+E564	graceNoteSlashStemUp
U+E565	graceNoteSlashStemDown

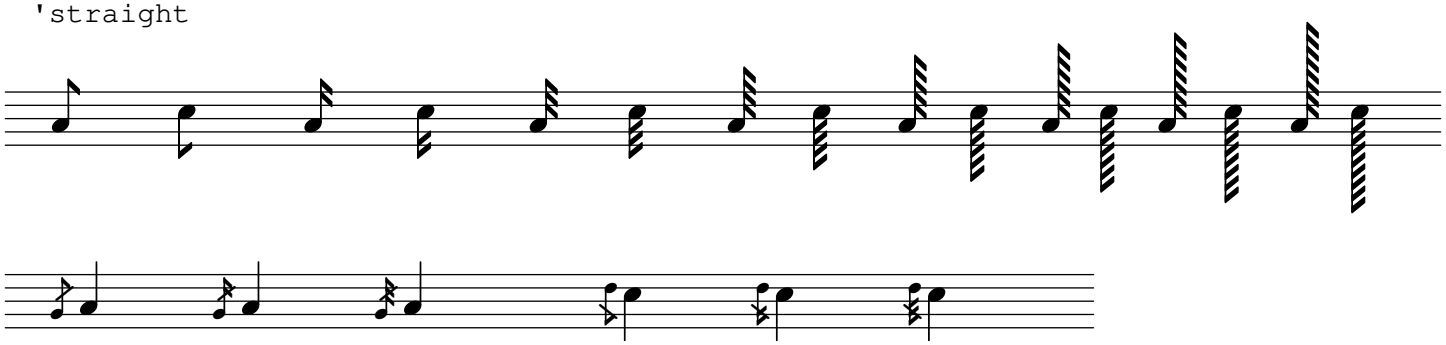


Bravura and Ekmelos

'short



'straight



Rests

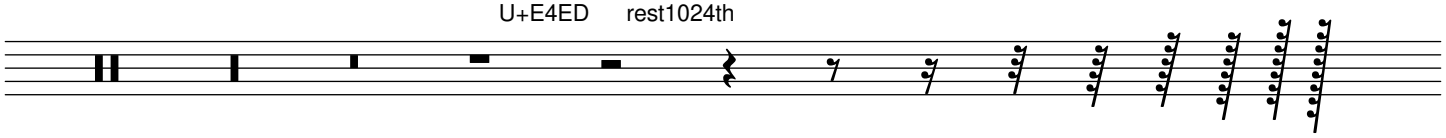
`\ekmSmuflOn #'rest`

Draw SMuFL rests and multi-measure rests, as well as SMuFL time signature digits for multi-measure rest numbers.

Styles

`'default`

U+E4E0	restMaxima
U+E4E1	restLonga
U+E4E2	restDoubleWhole
U+E4E3	restWhole
U+E4E4	restHalf
U+E4E5	restQuarter
U+E4E6	rest8th
U+E4E7	rest16th
U+E4E8	rest32nd
U+E4E9	rest64th
U+E4EA	rest128th
U+E4EB	rest256th
U+E4EC	rest512th
U+E4ED	rest1024th

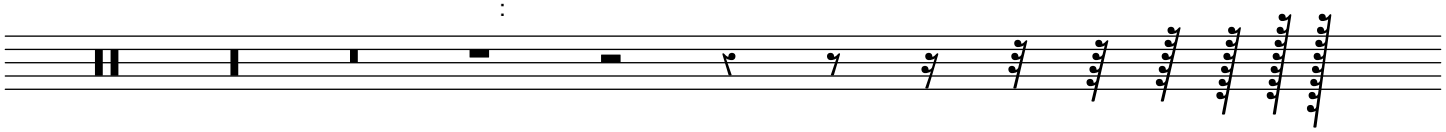


`'classical`

:

U+E4F2	restQuarterOld

:

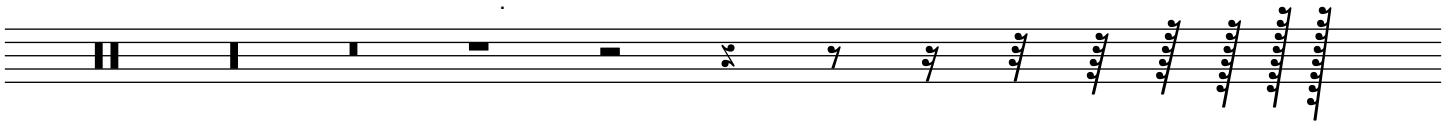


`'z`

:

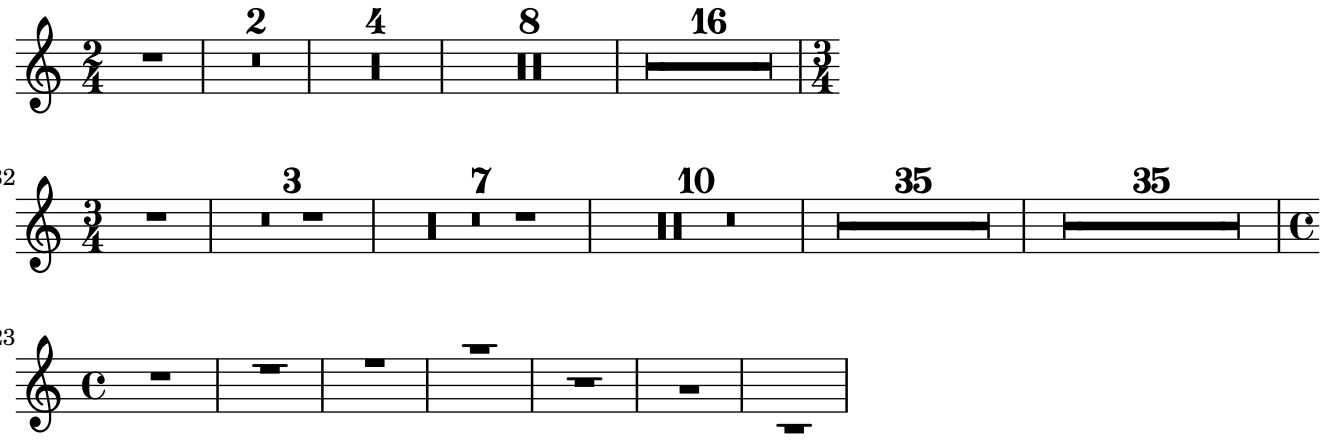
U+E4F6	restQuarterZ

:



In the following example, the time signatures are LilyPond's Emmentaler glyphs while the multi-measure rest numbers are SMuFL glyphs.

```
\relative c'' {
  \ekmSmuflOn #'rest

  \compressMMRests {
    \time 2/4
    R2 R1 R\breve R\longa R\maxima
    \break

    \time 3/4
    R2. R2.*3 R2.*7 R2.*10
    R2.*35
    \override MultiMeasureRest.space-increment = 2.5
    R2.*35
    \break
  }

  \time 4/4
  R1
  \override MultiMeasureRest.staff-position = #1
  R1
  \override MultiMeasureRest.staff-position = #2
  R1
  \override MultiMeasureRest.staff-position = #4
  R1
  \override MultiMeasureRest.staff-position = #-1
  R1
  \override MultiMeasureRest.staff-position = #-2
  R1
  \override MultiMeasureRest.staff-position = #-8
  R1
}
```



`\ekm-rest-by-number LOG DOTS`

Draw a rest with augmentation dots as markup. LOG is in the range -3 to 10. The dots are vertically centered, contrary to `\rest-by-number` .

Used properties:

- `font-size` (0)
- `ledgers` ('(-1 0 1))
- `style` ('())

`\ekm-multi-measure-rest-by-number MEASURES`

Draw a multi-measure rest as markup, with the number placed centered above unless it is 1.

Used properties:

- `font-size` (0)
- `expand-limit` (10)
- `style` ('())
- `word-space`
- `width` (8)
- `multi-measure-rest-number` (#t)

`\ekm-rest DURATION`

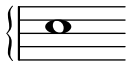
Draw either a rest or a multi-measure rest as markup.

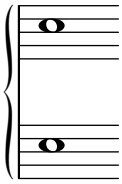
```
\ekm-rest-by-number #-1 #1
\ekm-rest-by-number #2 #2
\ekm-rest-by-number #3 #1

\ekm-multi-measure-rest-by-number #7
\ekm-multi-measure-rest-by-number #16

\ekm-rest { \breve. }
\ekm-rest { 4.. }
\ekm-rest { 8. }

\override #'(multi-measure-rest . #t)
\override #'(multi-measure-rest-number . #f)
\ekm-rest { 1*7 }

\override #'(multi-measure-rest . #t)
\ekm-rest { 1*16 }
```

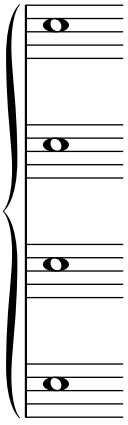
System start delimiters

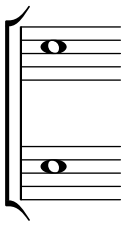
`\ekmSmuflOn #'systemstart`

Draw SMuFL system start delimiters, using `\ekm-system-start` .









`\ekm-system-start STYLE HEIGHT`

Draw a system start delimiter of HEIGHT in staff units as markup.
Used property:

- `font-size` (0)

Styles

`'brace`		U+E000	brace
`'bracket`		U+E003	bracketTop
		U+E004	bracketBottom

Bravura and Ekmelos

`'brace` makes use of Bravura's stylistic alternates or Ekmelos' size variants, each intended for a specific size range.

Dynamics

`\ekmSmuflOn #'dynamic`

Draw SMuFL absolute dynamic marks, using `\ekm-dynamic` .

`\ekm-dynamic DEFINITION`

Draw a dynamic symbol according to DEFINITION as markup.
DEFINITION must be either a single token or a sequence of the letters f, m, n, p, r, s, z, whose corresponding symbols are concatenated. This is slightly different from the usual interpretation of definition strings.

p	𝆐	U+E520	dynamicPiano
m	𝆑	U+E521	dynamicMezzo
f	𝆑	U+E522	dynamicForte
r	𝆒	U+E523	dynamicRinforzando
s	𝆓	U+E524	dynamicSforzando
z	𝆔	U+E525	dynamicZ
n	𝆕	U+E526	dynamicNiente
mp	𝆏	U+E52C	dynamicMP
mf	𝆎	U+E52D	dynamicMF
pf	𝆖	U+E52E	dynamicPF
fp	𝆗	U+E534	dynamicFortePiano
pppppp	𝆯	U+E527	dynamicPPPPPP
	:		
pp	𝆏	U+E52B	dynamicPP
ff	𝆑	U+E52F	dynamicFF
	:		
ffffff	𝆝	U+E533	dynamicFFFFFF
fz	𝆘	U+E535	dynamicForzando
sf	𝆓	U+E536	dynamicSforzando1
sfp	𝆔	U+E537	dynamicSforzandoPiano
sfpp	𝆕	U+E538	dynamicSforzandoPianissimo
sfz	𝆓	U+E539	dynamicSforzato
sfzp	𝆔	U+E53A	dynamicSforzatoPiano
sffz	𝆑	U+E53B	dynamicSforzatoFF
rf	𝆒	U+E53C	dynamicRinforzando1
rfz	𝆒	U+E53D	dynamicRinforzando2

Ekmelos

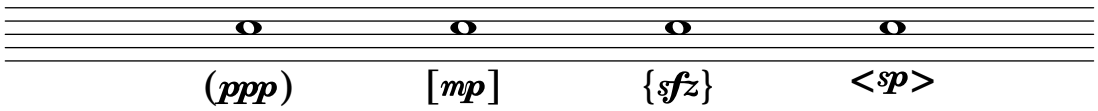
sff	𝆑	U+F645	dynamicSforzandoFF
sp	𝆐	U+F646	dynamicSP
spp	𝆏	U+F647	dynamicSPP
sfffz	𝆝	U+F6F4	dynamicSforzatoFFF
sffffz	𝆝	U+F6F5	dynamicSforzatoFFFF

`\ekmParensDyn STYLE DYNAMIC-MARK`

Draw the absolute dynamic mark parenthesized.

Styles

```
'default
'bracket
'brace
'angle
```

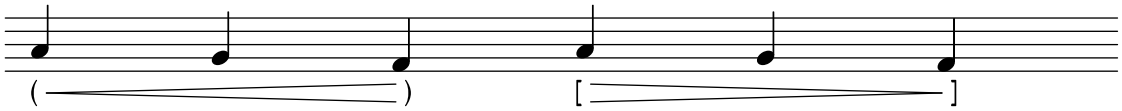


`\ekmParensHairpin STYLE`

Draw the subsequent hairpin parenthesized.

Styles

`'default`	U+E542	dynamicHairpinParenthesisLeft
	U+E543	dynamicHairpinParenthesisRight
`'bracket`	U+E544	dynamicHairpinBracketLeft
	U+E545	dynamicHairpinBracketRight



Scripts - Expressive marks

`\ekmSmuflOn #'script`

Draw SMuFL scripts for expressive marks like articulations, ornamentations, performance indications, fermatas, repeat signs, etc.

```
\ekmScript SCRIPT-NAME #'(EXTEXT-UP . EXTEXT-DOWN)
\ekmScript SCRIPT-NAME EXTEXT
```

Create a script from EXTEXT, either a pair for up and down, or a single value for both directions. If the latter is a list it must be enclosed in a list. SCRIPT-NAME is the symbol of an existing script like `accent`, `marcato`, `trill`, `turn`, `upbow`, `open`, `segno`, etc. It determines the vertical positioning of the script. [Ly]

```
\ekmScriptSmall SCRIPT-NAME #'(EXTEXT-UP . EXTEXT-DOWN)
\ekmScriptSmall SCRIPT-NAME EXTEXT
```

Create a script with a 3 steps smaller font size. [Ly]

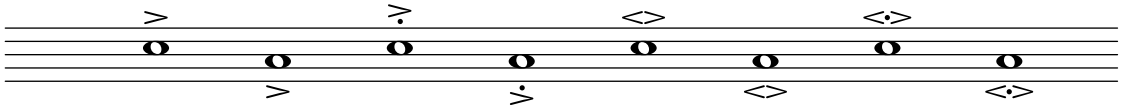
Articulations

`\accent` U+E4A0 articAccentAbove
U+E4A1 articAccentBelow

`\ekmScript #'accent #'(#xE4B0 . #xE4B1)`

`\espressivo` U+ED40 articSoftAccentAbove
U+ED41 articSoftAccentBelow

`\ekmScript #'espressivo #'(#xED42 . #xED43)`

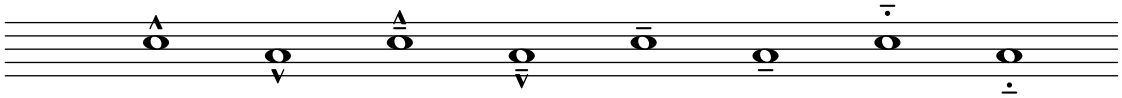


`\marcato` U+E4AC articMarcatoAbove
U+E4AD articMarcatoBelow

`\ekmScript #'marcato #'(#xE4BC . #xE4BD)`

`\tenuto` U+E4A4 articTenutoAbove
U+E4A5 articTenutoBelow

`\portato` U+E4B2 articTenutoStaccatoAbove
U+E4B3 articTenutoStaccatoBelow

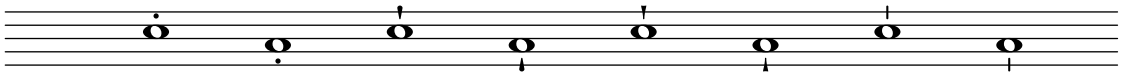


`\staccato` U+E4A2 articStaccatoAbove
U+E4A3 articStaccatoBelow

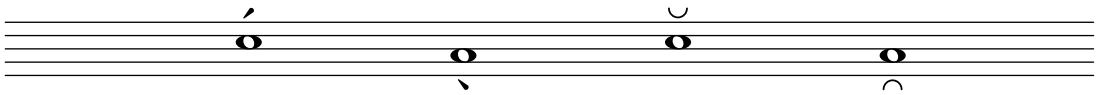
`\staccatissimo` U+E4A6 articStaccatissimoAbove
U+E4A7 articStaccatissimoBelow

`\ekmScript #'staccatissimo #'(#xE4A8 . #xE4A9)`

`\ekmScript #'staccatissimo #'(#xE4AA . #xE4AB)`

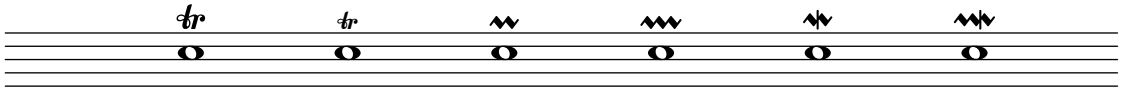


```
\ekmScript #'accent #'(#xE4B6 . #xE4B7)
\ekmScript #'accent #'(#xE4B8 . #xE4B9)
```



Ornamentations

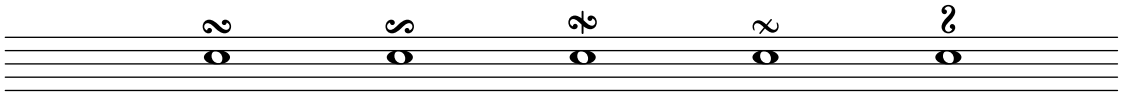
`\trill`		U+E566	ornamentTrill
`\ekmScriptSmall #'trill ##xE566`			
`\prall`		U+E56C	ornamentShortTrill
`\prallprall`		U+E56E	ornamentTremblement
`\mordent`		U+E56D	ornamentMordent
`\prallmordent`		U+E5BD	ornamentPrecompTrillWithMordent



`\upprall`		U+E59A	ornamentBottomLeftConcaveStroke
	2 ×	U+E59D	ornamentZigZagLineNoRightEnd
		U+E59E	ornamentZigZagLineWithRightEnd
`\downprall`		U+E5C6	ornamentPrecompMordentUpperPrefix
`\upmordent`		U+E5B8	ornamentPrecompSlideTrillBach
`\downmordent`		U+E5C7	ornamentPrecompInvertedMordentUpperPrefix
`\prallup`	3 ×	U+E59D	ornamentZigZagLineNoRightEnd
		U+E5A4	ornamentRightVerticalStroke
`\pralldown`		U+E5C8	ornamentPrecompTrillLowerSuffix
`\lineprall`		U+E5B2	ornamentPrecompAppoggTrill

`\turn`	U+E567	ornamentTurn
`\reverseturn`	U+E568	ornamentTurnInverted
`\slashturn`	U+E569	ornamentTurnSlash
`\haydnturn`	U+E56F	ornamentHaydn

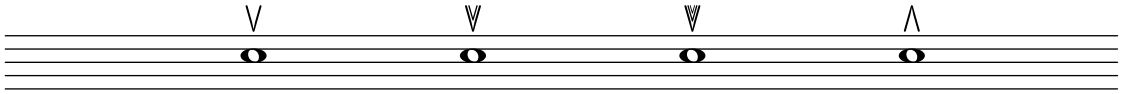
```
\ekmScript #'turn ##xE56A
```



Performance indications

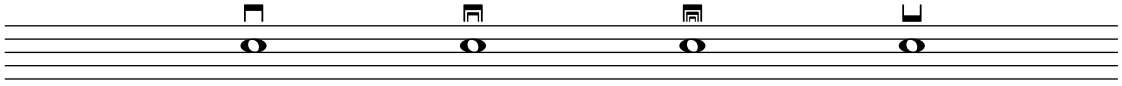
\upbow	U+E612	stringsUpBow

```
\ekmScript #'upbow ##xE61C
\ekmScript #'upbow ##xE61E
\ekmScript #'upbow ##xE613
```

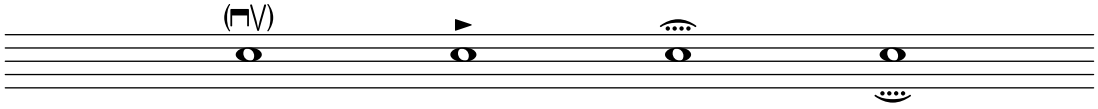


\downbow	U+E610	stringsDownBow

```
\ekmScript #'downbow ##xE61B
\ekmScript #'downbow ##xE61D
\ekmScript #'downbow ##xE611
```



```
\ekmScript #'downbow ##xE626
\ekmScript #'upbow ##xE61F
\ekmScript #'downbow #'(#xE620 . #xE621)
```

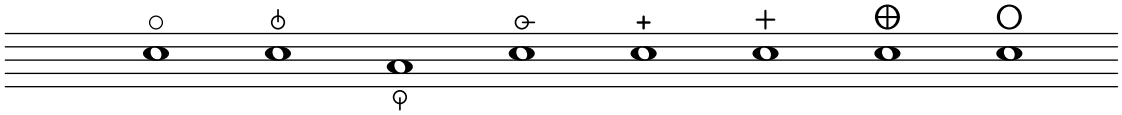


\flageolet	U+E614	stringsHarmonic
\snappizzicato	U+E631	pluckedSnapPizzicatoAbove
	U+E630	pluckedSnapPizzicatoBelow

```
\ekmScript #'snappizzicato ##xE632
```

\stopped	U+E633	pluckedLeftHandPizzicato

```
\ekmScript #'stopped ##xE5E5
\ekmScript #'halfopen ##xE5E6
\ekmScript #'open ##xE5E7
```

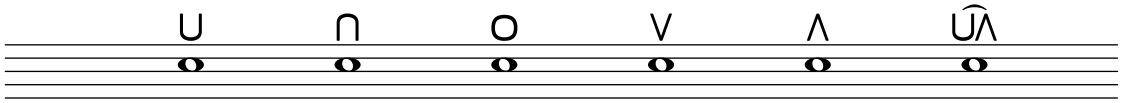


\lheel	U+E661	keyboardPedalHeel1
\rheel	U+E662	keyboardPedalHeel2

```
\ekmScript #'lheel ##xE663
```

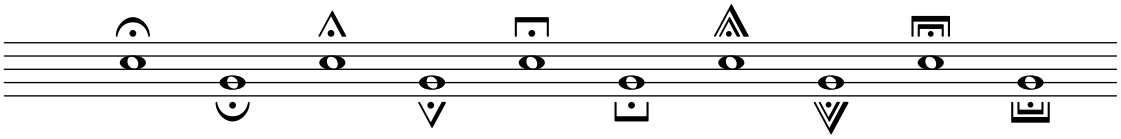
\ltoe	U+E664	keyboardPedalToe1
\rtoe	U+E665	keyboardPedalToe2

```
\ekmScript #'rtoe ##xE674
```



Fermatas

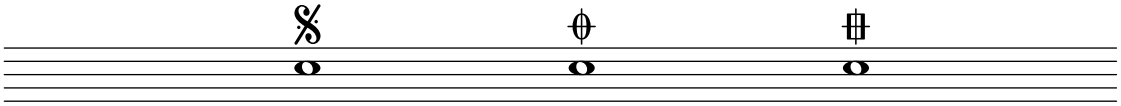
`\fermata`	U+E4C0	fermataAbove
	U+E4C1	fermataBelow
`\shortfermata`	U+E4C4	fermataShortAbove
	U+E4C5	fermataShortBelow
`\longfermata`	U+E4C6	fermataLongAbove
	U+E4C7	fermataLongBelow
`\veryshortfermata`	U+E4C2	fermataVeryShortAbove
	U+E4C3	fermataVeryShortBelow
`\verylongfermata`	U+E4C8	fermataVeryLongAbove
	U+E4C9	fermataVeryLongBelow



`\henzeshortfermata`	U+E4CC	fermataShortHenzeAbove
	U+E4CD	fermataShortHenzeBelow
`\henzelongfermata`	U+E4CA	fermataLongHenzeAbove
	U+E4CB	fermataLongHenzeBelow

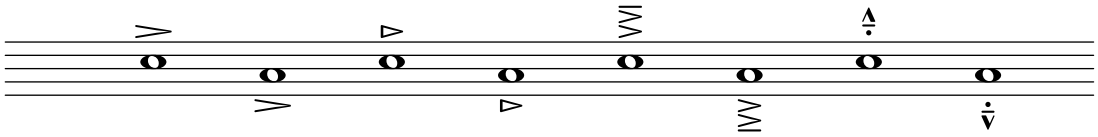
Repeat signs

`\segno`	U+E047	segno
`\coda`	U+E048	coda
`\varcoda`	U+E049	codaSquare

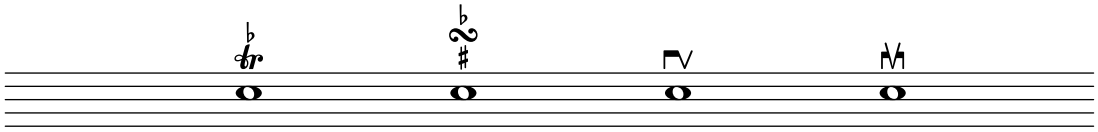


Ekmelos (Examples)

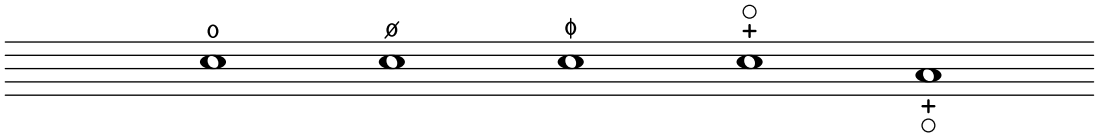
```
\ekmScript #'accent #'((#xE4A0 1) . (#xE4A1 1))
\ekmScript #'accent #'((#xE4A0 2) . (#xE4A1 2))
\ekmScript #'accent #'((#xE4A4 #xE4A0 #xE4A0) .
                       (#xE4A5 #xE4A1 #xE4A1))
\ekmScript #'portato #'((#xE4AC #xE4A4 #xE4A2) .
                        (#xE4AD #xE4A5 #xE4A3))
```



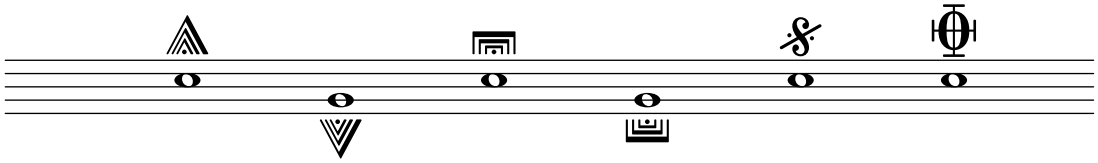
```
\ekmScript #'trill #'((#xE260 #xE566))
\ekmScript #'turn #'((#xE260 #xE567 #xE262))
\ekmScript #'downbow #'((#xE626 1))
\ekmScript #'downbow #'((#xE626 2))
```



```
\open
\halfopen
#(make-articulation 'halfopenvertical)
\ekmScript #'stopped #'((#xE614 #xE633) . (#xE633 #xE614))
```



```
\ekmScript #'veryshortfermata #'(#xF69E . #xF69F)
\ekmScript #'verylongfermata #'(#xF6A0 . #xF6A1)
\ekmScript #'segno #'((#xE047 1))
\ekmScript #'coda #'((#xE048 1))
```



Multi-segment spanner

`\ekmSmuflOn #'textspan`

Draw text spans assembled from SMuFL multi-segment glyphs.

`\ekmSmuflOn #'trill`

Draw trill spans assembled from SMuFL multi-segment glyphs, and SMuFL trill pitches.
See also Trill spans and pitches.

`\ekmStartSpan STYLE TEMPO ATTACHMENT`

Start a text span or trill span. [Ly]
The style `trill` starts a trill span. A second style can be added to draw other glyphs, e.g. `trill-vibrato-large` . Any other style starts a text span. An undefined style like `dashed-line` produces normal LilyPond output.
TEMPO is a number or a pair of numbers (rounded to integer) for the segments of the spanner. 0 is the main (medium) segment. Positive values mean faster (narrower) segments. Negative values mean slower (wider) segments. A pair `'(A . B)` draws all segments from A through B, evenly distributed over the spanner. The available range of numbers depends on the style.
ATTACHMENT is an EXTEXT or a pair `'(EXTEXT-LEFT . EXTEXT-RIGHT)` for the edge symbols. A single EXTEXT is equivalent to `'(EXTEXT . 0)`. It must be specified in a pair if itself is a list.
`#f` draws the standard glyph on the left and right edge according to the style.

`\ekmStartSpanMusic STYLE TEMPO ATTACHMENT MUSIC`

Start a text span or trill span at MUSIC. [Ly]
This is a music function that doesn't need the `textspan` or `trill` SMuFL switch turned on.

Styles

`'trill`			
left	𝆗𝅥𝆖	U+E566	ornamentTrill
4	𝅘𝅥𝅯	U+EAA0	wiggleTrillFastest
	:		
0	𝅘𝅥𝅰	U+EAA4	wiggleTrill
	:		
−4	𝅘𝅥𝅱	U+EAA8	wiggleTrillSlowest
`'vibrato`			
left	𝅘𝅥𝅲	U+EACC	wiggleVibratoStart
3	𝅘𝅥𝅳	U+EADB	wiggleVibratoMediumFastest
	:		
0	𝅘𝅥𝅴	U+EADE	wiggleVibratoMediumFast
	:		
−3	𝅘𝅥𝅵	U+EAE1	wiggleVibratoMediumSlowest

'vibrato-small

left		U+EACC	wiggleVibratoStart
3		U+EAD4	wiggleVibratoSmallFastest
	:		
0		U+EAD7	wiggleVibratoSmallFast
	:		
−3		U+EADA	wiggleVibratoSmallSlowest

'vibrato-smallest

left		U+EACC	wiggleVibratoStart
3		U+EACD	wiggleVibratoSmallestFastest
	:		
0		U+EAD0	wiggleVibratoSmallestFast
	:		
−3		U+EAD3	wiggleVibratoSmallestSlowest

'vibrato-large

left		U+EACC	wiggleVibratoStart
3		U+EAE2	wiggleVibratoLargeFastest
	:		
0		U+EAE5	wiggleVibratoLargeFast
	:		
−3		U+EAE8	wiggleVibratoLargeSlowest

'vibrato-largest

left		U+EACC	wiggleVibratoStart
3		U+EAE9	wiggleVibratoLargestFastest
	:		
0		U+EAEC	wiggleVibratoLargestFast
	:		
−3		U+EAEF	wiggleVibratoLargestSlowest

'circular

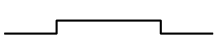
left		U+EAC4	wiggleCircularStart
4		U+EACA	wiggleCircularSmall
	:		
0		U+EAC9	wiggleCircular
	:		
−4		U+EAC5	wiggleCircularLargest
right		U+EACB	wiggleCircularEnd

'circular-constant

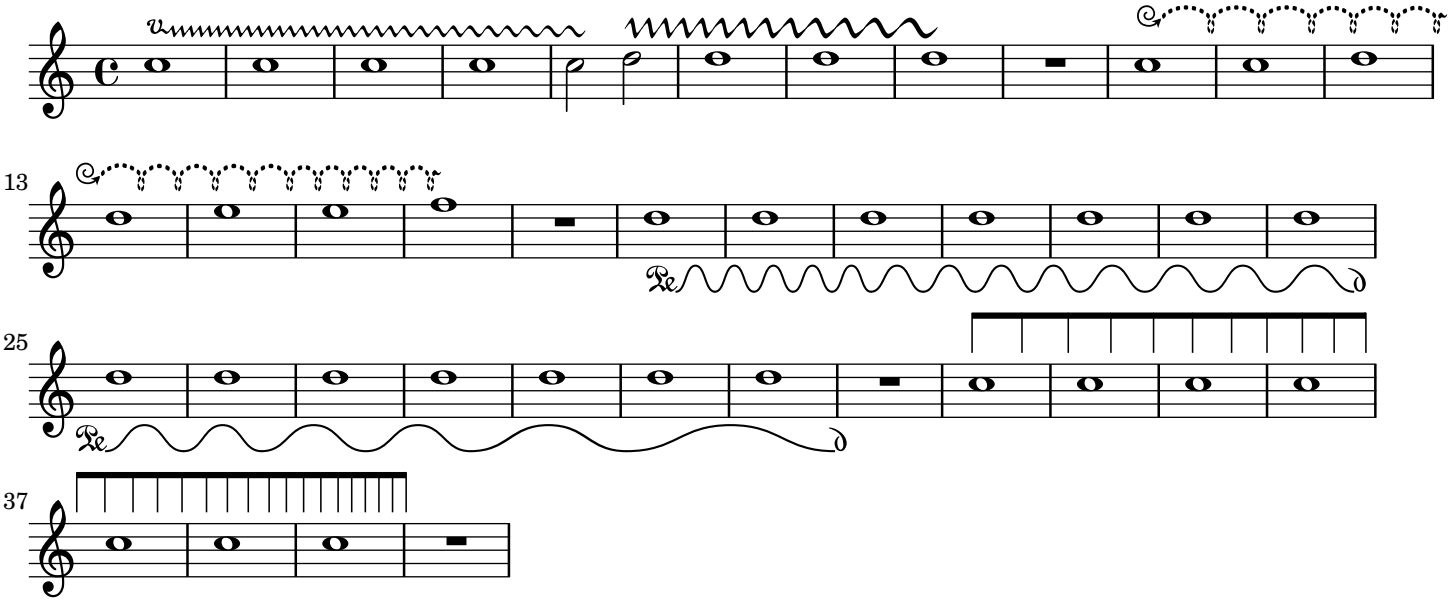
2		U+EAC2	wiggleCircularConstantLarge
1		U+EAC0	wiggleCircularConstant
0		U+EAC0	wiggleCircularConstant
−1		U+EAC1	wiggleCircularConstantFlipped
−2		U+EAC3	wiggleCircularConstantFlippedLarge

'wavy			
1		U+EAB4	wiggleWavyNarrow
0		U+EAB5	wiggleWavy
−1		U+EAB6	wiggleWavyWide
'square			
1		U+EAB7	wiggleSquareWaveNarrow
0		U+EAB8	wiggleSquareWave
−1		U+EAB9	wiggleSquareWaveWide
'sawtooth			
1		U+EABA	wiggleSawtoothNarrow
0		U+EABB	wiggleSawtooth
−1		U+EABC	wiggleSawtoothWide
'beam			
7		U+EB02	beamAccelRit15
	:		
0		U+EAFB	beamAccelRit8
	:		
−7		U+EAF4	beamAccelRit1
right		U+EB03	beamAccelRitFinal

Ekmelos

`'wavy`			
`6 .. 2`		U+F6B3	wiggleWavyNarrower
`1`		U+EAB4	wiggleWavyNarrow
`0`		U+EAB5	wiggleWavy
`-1`		U+EAB6	wiggleWavyWide
`-2`		U+F6B4	wiggleWavyWider
	:		
`-6`		U+F727	wiggleWavyQuadrupleWide
`'square`			
`6 .. 2`		U+F6B5	wiggleSquareWaveNarrower
`1`		U+EAB7	wiggleSquareWaveNarrow
`0`		U+EAB8	wiggleSquareWave
`-1`		U+EAB9	wiggleSquareWaveWide
`-2`		U+F6B6	wiggleSquareWaveWider
	:		
`-6`		U+F72B	wiggleSquareWaveQuadrupleWide
`'sawtooth`			
`6 .. 2`		U+F6B7	wiggleSawtoothNarrower
`1`		U+EABA	wiggleSawtoothNarrow
`0`		U+EABB	wiggleSawtooth
`-1`		U+EABC	wiggleSawtoothWide
`-2`		U+F6B8	wiggleSawtoothWider
	:		
`-6`		U+F72F	wiggleSawtoothQuadrupleWide

```
\relative c'' {
  \ekmSmuflOn #'textspan

  c1 \ekmStartSpan
       #'vibrato
       #'(3 . -3)
       ##f
  c c c c2 \stopTextSpan

  d2 \ekmStartSpan
       #'vibrato-large
       #'(3 . -3)
       #0
  d1 d d \stopTextSpan
  R1

  c1 \ekmStartSpan
       #'circular
       #'(-4 . -1)
       ##f
  c d d e e f \stopTextSpan
  R1

  \textSpannerDown
  d1 \ekmStartSpan
       #'wavy
       #'(0 . -6)  % this range uses Ekmelos glyphs
       #'((#xE651 #xE652) . #xE653)
  d d d d d d d d d d d d
  R1 \stopTextSpan
  \textSpannerNeutral

  c1 \ekmStartSpan
       #'beam
       #'(-5 . 5)
       ##f
  c c c c c c
  R1 \stopTextSpan
}
```


13
Re
25
Re
37

Trill spans and pitches

`\ekmSmuflOn #'trill`

Draw trill spans assembled from SMuFL multi-segment glyphs, and SMuFL trill pitches.

`\ekmStartTrillSpan TEMPO`

Start a trill span with the style `trill` . See Multi-segment spanner . [Ly]
TEMPO is a number or a pair of numbers (rounded to integer) for the segments of the spanner, usually in the range 4 to -4.

`\startTrillSpan` is equivalent to `\ekmStartTrillSpan #0`
and to `\ekmStartSpan #'trill #0 ##f`

`\ekmPitchedTrill NOTEHEAD-STYLE PARENS-STYLE MAIN-NOTE AUXILIARY-NOTE`

Draw a trill pitch. For NOTEHEAD-STYLE see Noteheads . For variable accidentals on AUXILIARY-NOTE see Ekmelily . [Err]

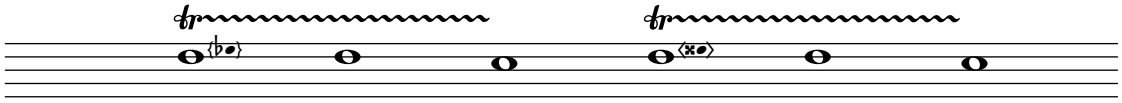
`\pitchedTrill` is equivalent to `\ekmPitchedTrill #'default #'default`

Parenthesis styles

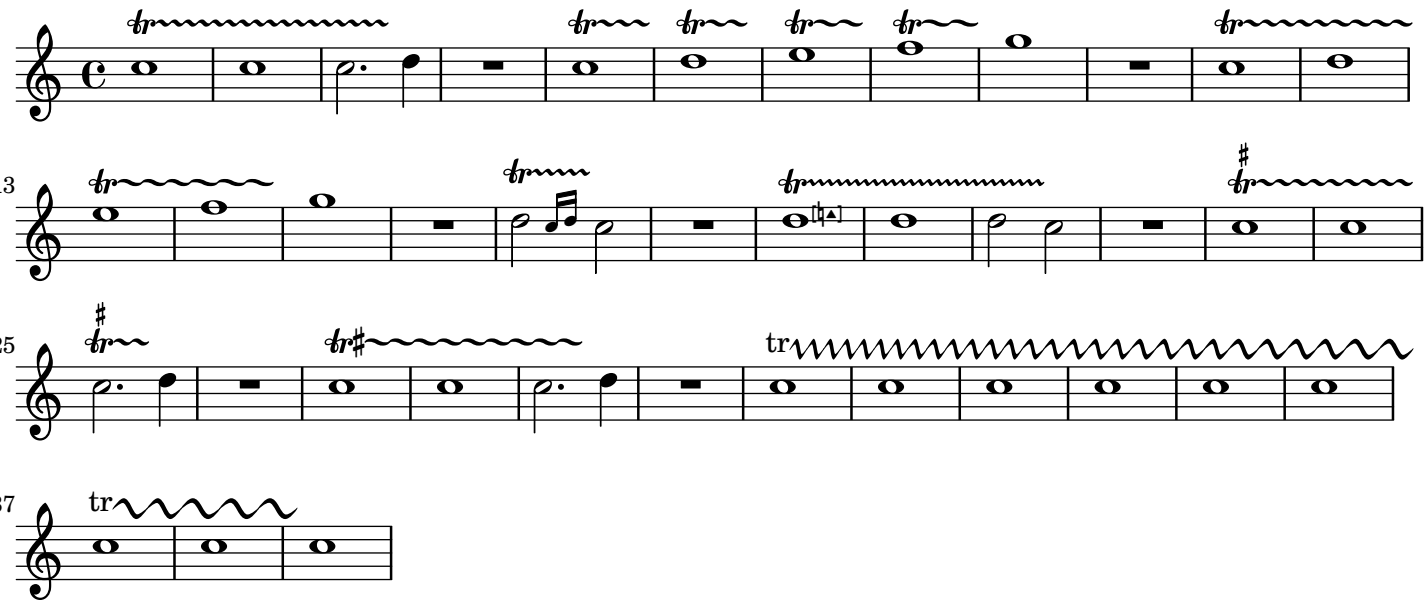
`'default`	U+E26A	accidentalParensLeft
	U+E26B	accidentalParensRight
`'bracket`	U+E26C	accidentalBracketLeft
	U+E26D	accidentalBracketRight

Ekmelos

`'brace`	U+F6D4	accidentalBraceLeft
	U+F6D5	accidentalBraceRight
`'angle`	U+F6D6	accidentalAngleLeft
	U+F6D7	accidentalAngleRight



```
\relative c'' {
  \ekmSmuflOn #'trill

  c1 \ekmStartTrillSpan #0
  c c2. d4 \stopTrillSpan
  R1

  c \ekmStartTrillSpan #-1
  d \ekmStartTrillSpan #-2
  e \ekmStartTrillSpan #-3
  f \ekmStartTrillSpan #-4
  g \stopTrillSpan
  R1

  c, \ekmStartTrillSpan #'(-1 . -4)
  d e f g \stopTrillSpan
  R1

  \afterGrace
  d2 \ekmStartTrillSpan #2
  { c16[ d] }
  c2 \stopTrillSpan
  R1

  \ekmPitchedTrill #'triangle #'bracket
  d1 \ekmStartTrillSpan #4
  e
  d d2 c \stopTrillSpan
  R1

  c1 \ekmStartSpan
       #'trill
       #-1
       #'((#xE262 #xE566) . 0) % draws an Ekmelos ligature
  c c2. d4 \stopTrillSpan
  R1

  c1 \ekmStartSpan
       #'trill
       #-3
       #`(,(markup #:concat
             (#:ekm-char #xE566
              #:general-align Y -0.7
                #:fontsize -2
                  #:ekm-char #xE262))
           . 0)
  c c2. d4 \stopTrillSpan
  R1

  c1 \ekmStartSpan
       #'trill-vibrato-large
       #'(3 . -3)
       #"tr"
  c c c c c c c c \stopTrillSpan
}
```


13
25
tr
37
tr

Laissez vibrer

`\ekmSmuflOn #'lv`

Draw SMuFL laissez vibrer ties.

`\ekmLaissezVibrer SIZE`

Draw a laissez vibrer tie after a note, corresponding to SIZE. [Ly]

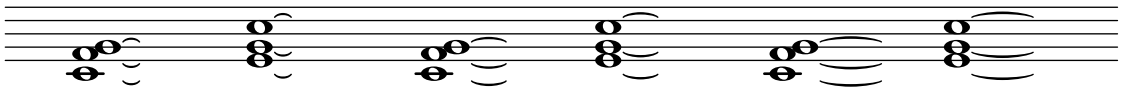
`\laissezVibrer` is equivalent to `\ekmLaissezVibrer #0`

Size ranges

any number	U+E4BA	articLaissezVibrerAbove
	U+E4BB	articLaissezVibrerBelow

Ekmelos

≤ 5.5	U+E4BA	articLaissezVibrerAbove
	U+E4BB	articLaissezVibrerBelow
≤ 8	U+F6FC	articLaissezVibrerAboveLong
	U+F6FD	articLaissezVibrerBelowLong
> 8	U+F6FE	articLaissezVibrerAboveExtraLong
	U+F6FF	articLaissezVibrerBelowExtraLong



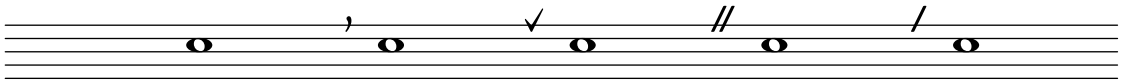
Breathing signs and Caesuras

`\ekmBreathing EXTEXT`

Draw a breathing sign or caesura from EXTEXT .

`#xE4CE`	U+E4CE	breathMarkComma
`#xE4CF`	U+E4CF	breathMarkTick
`#xE4D1`	U+E4D1	caesura

```
'(#xE4D1 1)
```



Colon and Segno bar lines

`\ekmSmuflOn #'colon`

Draw SMuFL colon bar lines.

`\ekmSmuflOn #'segno`

Draw SMuFL segno bar lines.

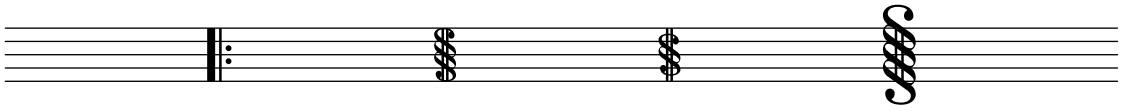
Note that both, `colon` and `segno` are set independently of a context and cannot be turned off.

Bar glyphs

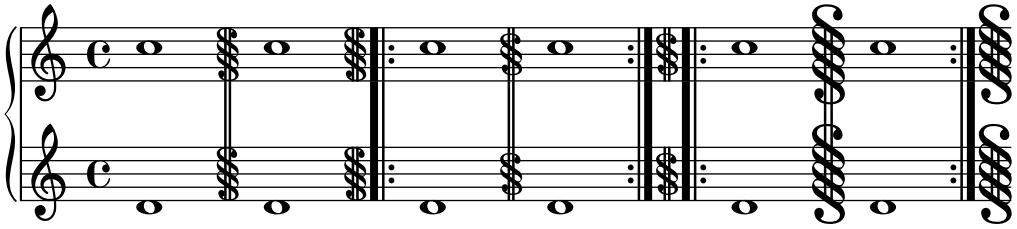
`:`	U+E043	repeatDots
`S`	U+E04A	segnoSerpent1

Ekmelos

`s`	U+F6C8	segnoSerpentSmall1
`$`	U+F6CA	segnoSerpentLarge1



```
\new PianoStaff \with {
  \ekmSmuflOn #'segno
}
<<
  \new Staff \relative c'' {
    c1 \bar "S"
    c \bar "S.|:-S"

    % Ekmelos bar glyphs
    c \bar "s"
    c \bar ":|.s.|:-s"

    c \bar "$"
    c \bar ":|.$-$"
  }
  \new Staff \relative c' {
    d1 d d d d d
  }
>>
```



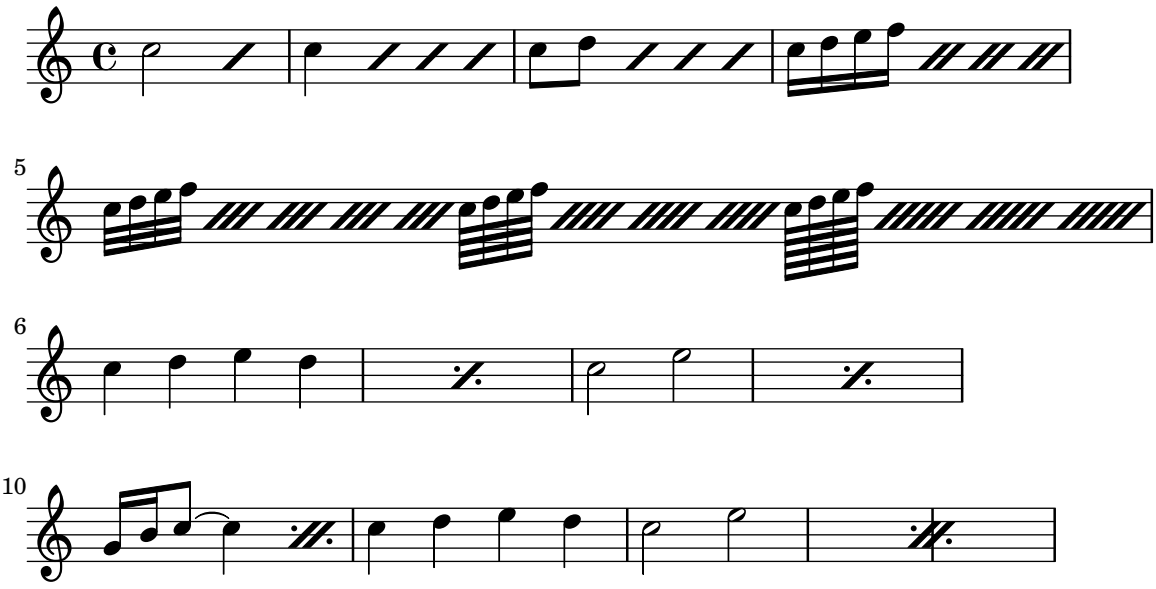
Percent repeats

`\ekmSmuflOn #'percent`

Draw SMuFL percent repeats.

U+E504	repeatBarSlash
U+E500	repeat1Bar
U+E501	repeat2Bars

```
\relative c'' {
  \ekmSmuflOn #'percent

  \repeat percent 2 { c2 }
  \repeat percent 4 { c4 }
  \repeat percent 4 { c8 d }
  \repeat percent 4 { c16 d e f }
  \repeat percent 5 { c32 d e f }
  \repeat percent 4 { c64 d e f }
  \repeat percent 4 { c128 d e f }
  \break

  \repeat percent 2 { c4 d e d }
  \repeat percent 2 { c2 e }
  \break

  \repeat percent 2 { g,16 b c8~ c4 }
  \repeat percent 2 { c4 d e d | c2 e }
}
```



Tremolo marks

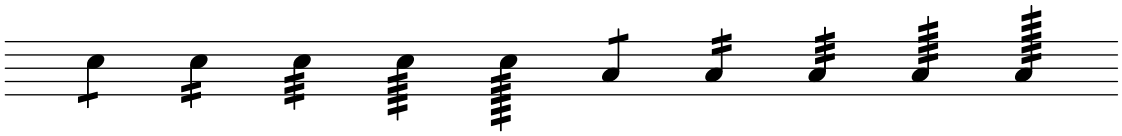
`\ekmSmuflOn #'tremolo`

Draw SMuFL tremolo marks on stems.
The style (shape) can be set with

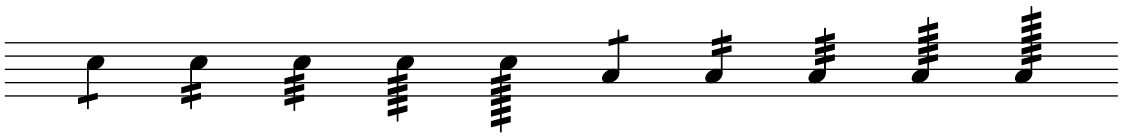
```
\override StemTremolo.shape = #STYLE
```

Styles

`'beam-like`	:8	U+E220	tremolo1
	:16	U+E221	tremolo2
	:32	U+E222	tremolo3
	:64	U+E223	tremolo4
	:128	U+E224	tremolo5



`'fingered`	:8	U+E225	tremoloFingered1
	:16	U+E226	tremoloFingered2
	:32	U+E227	tremoloFingered3
	:64	U+E228	tremoloFingered4
	:128	U+E229	tremoloFingered5

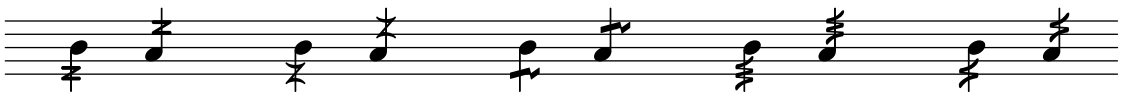


`\ekmTremolo EXTEXT MUSIC`

Draw a tremolo mark from EXTEXT on the stems of the tremolo notes in MUSIC, independent of the subdivision `:N`, and independent of the `tremolo` switch. A list of code points or a markup is centered horizontally, while a single code point is assumed being a centered stem decoration. Some strings are defined as names of symbols.

Names

`"buzzroll"`	U+E22A	buzzRoll
`"penderecki"`	U+E22B	pendereckiTremolo
`"stockhausen"`	U+E232	stockhausenTremolo
`"unmeasured"`	U+E22C	unmeasuredTremolo
`"unmeasuredS"`	U+E22D	unmeasuredTremoloSimple



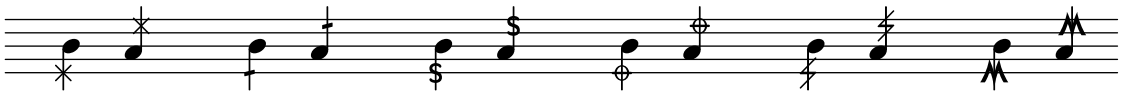
Symbols on stem (Stem decorations)

`\ekmStem EXTEXT MUSIC`

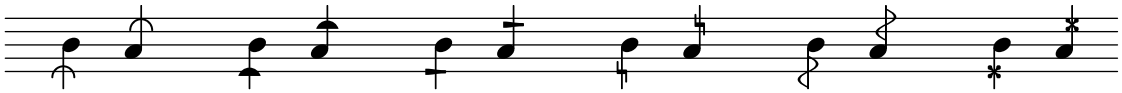
Draw a symbol from EXTEXT vertically centered on the stems in MUSIC. A list of code points or a markup is centered horizontally, while a single code point is assumed being a centered stem decoration. Some strings are defined as names of symbols.

Names

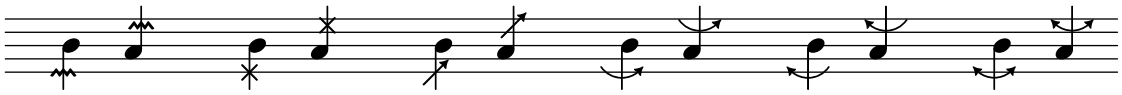
`"sprechgesang"`	U+E645	vocalSprechgesang
`"halbGesungen"`	U+E64B	vocalHalbGesungen
`"sussurando"`	U+E646	vocalsSussurando
`"damp"`	U+E63B	pluckedDampOnStem
`"stringNoise"`	U+E694	harpStringNoiseStem
`"multiphonics"`	U+E607	windMultiphonicsBlackStem



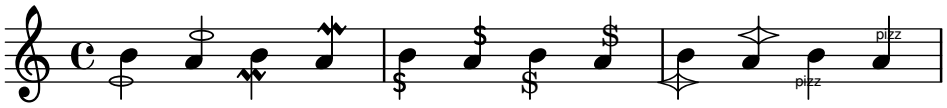
`"bowBehindBridge"`	U+E618	stringsBowBehindBridge
`"bowOnBridge"`	U+E619	stringsBowOnBridge
`"bowOnTailpiece"`	U+E61A	stringsBowOnTailpiece
`"fouette"`	U+E622	stringsFouette
`"vibrato"`	U+E623	stringsVibratoPulse
`"rimShot"`	U+E7FD	pictRimShotOnStem



`"crush"`	U+E80C	pictCrushStem
`"deadNote"`	U+E80D	pictDeadNoteStem
`"swish"`	U+E808	pictSwishStem
`"turnRight"`	U+E809	pictTurnRightStem
`"turnLeft"`	U+E80A	pictTurnLeftStem
`"turnRightLeft"`	U+E80B	pictTurnRightLeftStem



```
\relative c'' {
  \ekmTremolo ##xE233
  { b4:8 a: }

  % uses a list to center horizontally
  \ekmTremolo #'(#xE56C)
  { b4:8 a: }

  \ekmStem sussurando
  { b4 a }

  \ekmStem "S"
  { b a }

  \ekmStem #'(#xE843 #xE844)
  { b a }

  \ekmStem \markup { \fontsize #-6 \sans "pizz" }
  { b a }
}
```



Arpeggios

`\ekmSmuflOn #'arpeggio`

Draw SMuFL arpeggios.
The style can be set with

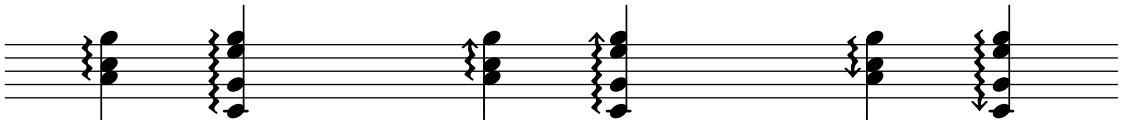
```
\override Arpeggio.style = #STYLE
```

Use `\ekmArpeggioNormal` , `\ekmArpeggioArrowUp` , and `\ekmArpeggioArrowDown` instead of `\arpeggioNormal`, etc. which turn off the SMuFL support (they revert the stencil).

Styles

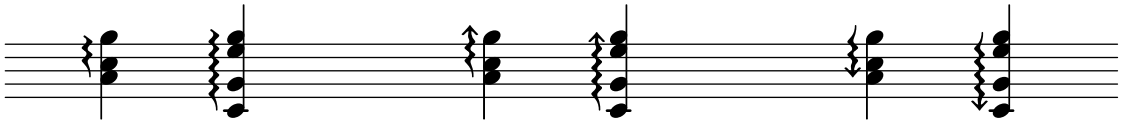
`'default`

U+EAA9	wiggleArpeggiatoUp
U+EAAA	wiggleArpeggiatoDown
U+EAAD	wiggleArpeggiatoUpArrow
U+EAAE	wiggleArpeggiatoDownArrow

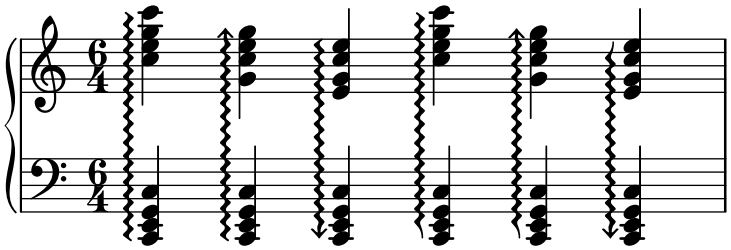


`'swash` :

U+EAAB	wiggleArpeggiatoUpSwash
U+EAAC	wiggleArpeggiatoDownSwash



```
\new PianoStaff \with {
  \ekmSmuflOn #'arpeggio
}
<<
  \set PianoStaff.connectArpeggios = ##t

  \new Staff \relative c'' {
    \time 6/4
    <c e g c>4 \arpeggio
    \once \override PianoStaff.Arpeggio.arpeggio-direction = #UP
    <g c e g> \arpeggio
    \once \override PianoStaff.Arpeggio.arpeggio-direction = #DOWN
    <e g c e> \arpeggio

    \override PianoStaff.Arpeggio.style = #'swash
    <c' e g c> \arpeggio
    \once \override PianoStaff.Arpeggio.arpeggio-direction = #UP
    <g c e g> \arpeggio
    \once \override PianoStaff.Arpeggio.arpeggio-direction = #DOWN
    <e g c e> \arpeggio
  }
  \new Staff \relative c, {
    \clef bass
    <c e g c>4 \arpeggio
    <c e g c> \arpeggio
    <c e g c> \arpeggio

    <c e g c> \arpeggio
    <c e g c> \arpeggio
    <c e g c> \arpeggio
  }
>>
```



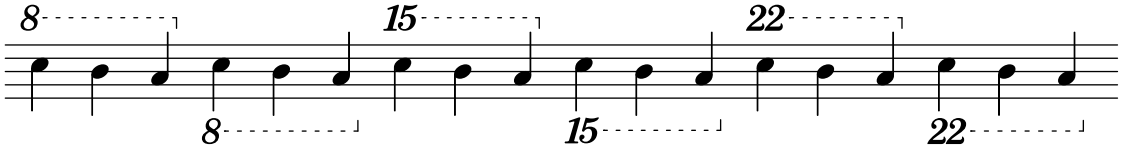
Ottavation

The ottavation style (markups) can be set with

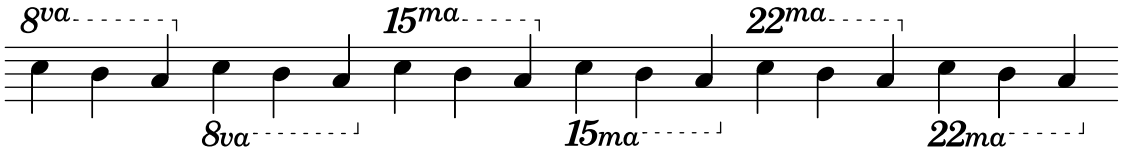
```
\set Staff.ottavationMarkups = #(ekm-ottavation STYLE)
```

Styles

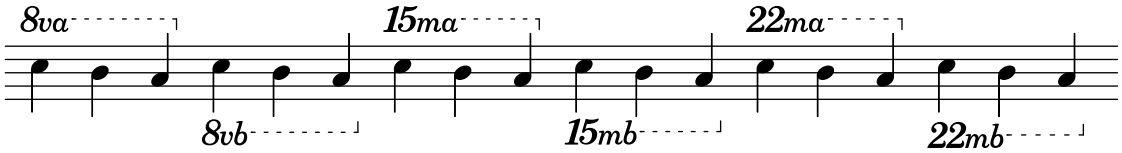
'numbers	±1	U+E510	ottava
	±2	U+E514	quindicesima
	±3	U+E517	ventiduesima



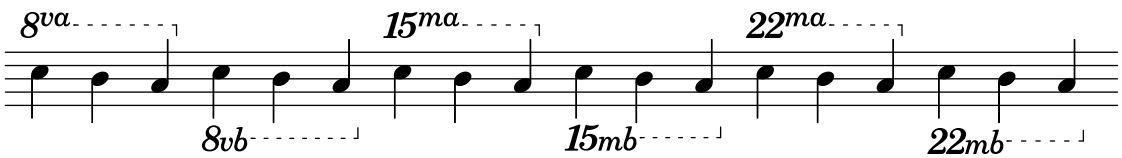
'ordinals	1	U+E511	ottavaAlta
	-1	U+E512	ottavaBassa
	2	U+E515	quindicesimaAlta
	-2	U+E516	quindicesimaBassa
	3	U+E518	ventiduesimaAlta
	-3	U+E519	ventiduesimaBassa



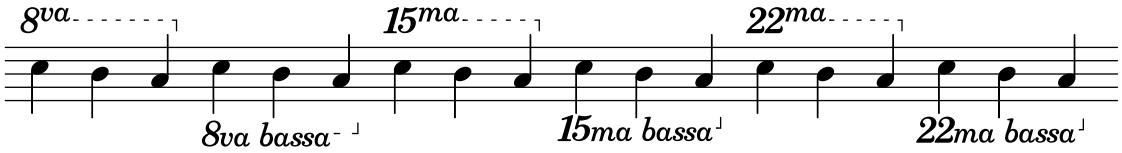
'simple-ordinals	1	U+E510	ottava
		U+EC97	octaveBaselineV
		U+EC91	octaveBaselineA
	-1	U+E51C	ottavaBassaVb
	2	U+E514	quindicesima
		U+EC95	octaveBaselineM
		U+EC91	octaveBaselineA
	-2	U+E51D	quindicesimaBassaMb
	3	U+E517	ventiduesima
		U+EC95	octaveBaselineM
		U+EC91	octaveBaselineA
	-3	U+E51E	ventiduesimaBassaMb



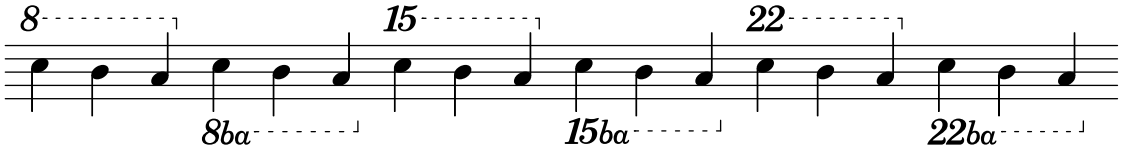
'ordinals-b	1	U+E511	ottavaAlta
	-1	U+E51C	ottavaBassaVb
	2	U+E515	quindicesimaAlta
	-2	U+E51D	quindicesimaBassaMb
	3	U+E518	ventiduesimaAlta
	-3	U+E51E	ventiduesimaBassaMb



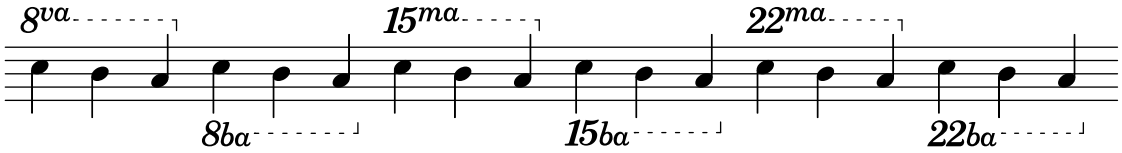
'ordinals-bassa	1	U+E511	ottavaAlta
	-1	U+E512	ottavaBassa
		U+E51F	octaveBassa
	2	U+E515	quindicesimaAlta
	-2	U+E516	quindicesimaBassa
		U+E51F	octaveBassa
	3	U+E518	ventiduesimaAlta
	-3	U+E519	ventiduesimaBassa
		U+E51F	octaveBassa



'numbers-ba	1	U+E510	ottava
	-1	U+E513	ottavaBassaBa
	2	U+E514	quindicesima
	-2	U+E514	quindicesima
		U+EC93	octaveBaselineB
		U+EC91	octaveBaselineA
	3	U+E517	ventiduesima
	-3	U+E517	ventiduesima
		U+EC93	octaveBaselineB
		U+EC91	octaveBaselineA



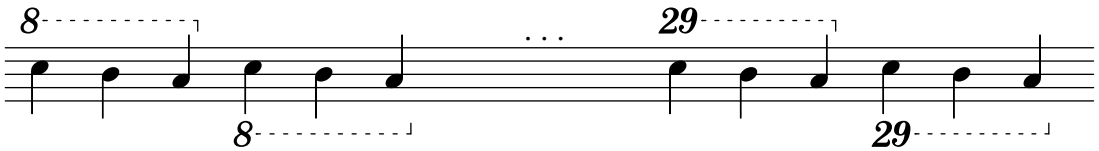
'ordinals-ba	1	U+E511	ottavaAlta
	-1	U+E513	ottavaBassaBa
	2	U+E515	quindicesimaAlta
	-2	U+E514	quindicesima
		U+EC93	octaveBaselineB
		U+EC91	octaveBaselineA
	3	U+E518	ventiduesimaAlta
	-3	U+E517	ventiduesima
		U+EC93	octaveBaselineB
		U+EC91	octaveBaselineA



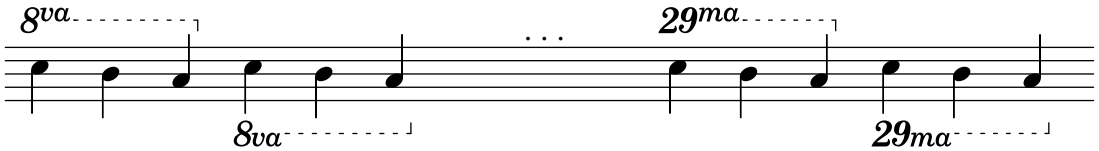
Note:
According to the implementation notes of SMuFL Octaves, the suffixes *vb* and *mb* as in `simple-ordinals` and `ordinals-b` are corruptions of the more correct forms *va bassa* and *ma bassa* as in `ordinals-bassa`. The recommended abbreviation for *8va bassa* is *8ba* as in `numbers-ba` and `ordinals-ba`.

Ekmelos

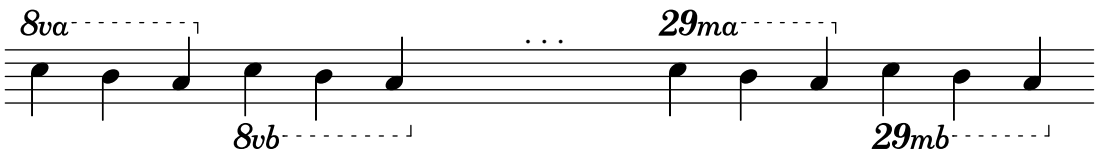
'numbers



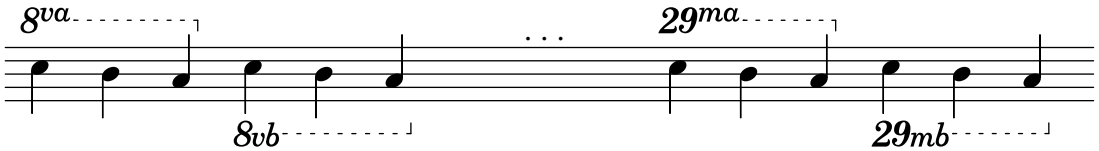
'ordinals



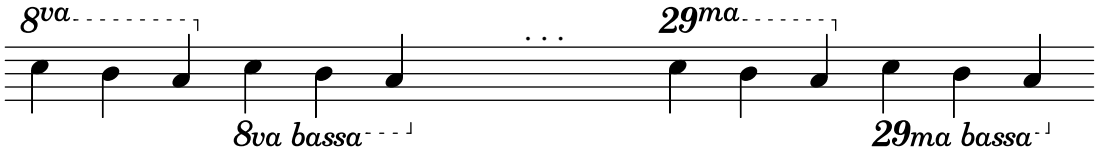
'simple-ordinals



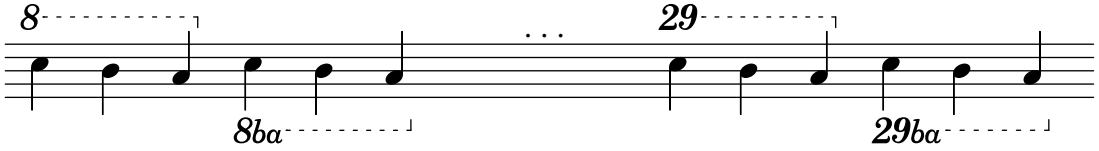
'ordinals-b



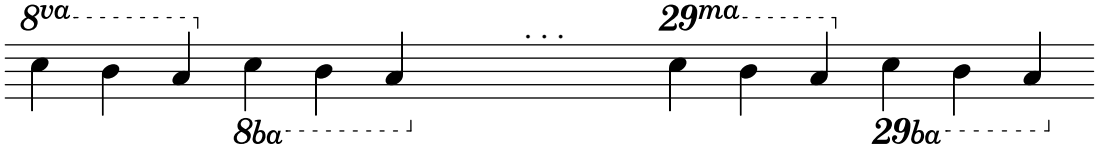
'ordinals-bassa



'numbers-ba



'ordinals-ba



`\ekm-ottavation DEFINITION`

Draw an ottavation text according to DEFINITION as markup.

`8`		U+E510	ottava
`8^va`		U+E511	ottavaAlta
`8va`		U+E512	ottavaBassa
`8ba`		U+E513	ottavaBassaBa
`8vb`		U+E51C	ottavaBassaVb
`15`		U+E514	quindicesima
`15^ma`		U+E515	quindicesimaAlta
`15ma`		U+E516	quindicesimaBassa
`15mb`		U+E51D	quindicesimaBassaMb
`22`		U+E517	ventiduesima
`22^ma`		U+E518	ventiduesimaAlta
`22ma`		U+E519	ventiduesimaBassa
`22mb`		U+E51E	ventiduesimaBassaMb
`(`		U+E51A	octaveParensLeft
`)`		U+E51B	octaveParensRight
`bassa`		U+E51F	octaveBassa
`loco`		U+EC90	octaveLoco
`^a`		U+EC92	octaveSuperscriptA
`^b`		U+EC94	octaveSuperscriptB
`^m`		U+EC96	octaveSuperscriptM
`^v`		U+EC98	octaveSuperscriptV
`a`		U+EC91	octaveBaselineA
`b`		U+EC93	octaveBaselineB
`m`		U+EC95	octaveBaselineM
`v`		U+EC97	octaveBaselineV

Ekmelos

`8^vb`		U+F652	ottavaBassaSupVb
`15^mb`		U+F653	quindicesimaBassaSupMb
`22^mb`		U+F654	ventiduesimaBassaSupMb
`29`		U+F6F8	ventinovesima
`29^ma`		U+F6F9	ventinovesimaAlta
`29ma`		U+F6FA	ventinovesimaBassa
`29^mb`		U+F655	ventinovesimaBassaSupMb
`29mb`		U+F6FB	ventinovesimaBassaMb

`\ekmSmuflOn #'tuplet`

Draw SMuFL tuplet numbers as numerator only. Set the first formatting function below, so this switch is not required if one of these functions is set explicitly.

0	𝅘𝅥𝅯	U+E880	tuplet0
	:		
9	𝅘𝅥𝅯	U+E889	tuplet9
:	:	U+E88A	tupletColon

```
ekm-tuplet-number::calc-denominator-text
ekm-tuplet-number::calc-fraction-text
(ekm-tuplet-number::non-default-tuplet-denominator-text NUM)
(ekm-tuplet-number::non-default-tuplet-fraction-text NUM DENOM)

(ekm-tuplet-number::append-note-wrapper
    FUNCTION DURATION)
(ekm-tuplet-number::fraction-with-notes
    NUM-DURATION DENOM-DURATION)
(ekm-tuplet-number::non-default-fraction-with-notes
    NUM NUM-DURATION DENOM DENOM-DURATION)
```

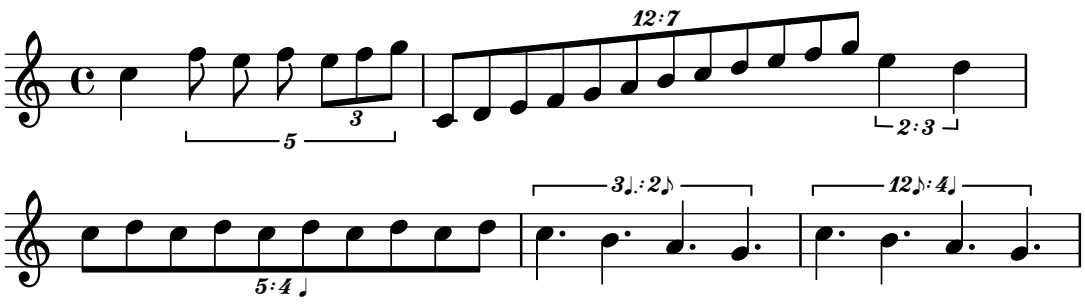
Tuplet formatting functions. The last three draw `metronome` style notes for the specified durations.

`(ekm-tuplet-number NUM DENOM)`

Draw NUM:DENOM, or NUM only if DENOM is 0. Use the actual tuplet fraction for NUM or DENOM if `#f` is specified. It is called by the first four functions above, i.e. they are equivalent to:

```
(ekm-tuplet-number #f 0)
(ekm-tuplet-number #f #f)
(ekm-tuplet-number NUM 0)
(ekm-tuplet-number NUM DENOM)
```

```
\relative c'' {
  \cadenzaOn

  c4
  \override TupletNumber.text =
    #ekm-tuplet-number::calc-denominator-text
  \tuplet 5/4 {
    f8 e f
    \tuplet 3/2 { e[ f g] }
  }
  \bar "|"

  \override TupletNumber.text =
    #ekm-tuplet-number::calc-fraction-text
  \tuplet 12/7 { c,,8[ d e f g a b c d e f g] }
  \tuplet 2/3 { e4 d }
  \bar "|"
  \break

  \once \override TupletNumber.text =
    #(ekm-tuplet-number::append-note-wrapper
       ekm-tuplet-number::calc-fraction-text
       (ly:make-duration 2 0))
  \tuplet 5/4 { c8[ d c d c d c d c d] }
  \bar "|"

  \once \override TupletNumber.text =
    #(ekm-tuplet-number::fraction-with-notes
      (ly:make-duration 2 1)
      (ly:make-duration 3 0))
  \tuplet 3/2 { c4. b a g }
  \bar "|"

  \once \override TupletNumber.text =
    #(ekm-tuplet-number::non-default-fraction-with-notes
      12 (ly:make-duration 3 0)
      4 (ly:make-duration 2 0))
  \tuplet 3/2 { c4. b a g }
  \bar "|"
}
```



`\ekmSmuflOn #'fingering`

Draw SMuFL fingering instructions specified with a digit or with `\finger` , as well as right-hand fingerings specified with `\rightHandFinger` , using `\ekm-finger` .
Note: The `\thumb` command always produces normal LilyPond output. Use `\finger "th"` to draw the corresponding SMuFL glyph.

`\ekm-finger DEFINITION`

Draw a fingering instruction according to DEFINITION as markup. If the first character is `*` the `italic` style symbols are drawn, else the `default` symbols.

`'default`

0	𝟎	U+ED10	fingering0
	:		
5	𝟓	U+ED15	fingering5
6	𝟔	U+ED24	fingering6
	:		
9	𝟗	U+ED27	fingering9
(	(	U+ED28	fingeringLeftParenthesis
)	)	U+ED29	fingeringRightParenthesis
[	[	U+ED2A	fingeringLeftBracket
]	]	U+ED2B	fingeringRightBracket
.	•	U+ED2C	fingeringSeparatorMiddleDot
'	◦	U+ED2D	fingeringSeparatorMiddleDotWhite
/	/	U+ED2E	fingeringSeparatorSlash
~~	⁀	U+ED20	fingeringSubstitutionAbove
~	‿	U+ED21	fingeringSubstitutionBelow
-	-	U+ED22	fingeringSubstitutionDash
M	[	U+ED23	fingeringMultipleNotes
th	ϙ	U+E624	stringsThumbPosition
ht	ϭ	U+E625	stringsThumbPositionTurned
T	𝐓	U+ED16	fingeringTUpper
t	𝑡	U+ED18	fingeringTLower
p	𝑝	U+ED17	fingeringPLower
i	𝑖	U+ED19	fingeringILower
m	𝑚	U+ED1A	fingeringMLower
a	𝑎	U+ED1B	fingeringALower
c	𝑐	U+ED1C	fingeringCLower
x	𝑥	U+ED1D	fingeringXLower
e	𝑒	U+ED1E	fingeringELower
o	𝑜	U+ED1F	fingeringOLower
q	𝑞	U+ED8E	fingeringQLower
s	𝑠	U+ED8F	fingeringSLower

R	⎣	U+E66E	keyboardPlayWithRH
RE	⎦	U+E66F	keyboardPlayWithRHEnd
L	⎡	U+E670	keyboardPlayWithLH
LE	⎤	U+E671	keyboardPlayWithLHEnd

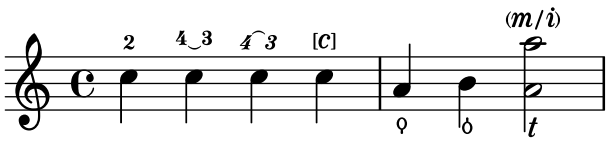
'italic

0	𝘰	U+ED80	fingering0Italic
	:		
9	𝘹	U+ED89	fingering9Italic
(	(	U+ED8A	fingeringLeftParenthesisItalic
)	)	U+ED8B	fingeringRightParenthesisItalic
[	[	U+ED8C	fingeringLeftBracketItalic
]	]	U+ED8D	fingeringRightBracketItalic
.	•	U+ED2C	fingeringSeparatorMiddleDot
,	◦	U+ED2D	fingeringSeparatorMiddleDotWhite
/	/	U+ED2E	fingeringSeparatorSlash
~~	⁀	U+ED20	fingeringSubstitutionAbove
~	‿	U+ED21	fingeringSubstitutionBelow
–	-	U+ED22	fingeringSubstitutionDash

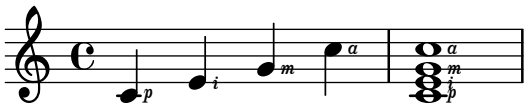
`\ekmPlayWith HAND START MUSIC`

Draw the symbol for the token `R`, `RE`, `L`, or `LE` (see above) alongside the notes in MUSIC. HAND is `RIGHT` or `LEFT`. START is `#t` for the start symbol placed to the left, or `#f` for the end symbol placed to the right.

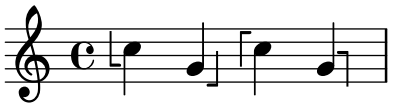
```
\relative c'' {
  \ekmSmuflOn #'fingering

  c4 - 2
  c - \finger "4~3"
  c - \finger "*4~~3"
  c - \finger "[c]"
  a _ \finger "th"
  b _ \finger "ht"
  < a - \finger "t"
    a' - \finger "(m_/_i)" >2
}
```



```
\relative c' {
  \ekmSmuflOn #'fingering

  c \rightHandFinger #1
  e \rightHandFinger #2
  g \rightHandFinger #3
  c \rightHandFinger #4
  < c, \rightHandFinger #1
    e  \rightHandFinger #2
    g  \rightHandFinger #3
    c  \rightHandFinger #4 >1
}
```



```
\relative c'' {
  \ekmSmuflOn #'fingering

  \ekmPlayWith #RIGHT ##t c
  \ekmPlayWith #RIGHT ##f g

  \ekmPlayWith #LEFT ##t c
  \ekmPlayWith #LEFT ##f g
}
```



String number indications

`\ekmSmuflOn #'stringnumber`

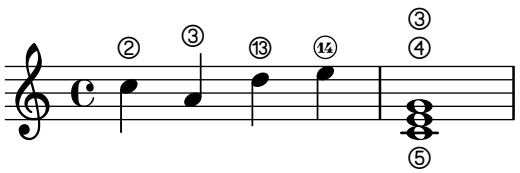
Draw SMuFL string number indications specified with \NUMBER, using `\ekm-string-number` .
Note: `\romanStringNumbers` overrides the SMuFL switch so that reverting with `\arabicStringNumbers` produces normal LilyPond output.

`\ekm-string-number ARG`

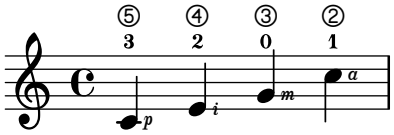
Draw a string number indication as markup. ARG is a number or string. For a number or a string representing a number, either the defined symbol is drawn or the number with the normal text font and a cicle around. Any other string, e.g. a Roman numeral, is drawn in italic style.

0	⓪	U+E833	guitarString0
	:		
9	⑨	U+E83C	guitarString9
10	⑩	U+E84A	guitarString10
	:		
13	⑬	U+E84D	guitarString13

```
\relative c'' {
  \ekmSmuflOn #'stringnumber

  c \2
  a \3
  d \13
  e \14
  < c,\5 e\4 g\3 >1
}
```



```
\relative c' {
  \ekmSmuflOn #'(fingering stringnumber)

  < c -3 \5 \rightHandFinger #1 >
  < e -2 \4 \rightHandFinger #2 >
  < g -0 \3 \rightHandFinger #3 >
  < c -1 \2 \rightHandFinger #4 >
}
```



Piano pedals

`\ekmSmuflOn #'pedal`

Draw SMuFL piano pedals for sustain, sostenuto, and una corda, using `\ekm-piano-pedal` .

`\ekm-piano-pedal DEFINITION`

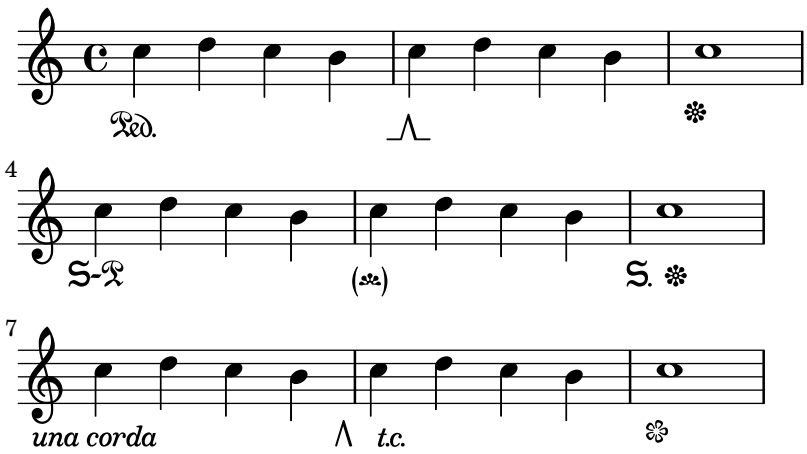
Draw piano pedal symbols according to DEFINITION as markup.

`Ped.`		U+E650	keyboardPedalPed
`P`		U+E651	keyboardPedalP
`e`		U+E652	keyboardPedalE
`d`		U+E653	keyboardPedalD
`Sost.`		U+E659	keyboardPedalSost
`S`		U+E65A	keyboardPedalS
`.`		U+E654	keyboardPedalDot
`-`		U+E658	keyboardPedalHyphen
`*`		U+E655	keyboardPedalUp
`o`		U+E65D	keyboardPedalUpSpecial
`,`		U+E65B	keyboardPedalHalf2
`'`		U+E65C	keyboardPedalHalf3
`H`		U+E656	keyboardPedalHalf
`^`		U+E657	keyboardPedalUpNotch
`l`		U+E65E	keyboardLeftPedalPictogram
`m`		U+E65F	keyboardMiddlePedalPictogram
`r`		U+E660	keyboardRightPedalPictogram
`(`		U+E676	keyboardPedalParensLeft
`)`		U+E677	keyboardPedalParensRight

Ekmelos

`Ped`		U+F434	keyboardPedalPedNoDot
`Sost`		U+F435	keyboardPedalSostNoDot
`Sos.`		U+F6D1	keyboardPedalSos2
`sos.`		U+F6D0	keyboardPedalSos
`unacorda`		U+F6CC	keyboardPedalUnaCorda
`trecorde`		U+F6CD	keyboardPedalTreCorde
`u.c.`		U+F6CE	keyboardPedalUC
`t.c.`		U+F6CF	keyboardPedalTC
`1/2Ped`		U+F6B0	keyboardPedalHalf4
`|1/4`		U+F6BA	keyboardPedalPosQuarter
`|1/2`		U+F6BB	keyboardPedalPosHalf
`|3/4`		U+F6BC	keyboardPedalPosThreeQuarters
`|1`		U+F6BD	keyboardPedalPosFull

```
\new Staff \with {
  \ekmSmuflOn #'pedal
}
\relative c'' {
  \set Staff.pedalSustainStrings = #'("Ped." "H" "*")
  c4 \sustainOn d c b c \sustainOff \sustainOn d c b c1 \sustainOff
  \break

  \set Staff.pedalSostenutoStyle = #'text
  \set Staff.pedalSostenutoStrings = #'("S-P" "(')" "S. *")
  c4 \sostenutoOn d c b c \sostenutoOff \sostenutoOn d c b c1 \sostenutoOff
  \break

  % draws Ekmelos glyphs "unacorda" and "t.c."
  \set Staff.pedalUnaCordaStyle = #'text
  \set Staff.pedalUnaCordaStrings = #'("unacorda" "^___t.c." "o")
  c4 \unaCorda d c b c \treCorde \unaCorda d c b c1 \treCorde
}
```



Harp pedals

`\ekm-harp-pedal DEFINITION`

Draw a harp pedal diagram according to DEFINITION as markup. Space between tokens is ignored.

`\ekm-harp-change EXTEXT Y`

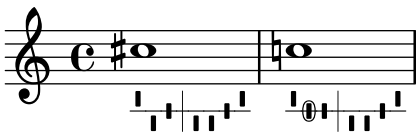
Draw EXTEXT as markup with an ellipse translated vertically by Y. Used by the following change tokens.

^		U+E680	harpPedalRaised
o^			
-		U+E681	harpPedalCentered
o-			
v		U+E682	harpPedalLowered
ov			
|		U+E683	harpPedalDivider

Ekmelos

o^		U+F648	harpPedalRaisedChange
o-		U+F649	harpPedalCenteredChange
ov		U+F64A	harpPedalLoweredChange

```
\relative c'' {
  \textLengthOn
  cis1 _ \markup \ekm-harp-pedal #"^v-|vv-^"
  c! _ \markup \ekm-harp-pedal #"^o--|vv-^"
}
```



Fret diagrams

`\ekm-fret-diagram-terse DEFINITION`

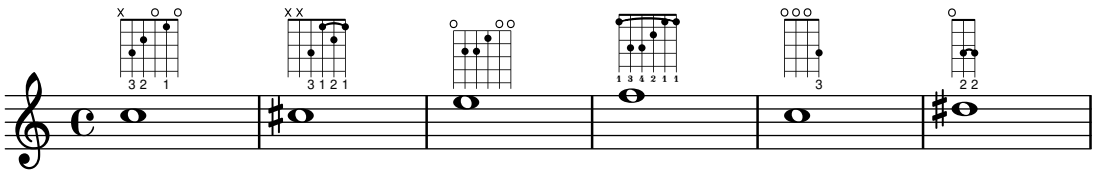
Draw a fret diagram according to DEFINITION as markup. Fingering is always placed below.

Used properties:

- `fret-diagram-details top-fret-thickness` (3): > 1 draws the …Nut glyph.
- `fret-diagram-details finger-code` (`#t`): `'none` draws no fingering.
- `fret-diagram-details finger-style` (`'sans`)

3 strings		U+E850	fretboard3String
		U+E851	fretboard3StringNut
4 strings		U+E852	fretboard4String
		U+E853	fretboard4StringNut
5 strings		U+E854	fretboard5String
		U+E855	fretboard5StringNut
6 strings		U+E856	fretboard6String
		U+E857	fretboard6StringNut
.		U+E858	fretboardFilledCircle
x		U+E859	fretboardX
o		U+E85A	fretboardO

```
\relative c'' {
  \textLengthOn

  c ^ \markup \ekm-fret-diagram-terse #"x;3-3;2-2;o;1-1;o;"

  cis ^ \markup \ekm-fret-diagram-terse #"x;x;3-3;1-1-(;2-2;1-1-);"

  e ^ \markup {
        \override #'(fret-diagram-details . (
                      (top-fret-thickness . 1)
                      (finger-code . none)))
        \ekm-fret-diagram-terse #"o;2-2;2-3;1-1;o;o;"
      }

  f ^ \markup {
        \override #'(fret-diagram-details . (
                      (finger-style . finger)))
        \ekm-fret-diagram-terse #"1-1-(;3-3;3-4;2-2;1-1;1-1-);"
      }

  c ^ \markup \ekm-fret-diagram-terse #"o;o;o;3-3;"

  dis ^ \markup \ekm-fret-diagram-terse #"o;3-2-(;3-2-);"
}
```



Accordion registers

`\ekm-accordion NAME`

Draw an accordion register symbol as markup. NAME has a prefix for the register style separated by a space. `"d"` (discant) is the default and can be omitted.
Most of the symbols use precomposed glyphs. The others are composed using accdnCombRH3RanksEmpty (U+E8C6) etc.
Note: The module `(scm accreg)` is not required.

`\ekmAccordion NAME`

Set an accordion register symbol as a standalone music event. This is equivalent to

```
<> ^ \markup \ekm-accordion NAME
```

Names

`"d …"`	Discant		
`"d 1"`		U+E8A4	accdnRH3RanksBassoon
`"d 10"`		U+E8A1	accdnRH3RanksClarinet
`"d 11"`		U+E8AB	accdnRH3RanksBandoneon
`"d 1+0"`		U+E8A2	accdnRH3RanksUpperTremolo8
`"d 1+1"`			
`"d 1-0"`		U+E8A3	accdnRH3RanksLowerTremolo8
`"d 1-1"`			
`"d 20"`		U+E8AE	accdnRH3RanksTwoChoirs
`"d 21"`		U+E8AF	accdnRH3RanksTremoloLower8ve
`"d 2+0"`		U+E8A6	accdnRH3RanksViolin
`"d 2+1"`		U+E8AC	accdnRH3RanksAccordion
`"d 2-0"`			
`"d 2-1"`			
`"d 30"`		U+E8A8	accdnRH3RanksAuthenticMusette
`"d 31"`		U+E8B1	accdnRH3RanksDoubleTremoloLower8ve
`"d 100"`		U+E8A0	accdnRH3RanksPiccolo
`"d 101"`		U+E8A9	accdnRH3RanksOrgan
`"d 110"`		U+E8A5	accdnRH3RanksOboe
`"d 111"`		U+E8AA	accdnRH3RanksHarmonium
`"d 11+0"`			
`"d 11+1"`			
`"d 11-0"`			
`"d 11-1"`			

"d 120"		U+E8B0	accdnRH3RanksTremoloUpper8ve
"d 121"		U+E8AD	accdnRH3RanksMaster
"d 12+0"		U+E8A7	accdnRH3RanksImitationMusette
"d 12+1"			
"d 12-0"			
"d 12-1"			
"d 130"		U+E8B2	accdnRH3RanksDoubleTremoloUpper8ve
"d 131"		U+E8B3	accdnRH3RanksFullFactory

"sb …" Standard bass

"sb Soprano"		U+E8B4	accdnRH4RanksSoprano
"sb Alto"		U+E8B5	accdnRH4RanksAlto
"sb Tenor"		U+E8B6	accdnRH4RanksTenor
"sb Master"		U+E8B7	accdnRH4RanksMaster
"sb Soft Bass"		U+E8B8	accdnRH4RanksSoftBass
"sb Soft Tenor"		U+E8B9	accdnRH4RanksSoftTenor
"sb Bass/Alto"		U+E8BA	accdnRH4RanksBassAlto

"sb4 …" Standard bass, four reed

"sb4 Soprano"		U+E8B4	accdnRH4RanksSoprano
"sb4 Alto"		U+E8B5	accdnRH4RanksAlto
"sb4 Tenor"			
"sb4 Master"			
"sb4 Soft Bass"			
"sb4 Bass/Alto"		U+E8BA	accdnRH4RanksBassAlto
"sb4 Soft Bass/Alto"			
"sb4 Soft Tenor"		U+E8B9	accdnRH4RanksSoftTenor

"sb5 …"	Standard bass, five reed		
"sb5 Bass/Alto"		U+E8BA	accdnRH4RanksBassAlto
"sb5 Soft Bass/Alto"			
"sb5 Alto"			
"sb5 Tenor"			
"sb5 Master"			
"sb5 Soft Bass"			
"sb5 Soft Tenor"		U+E8B9	accdnRH4RanksSoftTenor
"sb5 Soprano"		U+E8B4	accdnRH4RanksSoprano
"sb5 Sopranos"			
"sb5 Solo Bass"			

"sb6 …"	Standard bass, six reed		
"sb6 Soprano"		U+E8B4	accdnRH4RanksSoprano
"sb6 Alto"			
"sb6 Soft Tenor"		U+E8B9	accdnRH4RanksSoftTenor
"sb6 Master"		U+E8B7	accdnRH4RanksMaster
"sb6 Alto/Soprano"			
"sb6 Bass/Alto"		U+E8BA	accdnRH4RanksBassAlto
"sb6 Soft Bass"		U+E8B8	accdnRH4RanksSoftBass

"fb …"	Free bass		
"fb 10"		U+E8BB	accdnLH2Ranks8Round
"fb 1"		U+E8BC	accdnLH2Ranks16Round
"fb 11"		U+E8BD	accdnLH2Ranks8Plus16Round
"fb Master"		U+E8BE	accdnLH2RanksMasterRound
"fb Master 1"		U+E8BF	accdnLH2RanksMasterPlus16Round
"fb Master 11"		U+E8C0	accdnLH2RanksFullMasterRound

"sq …"	Square		
"sq 1"		U+E8C1	accdnLH3Ranks8Square
"sq 100"		U+E8C2	accdnLH3Ranks2Square
"sq 2"		U+E8C3	accdnLH3RanksDouble8Square
"sq 101"		U+E8C4	accdnLH3Ranks2Plus8Square
"sq 102"		U+E8C5	accdnLH3RanksTuttiSquare

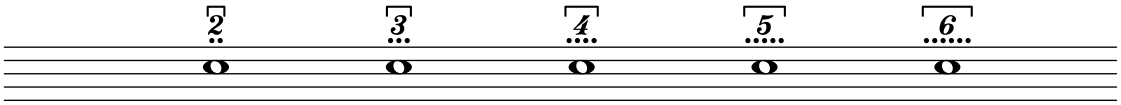
Accordion ricochet

`\ekmRicochet NUMBER`

Draw a ricochet symbol as an expressive mark (script). [Ly]

Numbers

2	U+E8CD	accdnRicochet2
3	U+E8CE	accdnRicochet3
4	U+E8CF	accdnRicochet4
5	U+E8D0	accdnRicochet5
6	U+E8D1	accdnRicochet6

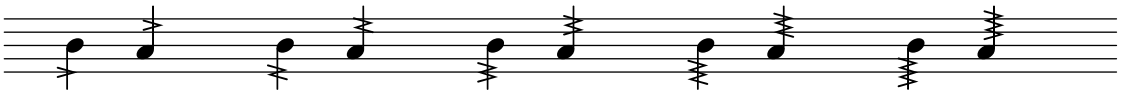


`\ekmStemRicochet NUMBER MUSIC`

Draw a ricochet symbol vertically centered on the stems in MUSIC.

Numbers

2	U+E8D2	accdnRicochetStem2
3	U+E8D3	accdnRicochetStem3
4	U+E8D4	accdnRicochetStem4
5	U+E8D5	accdnRicochetStem5
6	U+E8D6	accdnRicochetStem6



Falls and doits

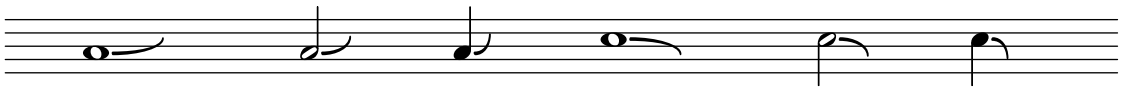
`\ekmBendAfter STYLE DIRECTION`

Draw a fall or doit (lift) symbol after a note.

Styles

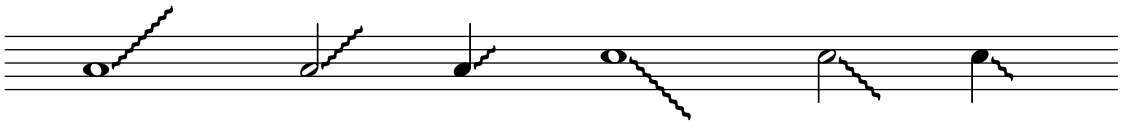
`'bend`

UP	U+E5D6	brassDoitLong
	U+E5D5	brassDoitMedium
	U+E5D4	brassDoitShort
DOWN	U+E5D9	brassFallLipLong
	U+E5D8	brassFallLipMedium
	U+E5D7	brassFallLipShort



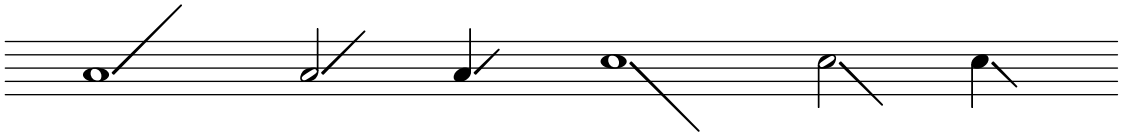
`'rough`

UP	U+E5D3	brassLiftLong
	U+E5D2	brassLiftMedium
	U+E5D1	brassLiftShort
DOWN	U+E5DF	brassFallRoughLong
	U+E5DE	brassFallRoughMedium
	U+E5DD	brassFallRoughShort



`'smooth`

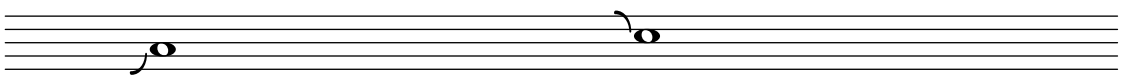
UP	U+E5EE	brassLiftSmoothLong
	U+E5ED	brassLiftSmoothMedium
	U+E5EC	brassLiftSmoothShort
DOWN	U+E5DC	brassFallSmoothLong
	U+E5DB	brassFallSmoothMedium
	U+E5DA	brassFallSmoothShort



`\ekmScoop DIRECTION MUSIC`

Draw a scoop or plop symbol to the left of each note in MUSIC.

UP	U+E5D0	brassScoop
DOWN	U+E5E0	brassPlop



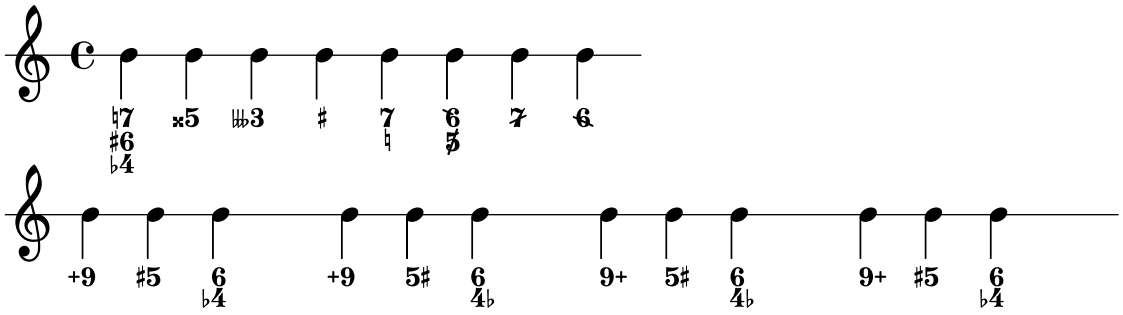
Figured bass

`\ekmSmuflOn #'fbass`

Draw SMuFL bass figures with `\figuremode` . Some raised / diminished figures use precomposed glyphs which ignore the property `figuredBassPlusDirection` .

0		U+EA50	figbass0
1		U+EA51	figbass1
2		U+EA52	figbass2
3		U+EA54	figbass3
4		U+EA55	figbass4
5		U+EA57	figbass5
6		U+EA5B	figbass6
7		U+EA5D	figbass7
8		U+EA60	figbass8
9		U+EA61	figbass9
!		U+EA65	figbassNatural
–		U+EA64	figbassFlat
+		U+EA66	figbassSharp
––		U+EA63	figbassDoubleFlat
++		U+EA67	figbassDoubleSharp
–––		U+ECC1	figbassTripleFlat
+++		U+ECC2	figbassTripleSharp
\+		U+EA6C	figbassPlus
/		U+EA6D	figbassCombiningRaising
\\		U+EA6E	figbassCombiningLowering
2\+		U+EA53	figbass2Raised
4\+		U+EA56	figbass4Raised
5\+		U+EA58	figbass5Raised1
5\\		U+EA59	figbass5Raised2
5/		U+EA5A	figbass5Raised3
6\\		U+EA5C	figbass6Raised
6\+		U+EA6F	figbass6Raised2
7\+		U+EA5E	figbass7Raised1
7\\		U+EA5F	figbass7Raised2
7/		U+ECC0	figbass7Diminished
9\\		U+EA62	figbass9Raised

```
\new Staff
<<
  \relative c'' {
    \cadenzaOn

    b b b b b b b b
    \break

    b b b s
    b b b s
    b b b s
    b b b s
  }
  \figures {
    <7! 6+ 4->  <5++>  <3--->  <_+>  <7 _!>  <6\+ 5/>  <7/>  <6\\>

    <9\+>  <5+>  <6 4->  r
    \set figuredBassAlterationDirection = #RIGHT
    <9\+>  <5+>  <6 4->  r
    \set figuredBassPlusDirection = #RIGHT
    <9\+>  <5+>  <6 4->  r
    \set figuredBassAlterationDirection = #LEFT
    <9\+>  <5+>  <6 4->  r
  }
>>
\layout {
  \context {
    \Score
    \ekmSmuflOn #'fbass
    \override StaffSymbol.line-count = #1
  }
}
```



`\ekmSmuflOn #'lyric`

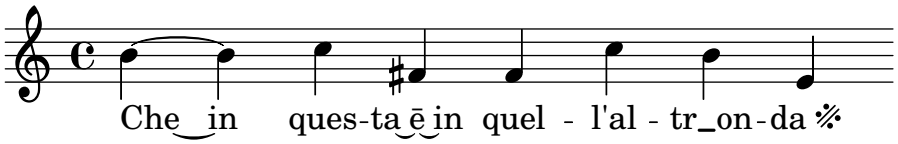
Draw the words in a lyric input mode (`\lyricmode` etc.) with `\ekm-tied-lyric` .
Note: The characters _ and `%` must be quoted in order to be passed on to this command.

`\ekm-tied-lyric STRING`

Draw STRING as markup, replacing tokens with the corresponding glyphs. The space between the adjoining words depends on the width of the respective glyph, while the property `word-space` is ignored.
In the token `~?~` (for narrow elision), `?` can be any Unicode character, not only ASCII.

~	‿	U+E551	lyricsElision
~?~	‿	U+E550	lyricsElisionNarrow
~~	‿	U+E552	lyricsElisionWide
_	–	U+E553	lyricsHyphenBaseline
%	𝄎	U+E555	lyricsTextRepeat

```
\relative {
  \cadenzaOn
  b'~ b c fis, fis c' b e,
}
\addlyrics {
  Che~~in ques -- ta~ē~in quel -- l'al -- "tr_on" -- "da %"
}
\layout {
  \context {
    \Score
    \ekmSmuflOn #'lyric
  }
}
```



Analytics symbols

`\ekm-analytics DEFINITION`

Draw analytics symbols according to DEFINITION as markup.

`H`		U+E860	analyticsHauptstimme
`CH`		U+E86A	analyticsChoralmelodie
`RH`		U+E86B	analyticsHauptrhythmus
`N`		U+E861	analyticsNebenstimme
`[`		U+E862	analyticsStartStimme
`]`		U+E863	analyticsEndStimme
`Th`		U+E864	analyticsTheme
`hT`		U+E865	analyticsThemeRetrograde
`ihT`		U+E866	analyticsThemeRetrogradeInversion
`iTh`		U+E867	analyticsThemeInversion
`T`		U+E868	analyticsTheme1
`iT`		U+E869	analyticsInversion1

Function theory symbols

`\ekm-func DEFINITION`

Draw a function theory symbol according to DEFINITION as markup. The definition string consists of several parts which are all optional:

Paren Function , Bass , Soprano ^ Extra ... Paren

The bass symbol is placed below the function symbol.
The soprano symbol is placed above the function symbol.
The extra symbols are stacked vertically and raised to the right of the function symbol.
A leading/trailing parenthesis, bracket, or brace is placed separately before/after the entire symbol.
Used properties:

- `font-size` (0) for the function symbol.
- `func-size` (-4) relative to the font size for bass, soprano, and extra symbols.
- `func-skip` (2.5) for vertical distances.
- `func-space` (0.3) for horizontal space around the function symbol.

`\ekmFunc DEFINITION`

Set a function theory symbol as a music expression, for use in a `Lyrics` context. The symbol is drawn with a 4 steps smaller font size compared to `\ekm-func` .
DEFINITION is a string as described above, with a further optional suffix:

- `-` Starts an extender line after the symbol.
- `.` Stops an extender line at the symbol.
- `+` Inserts the symbol between notes with `\set stanza` .
- `*` Dito but with the 4 steps larger font size of `\ekm-func` .

Note that the `Lyrics` context requires the `Text_spanner_engraver` to draw extender lines.

`\ekmFuncList DEFINITION-LIST`

Set a sequence of function theory symbols as music expressions, for use in a `Lyrics` context.
DEFINITION-LIST is a list of strings as for `\ekmFunc` .

`T`	T	U+EA8B	functionTUpper
`Tg`	Tg		
`Tp`	Tp		
`t`	t	U+EA8C	functionTLower
`D`	D	U+EA7F	functionDUpper
`/D`	Ø		
`Dp`	Dp		

DD		U+EA81	functionDD
/DD		U+EA82	functionSlashedDD
d		U+EA80	functionDLower
S		U+EA89	functionSUpper
Sg			
Sp			
SS		U+EA7D	functionSSUpper
s		U+EA8A	functionSLower
ss		U+EA7E	functionSSLower
F		U+EA99	functionFUpper
G		U+EA83	functionGUpper
g		U+EA84	functionGLower
I		U+EA9A	functionIUpper
i		U+EA9B	functionILower
K		U+EA9C	functionKUpper
k		U+EA9D	functionKLower
L		U+EA9E	functionLUpper
l		U+EA9F	functionLLower
M		U+ED00	functionMUpper
m		U+ED01	functionMLower
N		U+EA85	functionNUpper
n		U+EA86	functionNLower
P		U+EA87	functionPUpper
p		U+EA88	functionPLower
r		U+ED03	functionRLower
V		U+EA8D	functionVUpper
v		U+EA8E	functionVLower
0		U+EA70	functionZero
	:		
9		U+EA79	functionNine

<		U+EA7A	functionLessThan
>		U+EA7C	functionGreaterThan
–		U+EA7B	functionMinus
+		U+EA98	functionPlus
o		U+EA97	functionRing
(		U+EA91	functionParensLeft
)		U+EA92	functionParensRight
[		U+EA8F	functionBracketLeft
]		U+EA90	functionBracketRight
{		U+EA93	functionAngleLeft
}		U+EA94	functionAngleRight
..		U+EA95	functionRepetition1
..+		U+EA96	functionRepetition2
b		U+ED60	csymAccidentalFlat
#		U+ED62	csymAccidentalSharp
bb		U+ED64	csymAccidentalDoubleFlat
x		U+ED63	csymAccidentalDoubleSharp
=		U+ED61	csymAccidentalNatural
~			

The tokens `b # bb x =` draw standard accidentals for chord symbols.

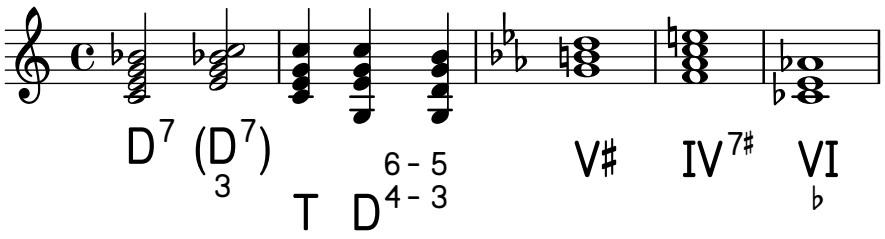
The token `~` draws a space with the dimensions of functionZero (U+EA70) . This is especially useful for empty extra symbols.

Ekmelos

/D		U+F644	functionSlashedD

The following example uses `\ekm-func` in text scripts to attach function theory symbols to chords and spacer rest. It sets `\textLengthOn` and `TextScript.staff-padding` for a consistent vertical alignment.

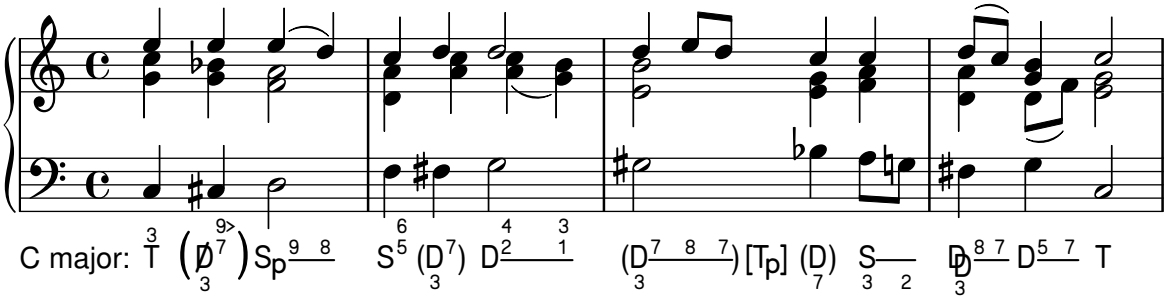
```
\relative c' {
  \textLengthOn

  \override TextScript.staff-padding = #6
  <c e g bes>2_\markup \ekm-func "D^7 "
  <e g bes! c>_\markup \ekm-func "(D,3^7)"

  \override TextScript.staff-padding = #11
  <c e g c>4_\markup \ekm-func "T___"
  <g e' g c>_\markup \ekm-func "D^4^6"
  s_\markup \ekm-func "^-^-"
  <g d' g b>_\markup \ekm-func "^3^5"

  \key es \major
  \override TextScript.staff-padding = #7
  <g' b d>1_\markup \ekm-func "V#"
  <f as c e>_\markup \ekm-func "IV^7#"
  <ces es as!>_\markup \ekm-func "VI,b"
}
```



The following example uses `\ekmFuncList` in a Lyrics context to synchronise function theory symbols to music. The Lyrics context requires the `Text_spanner_engraver` and is aligned to a NullVoice context. It is taken from lsr.di.unimi.it/LSR/Item?id=967 and adapted for Esmuflily.

```
funcSoprano = \relative c'' {
  e4 e e( d)
  c4 d d2
  d4 e8 d c4 c
  d8( c) <b g>4 c2
}

funcAltTenor = \relative c'' {
  <c g>4 <bes g> <a f>2
  <a d,>4 <c a> <c a>( <b g>)
  <b e,>2 <g e>4 <a f>
  <a d,>4 d,8( f) <g e>2
}

funcBass = \relative c {
  \clef bass
  c4 cis d2
  f4 fis g2
  gis2 bes4 a8 g
  fis4 g c,2
}

funcAligner = \relative c {
  c4 cis d d
  f4 fis g g
  gis4 gis8 gis bes4 a8 g
  fis8 fis g g c,2
}

funcSymbols = \lyricmode {
  \set stanza = #"C major:"
  \ekmFuncList #'(
    "T,,3"  " (*"  "/D,3^7^9>"  ")*"  "Sp^9-"  "^8."
    "S^5^6"  "(D,3^7)"  "D^2^4-"  "^1^3."
    "(D,3^7-"  "^8"  "^7."  "_) [Tp] +"  "(D,7)"  "S,3-"  " ,2."
    "DD,3^8-"  "^7."  "D^5-"  "^7."  "T"
  )
}

\layout {
  \context {
    \Lyrics
    \consists "Text_spanner_engraver"
    \override StanzaNumber.font-family = #'sans
    \override StanzaNumber.font-series = #'medium
  }
}
```

```
\new GrandStaff
<<
  \new Staff
    \new Voice \partCombine \funcSoprano \funcAltTenor

  \new Staff
  <<
    \new Voice \funcBass
    \new NullVoice = "funcaligner" \funcAligner
    \new Lyrics \lyricsto "funcaligner" \funcSymbols
  >>
>>
```



`\ekm-arrow STYLE ORIENTATION`

Draw an arrow or arrow head of STYLE according to ORIENTATION as markup.

Arrows can have glyphs for several orientations. The remaining orientations are achieved by flipping or rotating through 90° or 45°. All following six styles have glyphs for the 8 orientations `N` … `NW`.

Styles

`'black`	↑	U+EB60	arrowBlackUp
`'white`	⇧	U+EB68	arrowWhiteUp
`'open`	↑	U+EB70	arrowOpenUp
`'black-head`	▲	U+EB78	arrowheadBlackUp
`'white-head`	△	U+EB80	arrowheadWhiteUp
`'open-head`	∧	U+EB88	arrowheadOpenUp

Examples for all orientations

ORIENTATION	#N	#NE	#E	#SE	#S	#SW	#W	#NW	#NS	#NESW	#EW	#SENW
`'black`	↑	↗	→	↘	↓	↙	←	↖	↑	↗	→	↘
`'open-head`	∧	ㄱ	>	ㄴ	∨	ㄴ	<	ㄱ	∧	ㄱ	>	ㄴ

`\ekm-arrow-head AXIS DIRECTION FILLED`

Draw an arrow head as markup, i.e. `black-head` if FILLED is a true value, else `open-head` .

Ekmelos

`'simple`	↑	U+2191
`'double`	⇑	U+21D1
`'triple`	⟰	U+290A
`'quadruple`	⟰	U+27F0
`'black-wide`	⬆	U+2B06
`'white-wide`	⇧	U+21E7
`'triangle`	↑	U+2B61
`'triangle-bar`	↑	U+2B71
`'two-headed`	↟	U+2BED
`'dashed`	⇡	U+21E1
`'triangle-dashed`	⇡	U+2B6B
`'opposite`	⇅	U+21C5
`'triangle-opposite`	⇅	U+2B81
`'paired`	⇈	U+21C8
`'triangle-paired`	⇈	U+2B85
`'bent-tip`	↱	U+21B1
`'long-bent-tip`	⮡	U+2BA3
`'curving`	⤴	U+2934
`'equilateral-head`	▲	U+2B9D
`'three-d-head`	◭	U+2B99
`'black-triangle`	▲	U+25B2
`'white-triangle`	△	U+25B3
`'black-small-triangle`	▴	U+25B4
`'white-small-triangle`	▵	U+25B5
`'half-circle`	◒	U+2BCA
`'circle-half-black`	◓	U+25D3
`'square-half-black`	◧	U+2B12
`'diamond-half-black`	◈	U+2B18
`'circle-quarters`	◔	U+25D4

Percussion symbols

`\ekm-beater STYLE ORIENTATION`

Draw a percussion beater of STYLE according to ORIENTATION as markup.

Percussion beaters can have glyphs for several orientations. The remaining orientations are achieved by flipping or rotating through 90° or 30°. Most of the following styles have glyphs for the orientations `N,S,NE,NW` or `N,S`.

Styles

`'xyl-soft`		U+E770	pictBeaterSoftXylophoneUp
`'xyl-medium`		U+E774	pictBeaterMediumXylophoneUp
`'xyl-hard`		U+E778	pictBeaterHardXylophoneUp
`'xyl-wood`		U+E77C	pictBeaterWoodXylophoneUp
`'glsp-soft`		U+E780	pictBeaterSoftGlockenspielUp
`'glsp-hard`		U+E784	pictBeaterHardGlockenspielUp
`'timpani-soft`		U+E788	pictBeaterSoftTimpaniUp
`'timpani-medium`		U+E78C	pictBeaterMediumTimpaniUp
`'timpani-hard`		U+E790	pictBeaterHardTimpaniUp
`'timpani-wood`		U+E794	pictBeaterWoodTimpaniUp
`'yarn-soft`		U+E7A2	pictBeaterSoftYarnUp
`'yarn-medium`		U+E7A6	pictBeaterMediumYarnUp
`'yarn-hard`		U+E7AA	pictBeaterHardYarnUp
`'gum-soft`		U+E7BB	pictGumSoftUp
`'gum-medium`		U+E7BF	pictGumMediumUp
`'gum-hard`		U+E7C3	pictGumHardUp
`'bass-soft`		U+E798	pictBeaterSoftBassDrumUp
`'bass-medium`		U+E79A	pictBeaterMediumBassDrumUp
`'bass-hard`		U+E79C	pictBeaterHardBassDrumUp
`'bass-metal`		U+E79E	pictBeaterMetalBassDrumUp
`'bass-double`		U+E7A0	pictBeaterDoubleBassDrumUp
`'stick`		U+E7E8	pictDrumStick
`'stick-snare`		U+E7D1	pictBeaterSnareSticksUp
`'stick-jazz`		U+E7D3	pictBeaterJazzSticksUp
`'hammer-wood`		U+E7CB	pictBeaterHammerWoodUp
`'hammer-plastic`		U+E7CD	pictBeaterHammerPlasticUp
`'hammer-metal`		U+E7CF	pictBeaterHammerMetalUp

'wound-hard		U+E7B3	pictWoundHardUp
'wound-soft		U+E7B7	pictWoundSoftUp
'metal		U+E7C7	pictBeaterMetalUp
'brass-mallets		U+E7D9	pictBeaterBrassMalletsUp
'triangle		U+E7D5	pictBeaterTriangleUp
'triangle-plain		U+E7EF	pictBeaterTrianglePlain
'wire-brushes		U+E7D7	pictBeaterWireBrushesUp
'superball		U+E7AE	pictBeaterSuperballUp
'mallet		U+E7DF	pictBeaterMallet
'metal-hammer		U+E7E0	pictBeaterMetalHammer
'hammer		U+E7E1	pictBeaterHammer
'hand		U+E7E3	pictBeaterHand
'finger		U+E7E4	pictBeaterFinger
'fist		U+E7E5	pictBeaterFist
'fingernails		U+E7E6	pictBeaterFingernails

Examples for all orientations

ORIENTATION	#N	#NE	#E	#SE	#S	#SW	#W	#NW	#NS	#NESW	#EW	#SENW
'xyl-medium												
'bass-metal												
'finger												

`\ekm-fader LEVEL ORIENTATION`
`\ekm-midi LEVEL ORIENTATION`

Draw a fader (volume control) and a MIDI controller, respectively, as markup.
LEVEL ≥ 0 is a percent value.
LEVEL < 0 is a decibel (dB) value, e.g. -6.0 is equivalent to 50.
LEVEL is drawn as a label next to the control according to ORIENTATION . `#f` draws no label.
For the thumb position, LEVEL is rounded to the nearest integral percent value, limited to 100. If a symbol is defined exactly for this value, this symbol is drawn. Else if an empty control and a thumb symbol are defined, they are combined. Else the symbol for the value nearest to LEVEL is drawn.
Used properties:

- `label-format` (`#f`): `#f` uses `"~a%"` for percent and `"~adB"` for decibel values.
- `font-size` (0)
- `label-size` (-4) relative to the font size.
- `padding` (0.3)

`\ekm-fader`

`0`		U+EB2E	elecVolumeLevel0
	:		
`100`		U+EB33	elecVolumeLevel100
		U+EB2C	elecVolumeFader
		U+EB2D	elecVolumeFaderThumb

`\ekm-midi`

`0`		U+EB36	elecMIDIController0
	:		
`100`		U+EB3B	elecMIDIController100

Ekmelos

`\ekm-midi`		U+F6D2	elecMIDIController
		U+F6D3	elecMIDIControllerThumb

Examples for all orientations

ORIENTATION	#N	#NE	#E	#SE	#S	#SW	#W	#NW
\ekm-fader								
								
\ekm-midi								
								

Other symbols

`\ekm-fermata STYLE`

Draw a fermata as markup.

Used property:

- `direction`

Styles

`'default`		U+E4C0	fermataAbove
		U+E4C1	fermataBelow
`'short`		U+E4C4	fermataShortAbove
		U+E4C5	fermataShortBelow
`'long`		U+E4C6	fermataLongAbove
		U+E4C7	fermataLongBelow
`'veryshort`		U+E4C2	fermataVeryShortAbove
		U+E4C3	fermataVeryShortBelow
`'verylong`		U+E4C8	fermataVeryLongAbove
		U+E4C9	fermataVeryLongBelow
`'henzeshort`		U+E4CC	fermataShortHenzeAbove
		U+E4CD	fermataShortHenzeBelow
`'henzelong`		U+E4CA	fermataLongHenzeAbove
		U+E4CB	fermataLongHenzeBelow

Ekmelos

`'extrashort`		U+F69E	fermataExtraShortAbove
		U+F69F	fermataExtraShortBelow
`'extralong`		U+F6A0	fermataExtraLongAbove
		U+F6A1	fermataExtraLongBelow

`\ekm-eyeglasses DIRECTION`

Draw eyeglasses as markup.

		U+EC62	miscEyeglasses

Ekmelos

		U+F65F	miscEyeglassesRight

`\ekm-metronome COUNT`

Draw COUNT metronome strokes (tickings) as markup.
Used property:

- `word-space`

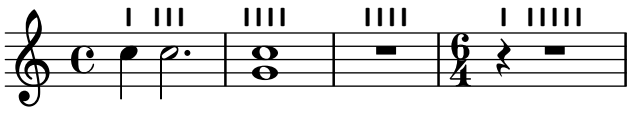
Ekmelos

U+F614 noteTick

`\ekmMetronome MUSIC`

Attach metronome strokes to each note, chord, or rest in MUSIC as a horizontally centered markup above the staff, using `\ekm-metronome` . The number of strokes equals the number of quarter note values of the respective duration (possibly rounded up).

```
\relative c'' {
  \ekmMetronome {
    c4
    c2.
    <g c>1
    R1
  }

  \time 6/4
  \ekmMetronome r4
  \ekmMetronome r1*5/4
}
```



They implement the underlying SMuFL output in Esmuflily.

`\ekm-char CODEPOINT`

Draw the glyph of CODEPOINT, or the point-stencil for zero.
Used property:

- `font-size` (0)

```
\ekm-char ##xE048
```


```
\ekm-char ##xEB27
```


`\ekm-charf CODEPOINT FEATURES`

Draw the glyph of CODEPOINT with font features.
FEATURES is either a list of one or more strings, or the number of a stylistic alternate, or a negative number to draw the path instead of the font glyph.
`#1` and `#'(1)` and `#'("salt 1")` are equivalent. `#0` and `#'()` do not set font features.
`#-1` and `#'(-1)` draw a filled path. Any other negative number `-N` draws the outline of the path with thickness `N` which is scaled to the current font size.
This command is independent of globally drawing paths.
Used property:

- `font-size` (0)

```
\ekm-charf ##xE242 #0
```


```
\ekm-charf ##xE242 #'("salt 1")
```


```
\ekm-charf ##xE242 #'(2)
```


```
\ekm-charf ##xE242 #-20
```


`\ekm-str STRING`

Draw STRING with the selected font.
This command is independent of globally drawing paths.

`\ekm-chars CODEPOINT-LIST`

Draw the glyphs of the CODEPOINTs in the list adjoined horizontally without padding, or the point-stencil for an empty list.
Used property:

- `font-size` (0)

```
\ekm-chars #'(#xE260 #xE2B4 #xE2B2)
```


```
\ekm-chars #'(#xE262 #xE566 #xEAA6 #xEAA5)
```


```
\ekm-chars #'(#xE1F0 #xE1F7 #xE1FC #xE1F7 #xE1F4)
```


`\ekm-text EXTEXT`

Draw EXTEXT . Depending on the argument type, it calls `\ekm-char`, `\ekm-charf`, or `\ekm-chars`, or it draws markup or the empty-stencil.

```
\ekm-text #'(#xE242 0)
```

```
\ekm-text #'(#xE242 "salt 1")
```

```
\ekm-text #'(#xE242 -20)
```

```
\ekm-text #'(#xE260 #xE2B4 #xE2B2)
```

`\ekm-concat EXTEXT-LIST`

Draw the EXTEXTs in the list stacked in a line without padding.

`\ekm-line EXTEXT-LIST`

Draw the EXTEXTs in the list stacked in a line.
Used properties:

- `word-space`
- `text-direction`

```
\ekm-line #'(#xE046 "al fine")
```

D.C. al fine

```
\ekm-line #'(#xE6D0 "with" #xE78E)
```


```
\ekm-line #'((#xE6D0 1) "with" #xE78E)
```

`\ekm-combine CODEPOINT X Y CODEPOINT2`

Combine the glyphs of CODEPOINT and CODEPOINT2, where CODEPOINT2 is translated scaled by X,Y.

```
\ekm-combine ##xECA5 #-0.5 #1.0 ##xE56E
```

```
\ekm-combine ##xEA7F #0.3 #0 ##xE87B
```

`\ekm-cchar CENTER CODEPOINT`

Draw the glyph of CODEPOINT, centered horizontally if CENTER is `CX` or `CXY` , and vertically if CENTER is `CY` or `CXY` .

`\ekm-ctext CENTER EXTEXT`

Draw EXTEXT . Markup is centered like `\ekm-cchar` . A list of code points is centered only horizontally. A single code point (possibly with font features) is never centered.
This command is intended to draw symbols on stem.

`\ekm-def MAP DEFINITION`

Draw a text according to DEFINITION .
MAP is an alist of EXTEXTs mapped onto tokens (strings). A token which is a prefix of other tokens must be arranged after them in MAP, i.e. the correct order is `"abc"` , `"ab"` , `"a"` . A shared token (`" "`, `"_"`, etc.) can be overridden. The special value `#f` draws nothing, i.e. the token is simply ignored.

```
#(define my-map `(
   (".|:" . #xE040)
   ("tr#~" . (#xE262 #xE566 #xEAA6 #xEAA5))
   ("timp" . (#xE6D0 1))
   (" " . #f)
   ("w" . "with")
   ("box/" . ,(markup #:box #:ekm-beater 'timpani-medium NE))
))

\ekm-def #my-map #".|:___tr#~"

\ekm-def #my-map #"timp w box/"
```

`\ekm-number STYLE NUMBER`

Draw the integer NUMBER according to STYLE, either as a sequence of decimal digit symbols or as a single number symbol (`string` and `scale`).

Styles

`'time`	4
`'time-turned`	4
`'time-reversed`	4
`'tuplet`	4
`'finger`	4
`'finger-italic`	4
`'fbass`	4
`'func`	4
`'string`	④
`'scale`	4
`'sans`	4
`'roman`	4
`'typewriter`	4

`\ekm-label ORIENTATION LABEL ARG`

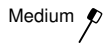
Combine a markup with another markup placed as a label next to it according to ORIENTATION (= `#f` ignores the label).

Used properties:

- `font-size` (0)
- `label-size` (-4) relative to the font size.
- `padding` (0.3)

```
\ekm-label #SE \ekm-char ##xE836 "G"
```


```
\ekm-label #NW "Medium" \ekm-char ##xE78E
```



`\ekm-orient TYPE STYLE ORIENTATION`

Draw the symbol of TYPE and STYLE according to ORIENTATION as markup.

```
\ekm-orient #'arrow #'black #NW
```



```
\ekm-orient #'beater #'timpani-medium #NE
```

Extended text

Some commands accept an EXTEXT value, or a pair or list of EXTEXT values.
EXTEXT can be:

- A single code point (integer). Calls `\ekm-char`.

  ```
  ##xE695
  ```

- A list of a single code point followed by font features, i.e. one or more strings or a number 0 to 31 of a stylistic alternate, or a negative number to draw the path instead of the font glyph. Higher values are treated as code points (see below). Calls `\ekm-charf`.

  ```
  #'(#xE626 "salt 2")
  ```

  ```
  #'(#xE626 2)
  ```

  ```
  #'(#xE626 -1)
  ```

- A list of one or more code points. Calls `\ekm-chars` .

  ```
  #'(#xE260 #xE567 #xE262)
  ```

- Any markup. Note that the commands `\ekmTremolo` and `\ekmStem` interpret some strings as names of symbols.

  ```
  #"poco a poco"
  ```

  ```
  #(markup #:box #:ekm-char #xED19)
  ```

- `#f`. Draws the empty-stencil.

Definition string

Some commands and properties accept a DEFINITION value. This is a string of one or more tokens, each consisting of one or more characters. Their corresponding symbols are stacked in a line. Any other character in the string produces a warning and only the text created so far is drawn.

Shared tokens

Additional tokens that are always applicable in DEFINITION values.
The shared tokens in the standard table defines space markup:

`<space>`	`\hspace #1`	SP
_	`\hspace #0.17`	HSP
__	`\hspace #0.78`	THSP
___	`\hspace #2`	ENSP
____	`\hspace #4`	EMSP
`	`#f`	ZWSP

Orientation

Some commands accept an ORIENTATION value. This is the sum of axis (0, 1, or ±0.5 for diagonal) and direction (±1). The following symbols are defined for the 12 possible values. The last four values are intended for “bilateral” orientations. An unsupported value is substituted with `N` .

`N`	2	Y	+ UP
`NE`	1.5	0.5	+ UP
`E`	1	X	+ RIGHT
`SE`	0.5	-0.5	+ RIGHT
`S`	0	Y	+ DOWN
`SW`	-0.5	0.5	+ DOWN
`W`	-1	X	+ LEFT
`NW`	-1.5	-0.5	+ LEFT
`NS`	-2	Y	+ -3
`NESW`	-2.5	0.5	+ -3
`EW`	-3	X	+ -3
`SENW`	-3.5	-0.5	+ -3

The commands `\ekm-arrow` and `\ekm-beater` support all 12 orientations. Missing symbols are completed by flipping or rotating. Missing bilateral symbols are substituted with the symbols for `N`,`NE`,`E`,`SE`. Currently, only the arrow style `simple` in Ekmelos has glyphs for all 12 orientations.