![<?xml version="1.0" encoding="utf-16"?> <ec version="1.5">   <title>Chant</title>   <ScoreWidth>160.02</ScoreWidth>   <ScoreHeight>247.015</ScoreHeight>   <ScoreP1Padding>-0.12347</ScoreP1Padding>   <LeftMargin>0</LeftMargin>   <RightMargin>0.9</RightMargin>   <TopMargin>0</TopMargin>   <BottomMargin>0</BottomMargin>   <OpeningQuote>„</OpeningQuote>   <ClosingQuote>”</ClosingQuote>   <VoiceName>   </VoiceName>   <LilyPond>\version "2.20.0"  \include "gregorian.ly"  \paper {   paper-width = 160.02\mm   paper-height = 247.015\mm   top-margin = 0\mm   bottom-margin = 0\mm   left-margin = 0\mm   right-margin = 2.0\mm   ragged-bottom = ##t   print-page-number = ##f   page-breaking = #ly:minimal-breaking   system-system-spacing =     #'((basic-distance . 12)        (minimum-distance . 12)        (padding . 0)        (stretchability . 0)) %  page-breaking-system-system-spacing = #'((padding . 2))   markup-system-spacing = #'((padding . 0))   top-markup-spacing = #'((basic-distance . 0) (minimum-distance . 0) (padding . 0)) }  #(set-global-staff-size 18)  #(define currBSign 0) #(define hideStems #t)  #(define (set-breathe bsign)  (begin   (set! currBSign bsign)  (let* ((columnOfBSign (ly:grob-parent (ly:grob-parent (ly:grob-parent currBSign X) X) X))    (system (ly:grob-parent columnOfBSign X))       (columns (ly:grob-object system 'columns))    (lastColumn (ly:grob-array-ref columns (- (ly:grob-array-length columns) 1)))   )   (if (eqv? columnOfBSign lastColumn)    (ly:grob-set-property! currBSign 'extra-offset '(0.69 . 0))   )  )  ) )  #(define (staff-print staffgrob)  (let* ((stencil (ly:staff-symbol::print staffgrob))    (Xextent (ly:stencil-extent stencil X))    (Yextent (ly:stencil-extent stencil Y))             (X0 (car Xextent))    (X1 (cdr Xextent))    (X1sc (+ X1 0.69))    )    (ly:make-stencil (ly:stencil-expr (ly:stencil-scale stencil (/ (- X1sc X0) (- X1 X0)) 1.0))                          (cons X0 X1sc)                          Yextent    )  ) )  #(define (drill-down-grob-add-note-head parentgrob noteheads)   (let*    (     (elems (ly:grob-object parentgrob 'elements))     (elems-count (ly:grob-array-length elems))    )    (let loopJ ((j 0)        )        (if (&lt; j elems-count)         (loopJ (let* ((grob (ly:grob-array-ref elems j))            (grobname (grob-name grob))           )           (if (eqv? grobname 'LedgerLineSpanner)            (let* (                (allnoteheadarray (ly:grob-object grob 'note-heads))                (allnoteheads (ly:grob-array-&gt;list allnoteheadarray))               )             (map (lambda (notehead)                (if (not (memq notehead allnoteheads))                 (ly:pointer-group-interface::add-grob grob 'note-heads notehead)                )               )               noteheads             )            )           )           (if (ly:grob-array? (ly:grob-object grob 'elements))            (drill-down-grob-add-note-head grob noteheads)           )           (1+ j)          )        )       )    )   ) )  #(define (staff-break-callback staffgrob)     (let*   (     (system (ly:grob-parent (ly:grob-parent staffgrob X) X))     (columns (ly:grob-object system 'columns))     (columns-count (ly:grob-array-length columns))   )   (if (ly:grob? currBSign)    (let* ((columnOfBSign (ly:grob-parent (ly:grob-parent (ly:grob-parent currBSign X) X) X))       (lastColumn (ly:grob-array-ref columns (- columns-count 1)))      )      (if (eqv? columnOfBSign lastColumn)       (ly:grob-set-property! currBSign 'extra-offset '(0.69 . 0))       (set! currBSign 0)      )      (set! currBSign 0)    )   )   (let loopI  ((i 0)      )        (if (&lt; i columns-count)       (loopI (let* ((column (ly:grob-array-ref columns i))           (columnname (grob-name column))          )          (if (and (eqv? columnname 'PaperColumn) (ly:grob-array? (ly:grob-object column 'elements)))           (let*            (             (elems (ly:grob-object column 'elements))             (elems-count (ly:grob-array-length elems))            )            (let loopJ ((j 0)                 )                 (if (&lt; j elems-count)                 (loopJ (let* ((grob (ly:grob-array-ref elems j))                    (grobname (grob-name grob))                   )                   (if (eqv? grobname 'NoteColumn)                    (show-notes grob)                   )                   (1+ j)                  )                )               )            )            (set! i columns-count)           )          )          (1+ i)         )       )      )   )  ) )  #(define (grob-name grob)    (if (ly:grob? grob)        (assoc-ref (ly:grob-property grob 'meta) 'name)        #f))  #(define (show-flag flag)  (ly:grob-set-property! flag 'transparent hideStems) )     #(define (show-stem stem)  (let*  ((beam (ly:grob-object stem 'beam))    )    (begin     (ly:grob-set-property! stem 'transparent hideStems)     (if (ly:grob? beam)(ly:grob-set-property! beam 'transparent hideStems))    )  ) )  #(define (show-note notehead)  (let* ((stem (ly:grob-object (ly:grob-parent notehead X) 'stem))         (beam (ly:grob-object stem 'beam))      (flag (ly:grob-object notehead 'flag))         (acci (ly:grob-object notehead 'accidental-grob))         (dots (ly:grob-object notehead 'dot)))  (begin   (ly:grob-set-property! notehead 'transparent #f)   (ly:grob-set-property! notehead 'no-ledgers #f)   (if (ly:grob? dots)(ly:grob-set-property! dots 'transparent #f))   (if (ly:grob? acci)(ly:grob-set-property! acci 'transparent #f))   (ly:grob-set-property! stem 'transparent hideStems)   (if (ly:grob? beam)(ly:grob-set-property! beam 'transparent hideStems))   (if (ly:grob? flag)(ly:grob-set-property! flag 'transparent hideStems))  )) )  #(define (show-notes note-column)   (begin    (let* (       (elems (ly:grob-object note-column 'elements))       (elems-count (ly:grob-array-length elems))       (Yparent (ly:grob-parent note-column Y))       (noteheads (list))      )         (let loop ((staff-pos-sum 0)       (i 0)         )         (if (&lt; i elems-count)        (loop (let* ((grob (ly:grob-array-ref elems i))            (grobname (grob-name grob))           )           (if (eqv? grobname 'NoteHead)            (begin             (show-note grob)             (set! noteheads (append noteheads (list grob)))            )            (if (eqv? grobname 'Stem)             (show-stem grob)             (if (eqv? grobname 'Flag)              (show-flag grob)             )            )           )          )         (1+ i)        )       )     )     (do ((stopit (eqv? (grob-name Yparent) 'VerticalAxisGroup) (set! stopit (eqv? (grob-name Yparent) 'VerticalAxisGroup))))      (stopit)      (set! Yparent (ly:grob-parent Yparent Y))     )     (if (ly:grob-array? (ly:grob-object Yparent 'elements))      (drill-down-grob-add-note-head Yparent noteheads)     )   )  ) )     % FAKE BREVE BEGIN fakeBreveStencil = #(             lambda(grob)     (let*           (            (note (let* ()                 (ly:grob-set-property! grob 'duration-log -1)                        (ly:note-head::print grob)               )            )           )           (             ly:make-stencil (ly:stencil-expr note)                             (ly:stencil-extent note X)                             (ly:stencil-extent note Y)           )     )  )  fakeBreve = { \once \override NoteHead #'stencil = #fakeBreveStencil \once \override Staff.Stem #'transparent = ##t \once \override Staff.Beam #'transparent = ##t \once \override Staff.Flag #'transparent = ##t \once \override Stem #'length = #0 \once \override Dots #'transparent = ##t } % FAKE BREVE END  leftAlign = {  \once \override LyricText #'self-alignment-X = #LEFT } centerAlign = {   \once \override LyricText #'self-alignment-X = #CENTER } rightAlign = {   \once \override LyricText #'self-alignment-X = #1 } hairpinToBarline = { \override Hairpin #'to-barline = ##t  } hairpinToNote = { \override Hairpin #'to-barline = ##f  }  alignAt = #(define-music-function      (parser location xOffset)      (number?)   (display xOffset)    #{       \once \override LyricText #'self-alignment-X = $xOffset    #})  stemOn = { \revert Staff.Stem #'transparent  \revert Staff.Beam #'transparent \revert Staff.Flag #'transparent \revert Voice.Stem #'length } stemOff = { \override Staff.Stem #'transparent = ##t \override Staff.Beam #'transparent = ##t \override Staff.Flag #'transparent = ##t \override Stem #'length = #0 }  bigNotes = { \override NoteHead #'font-size = #3 } smallNotes = { \override NoteHead #'font-size = #-3 } normalNotes = { \override NoteHead #'font-size = #0 }  bN = { \once \override NoteHead #'font-size = #3 } sN = { \once \override NoteHead #'font-size = #-3 } nN = { \once \override NoteHead #'font-size = #0 }  shOn = { \override NoteColumn #'force-hshift = #0.18 } shOff = { \revert NoteColumn #'force-hshift }  packedOffOnce = { \once \override Score.SpacingSpanner #'packed-spacing = ##f } packedOff = { \override Score.SpacingSpanner #'packed-spacing = ##f } packedOn = { \override Score.SpacingSpanner #'packed-spacing = ##t } packedRevert = { \revert Score.SpacingSpanner #'packed-spacing }  midRest = { \override Voice.Rest #'staff-position = #0 } shRest = { \revert Voice.Rest #'staff-position }  modernAccidentals = #(set-accidental-style 'modern) modernCautionaryAccidentals = #(set-accidental-style 'modern-cautionary) defaultAccidentals = #(set-accidental-style 'default) pianoAccidentals = #(set-accidental-style 'piano) voiceAccidentals = #(set-accidental-style 'voice) modernVoiceAccidentals = #(set-accidental-style 'modern-voice) modernVoiceCautionaryAccidentals = #(set-accidental-style 'modern-voice-cautionary) pianoCautionaryAccidentals = #(set-accidental-style 'piano-cautionary) neoModernAccidentals = #(set-accidental-style 'neo-modern) neoModernCautionaryAccidentals = #(set-accidental-style 'neo-modern-cautionary) dodecaphonicAccidentals = #(set-accidental-style 'dodecaphonic) choralAccidentals = #(set-accidental-style 'choral) teachingAccidentals = #(set-accidental-style 'teaching) noResetAccidentals = #(set-accidental-style 'no-reset) forgetAccidentals = #(set-accidental-style 'forget)  go = { \grace {s16 }  }    chant = { \key f \major c''4 bes'2 a'4 g' f' a' c'' c'' c''2 \divisioMaxima \fakeBreve c''4 \hideNotes \fakeBreve c'' \fakeBreve c'' \fakeBreve c'' \unHideNotes bes'4 a' bes'( c'') g'2 \hideNotes \grace {s16 } \finalis  }  verba = \lyricmode { "Í" -- "me" "most" "áld" -- "já" -- "tok" "az" "U" -- "rat" \alignAt #0 "ti" \alignAt #0 "az" \alignAt #-1.337 "Úr" \alignAt #-0.439 "min" -- "den" "szol" -- "gá" -- "i."  }   \score {   \new Staff &lt;&lt;     #(set-accidental-style 'modern-cautionary)     \new Voice = "melody" \chant     \new Lyrics = "one" \lyricsto melody \verba   &gt;&gt;   \layout {     ragged-right = ##t     line-thickness = #(/ (ly:output-def-lookup $defaultpaper 'staff-space) 10)     \context {       \Staff       \override BarLine  #'stencil = ##f       \remove Time_signature_engraver       \override Stem #'transparent = #hideStems       \override Beam #'transparent = #hideStems       \override Flag #'transparent = #hideStems       \override StaffSymbol #'after-line-breaking = #staff-break-callback       \override StaffSymbol #'thickness = #(/ (ly:output-def-lookup $defaultpaper 'staff-space) 3.0)       \override StaffSymbol #'stencil = #staff-print       \override InstrumentName #'font-name = #"Century Schoolbook L"     }     \context {       \Voice       \override Stem #'length = #0    \override BreathingSign #'after-line-breaking = #set-breathe     }     \context {       \Score       \override  RehearsalMark #'self-alignment-X = #LEFT       \override RehearsalMark #'break-align-symbols = #'(key-signature)       \override  NonMusicalPaperColumn #'padding = #0.8       \override SpacingSpanner #'packed-spacing = ##f       \override SpacingSpanner #'spacing-increment = #0       skipBars= ##t       barAlways = ##t       lyricMelismaAlignment = #CENTER     }     \context {       \Lyrics       \override LyricSpace #'minimum-distance = #0.8       \override LyricText #'font-size = #0       \override LyricText #'font-name = #"Century Schoolbook L"       \override LyricText #'color = #(rgb-color 0.000 0.000 0.000)       \override LyricHyphen #'color = #(rgb-color 0.000 0.000 0.000)       \override LyricExtender #'color = #(rgb-color 0.000 0.000 0.000)     }   } }  \header { tagline = ""  title = "" } </LilyPond>   <EasyChant>Í - me  most áld-já-tok az U-rat s4  f2  m4    r   d  m   s  s s2  ti az Úr min-den szol - gá - i. s0           f4  m      f  s r2</EasyChant>   <Clef>0</Clef>   <LyricsLocation>0</LyricsLocation>   <Key>8</Key>   <LyricsSpacing>0.8</LyricsSpacing>   <TimeOption>0</TimeOption>   <TimeText>Nincs (recitálás)</TimeText>   <StaffSizeOption>8</StaffSizeOption>   <StaffSizeText>18</StaffSizeText>   <LyricsSize>0</LyricsSize>   <LyricsFontName>Century Schoolbook L</LyricsFontName>   <LyricsFontBold>False</LyricsFontBold>   <LyricsFontItalic>False</LyricsFontItalic>   <LyricsFontColor>Color [A=255, R=0, G=0, B=0]</LyricsFontColor>   <LineSpacing>12</LineSpacing>   <LineThickness>10</LineThickness>   <Options>0,1,2,4,5,6,7,8,9,10</Options>   <Voices>1</Voices>   <pages>1</pages>   <page>1</page> </ec>]()

*Vers:*

![<?xml version="1.0" encoding="utf-16"?> <ec version="1.5">   <title>Chant1</title>   <ScoreWidth>160.02</ScoreWidth>   <ScoreHeight>247.015</ScoreHeight>   <ScoreP1Padding>43.0036</ScoreP1Padding>   <LeftMargin>0</LeftMargin>   <RightMargin>0.9</RightMargin>   <TopMargin>0</TopMargin>   <BottomMargin>0</BottomMargin>   <OpeningQuote>„</OpeningQuote>   <ClosingQuote>”</ClosingQuote>   <VoiceName>   </VoiceName>   <LilyPond>\version "2.20.0"  \include "gregorian.ly"  \paper {   paper-width = 160.02\mm   paper-height = 247.015\mm   top-margin = 0\mm   bottom-margin = 0\mm   left-margin = 0\mm   right-margin = 2.0\mm   ragged-bottom = ##t   print-page-number = ##f   page-breaking = #ly:minimal-breaking   system-system-spacing =     #'((basic-distance . 12)        (minimum-distance . 12)        (padding . 0)        (stretchability . 0)) %  page-breaking-system-system-spacing = #'((padding . 2))   markup-system-spacing = #'((padding . 27.09))   top-markup-spacing = #'((basic-distance . 0) (minimum-distance . 0) (padding . 0)) }  #(set-global-staff-size 18)  #(define currBSign 0) #(define hideStems #t)  #(define (set-breathe bsign)  (begin   (set! currBSign bsign)  (let* ((columnOfBSign (ly:grob-parent (ly:grob-parent (ly:grob-parent currBSign X) X) X))    (system (ly:grob-parent columnOfBSign X))       (columns (ly:grob-object system 'columns))    (lastColumn (ly:grob-array-ref columns (- (ly:grob-array-length columns) 1)))   )   (if (eqv? columnOfBSign lastColumn)    (ly:grob-set-property! currBSign 'extra-offset '(0.69 . 0))   )  )  ) )  #(define (staff-print staffgrob)  (let* ((stencil (ly:staff-symbol::print staffgrob))    (Xextent (ly:stencil-extent stencil X))    (Yextent (ly:stencil-extent stencil Y))             (X0 (car Xextent))    (X1 (cdr Xextent))    (X1sc (+ X1 0.69))    )    (ly:make-stencil (ly:stencil-expr (ly:stencil-scale stencil (/ (- X1sc X0) (- X1 X0)) 1.0))                          (cons X0 X1sc)                          Yextent    )  ) )  #(define (drill-down-grob-add-note-head parentgrob noteheads)   (let*    (     (elems (ly:grob-object parentgrob 'elements))     (elems-count (ly:grob-array-length elems))    )    (let loopJ ((j 0)        )        (if (&lt; j elems-count)         (loopJ (let* ((grob (ly:grob-array-ref elems j))            (grobname (grob-name grob))           )           (if (eqv? grobname 'LedgerLineSpanner)            (let* (                (allnoteheadarray (ly:grob-object grob 'note-heads))                (allnoteheads (ly:grob-array-&gt;list allnoteheadarray))               )             (map (lambda (notehead)                (if (not (memq notehead allnoteheads))                 (ly:pointer-group-interface::add-grob grob 'note-heads notehead)                )               )               noteheads             )            )           )           (if (ly:grob-array? (ly:grob-object grob 'elements))            (drill-down-grob-add-note-head grob noteheads)           )           (1+ j)          )        )       )    )   ) )  #(define (staff-break-callback staffgrob)     (let*   (     (system (ly:grob-parent (ly:grob-parent staffgrob X) X))     (columns (ly:grob-object system 'columns))     (columns-count (ly:grob-array-length columns))   )   (if (ly:grob? currBSign)    (let* ((columnOfBSign (ly:grob-parent (ly:grob-parent (ly:grob-parent currBSign X) X) X))       (lastColumn (ly:grob-array-ref columns (- columns-count 1)))      )      (if (eqv? columnOfBSign lastColumn)       (ly:grob-set-property! currBSign 'extra-offset '(0.69 . 0))       (set! currBSign 0)      )      (set! currBSign 0)    )   )   (let loopI  ((i 0)      )        (if (&lt; i columns-count)       (loopI (let* ((column (ly:grob-array-ref columns i))           (columnname (grob-name column))          )          (if (and (eqv? columnname 'PaperColumn) (ly:grob-array? (ly:grob-object column 'elements)))           (let*            (             (elems (ly:grob-object column 'elements))             (elems-count (ly:grob-array-length elems))            )            (let loopJ ((j 0)                 )                 (if (&lt; j elems-count)                 (loopJ (let* ((grob (ly:grob-array-ref elems j))                    (grobname (grob-name grob))                   )                   (if (eqv? grobname 'NoteColumn)                    (show-notes grob)                   )                   (1+ j)                  )                )               )            )            (set! i columns-count)           )          )          (1+ i)         )       )      )   )  ) )  #(define (grob-name grob)    (if (ly:grob? grob)        (assoc-ref (ly:grob-property grob 'meta) 'name)        #f))  #(define (show-flag flag)  (ly:grob-set-property! flag 'transparent hideStems) )     #(define (show-stem stem)  (let*  ((beam (ly:grob-object stem 'beam))    )    (begin     (ly:grob-set-property! stem 'transparent hideStems)     (if (ly:grob? beam)(ly:grob-set-property! beam 'transparent hideStems))    )  ) )  #(define (show-note notehead)  (let* ((stem (ly:grob-object (ly:grob-parent notehead X) 'stem))         (beam (ly:grob-object stem 'beam))      (flag (ly:grob-object notehead 'flag))         (acci (ly:grob-object notehead 'accidental-grob))         (dots (ly:grob-object notehead 'dot)))  (begin   (ly:grob-set-property! notehead 'transparent #f)   (ly:grob-set-property! notehead 'no-ledgers #f)   (if (ly:grob? dots)(ly:grob-set-property! dots 'transparent #f))   (if (ly:grob? acci)(ly:grob-set-property! acci 'transparent #f))   (ly:grob-set-property! stem 'transparent hideStems)   (if (ly:grob? beam)(ly:grob-set-property! beam 'transparent hideStems))   (if (ly:grob? flag)(ly:grob-set-property! flag 'transparent hideStems))  )) )  #(define (show-notes note-column)   (begin    (let* (       (elems (ly:grob-object note-column 'elements))       (elems-count (ly:grob-array-length elems))       (Yparent (ly:grob-parent note-column Y))       (noteheads (list))      )         (let loop ((staff-pos-sum 0)       (i 0)         )         (if (&lt; i elems-count)        (loop (let* ((grob (ly:grob-array-ref elems i))            (grobname (grob-name grob))           )           (if (eqv? grobname 'NoteHead)            (begin             (show-note grob)             (set! noteheads (append noteheads (list grob)))            )            (if (eqv? grobname 'Stem)             (show-stem grob)             (if (eqv? grobname 'Flag)              (show-flag grob)             )            )           )          )         (1+ i)        )       )     )     (do ((stopit (eqv? (grob-name Yparent) 'VerticalAxisGroup) (set! stopit (eqv? (grob-name Yparent) 'VerticalAxisGroup))))      (stopit)      (set! Yparent (ly:grob-parent Yparent Y))     )     (if (ly:grob-array? (ly:grob-object Yparent 'elements))      (drill-down-grob-add-note-head Yparent noteheads)     )   )  ) )     % FAKE BREVE BEGIN fakeBreveStencil = #(             lambda(grob)     (let*           (            (note (let* ()                 (ly:grob-set-property! grob 'duration-log -1)                        (ly:note-head::print grob)               )            )           )           (             ly:make-stencil (ly:stencil-expr note)                             (ly:stencil-extent note X)                             (ly:stencil-extent note Y)           )     )  )  fakeBreve = { \once \override NoteHead #'stencil = #fakeBreveStencil \once \override Staff.Stem #'transparent = ##t \once \override Staff.Beam #'transparent = ##t \once \override Staff.Flag #'transparent = ##t \once \override Stem #'length = #0 \once \override Dots #'transparent = ##t } % FAKE BREVE END  leftAlign = {  \once \override LyricText #'self-alignment-X = #LEFT } centerAlign = {   \once \override LyricText #'self-alignment-X = #CENTER } rightAlign = {   \once \override LyricText #'self-alignment-X = #1 } hairpinToBarline = { \override Hairpin #'to-barline = ##t  } hairpinToNote = { \override Hairpin #'to-barline = ##f  }  alignAt = #(define-music-function      (parser location xOffset)      (number?)   (display xOffset)    #{       \once \override LyricText #'self-alignment-X = $xOffset    #})  stemOn = { \revert Staff.Stem #'transparent  \revert Staff.Beam #'transparent \revert Staff.Flag #'transparent \revert Voice.Stem #'length } stemOff = { \override Staff.Stem #'transparent = ##t \override Staff.Beam #'transparent = ##t \override Staff.Flag #'transparent = ##t \override Stem #'length = #0 }  bigNotes = { \override NoteHead #'font-size = #3 } smallNotes = { \override NoteHead #'font-size = #-3 } normalNotes = { \override NoteHead #'font-size = #0 }  bN = { \once \override NoteHead #'font-size = #3 } sN = { \once \override NoteHead #'font-size = #-3 } nN = { \once \override NoteHead #'font-size = #0 }  shOn = { \override NoteColumn #'force-hshift = #0.18 } shOff = { \revert NoteColumn #'force-hshift }  packedOffOnce = { \once \override Score.SpacingSpanner #'packed-spacing = ##f } packedOff = { \override Score.SpacingSpanner #'packed-spacing = ##f } packedOn = { \override Score.SpacingSpanner #'packed-spacing = ##t } packedRevert = { \revert Score.SpacingSpanner #'packed-spacing }  midRest = { \override Voice.Rest #'staff-position = #0 } shRest = { \revert Voice.Rest #'staff-position }  modernAccidentals = #(set-accidental-style 'modern) modernCautionaryAccidentals = #(set-accidental-style 'modern-cautionary) defaultAccidentals = #(set-accidental-style 'default) pianoAccidentals = #(set-accidental-style 'piano) voiceAccidentals = #(set-accidental-style 'voice) modernVoiceAccidentals = #(set-accidental-style 'modern-voice) modernVoiceCautionaryAccidentals = #(set-accidental-style 'modern-voice-cautionary) pianoCautionaryAccidentals = #(set-accidental-style 'piano-cautionary) neoModernAccidentals = #(set-accidental-style 'neo-modern) neoModernCautionaryAccidentals = #(set-accidental-style 'neo-modern-cautionary) dodecaphonicAccidentals = #(set-accidental-style 'dodecaphonic) choralAccidentals = #(set-accidental-style 'choral) teachingAccidentals = #(set-accidental-style 'teaching) noResetAccidentals = #(set-accidental-style 'no-reset) forgetAccidentals = #(set-accidental-style 'forget)  go = { \grace {s16 }  }    chant = { \key f \major \fakeBreve g'4 \hideNotes \fakeBreve g' \fakeBreve g' \fakeBreve g' \fakeBreve g' \fakeBreve g' \fakeBreve g' \fakeBreve g' \fakeBreve g' \fakeBreve g' \unHideNotes \hideNotes \grace {s16 } \finalis  }  verba = \lyricmode { \alignAt #-0.698 "Kik" \alignAt #0 "az" \alignAt #-1.337 "Úr" \alignAt #-0.825 "házában" \alignAt #-1.18 "álltok," \alignAt #0 "a" \alignAt #-0.146 "mi" \alignAt #-1.161 "Istenünk" \alignAt #-0.822 "házának" \alignAt #-1.084 "udvaraiban."  }   \score {   \new Staff &lt;&lt;     #(set-accidental-style 'modern-cautionary)     \new Voice = "melody" \chant     \new Lyrics = "one" \lyricsto melody \verba   &gt;&gt;   \layout {     ragged-right = ##t     line-thickness = #(/ (ly:output-def-lookup $defaultpaper 'staff-space) 10)     \context {       \Staff       \override BarLine  #'stencil = ##f       \remove Time_signature_engraver       \override Stem #'transparent = #hideStems       \override Beam #'transparent = #hideStems       \override Flag #'transparent = #hideStems       \override StaffSymbol #'after-line-breaking = #staff-break-callback       \override StaffSymbol #'thickness = #(/ (ly:output-def-lookup $defaultpaper 'staff-space) 3.0)       \override StaffSymbol #'stencil = #staff-print       \override InstrumentName #'font-name = #"Century Schoolbook L"     }     \context {       \Voice       \override Stem #'length = #0    \override BreathingSign #'after-line-breaking = #set-breathe     }     \context {       \Score       \override  RehearsalMark #'self-alignment-X = #LEFT       \override RehearsalMark #'break-align-symbols = #'(key-signature)       \override  NonMusicalPaperColumn #'padding = #0.8       \override SpacingSpanner #'packed-spacing = ##f       \override SpacingSpanner #'spacing-increment = #0       skipBars= ##t       barAlways = ##t       lyricMelismaAlignment = #CENTER     }     \context {       \Lyrics       \override LyricSpace #'minimum-distance = #0.8       \override LyricText #'font-size = #0       \override LyricText #'font-name = #"Century Schoolbook L"       \override LyricText #'color = #(rgb-color 0.000 0.000 0.000)       \override LyricHyphen #'color = #(rgb-color 0.000 0.000 0.000)       \override LyricExtender #'color = #(rgb-color 0.000 0.000 0.000)     }   } }  \header { tagline = ""  title = "" } </LilyPond>   <EasyChant>Kik az Úr házában álltok, a mi Istenünk házának udvaraiban. r0</EasyChant>   <Clef>0</Clef>   <LyricsLocation>0</LyricsLocation>   <Key>8</Key>   <LyricsSpacing>0.8</LyricsSpacing>   <TimeOption>0</TimeOption>   <TimeText>Nincs (recitálás)</TimeText>   <StaffSizeOption>8</StaffSizeOption>   <StaffSizeText>18</StaffSizeText>   <LyricsSize>0</LyricsSize>   <LyricsFontName>Century Schoolbook L</LyricsFontName>   <LyricsFontBold>False</LyricsFontBold>   <LyricsFontItalic>False</LyricsFontItalic>   <LyricsFontColor>Color [A=255, R=0, G=0, B=0]</LyricsFontColor>   <LineSpacing>12</LineSpacing>   <LineThickness>10</LineThickness>   <Options>0,1,2,4,5,6,7,8,9,10</Options>   <Voices>1</Voices>   <pages>1</pages>   <page>1</page> </ec>]()