The xistercian package

Jonathan P. Spratte[*]

2021-12-20 v1.2

Contents

1 Documentation

The xistercian package provides Cistercian numerals for use in LaTeX. The name is chosen to be xistercian instead of cistercian because I've learned that a (currently) unpublished package by that name already exists and I don't want to be an evil name-stealer.

[*] jspratte@yahoo.de

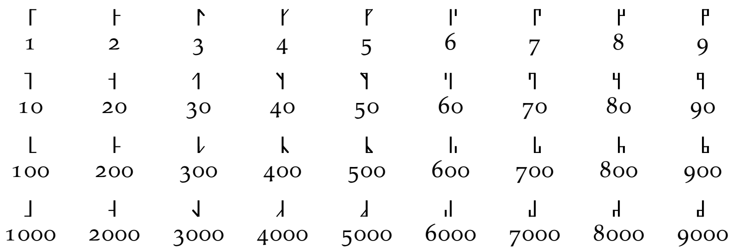


Figure 1: The different base glyphs in vertical mode

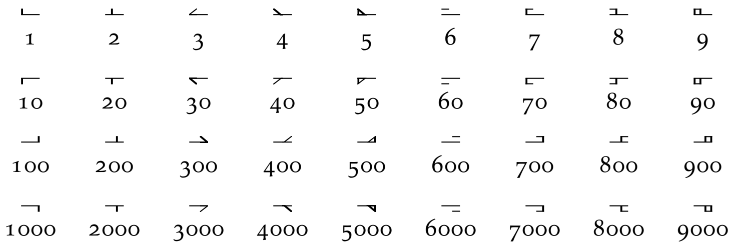


Figure 2: The different base glyphs in horizontal mode

1.1 Introduction

Cistercian numerals are a system to denote the numbers from 1 to 9999 with a single glyph. They use a stem (|) that is used as zero by this package, and add the digits to that using small tick marks, the place denoting the digits value. On the upper right means units (┌), upper left tens (┐), lower right hundreds (└), and lower left thousands (┘). Figure 1 gives an overview over the base glyphs.

In the medieval times it was quite common (according to Wikipedia more common than the vertical style) to use the Cistercian numerals horizontally instead. This package also supports that, in which case the numerals are rotated by 90 degrees counterclockwise, see figure 2.

Moreover some glyphs have a different shape in some manuscripts. The shapes in figures 1 and 2 are the most used ones nowadays (most likely because those are the ones Wikipedia shows vector graphics of). This package offers some (not all) alternative forms. Most notably some historic documents have swapped shapes of digits. To not further the confusion this package doesn't support swapping digit shapes.

All glyphs have the same bounding box, are drawn using the pgf package, and cached inside TEX box registers. Therefore the performance is quite well, at least well enough to actually consider using these numerals without a major performance hit. But it still slows down the compilation, especially if the glyphs have to be redrawn often. By default they will be redrawn if the font size changed since the last usage inside the current group scope, but you can specify which font feature changes should cause a redraw.

Since Cistercian numerals quite compactly cover a big range of natural numbers they might be used to compactly keep track of running numbers. For instance, to use Cistercian numerals as page numbers you can use:

```
\pagenumbering{cistercian}
```

or to use them as footnotes:

```
\renewcommand\thefootnote{\cistercian{footnotes}}
```

The package tries to play nice on hyperref if it is loaded, but bookmarks containing material that should've been a Cistercian numeral will have a leading `CISTER` in front of the numeric value to give unique strings compared to `\arabic` (to my knowledge there are no Unicode points for Cistercian numerals).

Negative numbers just keep a leading `-`, and numbers with an absolute number greater than 9999 are displayed with multiple Cistercian digits. For instance −12345678 is displayed as -. And in the case that four consecutive digits are zeros they get displayed as that: is 10000.

Full disclaimer: I'm neither a historian nor have I any authority on the matter. I'm just a bloke who was fascinated by this numeral system and wanted to provide LaTeX support for it. Most of my knowledge about this stems from Wikipedia and similar sources.

1.2 Macros

`\cistercian`

`\cistercian{⟨counter⟩}`

Prints the value of the LaTeX 2$_\varepsilon$ ⟨*counter*⟩ as a Cistercian numeral (similar to how `\roman` prints a ⟨*counter*⟩ as a Roman numeral). It can also be used as `\pagenumbering {cistercian}` to change the page numbering, *etc*.

`\cisterciannum`

`\cisterciannum{⟨integer⟩}`

Prints the ⟨*integer*⟩ as a Cistercian numeral. The ⟨*integer*⟩ has to be provided as a string containing only digits and optionally a single leading `-` (spaces are ignored). This can handle integers of arbitrary size.

`\cisterciannumE`

`\cisterciannum{⟨integer expression⟩}`

Evaluates the ⟨*integer expression*⟩ and prints the result as a Cistercian numeral.

`\cisterciansetup`

`\cisterciansetup{⟨key=value,...⟩}`

Can be used to locally change the options after the package was loaded. The glyphs of the Cistercian numerals aren't automatically updated when you change the setup.

`\cistercianredraw`
`\cistercianredrawlazy`

`\cistercianredraw`

When this is used the glyphs will be redrawn for the current group. The `lazy` variant will not immediately redraw the glyphs, but instead ensure that the next usage of a Cistercian numeral in the current scope will redraw the glyphs. The `lazy` variant could result in the glyphs being redrawn multiple times (if Cistercian numerals are used in different nested groups), while the normal variant might result in the glyphs being redrawn too eagerly.

`\cistercianstyle`

`\cistercianstyle⟨*⟩{⟨name⟩}{⟨key=value,...⟩}`

With this macro you can define a key called ⟨*name*⟩ that'll set the options in the ⟨*key=value*⟩ list if used inside `\cisterciansetup`. Only new names are allowed. If your style shouldn't take an argument (so is fixed) use the normal variant. If your style needs to take an argument use the variant with ⟨*⟩.

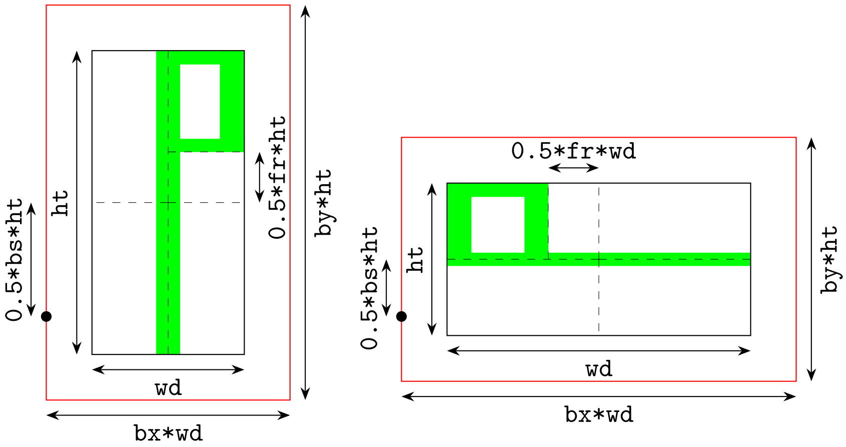


Figure 3: Measurements of a glyph. The red rectangle is the bounding box, the dot on the left shows the reference point placed on the surrounding baseline.

For example, the key `horizontal` is equivalently defined to and can be used as

```
\cistercianstyle{horizontal}{o=h,wd=0.775em,ht=1.13ex}
\cisterciansetup{horizontal}
```

And if you want to define a style that at the same time changes the stroke widths for vertical and horizontal strokes you could use

```
\cistercianstyle*{sh+v}{sh={#1},sv={#1}}
\cisterciansetup{sh+v=.12ex}
```

1.3 Options

The options described here are all usable as package options (except when explicitly stated otherwise). xistercian doesn't parse the global options provided to the documentclass. Additionally you can change the options (except for the `debug` option) using `\cisterciansetup`.

For the options `width` and `height`, and probably also the strokes, you should use dimensions using `ex` or `em` to get sizes depending on the current font size. A ⟨*dimension expression*⟩ is evaluated (using the setup specified in `font`) every time the font changed according to the `redraw` option's feature list (by default only when the size changed) since the last time they were drawn.

Initial values if present are printed on the right. A small graphic explaining most of the size related options is shown in figure 3.

`width`
`wd`

`width = {⟨dimension expression⟩}` `.5em`

Set the width of the Cistercian numeral digits.

`height`
`ht`

`height = {⟨dimension expression⟩}` `1.55ex`

Set the height of the Cistercian numeral digits.

`bound-x`
`bx`

`bound-x = {⟨float⟩}` `1.2`

Specifies the factor the bounding box in x-direction is bigger than the `width`. An empty ⟨*float*⟩ is considered the same as 1.

`bound-y`
`by`

`bound-y = {⟨float⟩}` *empty*

Specifies the factor the bounding box in y-direction is bigger than the `height`. An empty ⟨*float*⟩ is considered the same as 1.

`baseline`
`bs`

`baseline = {⟨float⟩}` *empty*

Sets the baseline of the symbols in multiples of the negative height. 1 (or empty) results in the symbol starting on the surrounding baseline, bigger values shift the symbols upwards, smaller values downwards.

`stroke-v`
`sv`

`stroke-v = {⟨dimension expression⟩}` `.18ex`

Sets the stroke thickness of vertical strokes.

`stroke-h`
`sh`

`stroke-h = {⟨dimension expression⟩}` `.1ex`

Sets the stroke thickness of horizontal strokes.

`stroke-du`
`sdu`

`stroke-du = {⟨dimension expression⟩}` `.1ex`

Sets the stroke thickness of diagonal strokes going from lower left to upper right.

`stroke-dd`
`sdd`

`stroke-dd = {⟨dimension expression⟩}` `.18ex`

Sets the stroke thickness of diagonal strokes going from upper left to lower right.

`stroke-s`
`ss`

`stroke-s = {⟨dimension expression⟩}` *empty*

Sets the stroke thickness of the zero stem. If this is set to an empty value the stroke thickness in vertical mode is that of `stroke-v` and in horizontal mode that of `stroke-h`.

`strokes`
`s`

`stroke = {⟨dimension expression⟩}`

Sets all stroke thicknesses at once.

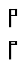
`fraction`
`fr`

`fraction = {⟨float⟩}` `/6`
`fraction = {⟨float⟩}/{⟨integer⟩}`

The Cistercian digits are drawn only in a fraction of the total bounding box of each symbol. The ones are drawn (with vertical Cistercians) in the upper right rectangle. The width of that rectangle is determined by `width`, and the upper border by `height`, and with this option you can set the lower border as a fraction of the `height`. You can either just give a factor as a ⟨*float*⟩, or just pass in a divisor as an ⟨*integer*⟩ (with a leading slash), or both. An empty ⟨*float*⟩ equals 1.

Consider the following example (result on the right):

```
{\cisterciansetup{fr=2/9,redraw}\cisterciannum{9}}\par
{\cisterciansetup{fr=0.6,redraw}\cisterciannum{9}}\par
```



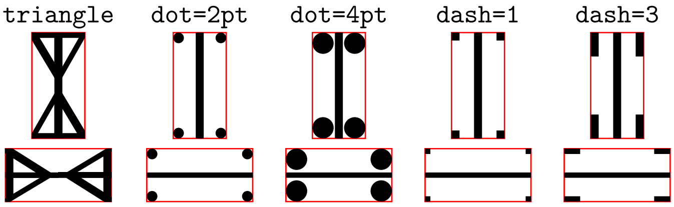


Figure 4: Alternate forms of the digit 5. The length of the dash=3 variant is too long in the normal setup of this document, as the difference between ⁝|⁝ and ⁞|⁞ is hard to notice. The tight bounding boxes using bx=1,by=1 are drawn red.

orientation
o

orientation = {⟨*choice*⟩} vertical

Sets the orientation of the Cistercian numerals. Choices are horizontal (or h for short) or vertical (or v for short). You'll most likely want to change the width and height of the symbols if you decide to change this.

horizontal

horizontal

Same as setting orientation = horizontal, width = 0.775em, height = 1.13ex.

vertical

vertical

Same as setting orientation = vertical, width = 0.5em, height = 1.55ex.

alternate-5
5

alternate-5 = {⟨*choice*⟩} triangle

There are alternative forms of the digit 5. While the most often used one nowadays seems to be the triangle form, this package also supports two other variants. Those are called dot and dash. While if you choose triangle you can't give an additional value, if your choice is dot or dash you can customise those using a second equals sign and some value. If you don't customise them they use their respective initial value (or last value if you changed the value at some point in time).

For dot you can also specify the radius as a dimension. The default is .09ex.

For dash you can specify the length of the dash in multiples of the used stroke thickness (which might differ for the horizontal and vertical symbols), an empty value is considered the same as 1. The default is *empty*. Please ensure that your readers can't confuse the results with the shape of 6.

A comparison of the three alternate forms is shown in figure 4.

The usage might look like any of the following:

```
\cisterciansetup
  {
    5=triangle, 5={triangle},
    5=dot,  5={dot},  5=dot=.1ex, 5={dot=.1ex},
    5=dash, 5={dash}, 5=dash=2,   5={dash=2}
  }
```

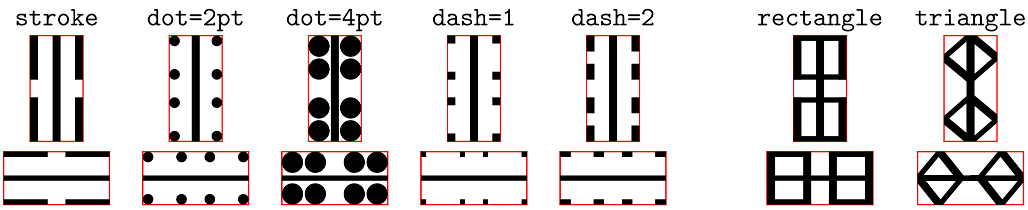


Figure 5: Alternate forms of the digits 6 (left) and 9 (right). The tight bounding boxes using `bx=1,by=1` are drawn red.

`alternate-6`
`6`

`\alternate-6 = {⟨choice⟩}` `stroke`

There are alternative forms of the digit 6. While today the usual form is `stroke`, there are also hiſtoric documents showing the 6 as two dots, which you can achieve using the `dot` choice, and to support squares or short dashes inſtead of round dots you can use the `dash` choice.

For `dot` you can also specify the radius as a dimension. The default is `.09ex`. And for `dash` you can also specify the length in multiples of the used ſtroke thickness, an empty value is considered the same as 1. The default is *empty*. The `stroke` has always the full length and doesn't accept an additional specification.

A comparison of the alternate forms is shown in figure 5.

`alternate-9`
`9`

`\alternate-9 = {⟨choice⟩}` `rectangle`

Another digit having alternative forms is the 9. For this two ⟨*choice*⟩s are available, `rectangle` and `triangle`. Neither of the two accept additional specifications.

A comparison of the alternate forms is shown in figure 5.

`font`
`f`

`font = {⟨font setup⟩}` `\normalfont`

Set the font which xiſtercian uses locally to evaluate the dimensions given to `width`, `height`, and `stroke` (only font switches work here, you can't use ſtuff like `\texttt`, inſtead use `\ttfamily`).

`redraw`
`r`

`redraw = {⟨feature-list⟩}` `size`
`redraw += {⟨feature-list⟩}`
`redraw`
`redraw!`

Selects which font features to consider when deciding to redraw the glyphs. If one of the features has changed (or the features to consider) the glyphs will be redrawn. This is especially handy if you change the `font` used while evaluating the given dimensions. The available features in the comma separated ⟨*feature-list*⟩ are `encoding`, `family`, `series`, `shape`, `size`, and `color`.

If you see that a Cistercian numeral doesn't have the correct colour of the surrounding text you need to add `color` to the font features (xistercian uses some hack to get the boxed glyphs to respect the surrounding colour, but that hack might fail in certain circumstances).

If you use the `+=` variant of this option the ⟨*feature-list*⟩ is added to the existing features, else they overwrite any existing configuration.

Without a value the key will issue `\cistercianredrawlazy`, and if you use the `!` variant it'll use `\cistercianredraw`. This usage is only available after the package was loaded.

`debug`

`debug`

Only usable as a package option. If this is used a bit of debugging information is printed in the log file and terminal, and some keys try to evaluate the given argument on the spot instead of lazily. This might help to locate faulty input.

1.4 Bold Cistercian Numerals

While the package doesn't directly support bold glyphs, it is possible to utilize the hooks provided by the NFSS to pick different stroke widths whenever the font is changed to `bfseries` and back to `mdseries`:

```
\AddToHook{bfseries}
  {%
    \cisterciansetup{sv=.27ex,sdd=.27ex,sh=.15ex,sdu=.15ex,wd=.625em}%
    \cistercianredrawlazy
  }
\AddToHook{mdseries}
  {%
    \cisterciansetup{sv=.18ex,sdd=.18ex,sh=.1ex,sdu=.1ex,wd=.5em}%
    \cistercianredrawlazy
  }
This results in \cisterciannum{2} different \enquote{series}.\par
\bfseries
This results in \cisterciannum{2} different \enquote{series}.
```

This results in ⊦ different “series”.
This results in ⊦ different “series”.

Of course, instead of doing `\cistercianredrawlazy` in the above code, we could as well do the following equivalent thing and let xistercian determine when to redraw:

```
\AddToHook{bfseries}
  {\cisterciansetup{sv=.27ex,sdd=.27ex,sh=.15ex,sdu=.15ex,wd=.625em}}
\AddToHook{mdseries}
  {\cisterciansetup{sv=.18ex,sdd=.18ex,sh=.1ex,sdu=.1ex,wd=.5em}}
\cisterciansetup{redraw+=series}
```

2 Implementation

First we say who we are and load the required packages

```
\ProvidesPackage{xistercian}[2021-12-20 v1.2 Cistercian numerals in LaTeX]
\RequirePackage{pgf,expkv-opt}
```

2.1 Variables

There are variables stored as macros to be evaluated later, as well as register types for faster access during the glyph drawing.

```
\newcommand*\xister@th{.1ex}
\newcommand*\xister@tv{.18ex}
\newcommand*\xister@tdu{.1ex}
\newcommand*\xister@tdd{.18ex}
\newcommand*\xister@ts{}
\newcommand*\xister@x{.5em}
\newcommand*\xister@y{1.55ex}
\newcommand*\xister@bx{1.2}
\newcommand*\xister@by{}
\newcommand*\xister@bs{}
\newcommand*\xister@font{\normalfont}
\newcommand*\xister@five@dot{.09ex}
\newcommand*\xister@five@dash{}
\newcommand*\xister@six@dot{.09ex}
\newcommand*\xister@six@dash{}
\newcommand*\xister@share@div{/6}
\newcommand*\xister@share@mul{}
\newif\ifxister@dbg@
\newif\ifxister@five@triangle@\xister@five@triangle@true
\newif\ifxister@five@dot@
\newif\ifxister@six@dot@
\newif\ifxister@six@stroke@\xister@six@stroke@true
\newif\ifxister@nine@triangle@
\newdimen\xister@X
\newdimen\xister@Y
\newdimen\xister@XY@share
\newdimen\xister@Th
\newdimen\xister@Tv
\newdimen\xister@Tdu
\newdimen\xister@Tdd
\newdimen\xister@Ts
\newdimen\xister@TMPA
\newdimen\xister@TMPB
\newdimen\xister@Five@dot
\newdimen\xister@Six@dot
```

We also need a bunch of box registers (37 of them to be precise). For the loop we borrow an expl3 function.

```
\ExplSyntaxOn
\int_step_inline:nn {9}
  {
    \expandafter\newsavebox\csname xister@digitbox@#1\endcsname
    \expandafter\newsavebox\csname xister@digitbox@#10\endcsname
    \expandafter\newsavebox\csname xister@digitbox@#100\endcsname
    \expandafter\newsavebox\csname xister@digitbox@#1000\endcsname
  }
\ExplSyntaxOff
\expandafter\newsavebox\csname xister@digitbox@0\endcsname
```

\xister@dbg@dim
\xister@dbg@float

These are initially just `\@gobble`, but the debug option might change things.

```
\let\xister@dbg@dim\@gobble
\let\xister@dbg@div\@gobble
\let\xister@dbg@float\@gobble
```

(*End definition for* `\xister@dbg@dim` *and* `\xister@dbg@float`.)

2.2 Options

Some macros are required to parse options.

\xister@if@slash

We need to test whether a slash is inside an argument to parse some user option. Since this isn't needed inside an inner loop we don't create an optimised version and only borrow a generic expl3 function.

```
\ExplSyntaxOn
\cs_new_protected:Npn \xister@if@slash #1 { \tl_if_in:nnTF {#1} { / } }
\ExplSyntaxOff
```

(*End definition for* `\xister@if@slash`.)

\xister@fraction

```
\protected\long\def\xister@fraction#1/#2\xister@stop
  {%
    \edef\xister@share@mul{\unexpanded{#1}}%
    \xister@dbg@float{#1}%
    \edef\xister@share@div{/\unexpanded{#2}}%
    \xister@dbg@div{#2}%
  }
```

(*End definition for* `\xister@fraction`.)

\xister@build
\xister@build@h
\xister@build@horizontal
\xister@build@v
\xister@build@vertical

First we set the length registers to the correct size, then we draw the glyphs batch wise. We have to flip the x-direction for the tenths and thousands, and the y-direction for the hundreds and thousands. The macro `\xister@sgn` is used to set the baseline in the correct direction. We locally disable `\pgfsetcolor` to allow building colour-unsafe boxes. This only works as long as luacolor (or something equivalent) isn't used, so consider this a crude hack.

```
\protected\def\xister@build@vertical
  {%
    \let\xister@pgfsetcolor\pgfsetcolor
    \let\pgfsetcolor\@gobble
    \xister@setlengths
    \def\xister@sgn{-}%
    \xister@drawzero@vertical
    \xister@drawdigits@vertical{}{}{}%
    \xister@X=-\xister@X
    \xister@Y=-\xister@Y
    \def\xister@sgn{+}%
    \xister@drawdigits@vertical{000}{-}{-}%
    \xister@TMPA=\xister@Tdd
    \xister@Tdd=\xister@Tdu
    \xister@Tdu=\xister@TMPA
    \xister@X=-\xister@X
    \xister@drawdigits@vertical{00}{-}{}%
    \xister@X=-\xister@X
    \xister@Y=-\xister@Y
    \def\xister@sgn{-}%
    \xister@drawdigits@vertical{0}{}{-}%
    \let\pgfsetcolor\xister@pgfsetcolor
  }
\protected\def\xister@build@horizontal
  {%
    \let\xister@pgfsetcolor\pgfsetcolor
    \let\pgfsetcolor\@gobble
    \xister@setlengths
    \def\xister@sgn{-}%
    \xister@drawzero@horizontal
    \xister@drawdigits@horizontal{}{}{}%
    \xister@X=-\xister@X
    \xister@Y=-\xister@Y
    \def\xister@sgn{+}%
    \xister@drawdigits@horizontal{000}{-}{-}%
    \xister@TMPA=\xister@Tdd
    \xister@Tdd=\xister@Tdu
    \xister@Tdu=\xister@TMPA
    \xister@X=-\xister@X
    \xister@drawdigits@horizontal{0}{-}{}%
    \xister@X=-\xister@X
    \xister@Y=-\xister@Y
    \def\xister@sgn{-}%
    \xister@drawdigits@horizontal{00}{}{-}%
    \let\pgfsetcolor\xister@pgfsetcolor
  }
\let\xister@build\xister@build@vertical
\let\xister@build@v\xister@build@vertical
\let\xister@build@h\xister@build@horizontal
```

(*End definition for `\xister@build` and others.*)

\xister@fontfeatures
\xister@add@fontfeature

This macro should store the different font features which should be kept track of. It will be redefined programmatically but the default is quite easy, just track the font size.

```
\protected\long\def\xister@add@fontfeature#1%
  {%
    \begingroup\expandafter\expandafter\expandafter\endgroup
    \expandafter\ifx
        \csname xister@add@fontfeature@\detokenize{#1}\endcsname
        \relax
      \PackageError{xistercian}{Unsupported font feature ‘\detokenize{#1}’.}{}%
    \else
      \csname xister@add@fontfeature@\detokenize{#1}\endcsname
    \fi
  }
\def\xister@fontfeatures{\unexpanded\expandafter{\f@size}/}
```

We know that the colour hack breaks if luacolor is loaded, so if that is found to be the case we add the color feature to the liſt of things we need to track. But we only want to add it if it's not yet in the liſt, so we run a check we borrow from expl3.

```
\ExplSyntaxOn
\AddToHook{begindocument/before}
  {
    \@ifpackageloaded { luacolor }
      {
        \exp_args:No \str_if_in:nnF \xister@fontfeatures { color }
          { \xister@add@fontfeature{color} }
      }
      {}
  }
\ExplSyntaxOff
```

(*End definition for* `\xister@fontfeatures` *and* `\xister@add@fontfeature`.)

\xister@add@fontfeature@encoding
\xister@add@fontfeature@family
\xister@add@fontfeature@series
\xister@add@fontfeature@shape
\xister@add@fontfeature@size

These macros all juſt add a specific macro to the liſt of things contained in `\xister@fontfeatures`, which determines what to consider deciding to redraw.

```
\def\xister@tmp#1#2%
  {%
    \protected\expandafter\def\csname xister@add@fontfeature@#2\endcsname
      {%
        \edef\xister@fontfeatures
          {%
            \unexpanded\expandafter
              {\xister@fontfeatures\unexpanded\expandafter{#1}}/%
          }%
      }%
  }
\expandafter\xister@tmp\csname\@backslashchar color@.\endcsname{color}
\xister@tmp\f@encoding{encoding}
\xister@tmp\f@family  {family}
\xister@tmp\f@series  {series}
\xister@tmp\f@shape   {shape}
\xister@tmp\f@size    {size}
```

(*End definition for* `\xister@add@fontfeature@encoding` *and others.*)

Now we define the keys

```
\ekvifdefinedset{xister}
  {\PackageError{xistercian}{keyval conflict detected. Aborting}{}\endinput}{}
\def\xister@tmp#1#2#3%
  {%
    \protected\long\ekvdef{xister}{#1}{#3}%
    \ekvletkv{xister}{#2}{xister}{#1}%
  }
\xister@tmp{wd}{width}{\edef\xister@x{\unexpanded{#1}}\xister@dbg@dim{#1}}
\xister@tmp{ht}{height}{\edef\xister@y{\unexpanded{#1}}\xister@dbg@dim{#1}}
\xister@tmp{bx}{bound-x}{\edef\xister@bx{\unexpanded{#1}}\xister@dbg@float{#1}}
\xister@tmp{by}{bound-y}{\edef\xister@by{\unexpanded{#1}}\xister@dbg@float{#1}}
\xister@tmp{bs}{baseline}{\edef\xister@bs{\unexpanded{#1}}\xister@dbg@float{#1}}
\xister@tmp{sv}{stroke-v}{\edef\xister@tv{\unexpanded{#1}}\xister@dbg@dim{#1}}
\xister@tmp{sh}{stroke-h}{\edef\xister@th{\unexpanded{#1}}\xister@dbg@dim{#1}}
\xister@tmp{sdu}{stroke-du}
```

```
  {\edef\xister@tdu{\unexpanded{#1}}\xister@dbg@dim{#1}}
\xister@tmp{sdd}{stroke-dd}
  {\edef\xister@tdd{\unexpanded{#1}}\xister@dbg@dim{#1}}
\xister@tmp{ss}{stroke-s}
  {%
    \edef\xister@ts{\unexpanded{#1}}%
    \unless\ifx\xister@ts\@empty\xister@dbg@dim{#1}\fi
  }
\xister@tmp{s}{strokes}
  {%
    \edef\xister@tv{\unexpanded{#1}}%
    \let\xister@th\xister@tv
    \let\xister@tdu\xister@tv
    \let\xister@tdd\xister@tv
    \let\xister@ts\xister@tv
    \xister@dbg@dim{#1}%
  }
\xister@tmp{f}{font}{\edef\xister@font{\unexpanded{#1}}}
\xister@tmp{o}{orientation}
  {%
    \begingroup\expandafter\expandafter\expandafter\endgroup
    \expandafter\ifx\csname xister@build@#1\endcsname\relax
      \PackageError{xister}%
        {Unaccepted orientation. Choices are horizontal/h or vertical/v}{}%
    \else
      \expandafter\let\expandafter\xister@build\csname xister@build@#1\endcsname
    \fi
  }
\xister@tmp{fr}{fraction}
  {%
    \xister@if@slash{#1}%
      {\xister@fraction#1\xister@stop}%
      {%
        \let\xister@share@div\@empty
        \edef\xister@share@mul{\unexpanded{#1}}%
        \xister@dbg@float{#1}%
      }%
  }
\xister@tmp{r}{redraw}
  {\let\xister@fontfeatures\@empty\ekvcsvloop\xister@add@fontfeature{#1}}
\xister@tmp{r+}{redraw+}{\ekvcsvloop\xister@add@fontfeature{#1}}
\ekvletkv{xister}{r +}{xister}{r+}
\ekvletkv{xister}{redraw +}{xister}{redraw+}
\protected\long\ekvdef{xister/5}{dot}
  {%
    \xister@five@triangle@false
    \xister@five@dot@true
    \edef\xister@five@dot{\unexpanded{#1}}%
    \xister@dbg@dim{#1}%
  }
\protected\long\ekvdef{xister/5}{dash}
  {%
    \xister@five@triangle@false
    \xister@five@dot@false
    \edef\xister@five@dash{\unexpanded{#1}}%
    \xister@dbg@float{#1}%
  }
\protected\ekvdefNoVal{xister/5}{triangle}{\xister@five@triangle@true}
\protected\ekvdefNoVal{xister/5}{dot}
  {%
    \xister@five@triangle@false
    \xister@five@dot@true
  }
\protected\ekvdefNoVal{xister/5}{dash}
  {%
    \xister@five@triangle@false
    \xister@five@dot@false
  }
\protected\ekvsetdef\xister@five@set{xister/5}
\ekvlet{xister}{5}\xister@five@set
\ekvlet{xister}{alternate-5}\xister@five@set
\let\xister@five@set\xister@undefined
\protected\long\ekvdef{xister/6}{dot}
  {%
    \xister@six@stroke@false
    \xister@six@dot@true
    \edef\xister@six@dot{\unexpanded{#1}}%
    \xister@dbg@dim{#1}%
  }
\protected\long\ekvdef{xister/6}{dash}
  {%
    \xister@six@stroke@false
    \xister@six@dot@false
    \edef\xister@six@dash{\unexpanded{#1}}%
    \xister@dbg@float{#1}%
  }
\protected\ekvdefNoVal{xister/6}{stroke}{\xister@six@stroke@true}
\protected\ekvdefNoVal{xister/6}{dot}
  {%
    \xister@six@stroke@false
    \xister@six@dot@true
  }
\protected\ekvdefNoVal{xister/6}{dash}
  {%
    \xister@six@stroke@false
    \xister@six@dot@false
  }
\protected\ekvsetdef\xister@six@set{xister/6}
\ekvlet{xister}{6}\xister@six@set
\ekvlet{xister}{alternate-6}\xister@six@set
\let\xister@six@set\xister@undefined
\protected\ekvdefNoVal{xister/9}{rectangle}{\xister@nine@triangle@false}
\protected\ekvdefNoVal{xister/9}{triangle}{\xister@nine@triangle@true}
\protected\ekvsetdef\xister@nine@set{xister/9}
\ekvlet{xister}{9}\xister@nine@set
\ekvlet{xister}{alternate-9}\xister@nine@set
\protected\ekvdefNoVal{xister}{horizontal}
  {%
```

```
    \let\xister@build\xister@build@horizontal
    \def\xister@x{.775em}%
    \def\xister@y{1.13ex}%
  }
\protected\ekvdefNoVal{xister}{vertical}
  {%
    \let\xister@build\xister@build@vertical
    \def\xister@x{.5em}%
    \def\xister@y{1.55ex}%
  }
```

The debug option redefines the debug macros to the correct definition.

```
\protected\ekvdefNoVal{xister}{debug}
  {%
    \xister@dbg@true
    \protected\long\def\xister@dbg@dim##1%
      {%
        \typeout
          {%
            Package xistercian Debug: Trying to use dimension
            ‘\unexpanded{##1}’.%
          }%
        \xister@TMPA=\dimexpr##1\relax
        \typeout{Package xistercian Debug: Done.}%
      }%
    \protected\long\def\xister@dbg@float##1%
      {%
        \typeout
          {Package xistercian Debug: Trying to use float ‘\unexpanded{##1}’.}%
        \xister@TMPA=##1\z@
        \typeout{Package xistercian Debug: Done.}%
      }%
    \protected\long\def\xister@dbg@div##1%
      {%
        \typeout
          {Package xistercian Debug: Trying to use divisor ‘\unexpanded{##1}’.}%
        \xister@TMPA=\dimexpr\z@/##1\relax
        \typeout{Package xistercian Debug: Done.}%
      }%
  }
```

and parse the package options (and since the used names `width`, `height`, and `debug` are quite frequent we don’t look at the global options).

```
\ekvoProcessLocalOptions{xister}
```

After the package was loaded we add the two `redraw` variants without a value.

```
\protected\ekvdefNoVal{xister}{r!}{\cistercianredraw}
\protected\ekvdefNoVal{xister}{r} {\cistercianredrawlazy}
\ekvletkvNoVal{xister}{redraw} {xister}{r}
\ekvletkvNoVal{xister}{redraw!}{xister}{r!}
```

The debug option is only available as a package option, we let it throw an error:

```
\ekvdefNoVal{xister}{debug}
  {\PackageError{xister}{‘debug’ is only available as a load time option}{}}
```

2.3 User macros

\cisterciansetup

Just as a simple way to pick options later on.

```
\protected\ekvsetdef\cisterciansetup{xister}
```

(*End definition for* \cisterciansetup*. This function is documented on page* ↾*.*)

\cistercianstyle
\xister@style

We allow users to define a style they can call with a single key name inside \cisterciansetup. To achieve that we just define a NoVal key that'll call a nested \cisterciansetup, or a normal key if their style needs to take a value.

```
\protected\def\cistercianstyle
  {%
    \@ifstar
      {\xister@style\ekvifdefined{\long\ekvdef}}%
      {\xister@style\ekvifdefinedNoVal\ekvdefNoVal}%
  }
\protected\long\def\xister@style#1#2#3#4%
  {%
    #1{xister}{#3}%
      {\PackageError{xistercian}{Key ‘#3’ already defined}{}}%
      {\protected#2{xister}{#3}{\cisterciansetup{#4}}}%
  }
```

(*End definition for* \cistercianstyle *and* \xister@style*. These functions are documented on page* ↾*.*)

\cistercian
\@cistercian

The macro is equivalent to other LaTeX counter formatting macros, so we let it build the counter name and forward that to the canonically named auxiliary macro \@cistercian.

```
\newcommand\cistercian[1]{\expandafter\@cistercian\csname c@#1\endcsname}
```

The internal just gets the current value of a counter and forwards it.

```
\newcommand\@cistercian[1]{\expandafter\cisterciannum\expandafter{\the#1}}
```

(*End definition for* \cistercian *and* \@cistercian*. These functions are documented on page* ↾*.*)

\cisterciannum
\cisterciannum@pdf

```
\edef\cisterciannum
  {%
    \unexpanded{\xister@texorpdf}%
    \unexpanded\expandafter
      {\csname cisterciannum \endcsname\cisterciannum@pdf}%
  }
\protected\expandafter\def\csname cisterciannum \endcsname#1{\xister@a#1@}
\def\cisterciannum@pdf#1{CISTER#1}
```

(*End definition for* \cisterciannum *and* \cisterciannum@pdf*. These functions are documented on page* ↾*.*)

\cisterciannumE

```
\newcommand\cisterciannumE[1]
  {\expandafter\cisterciannum\expandafter{\the\numexpr#1\relax}}
```

(*End definition for* \cisterciannumE*. This function is documented on page* ↾*.*)

\cistercianredraw
\cistercianredrawlazy

```
\protected\def\cistercianredraw
  {%
    \def\xister@last@font{\xister@last@font}%
    \xister@ensure@current
  }
\protected\def\cistercianredrawlazy{\def\xister@last@font{\xister@last@font}}
```

(*End definition for* `\cistercianredraw` *and* `\cistercianredrawlazy`*. These functions are documented on page* ᒥ.)

2.4 Parsing

2.4.1 Small Auxiliaries

\xister@gobbletoat
\xister@gobbletozero
\xister@gobblecopy
\xister@done

These are just small functions gobbling some tokens.

```
\def\xister@gobbletoat#1@{}
\def\xister@gobbletozero#10{}
\def\xister@gobblecopy0\rlap#1{}
\def\xister@done#1\xister@symbols{}
```

(*End definition for* `\xister@gobbletoat` *and others.*)

\xister@texorpdf

```
\let\xister@texorpdf\@firstoftwo
\AddToHook{package/hyperref/after}
  {%
    \ifdefined\texorpdfstring
      \def\xister@texorpdf{\texorpdfstring}%
    \fi
  }
\AddToHook{begindocument/before}
  {%
    \ifdefined\texorpdfstring
      \def\xister@texorpdf{\texorpdfstring}%
    \fi
  }
```

(*End definition for* `\xister@texorpdf`*.*)

2.4.2 Input Parsing

\xister@a
\xister@b
\xister@c
\xister@d

First we have to make sure that we're in horizontal mode, else the `\rlap`s used to insert the glyphs will create havoc. We also need to ensure that the boxes contain the correct glyphs according to the current size and colour. After that we check whether we have to step over a minus sign and go to the next step.

```
\def\xister@a#1%
  {%
    \leavevmode
    \xister@ensure@current
    \ifx-#1-\expandafter\xister@b
    \else\expandafter\xister@b\expandafter#1\fi
  }
```

Next we check whether the remaining number is empty. If that is the case we use \the\numexpr\relax to throw a missing number error and produce just a zero stem. Else we do a dry run over the input number to see how many leading zeros we must pad to get blocks of four. The two blocks of @s serve as end markers since each step of both \xister@c grabs eight and each of \xister@symbols will grab four digits. The \xister@c is used to put enough zeros in front of our number such that it consists of blocks of four digits.

```
\def\xister@b#1@%
  {%
    \if\relax\detokenize{#1}\relax
      \expandafter\xister@gobbletozero\the\numexpr\relax
      \xister@symbols0000@@@@%
      \expandafter\@gobbletwo
    \fi
    \@firstofone{\xister@c#1@@@@@@@@\relax#1@@@@}%
  }
\def\xister@c#1#2#3#4#5#6#7#8%
  {%
    \xister@gobbletoat
      #1\xister@d{}%
      #2\xister@d{000}%
      #3\xister@d{00}%
      #4\xister@d0%
      #5\xister@d{}%
      #6\xister@d{000}%
      #7\xister@d{00}%
      #8\xister@d0%
      @\xister@c
  }
\def\xister@d#1#2@\xister@c#3\relax{\xister@symbols#1}
```

(*End definition for* \xister@a *and others.*)

\xister@symbols

When this loop is done all arguments will be @, but we only check the first here. We generate the symbols by overlaying up to five boxes, one for each decade plus the zero stem. Each digit might be skipped if it is 0. Next grab the next four digits.

```
\def\xister@symbols#1%
  {%
    \protected\def\xister@symbols##1##2##3##4%
      {%
        \xister@gobbletoat##1\xister@done @%
        \xister@gobbletozero##1\xister@gobblecopy0%
          \rlap{\copy\csname xister@digitbox@##1000\endcsname}%
        \xister@gobbletozero##2\xister@gobblecopy0%
          \rlap{\copy\csname xister@digitbox@##200\endcsname}%
        \xister@gobbletozero##3\xister@gobblecopy0%
          \rlap{\copy\csname xister@digitbox@##30\endcsname}%
        \xister@gobbletozero##4\xister@gobblecopy0%
          \rlap{\copy\csname xister@digitbox@##4\endcsname}%
        \copy#1%
        \xister@symbols
      }%
  }
\expandafter\xister@symbols\csname xister@digitbox@0\endcsname
```

(*End definition for* `\xister@symbols`.)

2.5 Drawing the Digits

`\xister@clip@v`
`\xister@clip@h`

Some of the digit glyphs need a bit of clipping to look nice. Let's define a small auxiliary to select a clipping region.

```
\newcommand*\xister@clip@v[1]
  {%
    \pgfpathrectanglecorners
      {\pgfpoint{\dimexpr\ifdim\xister@X<\z@-\fi\xister@Ts\relax}{#1}}%
      {\pgfpoint\xister@X\xister@Y}%
    \pgfusepath{clip}%
  }
\newcommand*\xister@clip@h[1]
  {%
    \pgfpathrectanglecorners
      {\pgfpoint{#1}{-\dimexpr\ifdim\xister@Y<\z@-\fi\xister@Ts\relax}}%
      {\pgfpoint{-\xister@X}\xister@Y}%
    \pgfusepath{clip}%
  }
```

(*End definition for* `\xister@clip@v` *and* `\xister@clip@h`.)

`\xister@drawdigits@vertical`
`\xister@drawdigits@horizontal`

The digit glyphs. Since each numeral is build from the zero stem and some appendix to it this only specifies the form of the appendices. The zero stem is defined a bit down the road. #1 will determine the decade, and #2 will be either - or empty.

```
\newcommand\xister@drawdigits@vertical[3]
  {%
    \xister@XY@share=\dimexpr\xister@share@mul\xister@Y\xister@share@div\relax
    \xister@drawdigit{1#1}%
      {%
        \pgfsetlinewidth{2\xister@Th}%
        \xister@TMPA=\dimexpr\xister@Y\xister@sgn\xister@Th\relax
        \pgfpathmoveto{\pgfpoint\z@\xister@TMPA}%
        \pgfpathlineto{\pgfpoint\xister@X\xister@TMPA}%
      }%
    \xister@drawdigit{2#1}%
      {%
        \pgfsetlinewidth{2\xister@Th}%
        \xister@TMPA=\dimexpr\xister@XY@share+#2\xister@Th\relax
        \pgfpathmoveto{\pgfpoint\z@\xister@TMPA}%
        \pgfpathlineto{\pgfpoint\xister@X\xister@TMPA}%
      }%
    \xister@drawdigit{3#1}%
      {%
        \xister@TMPB=#3\dimexpr\xister@Ts-\xister@Tdd\relax
        \xister@clip@v\xister@XY@share
        \pgfsetlinewidth{2\xister@Tdd}%
        \pgfpathmoveto{\pgfpoint\xister@TMPB\xister@XY@share}%
        \pgfpathlineto{\pgfpoint\xister@TMPB\xister@Y}%
        \pgfpathlineto{\pgfpoint\xister@X\xister@XY@share}%
      }%
    \xister@drawdigit{4#1}%
      {%
        \xister@clip@v\z@
        \xister@TMPB=#3\dimexpr\xister@Ts-\xister@Tdu\relax
        \pgfsetlinewidth{2\xister@Tdu}%
        \pgfpathmoveto{\pgfpoint\xister@TMPB\xister@Y}%
        \pgfpathlineto{\pgfpoint\xister@TMPB\xister@XY@share}%
        \pgfpathlineto{\pgfpoint\xister@X\xister@Y}%
      }%
    \xister@drawdigit{5#1}%
      {%
        \ifxister@five@triangle@
          \xister@clip@v\z@
          \xister@TMPB=%
            #3\dimexpr\xister@Ts-\xister@Tdu\relax
          \pgfsetlinewidth{2\xister@Tdu}%
          \pgfpathmoveto{\pgfpoint\xister@TMPB\xister@Y}%
          \pgfpathlineto{\pgfpoint\xister@TMPB\xister@XY@share}%
          \pgfpathlineto{\pgfpoint\xister@X\xister@Y}%
          \pgfusepath{stroke,clip}%
          \pgfsetlinewidth{2\xister@Th}%
          \xister@TMPA=\dimexpr\xister@Y\xister@sgn\xister@Th\relax
          \pgfpathmoveto{\pgfpoint\z@\xister@TMPA}%
          \pgfpathlineto{\pgfpoint\xister@X\xister@TMPA}%
        \else\ifxister@five@dot@
          \pgfpathcircle
            {\pgfpoint{\dimexpr\xister@X-#3\xister@Five@dot\relax}%
                      {\dimexpr\xister@Y-#2\xister@Five@dot\relax}}%
            {\xister@Five@dot}%
          \pgfusepath{fill}%
        \else
          \xister@TMPB=\dimexpr\xister@X-#3\xister@Tv\relax
          \pgfsetlinewidth{2\xister@Tv}%
          \pgfpathmoveto{\pgfpoint\xister@TMPB\xister@Y}%
          \pgfpathlineto
            {%
              \pgfpoint
                \xister@TMPB
                {\dimexpr\xister@Y-#2\xister@five@dash\xister@Tv*2\relax}%
            }%
        \fi\fi
      }%
    \xister@drawdigit{6#1}%
      {%
        \pgfsetlinewidth{2\xister@Tv}%
        \xister@TMPB=\dimexpr\xister@X-#3\xister@Tv\relax
        \pgfpathmoveto{\pgfpoint\xister@TMPB\xister@Y}%
        \pgfpathlineto{\pgfpoint\xister@TMPB\xister@XY@share}%
      }%
    \xister@superimpose{7#1}%
      {%
        \rlap{\copy\csname xister@digitbox@1#1\endcsname}%
        \copy\csname xister@digitbox@6#1\endcsname
      }%
    \xister@superimpose{8#1}%
```

```
      {%
        \rlap{\copy\csname xister@digitbox@2#1\endcsname}%
        \copy\csname xister@digitbox@6#1\endcsname
      }%
    \ifxister@nine@triangle@
      \xister@drawdigit{9#1}%
        {%
          \xister@TMPA=.5\dimexpr\xister@XY@share+\xister@Y\relax
          \xister@TMPB=#3\dimexpr\xister@Ts-\xister@Tdd\relax
          \xister@clip@v\z@
          \pgfsetlinewidth{2\xister@Tdd}%
          \pgfpathmoveto{\pgfpoint\xister@TMPB\xister@XY@share}%
          \pgfpathlineto{\pgfpoint\xister@TMPB\xister@Y}%
          \pgfpathlineto{\pgfpoint\xister@X\xister@TMPA}%
          \pgfusepath{stroke}%
          \xister@TMPB=#3\dimexpr\xister@Ts-\xister@Tdu\relax
          \pgfsetlinewidth{2\xister@Tdu}%
          \pgfpathmoveto{\pgfpoint\xister@TMPB\xister@Y}%
          \pgfpathlineto{\pgfpoint\xister@TMPB\xister@XY@share}%
          \pgfpathlineto{\pgfpoint\xister@X\xister@TMPA}%
        }%
    \else
      \xister@superimpose{9#1}%
        {%
          \rlap{\copy\csname xister@digitbox@1#1\endcsname}%
          \rlap{\copy\csname xister@digitbox@2#1\endcsname}%
          \copy\csname xister@digitbox@6#1\endcsname
        }%
    \fi
    \unless\ifxister@six@stroke@
      \xister@drawdigit{6#1}%
        {%
          \ifxister@six@dot@
            \xister@TMPA=\dimexpr\xister@X-#3\xister@Six@dot\relax
            \pgfpathcircle
              {\pgfpoint\xister@TMPA
                        {\dimexpr\xister@Y-#2\xister@Six@dot\relax}}%
              {\xister@Six@dot}%
            \pgfpathcircle
              {\pgfpoint\xister@TMPA
                        {\dimexpr\xister@XY@share+#2\xister@Six@dot\relax}}%
              {\xister@Six@dot}%
            \pgfusepath{fill}%
          \else
            \xister@TMPB=\dimexpr\xister@X-#3\xister@Tv\relax
            \xister@TMPA=2\dimexpr#2\xister@six@dash\xister@Tv\relax
            \pgfsetlinewidth{2\xister@Tv}%
            \pgfpathmoveto{\pgfpoint\xister@TMPB\xister@Y}%
            \pgfpathlineto
              {\pgfpoint\xister@TMPB
                        {\dimexpr\xister@Y-\xister@TMPA\relax}}%
            \pgfpathmoveto{\pgfpoint\xister@TMPB\xister@XY@share}%
            \pgfpathlineto
              {\pgfpoint\xister@TMPB
```

```
                        {\dimexpr\xister@XY@share+\xister@TMPA\relax}}%
          \fi
        }%
    \fi
  }
\newcommand\xister@drawdigits@horizontal[3]
  {%
    \xister@XY@share=-\dimexpr\xister@share@mul\xister@X\xister@share@div\relax
    \xister@drawdigit{1#1}%
      {%
        \pgfsetlinewidth{2\xister@Tv}%
        \xister@TMPA=\dimexpr#3\xister@Tv-\xister@X\relax
        \pgfpathmoveto{\pgfpoint\xister@TMPA\z@}%
        \pgfpathlineto{\pgfpoint\xister@TMPA\xister@Y}%
      }%
    \xister@drawdigit{2#1}%
      {%
        \pgfsetlinewidth{2\xister@Tv}%
        \xister@TMPA=\dimexpr\xister@XY@share-#3\xister@Tv\relax
        \pgfpathmoveto{\pgfpoint\xister@TMPA\z@}%
        \pgfpathlineto{\pgfpoint\xister@TMPA\xister@Y}%
      }%
    \xister@drawdigit{3#1}%
      {%
        \xister@TMPB=#2\dimexpr\xister@Ts-\xister@Tdu\relax
        \xister@clip@h\xister@XY@share
        \pgfsetlinewidth{2\xister@Tdu}%
        \pgfpathmoveto{\pgfpoint\xister@XY@share\xister@TMPB}%
        \pgfpathlineto{\pgfpoint{-\xister@X}\xister@TMPB}%
        \pgfpathlineto{\pgfpoint\xister@XY@share\xister@Y}%
      }%
    \xister@drawdigit{4#1}%
      {%
        \xister@clip@h\z@
        \xister@TMPB=#2\dimexpr\xister@Ts-\xister@Tdd\relax
        \pgfsetlinewidth{2\xister@Tdd}%
        \pgfpathmoveto{\pgfpoint\xister@X\xister@TMPB}%
        \pgfpathlineto{\pgfpoint\xister@XY@share\xister@TMPB}%
        \pgfpathlineto{\pgfpoint{-\xister@X}\xister@Y}%
      }%
    \xister@drawdigit{5#1}%
      {%
        \ifxister@five@triangle@
          \xister@clip@h\z@
          \xister@TMPB=#2\dimexpr\xister@Ts-\xister@Tdd\relax
          \pgfsetlinewidth{2\xister@Tdd}%
          \pgfpathmoveto{\pgfpoint{-\xister@X}\xister@TMPB}%
          \pgfpathlineto{\pgfpoint\xister@XY@share\xister@TMPB}%
          \pgfpathlineto{\pgfpoint{-\xister@X}\xister@Y}%
          \pgfusepath{stroke,clip}%
          \pgfsetlinewidth{2\xister@Tv}%
          \xister@TMPA=\dimexpr#3\xister@Tv-\xister@X\relax
          \pgfpathmoveto{\pgfpoint\xister@TMPA\z@}%
          \pgfpathlineto{\pgfpoint\xister@TMPA\xister@Y}%
        \else\ifxister@five@dot@
          \pgfpathcircle
            {\pgfpoint{\dimexpr#3\xister@Five@dot-\xister@X\relax}%
                      {\dimexpr\xister@Y-#2\xister@Five@dot\relax}}%
            {\xister@Five@dot}%
          \pgfusepath{fill}%
        \else
          \pgfsetlinewidth{2\xister@Th}%
          \xister@TMPB=\dimexpr\xister@Y-#2\xister@Th\relax
          \pgfpathmoveto{\pgfpoint{-\xister@X}\xister@TMPB}%
          \pgfpathlineto
            {%
              \pgfpoint
                {\dimexpr#3\xister@five@dash\xister@Th*2-\xister@X\relax}%
                \xister@TMPB
            }%
        \fi\fi
      }%
    \xister@drawdigit{6#1}%
      {%
        \pgfsetlinewidth{2\xister@Th}%
        \xister@TMPB=\dimexpr\xister@Y-#2\xister@Th\relax
        \pgfpathmoveto{\pgfpoint{-\xister@X}\xister@TMPB}%
        \pgfpathlineto{\pgfpoint\xister@XY@share\xister@TMPB}%
      }%
    \xister@superimpose{7#1}%
      {%
        \rlap{\copy\csname xister@digitbox@1#1\endcsname}%
        \copy\csname xister@digitbox@6#1\endcsname
      }%
    \xister@superimpose{8#1}%
      {%
        \rlap{\copy\csname xister@digitbox@2#1\endcsname}%
        \copy\csname xister@digitbox@6#1\endcsname
      }%
    \ifxister@nine@triangle@
      \xister@drawdigit{9#1}%
        {%
          \xister@TMPA=.5\dimexpr\xister@XY@share-\xister@X\relax
          \xister@TMPB=#2\dimexpr\xister@Ts-\xister@Tdu\relax
          \xister@clip@h\z@
          \pgfsetlinewidth{2\xister@Tdu}%
          \pgfpathmoveto{\pgfpoint\xister@XY@share\xister@TMPB}%
          \pgfpathlineto{\pgfpoint{-\xister@X}\xister@TMPB}%
          \pgfpathlineto{\pgfpoint\xister@TMPA\xister@Y}%
          \pgfusepath{stroke}%
          \xister@TMPB=#2\dimexpr\xister@Ts-\xister@Tdd\relax
          \pgfsetlinewidth{2\xister@Tdd}%
          \pgfpathmoveto{\pgfpoint\xister@X\xister@TMPB}%
          \pgfpathlineto{\pgfpoint\xister@XY@share\xister@TMPB}%
          \pgfpathlineto{\pgfpoint\xister@TMPA\xister@Y}%
        }%
    \else
      \xister@superimpose{9#1}%
    {%
     \rlap{\copy\csname xister@digitbox@1#1\endcsname}%
     \rlap{\copy\csname xister@digitbox@2#1\endcsname}%
     \copy\csname xister@digitbox@6#1\endcsname
    }%
  \fi
  \unless\ifxister@six@stroke@
   \xister@drawdigit{6#1}%
    {%
     \ifxister@six@dot@
      \xister@TMPA=\dimexpr\xister@Y-#2\xister@Six@dot\relax
      \pgfpathcircle
       {\pgfpoint{\dimexpr#3\xister@Six@dot-\xister@X\relax}%
            \xister@TMPA}%
       {\xister@Six@dot}%
      \pgfpathcircle
       {\pgfpoint{\dimexpr\xister@XY@share-#3\xister@Six@dot\relax}%
            \xister@TMPA}%
       {\xister@Six@dot}%
      \pgfusepath{fill}%
     \else
      \pgfsetlinewidth{2\xister@Th}%
      \xister@TMPB=\dimexpr\xister@Y-#2\xister@Th\relax
      \xister@TMPA=2\dimexpr#3\xister@six@dash\xister@Th\relax
      \pgfpathmoveto{\pgfpoint{-\xister@X}\xister@TMPB}%
      \pgfpathlineto
       {\pgfpoint{\dimexpr\xister@TMPA-\xister@X\relax}%
            \xister@TMPB}%
      \pgfpathmoveto{\pgfpoint\xister@XY@share\xister@TMPB}%
      \pgfpathlineto
       {\pgfpoint{\dimexpr\xister@XY@share-\xister@TMPA\relax}%
            \xister@TMPB}%
     \fi
    }%
  \fi
 }
```

Each digit has the same bounding box and baseline which is specified here. #1 will be the name, #2 the digit specific path.

```
\newcommand\xister@drawdigit[2]
 {%
  \expandafter\setbox\csname xister@digitbox@#1\endcsname=\hbox
   {%
    \begin{pgfpicture}%
     \pgfpathrectanglecorners
      {\pgfpoint{-\xister@bx\xister@X}{-\xister@by\xister@Y}}%
      {\pgfpoint{\xister@bx\xister@X}{\xister@by\xister@Y}}%
     \pgfusepath{use as bounding box}%
     #2%
     \pgfusepath{stroke}%
     \pgfsetbaseline{\xister@sgn\xister@bs\xister@Y}%
    \end{pgfpicture}%
   }%
 }
```

(*End definition for* `\xister@drawdigits@vertical` *and* `\xister@drawdigits@horizontal`*.*)

\xister@drawzero@horizontal
\xister@drawzero@vertical

```
\newcommand*\xister@drawzero@vertical
  {%
    \xister@drawdigit{0}%
      {%
        \pgfsetlinewidth{2\xister@Ts}%
        \pgfpathmoveto{\pgfpoint\z@{-\xister@Y}}%
        \pgfpathlineto{\pgfpoint\z@\xister@Y}%
      }%
  }
\newcommand*\xister@drawzero@horizontal
  {%
    \xister@drawdigit{0}%
      {%
        \pgfsetlinewidth{2\xister@Ts}%
        \pgfpathmoveto{\pgfpoint{-\xister@X}\z@}%
        \pgfpathlineto{\pgfpoint\xister@X\z@}%
      }%
  }
```

(*End definition for* `\xister@drawzero@horizontal` *and* `\xister@drawzero@vertical`*.*)

\xister@superimpose

Some digits can be build from a few of the other ones. This is faster than drawing everything again, and this macro eases the process a bit.

```
\newcommand\xister@superimpose[2]
  {\expandafter\setbox\csname xister@digitbox@#1\endcsname=\hbox{#2}}
```

(*End definition for* `\xister@superimpose`*.*)

2.6 Rebuild the Glyphs

\xister@setlengths

Lengths are set depending on the current font size and the sizes of `\xister@font`. To keep the font changes local they are kept inside a group and a chain of `\expandafter`s is used to evaluate the user specified sizes while the `\xister@font` is still active.

```
\protected\def\xister@setlengths
  {%
    \begingroup
      \xister@font
      \edef\xister@tmp
        {%
          \endgroup
          \xister@X=\the\dimexpr.5\dimexpr\xister@x\relax\relax
          \xister@Y=\the\dimexpr.5\dimexpr\xister@y\relax\relax
          \xister@Th=\the\dimexpr.5\dimexpr\xister@th\relax\relax
          \xister@Tv=\the\dimexpr.5\dimexpr\xister@tv\relax\relax
          \xister@Tdu=\the\dimexpr.5\dimexpr\xister@tdu\relax\relax
          \xister@Tdd=\the\dimexpr.5\dimexpr\xister@tdd\relax\relax
          \xister@Five@dot=\the\dimexpr\xister@five@dot\relax
          \xister@Six@dot=\the\dimexpr\xister@six@dot\relax
          \ifx\xister@ts\@empty
            \xister@Ts=%
              \ifx\xister@build\xister@build@h\xister@Th\else\xister@Tv\fi
          \else
            \xister@Ts=\the\dimexpr.5\dimexpr\xister@ts\relax\relax
          \fi
        }%
    \xister@tmp
  }
```

(*End definition for* `\xister@setlengths`.)

`\xister@ensure@current` The glyphs have to be rebuilt if the font size or colour changed. The latter is an annoyance because we can build boxes in a way that they change colour to the surrounding colour, but PGF is too intelligent here and does issue the corresponding `\special`s to save the current colour for the box. Because of that we test which `\f@size` and colour (using `\\color@.`) was active when we last built the glyphs. If this matches we do nothing, else we rebuild the glyphs.

The `\edef` is just used to remove the debug code if it isn't necessary.

```
\protected\edef\xister@ensure@current
  {%
    \unexpanded{\edef\xister@tmp{\xister@fontfeatures}}%
    \unexpanded{\unless\ifx\xister@tmp\xister@last@font}%
      \ifxister@dbg@
        \unexpanded
          {%
            \typeout
              {%
                Package xistercian Debug: Font changed. Building glyphs for
                \xister@tmp.
              }%
          }%
      \fi
    \unexpanded
      {%
          \let\xister@last@font\xister@tmp
          \xister@build
      }%
      \ifxister@dbg@
        \unexpanded{\typeout{Package xistercian Debug: Done.}}%
      \fi
    \unexpanded{\fi}%
  }
\AtBeginDocument{\xister@ensure@current}
```

(*End definition for* `\xister@ensure@current`.)

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.