timechart — A package for drawing chronological charts[*]

Alan J. Cain

Released 2025-02-15

Abstract

This package provides for the easy creation of chronological charts which show visually the relative historical positions of people and events. Each event or period can be specified by a single line of LaTeX code comprising (possibly uncertain) start and finish dates and a label, and the package takes care of indicating the uncertainties and whether intervals extend beyond the specified bounds of the chart.

Contents

[*]This file describes v0.51, last revised 2025-02-15.

1 Introduction

The timechart package provides a system for the easy creation of chronological charts — of the type pioneered by Joseph Priestley (1733–1804) in his 'Chart of Biography' and more famously in his 'New Chart of History'[1] — which can show visually the relative historical positions of people and events. An example of what timechart can be used to produce is shown in Figure 1 on page 4.

Essentially (and this was a design requirement), *only one line* of LaTeX code is required for each interval (which, in the case of Figure 1, are mostly lifetimes). The timechart package takes care of indicating ranges of possible dates by suitable fading from or to transparency. It also handles indicators to show that intervals continue outside the specified scope of the chart.

timechart was developed from, and replaced, a set of macros used to create the chronological charts in the author's book *Form & Number: A History of Mathematical Beauty*, which is available on the Internet Archive under a Creative Commons licence.[2]

Licence. timechart is released under the LaTeX Project Public Licence v1.3c or later.[3]

Acknowledgements. The author thanks Tânia Paulista for reading and commenting on an earlier draft of the documentation.

2 Requirements

timechart requires PGF/Ti*k*Z and a LaTeX kernel with expl3 support (any kernel version since 2020-02-02 should suffice).

3 Installation

To install manually, run `tex timechart.ins` and copy the file `timechart.sty` to somewhere LaTeX can find it.

4 Getting started

The timechart package is loaded as usual via `\usepackage{timechart}`. There are no package options.

The small example in Section 5 illustrates the basic principles of timechart. Section 13 shows the full code used to produce the large example in Figure 1.

5 Example

This section illustrates how to create the small chart shown in Figure 3 on page 6.

The basic environment is `timechart`, which includes the specification of the start and finish years. The start year 50 BCE is specified as `-50`, the finish year 75 CE as `75`. Each interval in the chart is specified using the `\timechartinterval` command, which

[1] URL: https://commons.wikimedia.org/wiki/File:A_New_Chart_of_History_color.jpg

[2] URL: https://archive.org/details/cain_formandnumber_ebook_large

[3] URL: https://www.latex-project.org/lppl.txt

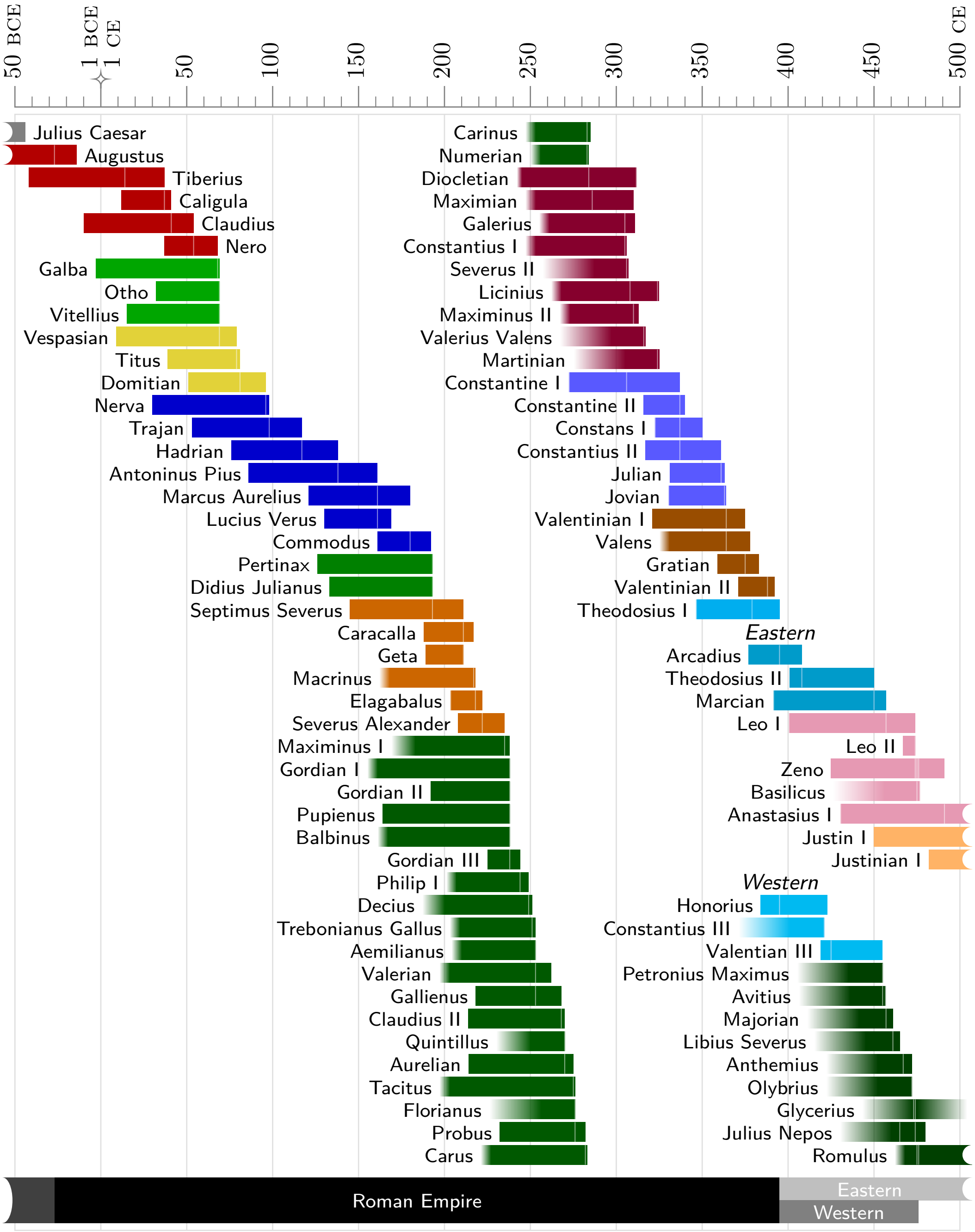

Figure 1: Timechart showing the lifetimes of Roman emperors from 50 BCE to 500 CE. Marks on each lifetime indicate the beginning (and, where relevant, the end) of that emperor's reign. Colours generally indicate dynasties, with shades of green indicating periods when the imperial power shifted between many short-reigning emperors.

takes three mandatory parameters: a start year, a finish year, and a label. The following code produces the flawed preliminary version shown in Figure 2.

```
\begin{timechart}{-50}{75}
  \timechartinterval{-63}{-14}{Augustus}
  \timechartinterval{-42}{37}{Tiberius}
  \timechartinterval{12}{41}{Caligula}
  \timechartinterval{-10}{54}{Claudius}
  \timechartinterval{37}{68}{Nero}
\end{timechart}
```

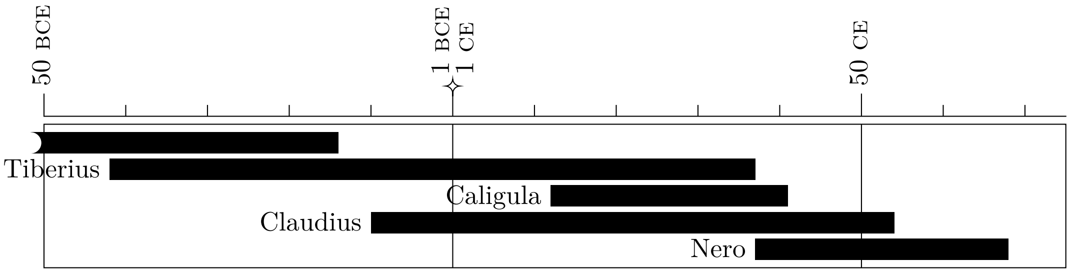

Figure 2: Flawed preliminary version of a chart showing the lifetimes of Roman emperors of the Julio-Claudian dynasty. (The final version is shown in Figure 3.)

This first attempt result illustrates some of the principles of timechart. Each interval has been placed on its own line. More precisely, the y coordinate of the first interval is 0 and each use of `\timechartinterval` increments the 'current y coordinate' by a specified amount. (Various commands are available to set the y coordinate manually or to reset it automatically when it passes certain bounds; see Subsection 9.4.) Horizontally, the chart starts and finishes at the specified years. The topmost interval, indicating Augustus' life, has been truncated with an indicator that it begins before the specified start year of the chart. Vertically, the chart has been sized to fit around the specified intervals.

But this version is hardly satisfactory, for many reasons. The chart finishes between two minor ticks on the axis, because the intervals between major and minor ticks default respectively to 10 years and 50 years. The black intervals and text do not contrast well with the black axis and grid. The serif font is perhaps not best suited to label the intervals. And the label 'Augustus' has been lost, since labels are by default placed on the left of intervals. To rectify these problems, some changes are necessary, all of which can be made using key-value syntax in an optional parameter to the `timechart` environment or the `\timechartinterval` command:

1. Set the intervals between major and minor ticks to 5 years and 25 years respectively, by setting `minor tick interval=5` and `major tick interval=25`.

2. Change the colour of the grid to light grey by appending `draw=lightgray` to the `grid` style

3. Change the colour of the axis to grey by appending `draw=gray` to the `axis line`, `minor tick`, and `major tick` styles.

4. Change the font used for interval labels to small san-serif by appending `node font=\sffamily\small` to the `interval label` style

5. Change the colour of the intervals by defining a style `julioclaudian` that sets `interval bar color=red!80!black` and applying it to each interval via its optional argument. (While `interval bar color` could be set locally for each interval, it is better to define a style that corresponds to the semantic meaning of the colour: in this case, a single dynasty.)

6. Use the `right` key to place some labels on the right

The result is the following code, which produces Figure 3.

```
\begin{timechart}[
    minor tick interval=5,
    major tick interval=25,
    grid/.append style={ draw=lightgray },
    axis line/.append style={ draw=gray },
    minor tick/.append style={ draw=gray },
    major tick/.append style={ draw=gray },
    interval label/.append style={ node font=\sffamily\small },
    julioclaudian/.style={ interval bar color=red!80!black },
  ]{-50}{75}
  \timechartinterval[right,julioclaudian]{-63}{-14}{Augustus}
  \timechartinterval[right,julioclaudian]{-42}{37}{Tiberius}
  \timechartinterval[julioclaudian]{12}{41}{Caligula}
  \timechartinterval[julioclaudian]{-10}{54}{Claudius}
  \timechartinterval[julioclaudian]{37}{68}{Nero}
\end{timechart}
```

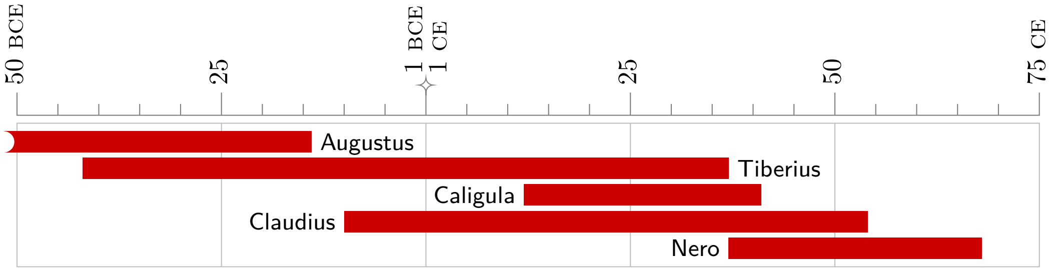

Figure 3: Chart showing the lifetimes of Roman emperors of the Julio-Claudian dynasty.

6 Configuration

All timechart configuration, global or local, is via PGF keys, so some familiarity with their use is beneficial; see the PGF/Ti*k*Z manual.

Configuration keys for timechart are contained in `/timechart/` in the PGF keys hierarchy. The ⟨*options*⟩ passed to the `timechart` environment or any of the commands `\timechartinterval`, `\timecharttext`, or `\timechartspace` are processed within `/timechart/` (since `/timechart/.cd` is executed before keys are processed).

The user may wish to define PGF styles for different kinds of interval within a chart. For example, one could define styles `science` and `art` that set a particular colour for the interval, and write `\timechartinterval[science]{`⟨*birth*⟩`}{`⟨*death*⟩`}{`⟨*name*⟩`}` or

`\timechartinterval[art]{`⟨*birth*⟩`}{`⟨*death*⟩`}{`⟨*name*⟩`}` to distinguish visually the lifetimes of various scientists and artists.

7 Specifying dates and date ranges

Using timechart requires specification of dates and date ranges for the start and finish of each interval, both of which may be uncertain.

The basic specification of a date uses ISO 8601 format `YYYY-MM-DD`. This format specifies a date with day-level precision; use `YYYY-MM` and `YYYY` for month- and year-level precision. If the date is prefixed by `-`, it is treated as the corresponding date before the epoch. (This is a difference with ISO 8601, in which `0` represents 1 BCE, `-1` represents 2 BCE, and so on.) So (assuming that one is using BCE/CE) one uses `-100` to indicate 100 BCE and `100` to indicate 100 CE. (The era indicators 'BCE' and 'CE' appear on the axis. Alternative era indicators — or a different epoch — can be specified; see Section 10.)

A date can be prefixed with a `c` to indicate 'circa', such as `c-100` for 'circa 100 BCE' and `c100` for 'circa 100 CE'. When an interval is drawn in a chart, 'circa' will be indicated by automatically creating (or extending) a range according to the value of the key `circa uncertainty` (see Subsection 9.1).

A date range comprises two dates (each with or without `c`) separated by a slash `/`, with the first date being earlier (or equal to) the second date. (The slash indicates a range of dates per ISO 8601.) A date range can be used to indicate a broader uncertainty than the default 'circa', or to indicate a definite range within which an interval starts or ends.

Examples of correctly formatted dates and date ranges: `-50`, `100`, `c-50`, `c100`, `-50/100`, `c-50 / +100`, `-50/c100`, `c-50/c100`, `-585-05-28`, `1947-12-01`, `1989-11`.

Examples of incorrectly formatted date and date ranges: `100?`, `100CE`, `100BCE`, `-50-100`, `100/-50`.

That is, the syntax for dates and date ranges is per the following (not-quite-formal) grammar:

⟨*cdate-or-crange*⟩	::=	⟨*cdate*⟩	
	|	⟨*crange*⟩	
⟨*cdate*⟩	::=	⟨*date*⟩	
	|	`c`⟨*date*⟩	
⟨*date*⟩	::=	⟨*pdate*⟩	
	|	`-`⟨*pdate*⟩	
⟨*pdate*⟩	::=	⟨*year*⟩`-`⟨*month*⟩`-`⟨*day*⟩	
	|	⟨*year*⟩`-`⟨*month*⟩	
	|	⟨*year*⟩	
⟨*crange*⟩	::=	$\langle date_1\rangle$`/`$\langle date_2\rangle$	
	|	`c`$\langle date_1\rangle$`/`$\langle date_2\rangle$	
	|	$\langle date_1\rangle$`/c`$\langle date_2\rangle$	
	|	`c`$\langle date_1\rangle$`/c`$\langle date_2\rangle$	(with $\text{date}_1 \leq \text{date}_2$)

The bounds of the `timechart` environment (see Section 8) must satisfy ⟨*cdate*⟩ in this grammar (although only the ⟨*year*⟩ is used); the start and finish dates of an `\timechartinterval` command (see Subsection 9.1) must satisfy ⟨*cdate-or-crange*⟩; the parameter of an `\timecharttext` command (see Subsection 9.2) must satisfy ⟨*date*⟩.

Note. For performance reasons, the date parser does only limited error-checking. Months outside the range from `01` to `12` or days outside the range of the specified month will be ignored. Otherwise malformed dates or date ranges may produce obscure error messages or unexpected results.

8 `timechart` environment

`timechart` `\begin{timechart}[⟨options⟩]{⟨start⟩}{⟨finish⟩}`
`⟨content⟩`
`\end{timechart}`

This is the main environment for creating a chronological chart. The mandatory arguments ⟨*start*⟩ and ⟨*finish*⟩ specify the first and last years of the chart. These can be dates with circa indicators (that is, they satisfy ⟨*cdate*⟩ in the grammar in Section 7), but the circa specifier has no effect and only the 'year' part of the date is used. The optional argument ⟨*options*⟩ supplies PGF keys that apply to the entire chart.

The ⟨*content*⟩ comprises commands like `\timechartinterval`, `\timecharttext`, `\timechartspace`, commands for positioning, as described in Section 9, and the user's own Ti*k*Z code.

8.1 General configuration of the `timechart` environment

`/timechart/width=⟨dimension⟩` (default `\textwidth`)

The width of the chart. This refers to the width of the grid and axis of the chart, not including intervals that pass beyond the specified limits of the chart, or axis labels that protrude beyond the width of the axis itself.

`/timechart/tolerance=⟨dimension⟩` (default `5pt`)

The length by which an interval is allowed to pass beyond the limits of the chart before it 'counts' as doing so and the appropriate indicator is drawn.

`/timechart/beyond length=⟨dimension⟩` (default `5pt`)

The length of the indicator that an interval passes beyond the limits of the chart.

`/timechart/beyond x radius=⟨dimension⟩` (default `4pt`)

The horizontal radius of the concave part of the indicator that an interval passes beyond the limits of the chart. (The vertical radius will be half the thickness of the bar.)

`/timechart/ystep=⟨dimension⟩` (default `-10pt`)

The default length (positive or negative) by which the current y coordinate is automatically adjusted after each interval, text, or space is placed.

`/timechart/minor tick interval=⟨number⟩` (default `10`)

The number of years (which must be positive) between each minor tick on the axis.

`/timechart/major tick interval=⟨number⟩` (default `50`)

The number of years (which must be positive) between each major tick on the axis and each vertical line in the grid.

8.2 Grid configuration

`/timechart/no grid`

Do not draw the grid.

`/timechart/grid top ysep=`⟨*dimension*⟩ (default `3pt`)

Distance between the top of the grid and the topmost interval or space.

`/timechart/grid bottom ysep=`⟨*dimension*⟩ (default `3pt`)

Distance between the bottom of the grid and the bottommost interval or space.

`/timechart/grid` (style, initially empty)

Style for drawing the grid.

8.3 Axis configuration

`/timechart/no axis`

Do not draw the axis.

`/timechart/axis line` (style, initially `line cap=rect`)

Style for drawing the axis line.

`/timechart/minor tick` (style, initially empty)

Style for drawing minor ticks.

`/timechart/minor tick length=`⟨*dimension*⟩ (default `1.5mm`)

Length of minor ticks.

`/timechart/major tick` (style, initially empty)

Style for drawing major ticks.

`/timechart/major tick length=`⟨*dimension*⟩ (default `3mm`)

Length of major ticks.

`/timechart/major tick label` (style, initially as described below)

Style for labels on the major ticks on the axis. The intial style essentially sets `inner sep=0`, `outer sep=0`, `anchor=mid west`, `rotate=90`.

9 Within the `timechart` environment

9.1 Intervals

`\timechartinterval`

`\timechartinterval[`⟨*options*⟩`]{`⟨*start*⟩`}{`⟨*finish*⟩`}{`⟨*label*⟩`}`

This command creates an interval in the chart at the current y coordinate between the specified ⟨*start*⟩ and ⟨*finish*⟩, with the given ⟨*label*⟩. These arguments are mandatory. Each of ⟨*start*⟩ and ⟨*finish*⟩ can be either a year or a range of years, possibly with circa markers. That is, each must satisfy ⟨*cyear-or-crange*⟩ in the grammar in Section 7.

The optional argument ⟨*options*⟩ specifies PGF keys within `/timechart/` that are applied locally to the interval.

The current y coordinate will be adjusted according to `/timechart/ystep` unless `/timechart/no autostep` has been set.

Interval configuration

`/timechart/no autostep`

Do not automatically alter the current y coordinate by the amount specified in `/timechart/ystep`.

`/timechart/ref=⟨label⟩` (default `none`)

Make the interval label a hyperlink to the position labelled by ⟨*label*⟩.

`/timechart/mark=⟨comma-separated list of years⟩` (default `empty`)

Draw marks in the interval at the years contained in the list. Each entry in the list must be a definite year (that is, must satisfy ⟨*year*⟩ in the grammar in Section 7). The colour of marks can be specified using `/timechart/interval mark color`. `/timechart/marks` is a synonym for this key.

`/timechart/circa uncertainty=⟨number⟩` (default `3`)

Treat a circa indicator `c` as indicating an uncertainity of ±⟨*number*⟩.

`/timechart/interval minimum width=⟨dimension⟩` (default `1pt`)

Ensure that any interval has a width of at least ⟨*dimension*⟩. This is useful to ensure that a single event is visible in the chart.

If an interval is specified with start and finish ranges, and with `start range=fade` and `finish range=fade`, then the 'certain' portion of the interval will also have width at least ⟨*dimension*⟩. (This restriction prevents a common rendering error where start and finish fadings around a 'certain' interval of length 0 would not quite meet.)

`/timechart/interval bar color=⟨color⟩` (default `black`)

Fill the interval bar with ⟨*color*⟩.

`/timechart/interval bar thickness=⟨dimension⟩` (default `8pt`)

Set the vertical thickness of the interval bar to ⟨*dimension*⟩.

`/timechart/interval bar node name=⟨string⟩` (default `bar node`)

Set the name of the node containing the interval bar to ⟨*string*⟩.

`/timechart/interval mark color=⟨color⟩` (default `gray`)

Draw marks using ⟨*color*⟩.

`/timechart/interval label` (style, initially empty)

Style to apply to an interval label.

`/timechart/interval label centered` (style, initially as below)

Style to apply to an interval label placed centrally. Initially, this style executes the style `/timechart/interval label` and sets `text=white`. The reason for a separate style for centred labels is that often a contrasting colour will be required. For instance, labels positioned to the left and right may be black, but if the bar is black, a centred label should be a light colour.

`/timechart/interval label centered background` (style, initially as below)

Style to apply to the 'background' interval label placed centrally. Initially, this style executes `/timechart/interval label`. The 'background' interval label is simply the usual label and is placed in the same location, but on a layer behind the bar and, unlike the label itself, is not clipped. The reason for this style is that if the bar is narrow, part of the label text (such as ascenders and/or descenders) may naturally extend beyond the bar itself and it may be useful that thse should appear in a different colour.

`/timechart/interval label baseline=⟨dimension⟩` (default `-3pt`)

Position the baseline of the interval label ⟨*dimension*⟩ below the current y coordinate (which is the midpoint of the interval bar).

`/timechart/interval label pos=`⟨*position*⟩ (default `left`)

Specify where to place the label relative to the interval bar: ⟨*position*⟩ may be `left`, `center`, or `right`. The position `center` places the label at the midpoint of *the visible segment of the solid part* of the interval bar (that is, not including fading at the start or finish of the bar, and not including part of the bart that would extend beyond the bounds of the chart). Further, a centred label is clipped to the size of the bar and a unclipped 'background' copy of it is drawn behind the bar, so that the portion appearing 'on' and 'outside' the bar can have different styles. (See the keys `/timechart/interval label centered` and `/timechart/interval label centered background`.)

`/timechart/interval label node name=`⟨*string*⟩ (default `label node`)

Set the name of the node containing the interval label to ⟨*string*⟩.

`/timechart/start range=`⟨*range-type*⟩ (default `fade`)

Type of indication of the range where an interval may start. ⟨*range-type*⟩ can be `fade`, which produces an indicator like ▬, or `slant`, which produces ▰.

`/timechart/finish range=`⟨*range-type*⟩ (default `fade`)

Type of indication of the range where an interval may finish. ⟨*range-type*⟩ can be `fade`, which produces an indicator like ▬, or `slant`, which produces ▰.

`/timechart/fade minimum width=`⟨*dimension*⟩ (default `0pt`)

No start or finish range indicator of type `fade` will be drawn if it is smaller than ⟨*dimension*⟩. Using this key is sometimes necessary because, under certain limited circumstances, a very narrow fading can cause PGF/Ti*k*Z to produce a `dimension too large` error.

9.2 Text

`\timecharttext` `\timecharttext[`⟨*options*⟩`]{`⟨*year*⟩`}{`⟨*text*⟩`}`

Place ⟨*text*⟩ in the time chart at the current y coordinate and at the horizontal position of ⟨*year*⟩, which must be a definite year (that is, must satsify ⟨*year*⟩ in the grammar in Section 7).

The optional argument ⟨*options*⟩ specifies PGF keys within `/timechart/` that are applied locally.

The current y coordinate will be adjusted according to `/timechart/ystep` unless `/timechart/no autostep` has been set.

Text configuration

`/timechart/text node name=`⟨*string*⟩ (default `text node`)

Set the name of the node containing the text to ⟨*string*⟩.

`/timechart/text` (style, initially empty)

Style to apply to the text.

`/timechart/text baseline=`⟨*dimension*⟩ (default `-3pt`)

Position the baseline of the text ⟨*dimension*⟩ below the current y coordinate.

`/timechart/text pos=`⟨*position*⟩ (default `left`)

Specify where to place the label relative to the given ⟨*year*⟩: ⟨*position*⟩ may be `left`, `center`, or `right`.

9.3 Space

`\timechartspace` `\timechartspace[⟨options⟩]`

Create a space in the time chart at the current y coordinate, with the same effect on vertical spacing as an interval. More precisely, the current y coordinate will be adjusted according to `/timechart/ystep` unless `/timechart/no autostep` has been set.

The optional argument ⟨*options*⟩ specifies PGF keys within `/timechart/` that are applied locally.

9.4 Positioning

The commands `\timechartinterval`, `\timecharttext`, and `\timechartspace` all act at the current y coordinate and change its value according to `/timechart/ystep` (unless `/timechart/no autostep` is used). There are several functions to set the current y coordinate and to have it reset automatically.

`\timechartsety` `\timechartsety{⟨dimension⟩}`

Set the current y coordinate to ⟨*dimension*⟩.

`\timechartsavey` `\timechartsavey`

Save the current y coordinate. If `\timechartresety` is used, the y coordinate resets to the last saved y coordinate. If the current y coordinate exceeds the minimum or maximum set by `\timechartsetyminimumautoreset` and `\timechartsetymaximumautoreset`, it will be reset to the last saved y coordinate.

`\timechartresety` `\timechartresety`

Reset the y coordinate to the last coordinate saved using `\timechartsavey`, or to 0 if there has been no use of `\timechartsavey` within the current `timechart` environment.

`\timechartsetyminimumautoreset` `\timechartsetyminimumautoreset{⟨min-coordinate⟩}`
`\timechartsetymaximumautoreset` `\timechartsetymaximumautoreset{⟨max-coordinate⟩}`

Set y coordinates that automatically trigger `\timechartresety` if the current y coordinate goes below ⟨*min-coordinate*⟩ or above ⟨*max-coordinate*⟩.

`\timechartstepy` `\timechartstepy[⟨count⟩]`

Manually step the current y coordinate by ⟨*count*⟩ times the value of `/timechart/ystep`. The default value of ⟨*count*⟩ is 1. (The `/timechart/no autostep` does not affect `\timechartstepy`.)

9.5 Completion

`\timechartfinish`

Signal that the chart is complete and that the grid and axis should be drawn (unless the keys `/timechart/no grid` and/or `/timechart/no axis` have been used). It is not necessary to use this command: if it is not given, the grid and axis will be drawn at the end of the `timechart` environment. But after this command, the Ti*k*Z nodes `grid` and `axis` are available, containing (respectively) the grid and the axis. These can be used in for further Ti*k*Z drawing.

Note that after `\timechartfinish`, none of the various `\timechart`... commands are available inside that `timechart` environment.

9.6 Shortcut keys

`/timechart/left`

Equivalent to setting `/timechart/interval label pos` and `/timechart/text pos` to `left`.

`/timechart/center`

Equivalent to setting `/timechart/interval label pos` and `/timechart/text pos` to `center`.

`/timechart/right`

Equivalent to setting `/timechart/interval label pos` and `/timechart/text pos`right to `right`.

10 Era indicators

`\timechartmakebeforeyear`
`\timechartmakeafteryear`

`\timechartmakebeforeyear{`⟨*number*⟩`}`
`\timechartmakeafteryear{`⟨*number*⟩`}`

Typeset ⟨*number*⟩ (which should be a positive whole number) as a year before or after the epoch. By default, `\timechartmakebeforeyear{`⟨*number*⟩`}` produces ⟨*number*⟩ BCE and `\timechartmakeafteryear{`⟨*number*⟩`}` produces ⟨*number*⟩ CE.

These commands are used for axis labels and can be re-defined by the user. For example, if BC/AD is preferred to BCE/CE, the user can define

```
\renewcommand*{\timechartmakebeforeyear}[1]{#1~\textsc{bc}}
\renewcommand*{\timechartmakeafteryear}[1]{\textsc{ad}~#1}
```

Similarly, if AH/BH is preferred, the user can define

```
\renewcommand*{\timechartmakebeforeyear}[1]{#1~\textsc{bh}}
\renewcommand*{\timechartmakeafteryear}[1]{\textsc{ah}~#1}
```

11 Legend

timechart supplies a number of auxiliary macros for creating a legend to explain, for example the significance of different colours of intervals. For example, Figure 4 shows a suitable legend for Figure 1.

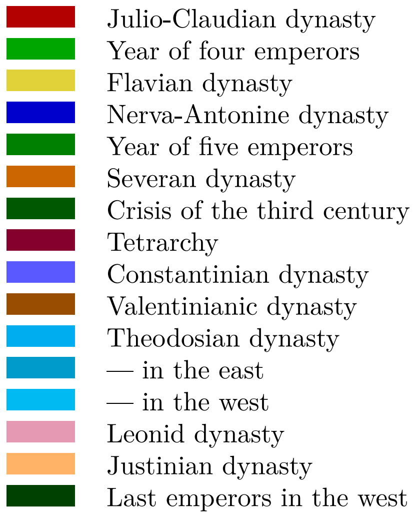

Figure 4: Example legend for the timechart shown in Figure 1.

The `\timechartlength`... macros are *not* meant to be used inside a `timechart` environment, but in locations such as running text or a `tabular` environment.

`\timechartlegenditem`

`\timechartlegenditem[⟨options⟩]`

Draw a bar suitable for use in a legend. ⟨*options*⟩ specifies PGF keys that are applied within `/timechart`. The same PGF keys that affect interval bars affect the drawn bar, as do the keys listed below.

`\timechartlegendstartrange`
`\timechartlegendfinishrange`

`\timechartlegendstartrange[⟨options⟩]`
`\timechartlegendfinishrange[⟨options⟩]`

Draw a bar suitable for use in a legend, with a start or finish range. ⟨*options*⟩ specifies PGF keys that are applied within `/timechart/`. The same PGF keys that affect interval bars affect the drawn bar, as do the keys listed below.

`/timechart/legend item width=⟨dimension⟩` (default `9mm`)

When using macros `\timechartlegenditem`, `\timechartlegendstartrange`, or `\timechartlegendfinishrange`, draw a bar of total width ⟨*dimension*⟩.

`/timechart/legend item range width=⟨dimension⟩` (default `3mm`)

When using `\timechartlegendstartrange` or `\timechartlegendfinishrange`, draw a bar with a range of width ⟨*dimension*⟩.

12 Usage notes

12.1 Additional Ti*k*Z code

The `timechart` environment is a `tikzpicture`. The user can add any Ti*k*Z code before, between, or after content created using the `\timechart`... commands. Each use of `\timechartinterval` defines two nodes. One, by default named `bar node`, contains the interval bar; the other, by default named `label node`, contains the interval label. Similarly, text added using `\timecharttext` is contained in a node, by default named `text node`. (The default names are re-used, but can be changed using the keys `/timechart/interval bar node name`, `/timechart/interval label node name`, and `/timechart/text node name`.) The user can use these nodes to position extra content.

If the `\timechartcomplete` command is used (after which the `\timechart`... commands are no longer available within the `timechart` environment) the nodes `grid` and `axis`, which contain the grid and the axis, are also available.

12.2 ‘year zero’

Although calendars typically do not admit a ‘year zero’ (for instance, 1 BCE is immediately followed by 1 CE, with no intervening ‘year zero’), timechart does allow `0` for the ⟨*start*⟩ or ⟨*finish*⟩ of the `timechart` environment or as the ⟨*start*⟩ or ⟨*finish*⟩ of `\timechartinterval` or the ⟨*year*⟩ of `\timecharttext`. But ‘year zero’ is indicated on the axis by a special epoch marker showing the last year before and first year after the epoch.

12.3 `dimension too large error`

Under certain limited circumstances, a very narrow fading can cause PGF/Ti*k*Z to produce a `dimension too large` error. This error can be triggered by a range of type `fade` at the start or end of an interval when the range is small compared to the width of the chart. In this case, the fading is unlikely to be visible, so one can simply set `/timechart/fade minimum width` to a suitable small positive value, which will stop the fading from being drawn and so prevent the error.

13 Appendix: full example source

This section contains the necessary source code to produce the example timechart and legend shown in Figures 1 and 4 on pages 4 and 14.

13.1 Setup source

```
\pgfkeys{
  /timechart/.cd,
  julioclaudian/.style={
    interval bar color=red!70!black,
    interval mark color=red!70!black!50!white
  },
  fouremperors/.style={
    interval bar color=green!65!black,
    interval mark color=green!65!black!50!white
  },
  flavian/.style={
    interval bar color=yellow!85!black,
    interval mark color=yellow!85!black!50!white
  },
  nervaantonine/.style={
    interval bar color=blue!80!black,
    interval mark color=blue!80!black!50!white
  },
  fiveemperors/.style={
    interval bar color=green!50!black,
    interval mark color=green!50!black!50!white
  },
  severan/.style={
    interval bar color=orange!80!black,
    interval mark color=orange!80!black!50!white
  },
  thirdcentury/.style={
    interval bar color=green!35!black,
    interval mark color=green!35!black!50!white
  },
  tetrarchy/.style={
    interval bar color=purple!70!black,
    interval mark color=purple!70!black!50!white
  },
  constantinian/.style={
    interval bar color=blue!65!white,
    interval mark color=blue!65!white!50!white
  },
  valentinianic/.style={
    interval bar color=orange!60!black,
    interval mark color=orange!60!black!50!white
  },
  theodosian/.style={
    interval bar color=cyan,
    interval mark color=cyan!50!white
  },
  theodosian-east/.style={
    interval bar color=cyan!80!black,
    interval mark color=cyan!80!black!50!white
  },
  theodosian-west/.style={
    interval bar color=cyan!80!white,
    interval mark color=cyan!80!white!50!white
  },
  leonid/.style={
    interval bar color=purple!40!white,
    interval mark color=purple!40!white!50!white
  },
  justinian/.style={
    interval bar color=orange!60!white,
    interval mark color=orange!60!white!50!white
  },
  lastwest/.style={
    interval bar color=green!25!black,
    interval mark color=green!25!black!50!white
  },
}
```

13.2 Timechart source

```
\begin{timechart}[
    axis line/.append style={ draw=gray, line width=.5pt },
    grid/.append style={ draw=lightgray!50!white, line width=.5pt },
  minor tick/.append style={ draw=gray },
  major tick/.append style={ line width=.5pt, draw=gray },
  major tick label/.append style={ node font=\small },
  interval label/.style={ node font=\sffamily\footnotesize },
  text/.style={ node font=\sffamily\small\itshape, },
  ystep=-3.25mm,
]{-50}{500}

\pgfmathsetmacro{\mainlinecount}{46}

\timechartsetyminimumautoreset{-3.25mm*\mainlinecount+1mm}

\timechartinterval[interval bar color=gray,right,mark=-45]{-100}{-44}{Julius Caesar}
\timechartinterval[julioclaudian,right,mark=-27]{-63}{-14}{Augustus}
\timechartinterval[julioclaudian,right,mark=14]{-42}{37}{Tiberius}
\timechartinterval[julioclaudian,right,mark=37]{12}{41}{Caligula}
\timechartinterval[julioclaudian,right,mark=41]{-10}{54}{Claudius}
\timechartinterval[julioclaudian,right,mark=54]{37}{68}{Nero}

\timechartinterval[fouremperors,mark=68]{-3}{69}{Galba}
\timechartinterval[fouremperors,mark=69]{32}{69}{Otho}
\timechartinterval[fouremperors,mark=69]{15}{69}{Vitellius}

\timechartinterval[flavian,mark=69]{9}{79}{Vespasian}
\timechartinterval[flavian,mark=79]{39}{81}{Titus}
\timechartinterval[flavian,mark=81]{51}{96}{Domitian}

\timechartinterval[nervaantonine,mark=96]{30}{98}{Nerva}
\timechartinterval[nervaantonine,mark=98]{53}{117}{Trajan}
\timechartinterval[nervaantonine,mark=117]{76}{138}{Hadrian}
\timechartinterval[nervaantonine,mark=138]{86}{161}{Antoninus Pius}
\timechartinterval[nervaantonine,mark=161]{121}{180}{Marcus Aurelius}
\timechartinterval[nervaantonine,mark=161]{130}{169}{Lucius Verus}
\timechartinterval[nervaantonine,mark=180]{161}{192}{Commodus}

\timechartinterval[fiveemperors,mark=193]{126}{193}{Pertinax}
\timechartinterval[fiveemperors,mark=193]{133}{193}{Didius Julianus}

\timechartinterval[severan,mark=193]{145}{211}{Septimus Severus}
\timechartinterval[severan,mark=211]{188}{217}{Caracalla}
\timechartinterval[severan,mark=211]{189}{211}{Geta}
\timechartinterval[severan,mark=217]{c165}{218}{Macrinus}
\timechartinterval[severan,mark=218]{203/204}{222}{Elagabalus}
\timechartinterval[severan,mark=222]{208}{235}{Severus Alexander}

\timechartinterval[thirdcentury,mark=235]{c172/c180}{238}{Maximinus I}
\timechartinterval[thirdcentury,mark=238]{c158}{238}{Gordian I}
\timechartinterval[thirdcentury,mark=238]{192}{238}{Gordian II}
\timechartinterval[thirdcentury,mark=238]{164}{238}{Pupienus}
\timechartinterval[thirdcentury,mark=238]{c164}{238}{Balbinus}
\timechartinterval[thirdcentury,mark=238]{225}{244}{Gordian III}
\timechartinterval[thirdcentury,mark=244]{c204}{249}{Philip I}
\timechartinterval[thirdcentury,mark=249]{c190/200}{251}{Decius}
```

```
\timechartinterval[thirdcentury,mark=251]{c206}{253}{Trebonianus Gallus}
\timechartinterval[thirdcentury,mark=253]{c207}{253}{Aemilianus}
\timechartinterval[thirdcentury,mark=253]{c200}{262}{Valerian}
\timechartinterval[thirdcentury,mark=253]{218}{268}{Gallienus}
\timechartinterval[thirdcentury,mark=268]{214}{270}{Claudius II}
\timechartinterval[thirdcentury,mark=270]{230/250}{270}{Quintillus}
\timechartinterval[thirdcentury,mark=270]{214}{275}{Aurelian}
\timechartinterval[thirdcentury,mark=275]{c200}{276}{Tacitus}
\timechartinterval[thirdcentury,mark=276]{226/256}{276}{Florianus}
\timechartinterval[thirdcentury,mark=276]{232}{282}{Probus}
\timechartinterval[thirdcentury,mark=282]{c224}{283}{Carus}
\timechartinterval[thirdcentury,mark=283]{c250}{285}{Carinus}
\timechartinterval[thirdcentury,mark=283]{c253}{284}{Numerian}

\timechartinterval[tetrarchy,mark=284]{242/245}{311/312}{Diocletian}
\timechartinterval[tetrarchy,mark=286]{c250}{310}{Maximian}
\timechartinterval[tetrarchy,mark=305]{c258}{311}{Galerius}
\timechartinterval[tetrarchy,mark=305]{c250}{306}{Constantius I}
\timechartinterval[tetrarchy,mark=306]{257/287}{307}{Severus II}
\timechartinterval[tetrarchy,marks={308,324}]{c265}{325}{Licinius}
\timechartinterval[tetrarchy,mark=310]{c270}{313}{Maximinus II}
\timechartinterval[tetrarchy,mark=316]{267/297}{317}{Valerius Valens}
\timechartinterval[tetrarchy,mark=324]{275/305}{325}{Martinian}

\timechartinterval[constantinian,mark=306]{272/273}{337}{Constantine I}
\timechartinterval[constantinian,mark=337]{316}{340}{Constantine II}
\timechartinterval[constantinian,mark=337]{322/323}{350}{Constans I}
\timechartinterval[constantinian,mark=337]{317}{361}{Constantius II}
\timechartinterval[constantinian,mark=361]{331}{363}{Julian}
\timechartinterval[constantinian,mark=363]{330/331}{364}{Jovian}

\timechartinterval[valentinianic,mark=364]{321}{375}{Valentinian I}
\timechartinterval[valentinianic,mark=364]{c328}{378}{Valens}
\timechartinterval[valentinianic,mark=375]{359}{383}{Gratian}
\timechartinterval[valentinianic,mark=388]{371}{392}{Valentinian II}

\timechartinterval[theodosian,mark=379]{346/347}{395}{Theodosius I}
\timecharttext[center]{395}{Eastern}
\timechartinterval[theodosian-east,mark=395]{377}{408}{Arcadius}
\timechartinterval[theodosian-east,mark=408]{401}{450}{Theodosius II}
\timechartinterval[theodosian-east,mark=450]{391/392}{457}{Marcian}

\timechartinterval[leonid,mark=457]{400/401}{474}{Leo I}
\timechartinterval[leonid,mark=474]{467}{474}{Leo II}
\timechartinterval[leonid,mark={474,475,476}]{425}{491}{Zeno}
\timechartinterval[leonid,mark=475]{426/456}{476/477}{Basilicus}
\timechartinterval[leonid,mark=491]{430/431}{518}{Anastasius I}

\timechartinterval[justinian,mark=518]{450}{527}{Justin I}
\timechartinterval[justinian,mark=527]{482}{565}{Justinian I}

\timecharttext[center]{395}{Western}
\timechartinterval[theodosian-west,mark=395]{384}{423}{Honorius}
\timechartinterval[theodosian-west,mark=421]{371/401}{421}{Constantius III}
```

```
  \timechartinterval[theodosian-west,mark=425]{419}{455}{Valentian III}

  \timechartinterval[lastwest,mark=455]{405/435}{455}{Petronius Maximus}
  \timechartinterval[lastwest,mark=455]{406/436}{456/457}{Avitius}
  \timechartinterval[lastwest,mark=457]{411/441}{461}{Majorian}
  \timechartinterval[lastwest,mark=461]{415/445}{465}{Libius Severus}
  \timechartinterval[lastwest,mark=467]{422/452}{472}{Anthemius}
  \timechartinterval[lastwest,mark=472]{422/452}{472}{Olybrius}
  \timechartinterval[lastwest,mark={473,474}]{443/473}{474/504}{Glycerius}
  \timechartinterval[lastwest,mark={474,465}]{430/460}{480}{Julius Nepos}
  \timechartinterval[lastwest,mark={475,476}]{c465}{507/527}{Romulus}

  \timechartsetyminimumautoreset{-\maxdimen}
  \timechartsety{(-\mainlinecount-1)*3.25mm}

  \timechartinterval[
    center,
    interval bar color=darkgray,
    interval text/.append style={ text=white },
    interval bar thickness=6.5mm
  ]{-509}{-27}{}
  \timechartstepy[-1]
  \timechartinterval[
    center,
    interval bar color=black,
    interval text/.append style={ node font=\sffamily\normalsize,text=white },
    interval bar thickness=6.5mm
  ]{-27}{395}{Roman Empire}
  \timechartstepy[-1.5]
  \timechartinterval[
    center,
    interval bar color=lightgray,
    interval bar thickness=3.25mm
  ]{395}{1453}{Eastern}
  \timechartinterval[
    center,
    interval bar color=gray,
    interval bar thickness=3.25mm
  ]{395}{476}{Western}
\end{timechart}
```

13.3 Legend source

```
\begin{tabular}{rl}
  \timechartlegenditem[julioclaudian]   & Julio-Claudian dynasty      \\
  \timechartlegenditem[fouremperors]    & Year of four emperors       \\
  \timechartlegenditem[flavian]         & Flavian dynasty             \\
  \timechartlegenditem[nervaantonine]   & Nerva-Antonine dynasty      \\
  \timechartlegenditem[fiveemperors]    & Year of five emperors       \\
  \timechartlegenditem[severan]         & Severan dynasty             \\
  \timechartlegenditem[thirdcentury]    & Crisis of the third century \\
  \timechartlegenditem[tetrarchy]       & Tetrarchy                   \\
  \timechartlegenditem[constantinian]   & Constantinian dynasty       \\
  \timechartlegenditem[valentinianic]   & Valentinianic dynasty       \\
  \timechartlegenditem[theodosian]      & Theodosian dynasty          \\
  \timechartlegenditem[theodosian-east] & --- in the east             \\
  \timechartlegenditem[theodosian-west] & --- in the west             \\
  \timechartlegenditem[leonid]          & Leonid dynasty              \\
  \timechartlegenditem[justinian]       & Justinian dynasty           \\
  \timechartlegenditem[lastwest]        & Last emperors in the west   \\
\end{tabular}
```

14 Implementation

```
⟨*package⟩
⟨@@=timechart⟩
```

14.1 Coding standard

This package makes extensive use of pgfmath computations. The usual expl3 standard of ending variables with a type indicator (`_bool`, `_int`, etc.) is therefore adapted as follows:

`_year` Stores a year, which could in principle be fractional.

`_pgf` Stores a length calculated by pgfmath. (Unlike `_dim`, there is no underlying dimension register.)

`_x` Stores a raw x coordinate (not in Ti*k*Z's XY-coordinate system).

`_y` Stores a raw y coordinate (not in Ti*k*Z's XY-coordinate system).

`_text` Stores text (not an expl3 string).

14.2 Initial set-up

Package identification/version information.

```
\NeedsTeXFormat{LaTeX2e}[2020-02-02]
\ProvidesExplPackage{timechart}{2025-02-15}{0.51}
  {Typesetting chronological charts}
```

14.3 Load TikZ

```
\RequirePackage{tikz}
```

In the remainder of the package, only a limited subset of Ti*k*Z is used, and PGF code is preferred. For PGF keys, it is necessary to use ~ in place of a space.

14.4 Scratch variables

\l__timechart_tmpa_bool
\l__timechart_tmpb_bool
\l__timechart_tmpc_bool
\l__timechart_tmpd_bool

Scratch boolean variables.

```
\bool_new:N\l__timechart_tmpa_bool
\bool_new:N\l__timechart_tmpb_bool
\bool_new:N\l__timechart_tmpc_bool
\bool_new:N\l__timechart_tmpd_bool
```

(*End of definition for* `\l__timechart_tmpa_bool` *and others.*)

`\l__timechart_tmpa_dim`
`\l__timechart_tmpb_dim`
`\l__timechart_tmpc_dim`
`\l__timechart_tmpd_dim`

Scratch dimension variables, reusing `\l_tmpa_dim` and `\l_tmpb_dim` with uniform names.

```
\cs_set_eq:NN\l__timechart_tmpa_dim\l_tmpa_dim
\cs_set_eq:NN\l__timechart_tmpb_dim\l_tmpb_dim
\dim_new:N\l__timechart_tmpc_dim
\dim_new:N\l__timechart_tmpd_dim
```

(*End of definition for* `\l__timechart_tmpa_dim` *and others.*)

14.5 Generic auxiliary functions

`\__timechart_make_ref:NN`

Make hyperreference from text, if the supplied target is non-empty.
#1 : Reference for hyperlink target, or empty.
#2 : Text.

```
\cs_new:Npn\__timechart_make_ref:NN #1#2
  {
    \str_if_empty:NTF #1
      { #2 }
      { \hyperref[#1]{#2} }
  }
```

(*End of definition for* `\__timechart_make_ref:NN`*.*)

14.6 PGF auxiliary functions

`__timechart_pgfmathsetbool:nn`

Set an expl3 boolean variable to the outcome of a pgfmath comparison. This macro is simply a wrapper around `\pgfmathsetmacro` using `ifthenelse` and returning the boolean literal true or false.

```
\cs_new:Npn\__timechart_pgfmathsetbool:nn #1#2
  {
    \pgfmathsetmacro{#1}{ifthenelse(#2,"\c_true_bool","\c_false_bool")}
  }
```

(*End of definition for* `__timechart_pgfmathsetbool:nn`*.*)

`\__timechart_if_equal:nnF`

Use pgfmath to check whether #1 and #2 are equal. If not, execute #3.

```
\cs_new:Npn\__timechart_if_equal:nnF #1#2#3
  {
    \__timechart_pgfmathsetbool:nn{\l__timechart_tmpa_bool}{#1==#2}
    \bool_if:NF\l__timechart_tmpa_bool{#3}
  }
```

(*End of definition for* `\__timechart_if_equal:nnF`*.*)

`\__timechart_pgfextractxy:nnn`

Extract coordinates of #3 (a PGF point) to dimension variables #1 and #2. This macro simply combines the functionality of `\pgfextractx` and `\pgfextracty`.

```
\cs_new:Npn\__timechart_pgfextractxy:nnn #1#2#3
  {
    \pgf@process{#3}
    #1=\pgf@x\relax
    #2=\pgf@y\relax
  }
```

(*End of definition for* `\__timechart_pgfextractxy:nnn`*.*)

__timechart_hsmash_pgfnode:nnnnn

Do the same as \pgfnode but only update the bounding box 'vertically'.

```
\cs_new:Npn\__timechart_hsmash_pgfnode:nnnnn #1#2#3#4#5
  {
    \pgfinterruptboundingbox
    \pgfnode{#1}{#2}{#3}{#4}{#5}
    \pgfcoordinate
      {__timechart_tmpa_coord}
      {\pgfpointanchor{current~bounding~box}{south}}
    \pgfcoordinate
      {__timechart_tmpb_coord}
      {\pgfpointanchor{current~bounding~box}{north}}
    \endpgfinterruptboundingbox
    \pgfextractx
      {\l__timechart_tmpa_dim}
      {\pgfpointanchor{__timechart_tmpa_coord}{center}}
    \pgfextractx
      {\l__timechart_tmpb_dim}
      {\pgfpointanchor{__timechart_tmpb_coord}{center}}
    \pgfpathmoveto{\pgfpoint{\l__timechart_tmpa_dim}{0}}
    \pgfpathmoveto{\pgfpoint{\l__timechart_tmpb_dim}{0}}
    \pgfusepath{discard}
  }
```

(*End of definition for* __timechart_hsmash_pgfnode:nnnnn*.*)

__timechart_make_rectangle_node:nnnn

Make a node with south west corner #1, north east corner #2, and name #3. #1 and #2 should be given as PGF points. #4 is a boolean literal indicating whether the path should be stroked.

```
\cs_new:Npn\__timechart_make_rectangle_node:nnnn #1#2#3#4
  {
    \group_begin:
    \__timechart_pgfextractxy:nnn
      {\l__timechart_tmpa_dim}{\l__timechart_tmpb_dim}{#1}
    \__timechart_pgfextractxy:nnn
      {\l__timechart_tmpc_dim}{\l__timechart_tmpd_dim}{#2}
    \pgftransformshift{#1}
    \pgfset{
      minimum~width=\l__timechart_tmpc_dim-\l__timechart_tmpa_dim,
      minimum~height=\l__timechart_tmpd_dim-\l__timechart_tmpb_dim,
      inner~sep=0,
      outer~sep=0,
    }
    \bool_if:NTF #4
      { \pgfnode{rectangle}{south~west}{}{#3}{\pgfusepath{draw}} }
      { \pgfnode{rectangle}{south~west}{}{#3}{\pgfusepath{discard}} }
    \group_end:
  }
```

(*End of definition for* __timechart_make_rectangle_node:nnnn*.*)

\l__timechart_left_nonclip_x
\l__timechart_right_nonclip_x

In order to clip 'only on one side', define coordinates for specifying the 'other side' of the clipping path.

```
\pgfmathsetmacro{\l__timechart_left_nonclip_x}{-16000pt}
\pgfmathsetmacro{\l__timechart_right_nonclip_x}{16000pt}
```

(*End of definition for* `\l__timechart_left_nonclip_x` *and* `\l__timechart_right_nonclip_x`*.*)

__timechart_set_style_line_width:nn

Set macro `#1` to be the line width set by the PGF style `#2`. Note that `\begingroup` and `\endgroup` are used here because of the definition of `\pgfmathsmuggle`.

```
\cs_new:Npn\__timechart_set_style_line_width:nn #1#2
  {
    \begingroup
    \tikzset{#2}
    \pgfmathsetlengthmacro{#1}{\pgflinewidth}
    \pgfmathsmuggle #1
    \endgroup
  }
```

(*End of definition for* `\__timechart_set_style_line_width:nn`*.*)

14.7 PGF keys

All PGF keys for this package are under `/timechart/`.

```
\pgfkeys{
  /timechart/.cd,
```

Keys applicable to whole chart.

```
  width/.initial=\textwidth,
  tolerance/.initial=5pt,
  beyond~length/.initial=5pt,
  beyond~x~radius/.initial=4pt,
  ystep/.initial=-10pt,
  minor~tick~interval/.initial=10,
  major~tick~interval/.initial=50,
```

Keys applicable to the grid.

```
  no~grid/.code = { \bool_set_false:N\l__timechart_grid_bool },
  grid~top~ysep/.initial={3pt},
  grid~bottom~ysep/.initial={3pt},
  grid/.style={},
```

Keys applicable to the axis.

```
  no~axis/.code = { \bool_set_false:N\l__timechart_axis_bool },
  axis~line/.style={
    line~cap=rect,
  },
  axis~ysep/.initial=3pt,
  minor~tick/.style={},
  minor~tick~length/.initial=1.5mm,
  major~tick/.style={},
  major~tick~length/.initial=3mm,
  major~tick~label/.style={
    inner~sep=0,
    outer~sep=0,
    anchor=mid~west,
    rotate=90,
  },
```

Keys applicable to intervals, texts, spaces, and legends.

```
no~autostep/.code = { \bool_set_false:Nz\l__timechart_autostep_bool },
ref/.initial={},
mark/.initial={},
marks/.forward~to=/timechart/mark,
circa~uncertainty/.initial=3,
interval~minimum~width/.initial=1pt,
interval~bar~color/.initial=black,
interval~bar~thickness/.initial=8pt,
interval~bar~node~name/.initial = {bar~node},
interval~mark~color/.initial=gray,
interval~label/.style={},
interval~label~centered/.style={/timechart/interval~label,text=white},
interval~label~centered~background/.style={/timechart/interval~label},
interval~label~baseline/.initial=-3pt,
interval~label~pos/.is~choice,
interval~label~pos/left/.code
  = { \int_set:Nn\l__timechart_label_pos_int{0} },
interval~label~pos/center/.code
  = { \int_set:Nn\l__timechart_label_pos_int{1} },
interval~label~pos/right/.code
  = { \int_set:Nn\l__timechart_label_pos_int{2} },
interval~label~node~name/.initial = {label~node},
start~range/.is~choice,
start~range/fade/.code
  = { \int_set:Nn\l__timechart_start_range_type_int{0} },
start~range/slant/.code
  = { \int_set:Nn\l__timechart_start_range_type_int{1} },
finish~range/.is~choice,
finish~range/fade/.code
  = { \int_set:Nn\l__timechart_finish_range_type_int{0} },
finish~range/slant/.code
  = { \int_set:Nn\l__timechart_finish_range_type_int{1} },
fade~minimum~width/.initial = 0pt,
```

Keys applicable only to texts.

```
text~node~name/.initial = {text~node},
text/.style={},
text~baseline/.initial=-3pt,
text~pos/.is~choice,
text~pos/left/.code = { \int_set:Nn\l__timechart_text_pos_int{0} },
text~pos/center/.code = { \int_set:Nn\l__timechart_text_pos_int{1} },
text~pos/right/.code = { \int_set:Nn\l__timechart_text_pos_int{2} },
```

Keys applicable only to legends.

```
legend~item~width/.initial=9mm,
legend~item~range~width/.initial=3mm,
```

Shortcuts for positioning.

```
left/.code = {
  \int_set:Nn\l__timechart_label_pos_int{0}
  \int_set:Nn\l__timechart_text_pos_int{0}
},
center/.code = {
  \int_set:Nn\l__timechart_label_pos_int{1}
```

```
    \int_set:Nn\l__timechart_text_pos_int{1}
  },
  right/.code = {
    \int_set:Nn\l__timechart_label_pos_int{2}
    \int_set:Nn\l__timechart_text_pos_int{2}
  },
}
```

\l__timechart_grid_bool

Boolean indicating whether the grid will be drawn. This variable is by default true but can be set false via the `/timechart/no grid` PGF key.

```
\bool_new:N\l__timechart_grid_bool
\bool_set_true:N\l__timechart_grid_bool
```

(*End of definition for* `\l__timechart_grid_bool`.)

\l__timechart_axis_bool

Boolean indicating whether the axis will be drawn. This variable is by default true but can be set false via the `/timechart/no axis` PGF key.

```
\bool_new:N\l__timechart_axis_bool
\bool_set_true:N\l__timechart_axis_bool
```

(*End of definition for* `\l__timechart_axis_bool`.)

\l__timechart_autostep_bool

Boolean indicating whether to automatically step the y coordinate after an interval, text, or space. This variable is by default true but can be set false via the `/timechart/no autostep` PGF key.

```
\bool_new:N\l__timechart_autostep_bool
\bool_set_true:N\l__timechart_autostep_bool
```

(*End of definition for* `\l__timechart_autostep_bool`.)

\l__timechart_label_pos_int

An integer to hold the interval label position. This is set via the `/timechart/interval label pos` PGF key.

```
\int_new:N \l__timechart_label_pos_int
```

(*End of definition for* `\l__timechart_label_pos_int`.)

\l__timechart_text_pos_int

An integer to hold the text position. This is set via the `/timechart/text pos` PGF key.

```
\int_new:N \l__timechart_text_pos_int
```

(*End of definition for* `\l__timechart_text_pos_int`.)

\l__timechart_start_range_type_int
\l__timechart_finish_range_type_int

Integers to hold the type of the start/end ranges. These are set via the `/timechart/start range` and `/timechart/finish range` PGF keys.

```
\int_new:N \l__timechart_start_range_type_int
\int_new:N \l__timechart_finish_range_type_int
```

(*End of definition for* `\l__timechart_start_range_type_int` *and* `\l__timechart_finish_range_type_int`.)

14.8 Main environment

timechart The main environment.

#1 : PGF keys to apply.

#2 : Start year.

#3 : End year.

```
\NewDocumentEnvironment{timechart}{ O{} m m }
  { \__timechart_main_begin:nnn{#1}{#2}{#3} }
  { \__timechart_main_end: }
```

(*End of definition for* `timechart`. *This function is documented on page 8.*)

__timechart_main_begin:nnn This command uses values specified by PGF keys to make some necessary calculations to begin the chart.

```
\cs_new:Npn \__timechart_main_begin:nnn #1#2#3
  {
```

Process the supplied PGF keys and retrieve values that affect the chart as a whole.

```
    \pgfkeys{
      /timechart/.cd,
      #1,
      width/.get=\l__timechart_width_pgf,
      tolerance/.get=\l__timechart_tolerance_pgf,
      ystep/.get=\l__timechart_ystep_pgf,
      grid~top~ysep/.get=\l__timechart_grid_top_ysep_pgf,
      grid~bottom~ysep/.get=\l__timechart_grid_bottom_ysep_pgf,
      beyond~length/.get=\l__timechart_beyond_length_pgf,
      beyond~x~radius/.get=\l__timechart_beyond_x_radius_pgf,
      minor~tick~interval/.get=\l__timechart_minor_tick_interval_year,
      major~tick~interval/.get=\l__timechart_major_tick_interval_year,
    }
```

Start the Ti*k*Z picture and set up the necessary layers.

```
    \tikzpicture
    \pgfdeclarelayer{grid}
    \pgfdeclarelayer{labelbg}
    \pgfsetlayers{grid,labelbg,main}
```

Store the line width of the grid and axis, treating them as 0 pt if they are disabled.

```
    \bool_if:NTF\l__timechart_grid_bool
      {
        \__timechart_set_style_line_width:nn
          {\l__timechart_grid_line_width}
          {/timechart/grid}
      }
      { \pgfmathsetlengthmacro{\l__timechart_grid_line_width}{0} }
    \bool_if:NTF\l__timechart_axis_bool
      {
        \__timechart_set_style_line_width:nn
          {\l__timechart_axis_line_width}
          {/timechart/axis~line}
        \__timechart_set_style_line_width:nn
          {\l__timechart_major_tick_line_width}
          {/timechart/major~tick}
        \__timechart_set_style_line_width:nn
          {\l__timechart_minor_tick_line_width}
          {/timechart/minor~tick}
      }
      {
        \pgfmathsetlengthmacro{\l__timechart_axis_line_width}{0}
        \pgfmathsetlengthmacro{\l__timechart_major_tick_line_width}{0}
        \pgfmathsetlengthmacro{\l__timechart_minor_tick_line_width}{0}
      }
```

Store the start and finish years (ignoring circa, month, day), and then set up the conversion from years to x coordinates. `\l__timechart_x` is the x-distance corresponding to one year, and `yeartox` is the `pgfmath` function that does the conversion.

```
\__timechart_parse_date:NNn\l_tmpa_bool\l__timechart_start_year{#2}
\__timechart_parse_date:NNn\l_tmpa_bool\l__timechart_finish_year{#3}
\pgfmathsetmacro{\l__timechart_start_year}
  {floor(\l__timechart_start_year)}
\pgfmathsetmacro{\l__timechart_finish_year}
  {floor(\l__timechart_finish_year)}
\pgfmathsetmacro{\l__timechart_x}
  {
    (
      \l__timechart_width_pgf
      - max(
        \l__timechart_grid_line_width,
        \l__timechart_axis_line_width,
        \l__timechart_major_tick_line_width,
        \l__timechart_minor_tick_line_width
      )
    )/(\l__timechart_finish_year-\l__timechart_start_year)
  }
\pgfkeys{
  /pgf/declare~function={
    yeartox(\n)=\l__timechart_x*(\n-\l__timechart_start_year);
  },
}
```

Calculate the start and finish x coordinates.

```
\pgfmathsetmacro{\l__timechart_start_x}
  {yeartox(\l__timechart_start_year)}
\pgfmathsetmacro{\l__timechart_finish_x}
  {yeartox(\l__timechart_finish_year)}
\pgfmathsetmacro{\l__timechart_start_tolerance_x}{
  \l__timechart_start_x-(\l__timechart_tolerance_pgf)
}
\pgfmathsetmacro{\l__timechart_finish_tolerance_x}{
  \l__timechart_finish_x+(\l__timechart_tolerance_pgf)
}
\pgfmathsetmacro{\l__timechart_start_beyond_x}{
  \l__timechart_start_x-(\l__timechart_beyond_length_pgf)
}
\pgfmathsetmacro{\l__timechart_finish_beyond_x}{
  \l__timechart_finish_x+(\l__timechart_beyond_length_pgf)
}
```

Set up tracking of current y coordinate.

```
    \pgfmathsetmacro{\l__timechart_current_y}{0}
    \pgfmathsetmacro{\l__timechart_saved_y}{0}
    \pgfmathsetmacro{\l__timechart_auto_reset_minimum_y}{-16000pt}
    \pgfmathsetmacro{\l__timechart_auto_reset_maximum_y}{16000pt}
```

Calculate some years used in loops.

```
    \pgfmathsetmacro{\l__timechart_start_plus_year}{
      \l__timechart_start_year+\l__timechart_minor_tick_interval_year
    }
    \pgfmathsetmacro{\l__timechart_start_plusplus_year}{
      \l__timechart_start_year+(2*\l__timechart_minor_tick_interval_year)
    }
    \pgfmathsetmacro{\l__timechart_end_minus_year}{
      \l__timechart_finish_year-\l__timechart_minor_tick_interval_year
    }
```

Begin a group and make available the user commands \timechart.... (The group will be ended by __timechart_main_end_user:.)

```
    \group_begin:
    \cs_set_eq:NN\timechartinterval\__timechart_interval_user:Ommm
    \cs_set_eq:NN\timecharttext\__timechart_text_user:Omm
    \cs_set_eq:NN\timechartspace\__timechart_space_user:O
    \cs_set_eq:NN\timechartsety\__timechart_set_y_user:m
    \cs_set_eq:NN\timechartsavey\__timechart_save_y_user:
    \cs_set_eq:NN\timechartresety\__timechart_reset_y_user:
    \cs_set_eq:NN\timechartsetyminimumautoreset
      \__timechart_set_y_minimum_auto_reset_user:m
    \cs_set_eq:NN\timechartsetymaximumautoreset
      \__timechart_set_y_maximum_auto_reset_user:m
    \cs_set_eq:NN\timechartstepy\__timechart_step_y_user:O
    \cs_set_eq:NN\timechartfinish\__timechart_main_end_user:
  }
```

(*End of definition for* __timechart_main_begin:nnn*.*)

__timechart_main_end:

Make sure the chart is complete and end the Ti*k*Z picture. __timechart_main_end_user: ends the group begun by __timechart_main_begin:nnn, so whether the user has *not* called it (as \timechartfinish) is equivalent to it being equal to \timechartfinish.

```
\cs_new:Npn\__timechart_main_end:
  {
    \cs_if_eq:NNT\timechartfinish\__timechart_main_end_user:
      { \__timechart_main_end_user: }
    \endtikzpicture
  }
```

(*End of definition for* __timechart_main_end:*.*)

__timechart_main_end_user:

End the group begun by __timechart_main_begin:nnn, draw the axis and grid, and set the bounding box. This macro is made available as \timechartfinish inside the timechart environment.

```
\cs_new:Npn\__timechart_main_end_user:
  {
```

The aim here is to set the bounding box (1) to fit horizontally the axis *not* including labels and the grid and (2) to fit vertically the axis including labels and the grid . All the 'horizonatal' data is already known, and the 'vertical' data is determined by the *current* bounding box. So extract the 'vertical' data and then reset the bounding box.

```
    \pgfextracty{\l__timechart_tmpa_dim}
      { \pgfpointanchor{current~bounding~box}{south} }
    \pgfextracty{\l__timechart_tmpb_dim}
      { \pgfpointanchor{current~bounding~box}{north} }
    \pgfresetboundingbox
```

If the timechart is empty, then the extracted y coordinates of 'north' and 'south' anchors of the bounding box will be $-16\,000$ pt and $16\,000$ pt respectively. Test for this and treat them as both having y coordinate 0 pt in this case.

```
    \dim_compare:nNnTF{\l__timechart_tmpa_dim}>{\l__timechart_tmpb_dim}
      {
        \pgfmathsetmacro{\l__timechart_content_bottom_y}{0pt}
        \pgfmathsetmacro{\l__timechart_content_top_y}{0pt}
      }
      {
        \pgfmathsetmacro{\l__timechart_content_bottom_y}
          {\l__timechart_tmpa_dim}
        \pgfmathsetmacro{\l__timechart_content_top_y}
          {\l__timechart_tmpb_dim}
      }
```

Now draw the grid and axis if necessary and set the bounding box if not.

```
    \bool_if:NTF{\l__timechart_grid_bool}
      {
        \__timechart_grid_draw:
        \pgfmathsetmacro{\l__timechart_axis_y}
          {
            \l__timechart_content_top_y
            + \l__timechart_grid_top_ysep_pgf
            + \pgfkeysvalueof{/timechart/axis~ysep}
          }
      }
      {
        \__timechart_nogrid_bounding_box_set:
        \pgfmathsetmacro{\l__timechart_axis_y}
          {
            \l__timechart_content_top_y
            +\pgfkeysvalueof{/timechart/axis~ysep}
          }
      }
    \bool_if:NT{\l__timechart_axis_bool}
      { \__timechart_axis_draw: }
```

Finally, end the group begun by `\__timechart_main_begin:nnn`.

```
    \group_end:
  }
```

(*End of definition for* `\__timechart_main_end_user:`.)

14.9 Grid drawing

__timechart_grid_draw: Draw the grid of the chart, assuming that the y coordinates of the top and bottom of the content have been calculated and stored in \l__timechart_content_top_y and \l__timechart_content_bottom_y.

```
\cs_new:Npn\__timechart_grid_draw:
  {
    \pgfmathsetmacro{\l__timechart_grid_bottom_y}{
      \l__timechart_content_bottom_y-\l__timechart_grid_bottom_ysep_pgf
    }
    \pgfmathsetmacro{\l__timechart_grid_top_y}{
      \l__timechart_content_top_y+\l__timechart_grid_top_ysep_pgf
    }
    \pgfonlayer{ grid }
    \scope[/timechart/grid]
    \foreach \year in {
      \l__timechart_start_plus_year,
      \l__timechart_start_plusplus_year,
      ...,
      \l__timechart_end_minus_year
    } {
      \group_begin:
```

Only draw gridlines at major ticks.

```
      \__timechart_pgfmathsetbool:nn{\l__timechart_tmpa_bool}
        { Mod(\year,\l__timechart_major_tick_interval_year)==0 }
      \bool_if:NT\l__timechart_tmpa_bool
        {
          \pgftransformshift{ \pgfpoint{yeartox(\year)}{0} }
          \pgfpathmoveto{ \pgfpoint{0}{\l__timechart_grid_top_y} }
          \pgfpathlineto{ \pgfpoint{0}{\l__timechart_grid_bottom_y} }
          \pgfusepath{ draw }
        }
      \group_end:
    }
```

Define and draw the grid node.

```
    \__timechart_make_rectangle_node:nnnn
      { \pgfpoint{\l__timechart_start_x}{\l__timechart_grid_bottom_y} }
      { \pgfpoint{\l__timechart_finish_x}{\l__timechart_grid_top_y} }
      { grid }
      { \c_true_bool }
    \endscope
    \endpgfonlayer
  }
```

(*End of definition for* __timechart_grid_draw:*.*)

14.10 Axis drawing

__timechart_axis_draw: Draw the axis, with large/small ticks and labels on appropriate years, assuming that the y coordinate has been calculated and stored in \l__timechart_axis_y.

```
\cs_new:Npn\__timechart_axis_draw:
  {
    \group_begin:
```

```
    \pgfkeys{
      /timechart/minor~tick~length/.get=\__timechart_minor_tick_length_pgf,
      /timechart/major~tick~length/.get=\__timechart_major_tick_length_pgf,
    }
```

Shift to the y coordinate of the axis line.

```
    \pgftransformshift{ \pgfpoint{0}{\l__timechart_axis_y} }
```

Work out the first and last years which will have a major tick (since these are marked with the era).

```
    \pgfmathsetmacro{\__timechart_start_major_tick_year}
      {
        \l__timechart_start_year
        -Mod(
          \l__timechart_start_year,
          \l__timechart_major_tick_interval_year
        )
      }
    \pgfmathsetmacro{\__timechart_start_major_tick_year}
      {
        ifthenelse(
          \__timechart_start_major_tick_year<\l__timechart_start_year,
          \__timechart_start_major_tick_year
            +\l__timechart_major_tick_interval_year,
          \__timechart_start_major_tick_year
        )
      }
    \pgfmathsetmacro{\__timechart_finish_major_tick_year}
      {
        \l__timechart_finish_year
        -Mod(
          \l__timechart_finish_year,
          \l__timechart_major_tick_interval_year
        )
      }
```

Loop over years and draw the appropriate ticks.

```
    \foreach \year in {
      \l__timechart_start_year,
      \l__timechart_start_plus_year,
      ...,
      \l__timechart_finish_year
    } {
      \pgfmathsetmacro{\x}{yeartox(\year)}
      \__timechart_pgfmathsetbool:nn
        {\l__timechart_tmpa_bool}
        { Mod(\year,\l__timechart_major_tick_interval_year)==0 }
      \bool_if:NTF\l__timechart_tmpa_bool
        { \__timechart_axis_draw_labelled_major_tick:NN\x\year }
        { \__timechart_axis_draw_minor_tick:N\x }
    }
    \__timechart_draw_axis_line
```

Define the axis node.

```
    \pgfextracty{\l__timechart_tmpa_dim}
      { \pgfpointanchor{current~bounding~box}{north} }
    \__timechart_make_rectangle_node:nnnn
      { \pgfpoint{\l__timechart_start_x}{0} }
      { \pgfpoint{\l__timechart_finish_x}{\l__timechart_tmpa_dim} }
      { axis }
      { \c_false_bool }
    \group_end:
  }
```

(*End of definition for* `\__timechart_axis_draw:`.)

All the remaining axis-related macros (which begin `\__timechart_axis_draw_`) assume that a transformation has been applied so that the axis line is at $y = 0$.

__timechart_axis_draw_minor_tick:N

Draw an unlabelled tick at x coordinate `#1`.

```
\cs_new:Npn\__timechart_axis_draw_minor_tick:N #1
  {
    \scope[/timechart/minor~tick]
    \pgfpathmoveto{ \pgfpoint{#1}{0} }
    \pgfpathlineto{ \pgfpoint{#1}{\__timechart_minor_tick_length_pgf} }
    \pgfusepath{draw}
    \endscope
  }
```

(*End of definition for* `\__timechart_axis_draw_minor_tick:N`.)

_timechart_axis_draw_labelled_major_tick:NN

Draw a labelled major tick at x coordinate `#1`, with label for year `#2`, using the special epoch marker if the year is 0, and showing the era if and only if the year is for the first or last major tick. This macro assumes that `\__timechart_start_major_tick_year` and `\__timechart_finish_major_tick_year` have been calculated.

```
\cs_new:Npn\__timechart_axis_draw_labelled_major_tick:NN #1#2
  {
    \__timechart_pgfmathsetbool:nn{\l__timechart_tmpa_bool}{#2==0}
    \bool_if:NTF \l__timechart_tmpa_bool
      { \__timechart_axis_draw_zero_tick:N #1 }
      {
        \__timechart_pgfmathsetbool:nn{\l__timechart_tmpb_bool}
          { #2==\__timechart_start_major_tick_year }
        \__timechart_pgfmathsetbool:nn{\l__timechart_tmpc_bool}
          { #2==\__timechart_finish_major_tick_year }
        \bool_set:Nn\l__timechart_tmpd_bool
          { \l__timechart_tmpb_bool || \l__timechart_tmpc_bool }
        \__timechart_axis_draw_major_tick:N #1
        \__timechart_axis_draw_year_label:nnnnnn
          { #1 }
          { #2 }
          { \l__timechart_tmpd_bool }
          { mid~west }
          { 0 }
          { \__timechart_major_tick_length_pgf+1mm }
      }
  }
```

(*End of definition for* `\__timechart_axis_draw_labelled_major_tick:NN`.)

__timechart_axis_draw_major_tick:N Draw a major tick at x coordinate #1.

```
\cs_new:Npn\__timechart_axis_draw_major_tick:N #1
  {
    \scope[/timechart/major~tick]
    \pgfpathmoveto{ \pgfpoint{#1}{0} }
    \pgfpathlineto{ \pgfpoint{#1}{\__timechart_major_tick_length_pgf} }
    \pgfusepath{draw}
    \endscope
  }
```

(*End of definition for* __timechart_axis_draw_major_tick:N*.*)

__timechart_axis_draw_zero_tick:N Draw a special (labelled) tick for year zero at $x =$ #1.

```
\cs_new:Npn\__timechart_axis_draw_zero_tick:N #1
  {
    \group_begin:
    \pgftransformshift{ \pgfpoint{#1}{0} }
```

The mark is a cross made up of four arcs and the miter joins between them. The parameter \r is the arc radius. The parameter \a is how many degrees should be trimmed from the start/end of a quarter-circle to form each arc. Thus the length of the miter is dependent on \a.

```
    \pgfmathsetmacro{\a}{5}
    \pgfmathsetlengthmacro{\r}{1mm}
    \pgfmathsetlengthmacro{\t}{\r*(cos(\a)-cos(90-\a))/(1-cos(90-\a))}
```

The drawing process is: move to the start of the tick, draw the tick, draw the four arcs, then draw a small part of the tick again. The last step is not mathematically necessary for a smooth join, but ensures that the join *appears* smooth.

```
    \scope[/timechart/major~tick,line~join=miter]
    \pgfpathmoveto{\pgfpointorigin}
    \pgfpathlineto{\pgfpoint{0}{\__timechart_major_tick_length_pgf}}
    \pgfpatharc{0}{90-\a}{\t~and~\r}
    \pgfpatharc{270+\a}{360-\a}{\r}
    \pgfpatharc{180+\a}{270-\a}{\r}
    \pgfpatharc{90+\a}{180-\a}{\t~and~\r}
    \pgfpathlineto{\pgfpoint{0}{\__timechart_major_tick_length_pgf-1pt}}
    \pgfusepath{draw}
    \endscope
```

There is no year 0, so label the zero mark with the 1 before and 1 after the epoch.

```
    \__timechart_axis_draw_year_label:nnnnnn
      { 0 }
      { -1 }
      { \c_true_bool }
      { base~west }
      { -.5mm }
      { 5.5mm }
    \__timechart_axis_draw_year_label:nnnnnn
      { 0 }
      { 1 }
      { \c_true_bool }
      { north~west }
      { .5mm }
      { 5.5mm }
    \group_end:
  }
```

(*End of definition for* `\__timechart_axis_draw_zero_tick:N`.)

`\__timechart_axis_draw_line`

Draw the axis line itself.

```
\cs_new:Npn\__timechart_draw_axis_line
  {
    \scope[/timechart/axis~line]
    \pgfpathmoveto{ \pgfpoint{\l__timechart_start_x}{0} }
    \pgfpathlineto{ \pgfpoint{\l__timechart_finish_x}{0} }
    \pgfusepath{draw}
    \endscope
  }
```

(*End of definition for* `\__timechart_axis_draw_line`.)

`\__timechart_axis_draw_year_label:nnnnnn`

Draw a year label.

#1 : x coordinate.
#2 : Year for label.
#3 : Boolean literal indicating whether the era should be shown.
#4 : Anchor for node.
#5 : x offset (dimension).
#6 : y offset (dimension).

```
\cs_new:Npn\__timechart_axis_draw_year_label:nnnnnn #1#2#3#4#5#6
  {
    \group_begin:
    \pgftransformshift{ \pgfpoint{#1+#5}{#6} }
    \pgfmathtruncatemacro{\absyear}{ abs(#2) }
    \scope[/timechart/major~tick~label]
    \bool_if:NTF #3
      {
        \__timechart_pgfmathsetbool:nn{\l__timechart_tmpa_bool}{#2<0}
        \bool_if:NTF\l__timechart_tmpa_bool
          { \cs_set_eq:NN\__timechart_make_year:n\timechartmakebeforeyear }
          { \cs_set_eq:NN\__timechart_make_year:n\timechartmakeafteryear  }
        \__timechart_hsmash_pgfnode:nnnnn
          {rectangle}
          {#4}
          {\__timechart_make_year:n{\absyear}}
          {}
          {}
      }
      { \pgfnode{rectangle}{#4}{\absyear}{}{} }
    \endscope
    \group_end:
  }
```

(*End of definition for* `\__timechart_axis_draw_year_label:nnnnnn`.)

`\timechartmakebeforeyear`
`\timechartmakeafteryear`

User-redefineable macros to format a year as before or after the epoch.

```
\cs_new:Npn\timechartmakebeforeyear #1
  {
    #1\nobreakspace\textsc{bce}
  }
\cs_new:Npn\timechartmakeafteryear #1
  {
    #1\nobreakspace\textsc{ce}
  }
```

(*End of definition for* `\timechartmakebeforeyear` *and* `\timechartmakeafteryear`*. These functions are documented on page 13.*)

14.11 Bounding box

__timechart_nogrid_bounding_box_set: Set the bounding box when no grid is being drawn.

```
\cs_new:Npn\__timechart_nogrid_bounding_box_set:
  {
    \pgfpathmoveto
      { \pgfpoint{\l__timechart_start_x}{\l__timechart_content_bottom_y} }
    \pgfpathmoveto
      { \pgfpoint{\l__timechart_finish_x}{\l__timechart_content_top_y} }
    \pgfusepath{discard}
  }
```

(*End of definition for* `\__timechart_nogrid_bounding_box_set:`*.*)

14.12 Positioning

__timechart_set_y_user:m Set current y coordinate to `#1`. This macro will be made available as `\timechartsety` in the `timechart` environment.

```
\cs_new:Npn\__timechart_set_y_user:m #1
  {
    \pgfmathsetmacro{\l__timechart_current_y}{#1}
  }
```

(*End of definition for* `\__timechart_set_y_user:m`*.*)

__timechart_save_y_user: Save the current y coordinate. This macro will be made available as `\timechartsavey` in the `timechart` environment.

```
\cs_new:Npn\__timechart_save_y_user:
  {
    \pgfmathsetmacro{\l__timechart_saved_y}{\l__timechart_current_y}
  }
```

(*End of definition for* `\__timechart_save_y_user:`*.*)

__timechart_reset_y_user: Set the current y coordinate to the last saved coordinate. This macro will be made available as `\timechartsresety` in the `timechart` environment.

```
\cs_new:Npn\__timechart_reset_y_user:
  {
    \pgfmathsetmacro{\l__timechart_current_y}{\l__timechart_saved_y}
  }
```

(*End of definition for* `\__timechart_reset_y_user:`*.*)

__timechart_set_y_minimum_auto_reset_user:m

Set a y coordinate below which `\__timechart_step_y_user:` will automatically reset the currrent y coordinate to the last saved y coordinate. This macro will be made available as `\timechartssetyminimumautoreset` in the `timechart` environment.

```
\cs_new:Npn\__timechart_set_y_minimum_auto_reset_user:m #1
  {
    \pgfmathsetmacro{\l__timechart_auto_reset_minimum_y}{#1}
  }
```

(*End of definition for* `\__timechart_set_y_minimum_auto_reset_user:m`.)

__timechart_set_y_maximum_auto_reset_user:m

Set a y coordinate above which `\__timechart_step_y_user:` will automatically reset the currrent y coordinate to the last saved y coordinate. This macro will be made available as `\timechartssetymaximumautoreset` in the `timechart` environment.

```
\cs_new:Npn\__timechart_set_y_maximum_auto_reset_user:m #1
  {
    \pgfmathsetmacro{\l__timechart_auto_reset_maximum_y}{#1}
  }
```

(*End of definition for* `\__timechart_set_y_maximum_auto_reset_user:m`.)

__timechart_step_y_user:O

Increment the current y coordinate by `#1` times the length specified in `/timechart/ystep`. This macro will be made available as `\timechartstepy` in the `timechart` environment.

```
\NewDocumentCommand{\__timechart_step_y_user:O}{ O{1} }
  {
    \pgfmathsetmacro{\l__timechart_current_y}
      {\l__timechart_current_y+#1*\l__timechart_ystep_pgf}
    \__timechart_pgfmathsetbool:nn{\l__timechart_tmpa_bool}
      {
        or(
          \l__timechart_current_y<\l__timechart_auto_reset_minimum_y,
          \l__timechart_current_y>\l__timechart_auto_reset_maximum_y
        )
      }
    \bool_if:nT{\l__timechart_tmpa_bool}
      { \pgfmathsetmacro{\l__timechart_current_y}{\l__timechart_saved_y)} }
  }
```

(*End of definition for* `\__timechart_step_y_user:O`.)

14.13 Bounds checking

__timechart_if_x_in_bounds:nT

Check if x coordinate `#1` is (strictly) within the bounds of the chart; if so, execute `#2`.

```
\cs_new:Npn\__timechart_if_x_in_bounds:nT #1#2
  {
    \__timechart_pgfmathsetbool:nn{\l__timechart_tmpa_bool}{
      and(
        #1>=\l__timechart_start_x,
        #1<=\l__timechart_finish_x
      )
    }
    \bool_if:NT\l__timechart_tmpa_bool
      {#2}
  }

```

(*End of definition for* `\__timechart_if_x_in_bounds:nT`.)

__timechart_if_x_in_tolerance_bounds_x:nT

Check if x coordinate #1 is within the specified tolerance of the bounds of the chart; if so, execute #2.

```
\cs_new:Npn\__timechart_if_x_in_tolerance_bounds:nT #1#2
  {
    \__timechart_pgfmathsetbool:nn{\l__timechart_tmpa_bool}{
      and(
        #1>=\l__timechart_start_tolerance_x,
        #1<=\l__timechart_finish_tolerance_x
      )
    }
    \bool_if:NT\l__timechart_tmpa_bool{#2}
  }
```

(*End of definition for* `\__timechart_if_x_in_tolerance_bounds_x:nT`.)

if_x_range_intersect_tolerance_bounds_x:nnT

Check if the range between x coordinates #1 and #2 intersects the range of the bounds of the chart plus the specified tolerance; if so, execute #3.

```
\cs_new:Npn\__timechart_if_x_range_intersect_tolerance_bounds:nnT #1#2#3
  {
    \__timechart_pgfmathsetbool:nn{\l__timechart_tmpa_bool}{
      or(
        or(
          and(
            #2>=\l__timechart_start_tolerance_x,
            #2<=\l__timechart_finish_tolerance_x
          ),
          and(
            #1>=\l__timechart_start_tolerance_x,
            #1<=\l__timechart_finish_tolerance_x
          )
        ),
        and(
          #1<\l__timechart_start_tolerance_x,
          #2>\l__timechart_finish_tolerance_x
        )
      )
    }
    \bool_if:NT\l__timechart_tmpa_bool
      {#3}
  }
```

(*End of definition for* `\__timechart_if_x_range_intersect_tolerance_bounds_x:nnT`.)

14.14 Date and date range parsing

__timechart_parse_date_or_daterange:NNNNNn

Parse the text in #6, which should represent a date or date range, into parameters #1–#5.
#1 : range indicator boolean variable.
#2 : minimum circa indicator boolean variable.
#3 : minimum variable.
#4 : maximum circa indicator boolean variable.
#5 : maximum variable.
#6 : text to parse.

```
\cs_new:Npn\__timechart_parse_date_or_daterange:NNNNNn #1#2#3#4#5#6
{
  \bool_set:Nn #1 {\__timechart_is_nondaterange_p:w #6/\q_stop}
  \bool_set_inverse:N #1
  \bool_if:nTF #1
    { \__timechart_parse_range:w #2#3#4#5\q_mark #6\q_stop }
    {
      \__timechart_parse_date:NNn #2#3{#6}
      \bool_set_eq:NN #4#2
      \pgfmathsetmacro{#5}{#3}
    }
}
```

(*End of definition for* `\__timechart_parse_date_or_daterange:NNNNNn`.)

__timechart_is_nondaterange_p:w

To be called in the form `\__timechart_is_nondaterange_p:w`⟨*text*⟩`/\q_stop`. Return boolean true if and only if ⟨*text*⟩ (known to be either a date or date range) contains a range marker.

```
\cs_new:Npn\__timechart_is_nondaterange_p:w #1/#2\q_stop
{
  \tl_if_empty_p:n{#2}
}
```

(*End of definition for* `\__timechart_is_nondaterange_p:w`.)

__timechart_parse_range:w

To be called in the form `\__timechart_parse_range:w`⟨*cmin*⟩⟨*min*⟩⟨*cmax*⟩⟨*max*⟩`\q_mark`⟨*text*⟩`\q_stop`. Parse ⟨*text*⟩ (known to represent a date range) into minimum circa indicator boolean variable ⟨*cmin*⟩, minimum variable ⟨*min*⟩, maximum circa indicator boolean variable ⟨*cmax*⟩, maximum variable ⟨*max*⟩.

```
\cs_new:Npn\__timechart_parse_range:w #1#2#3#4\q_mark #5/#6\q_stop
{
  \__timechart_parse_date:NNn #1#2{#5}
  \__timechart_parse_date:NNn #3#4{#6}
}
```

(*End of definition for* `\__timechart_parse_range:w`.)

__timechart_parse_date:NNn

Parse text (known to represent a date) into the supplied variables. Parameters `#1` and `#2` are the variables for (respectively) circa indicator boolean and date, and `#3` is the text to be parsed:

`#1` : circa indicator boolean variable.
`#2` : date variable.
`#3` : text to parse.

```
\cs_new:Npn\__timechart_parse_date:NNn #1#2#3
{
  \bool_set:Nn #1 { \__timechart_is_circa_p:w #3c\q_stop }
  \bool_if:NTF #1
    { \__timechart_parse_circa_date:w #2\q_mark #3\q_stop }
    { \__timechart_parse_noncirca_date:Nn #2{#3} }
}
```

(*End of definition for* `\__timechart_parse_date:NNn`.)

__timechart_is_circa_p:w

To be called in the form __timechart_is_circa_p:w⟨*text*⟩c\q_stop. Return boolean true if and only if ⟨*text*⟩ (known to be either a date or a date with a circa indicator) has a circa indicator.

```
\cs_new:Npn\__timechart_is_circa_p:w #1c#2\q_stop
{
  \tl_if_empty_p:n{#1}
}
```

(*End of definition for __timechart_is_circa_p:w.*)

__timechart_parse_circa_date:w

To be called in the form __timechart_parse_circa_date:w⟨*var*⟩\q_mark⟨*text*⟩\q_stop. Parse ⟨*text*⟩ (known to represent a circa date) into the supplied variable. #1 is the variable for the date and #2 is the text to be parsed.

```
\cs_new:Npn\__timechart_parse_circa_date:w #1\q_mark c#2\q_stop
{
  \__timechart_parse_noncirca_date:Nn #1{#2}
}
```

(*End of definition for __timechart_parse_circa_date:w.*)

__timechart_parse_noncirca_date:w

To be called in the form __timechart_parse_noncirca_date:w⟨*var*⟩\q_mark⟨*text*⟩ \q_stop. Parse ⟨*text*⟩ (known to represent a non-circa date) into the supplied variable. #1 is the variable for the date and #2 is the text to be parsed.

```
\cs_new:Npn\__timechart_parse_noncirca_date:Nn #1#2
{
  \bool_if:nTF { \__timechart_is_before_p:w #2-\q_stop }
    { \__timechart_parse_before_date:w #1\q_mark #2\q_stop }
    { \__timechart_parse_signed_date:w #1\q_mark #2-0-0\q_stop }
}
```

(*End of definition for __timechart_parse_noncirca_date:w.*)

__timechart_is_before_p:w

To be called in the form __timechart_is_before_p:w⟨*text*⟩-\q_stop. Return boolean true if and only if ⟨*text*⟩ (known to be a date without a circa indicator) begins with a -.

```
\cs_new:Npn\__timechart_is_before_p:w #1-#2\q_stop
{
  \tl_if_empty_p:n{#1}
}
```

(*End of definition for __timechart_is_before_p:w.*)

__timechart_parse_before_date:w

To be called in the form __timechart_parse_before_date:w⟨*var*⟩\q_mark⟨*text*⟩\q_stop. Parse ⟨*text*⟩ (known to represent a date with a leading -) into the supplied variable. #1 is the variable for the date and #2 is the text to be parsed.

```
\cs_new:Npn\__timechart_parse_before_date:w #1\q_mark-#2\q_stop
{
  \__timechart_parse_signed_date:w #1-\q_mark #2-0-0\q_stop
}
```

(*End of definition for* `\__timechart_parse_before_date:w`*.*)

Now comes that actual parsing of an ISO-format date `YYYY-MM-DD`. The following macros serve as lookup tables for the number of days in the n-th month and the number of days in the year up to the start of the n-th month.

```
\cs_new:cpn{c__timechart_year_days_pgf}{365}
\cs_new:cpn{c__timechart_month_days_1_pgf}{31}
\cs_new:cpn{c__timechart_month_days_2_pgf}{28}
\cs_new:cpn{c__timechart_month_days_3_pgf}{31}
\cs_new:cpn{c__timechart_month_days_4_pgf}{30}
\cs_new:cpn{c__timechart_month_days_5_pgf}{31}
\cs_new:cpn{c__timechart_month_days_6_pgf}{30}
\cs_new:cpn{c__timechart_month_days_7_pgf}{31}
\cs_new:cpn{c__timechart_month_days_8_pgf}{31}
\cs_new:cpn{c__timechart_month_days_9_pgf}{30}
\cs_new:cpn{c__timechart_month_days_10_pgf}{31}
\cs_new:cpn{c__timechart_month_days_11_pgf}{30}
\cs_new:cpn{c__timechart_month_days_12_pgf}{31}
\cs_new:cpn{c__timechart_cumulative_days_1_pgf}{0}
\cs_new:cpn{c__timechart_cumulative_days_2_pgf}{31}
\cs_new:cpn{c__timechart_cumulative_days_3_pgf}{59}
\cs_new:cpn{c__timechart_cumulative_days_4_pgf}{90}
\cs_new:cpn{c__timechart_cumulative_days_5_pgf}{120}
\cs_new:cpn{c__timechart_cumulative_days_6_pgf}{151}
\cs_new:cpn{c__timechart_cumulative_days_7_pgf}{181}
\cs_new:cpn{c__timechart_cumulative_days_8_pgf}{212}
\cs_new:cpn{c__timechart_cumulative_days_9_pgf}{243}
\cs_new:cpn{c__timechart_cumulative_days_10_pgf}{273}
\cs_new:cpn{c__timechart_cumulative_days_11_pgf}{304}
\cs_new:cpn{c__timechart_cumulative_days_12_pgf}{334}
```

`\__timechart_parse_signed_date:w`

To be called in the form `\__timechart_parse_positive_date:w`⟨*var*⟩⟨*sign*⟩`\q_mark` ⟨*text*⟩`-0-0\q_stop`. Parse ⟨*text*⟩ (known to represent a non-circa date) into the supplied variable. `#1` is the variable for the date and `#2` is possibly `-`.

There is a trick in the parsing:

1. If ⟨*text*⟩ has the form ⟨*year*⟩-⟨*month*⟩-⟨*day*⟩, then parameters `#3`, `#4`, and `#5` will be, respectively, ⟨*year*⟩, ⟨*month*⟩, and ⟨*day*⟩`-0-0`. Thus `#5` will be evaulated by pgfmath to ⟨*day*⟩.

2. If ⟨*text*⟩ has the form ⟨*year*⟩-⟨*month*⟩, then parameters `#3`, `#4`, and `#5` will be, respectively, ⟨*year*⟩, ⟨*month*⟩, and `0-0`. Thus `#5` will be evaulated by pgfmath to 0.

3. If ⟨*text*⟩ is simply ⟨*year*⟩, then parameters `#3`, `#4`, and `#5` will be, respectively, ⟨*year*⟩, `0`, and `0`.

```
\cs_new:Npn\__timechart_parse_signed_date:w #1#2\q_mark #3-#4-#5\q_stop
{
  \pgfmathtruncatemacro{\__timechart_parsed_year_pgf}{#2#3}
  \pgfmathtruncatemacro{\__timechart_parsed_month_pgf}{#4}
  \pgfmathtruncatemacro{\__timechart_parsed_day_pgf}{#5}
  \__timechart_pgfmathsetbool:nn{\l_tmpa_bool}{
    or(
      \__timechart_parsed_month_pgf < 1,
      \__timechart_parsed_month_pgf > 12,
  )
}
\bool_if:NTF\l_tmpa_bool
{
```

Case: no valid month is given. Use only the year.

```
  \pgfmathsetmacro{#1}{#2#3}
}
{
```

Case: a valid month is given. Get the number of days in the year, in the month, and in the year up to the month. Then check if the year is a leap year and, if so, make the appropriate adjustments.

```
  \cs_set_eq:NN\l__timechart_year_days_pgf\c__timechart_year_days_pgf
  \cs_set_eq:Nc\l__timechart_month_days_pgf
    { c__timechart_month_days_\__timechart_parsed_month_pgf _pgf }
  \cs_set_eq:Nc\l__timechart_cumulative_days_pgf
    { c__timechart_cumulative_days_\__timechart_parsed_month_pgf _pgf }
  \__timechart_pgfmathsetbool:nn{\l_tmpa_bool}{
    or(
      Mod(\__timechart_parsed_year_pgf,400) == 0,
      and(
        Mod(\__timechart_parsed_year_pgf,4) == 0,
        Mod(\__timechart_parsed_year_pgf,100) != 0
      )
    )
  }
  \bool_if:NT\l_tmpa_bool
    {
      \pgfmathsetmacro{\l__timechart_year_days_pgf}
        { \l__timechart_year_days_pgf+1 }
      \__timechart_pgfmathsetbool:nn{\l_tmpb_bool}
        { \__timechart_parsed_month == 1 }
      \bool_if:NF\l_tmpb_bool
        {
          \__timechart_pgfmathsetbool:nn{\l_tmpb_bool}
            { \__timechart_parsed_month == 2 }
          \bool_if:NF\l_tmpb_bool
            {
              \pgfmathsetmacro{\l__timechart_month_days_pgf}
                { \l__timechart_month_days_pgf + 1 }
            }
            {
              \pgfmathsetmacro{\l__timechart_cumulative_days_pgf}
                { \l__timechart_cumulative_days_pgf + 1 }
            }
        }
    }
  \__timechart_pgfmathsetbool:nn{\l_tmpa_bool}{
    or(
      \__timechart_parsed_day_pgf < 1,
      \__timechart_parsed_day_pgf > \l__timechart_month_days_pgf,
    )
  }
  \bool_if:NTF\l_tmpa_bool
```

```
      {
```

Sub-case: no valid day is given. Use only the year and month.

```
       \pgfmathsetmacro{#1}
         {
           #2#3
           + \l__timechart_cumulative_days_pgf/\l__timechart_year_days_pgf
         }
      }
      {
```

Sub-case: a valid day is given. Use the year, month, and day.

```
       \pgfmathsetmacro{#1}
         {
           #2#3
           + (
               \l__timechart_cumulative_days_pgf
               + \__timechart_parsed_day_pgf
             )/\l__timechart_year_days_pgf
         }
      }
  }
}
```

(*End of definition for* `\__timechart_parse_signed_date:w`.)

14.15 Interval drawing

14.15.1 Preliminaries

\l__timechart_start_is_range_bool
\l__timechart_startmin_circa_bool
\l__timechart_startmax_circa_bool
\l__timechart_finish_is_range_bool
\l__timechart_finishmin_circa_bool
\l__timechart_finishmax_circa_bool

These boolean variables will be used to hold parsed data for the start and finish of an interval: whether it is a range, whether the beginning of that range is qualified by 'circa', and whether the end of that range is qualified by 'circa'.

```
\bool_new:N\l__timechart_start_is_range_bool
\bool_new:N\l__timechart_startmin_circa_bool
\bool_new:N\l__timechart_startmax_circa_bool
\bool_new:N\l__timechart_finish_is_range_bool
\bool_new:N\l__timechart_finishmin_circa_bool
\bool_new:N\l__timechart_finishmax_circa_bool
```

(*End of definition for* `\l__timechart_start_is_range_bool` *and others.*)

14.15.2 Error message definition

```
\msg_new:nnn{timechart}{interval_dates_invalid}
  { Invalid~interval~dates:~#1~to~#2 }
```

14.15.3 Main macros

__timechart_interval_user:Ommm

Draw an interval. This macro will be made available as `\timechartinterval` inside the `timechart` environment.

`#1` : PGF keys under `/timechart/` to apply.
`#2` : Start year.
`#3` : Finish year.
`#4` : Label.

```
\NewDocumentCommand{\__timechart_interval_user:Ommm}{ O{} m m m }
  {
    \group_begin:
```

Parse the start and finish dates or date ranges.

```
    \__timechart_parse_date_or_daterange:NNNNNn
      \l__timechart_start_is_range_bool
      \l__timechart_startmin_circa_bool\l__timechart_startmin_year
      \l__timechart_startmax_circa_bool\l__timechart_startmax_year
      {#2}
    \__timechart_parse_date_or_daterange:NNNNNn
      \l__timechart_finish_is_range_bool
      \l__timechart_finishmin_circa_bool\l__timechart_finishmin_year
      \l__timechart_finishmax_circa_bool\l__timechart_finishmax_year
      {#3}
```

Check the results of parsing and only proceed if they are valid.

```
    \__timechart_pgfmathsetbool:nn{\l_tmpa_bool}{
      and(
        \l__timechart_startmin_year <= \l__timechart_startmax_year,
        and(
          \l__timechart_startmax_year <= \l__timechart_finishmin_year,
          \l__timechart_finishmin_year <= \l__timechart_finishmax_year
        )
      )
    }
    \bool_if:NTF\l_tmpa_bool
      {
        \__timechart_interval_checked:nn{#1}{#4}
        \group_end:
        \bool_if:NT\l__timechart_autostep_bool
          { \__timechart_step_y_user:O }
      }
      {
        \msg_error:nnnn{timechart}{interval_dates_invalid}{#2}{#3}
        \group_end:
      }
  }
```

(*End of definition for* `\__timechart_interval_user:Ommm`.)

`\__timechart_interval_checked:nn`

Draw an interval using the parsed and checked years or year ranges.
`#1` : PGF keys under `/timechart/` to apply.
`#2` : Label.

```
\cs_new:Npn \__timechart_interval_checked:nn #1#2
  {
```

Process keys supplied locally and retrieve the only value needed at this stage.

```
    \pgfqkeys{/timechart}{
      #1,
      circa~uncertainty/.get=\l__timechart_circa_uncertainty_year
    }
```

Do the minimum amount of calculation necessary to check whether any part of interval is visible.

```
    \bool_if:NTF\l__timechart_startmin_circa_bool
      {
        \pgfmathsetmacro{\l__timechart_start_extreme_x}
          { yeartox(\l__timechart_startmin_year
            - \l__timechart_circa_uncertainty_year) }
      }
      {
        \pgfmathsetmacro{\l__timechart_start_extreme_x}
          { yeartox(\l__timechart_startmin_year) }
      }
    \bool_if:NTF\l__timechart_finishmax_circa_bool
      {
        \pgfmathsetmacro{\l__timechart_finish_extreme_x}
          { yeartox(\l__timechart_finishmax_year
            + \l__timechart_circa_uncertainty_year) }
      }
      {
        \pgfmathsetmacro{\l__timechart_finish_extreme_x}
          { yeartox(\l__timechart_finishmax_year) }
      }
```

Draw the interval if some part of it is visible.

```
    \__timechart_if_x_range_intersect_tolerance_bounds:nnT
      {\l__timechart_start_extreme_x}{\l__timechart_finish_extreme_x}
      { \__timechart_draw_visible_interval:nn{#1}{#2} }
  }
```

(*End of definition for* `\__timechart_interval_checked:nn`*.*)

__timechart_draw_visible_interval:nn

Draw an interval of which some part is known to be visible.

#1 : PGF keys

#2 : Label

```
\cs_new:Npn\__timechart_draw_visible_interval:nn #1#2
  {
```

Retrieve PGF key values.

```
    \pgfqkeys{/timechart}{
      ref/.get=\l__timechart_ref_text,
      mark/.get=\l__timechart_mark_text,
      interval~minimum~width/.get=\l__timechart_minimum_width_pgf,
      interval~bar~color/.get=\l__timechart_bar_color,
      interval~bar~thickness/.get=\l__timechart_bar_thickness_pgf,
      interval~mark~color/.get=\l__timechart_mark_color,
      interval~label~baseline/.get=\l__timechart_text_baseline_pgf,
      interval~label~node~name/.get=\l__timechart_interval_label_node_name,
    }
```

Do the remaining calculations.

```
    \pgfmathsetlengthmacro{\l__timechart_bar_half_thickness_pgf}
      {.5*\l__timechart_bar_thickness_pgf}
    \cs_set:Npn\l__timechart_label_text{#2}
    \bool_if:NTF\l__timechart_startmax_circa_bool
      {
        \pgfmathsetmacro{\l__timechart_start_definite_x}
          { yeartox(\l__timechart_startmax_year
            + \l__timechart_circa_uncertainty_year) }
      }
      {
        \pgfmathsetmacro{\l__timechart_start_definite_x}
          { yeartox(\l__timechart_startmax_year) }
      }
    \bool_if:NTF\l__timechart_finishmin_circa_bool
      {
        \pgfmathsetmacro{\l__timechart_finish_definite_x}
          { yeartox(\l__timechart_finishmin_year
            - \l__timechart_circa_uncertainty_year) }
      }
      {
        \pgfmathsetmacro{\l__timechart_finish_definite_x}
          { yeartox(\l__timechart_finishmin_year) }
      }
```

It is possible that circa indicators have made `\l__timechart_start_definite_x` greater than `\l__timechart_finish_definite_x`. Check for this; if so, set them both to their average.

```
    \__timechart_pgfmathsetbool:nn{\l__timechart_tmpa_bool}
      {
        \l__timechart_start_definite_x > \l__timechart_finish_definite_x
      }
    \bool_if:NT\l__timechart_tmpa_bool
      {
        \pgfmathsetmacro{\l__timechart_start_definite_x}
          {
            .5*(
              \l__timechart_start_definite_x
              + \l__timechart_finish_definite_x
            )
          }
        \pgfmathsetmacro{\l__timechart_finish_definite_x}
          { \l__timechart_start_definite_x }
      }
```

Calculate whether it is necessary to make an adjustment to ensure that the minimum width requirement is satisfied, and store in `\l__timechart_tmpa_bool`.

```
    \__timechart_pgfmathsetbool:nn{\l__timechart_tmpa_bool}
      {
        (\l__timechart_finish_extreme_x-\l__timechart_start_extreme_x)
        < \l__timechart_minimum_width_pgf
      }
```

Calculate whether it is necessary to make an adjustment to prevent a rendering glitch when there are two fadings and a zero-width (or very small) definite part, and store in `\l__timechart_tmpb_bool`. Note the quick test for both ranges being fadings.

```
    \int_if_zero:nTF
      {
        \l__timechart_start_range_type_int
        + \l__timechart_finish_range_type_int
      }
      {
        \__timechart_pgfmathsetbool:nn{\l__timechart_tmpb_bool}
          {
            (
              \l__timechart_finish_definite_x
              - \l__timechart_start_definite_x
            )
            < \l__timechart_minimum_width_pgf
          }
      }
      { \bool_set_false:N\l__timechart_tmpb_bool }
```

Make the adjustment if necessary.

```
    \bool_if:nT{
      \l__timechart_tmpa_bool || \l__timechart_tmpb_bool
    }{
      \pgfmathsetmacro{\l__timechart_width_adjust}
        { .5*\l__timechart_minimum_width_pgf }
      \pgfmathsetmacro{\l__timechart_start_definite_x}
        { \l__timechart_start_definite_x-\l__timechart_width_adjust }
      \pgfmathsetmacro{\l__timechart_finish_definite_x}
        { \l__timechart_finish_definite_x+\l__timechart_width_adjust }
      \pgfmathsetmacro{\l__timechart_start_extreme_x}
        {
          min(
            \l__timechart_start_extreme_x,
            \l__timechart_start_definite_x
          )
        }
      \pgfmathsetmacro{\l__timechart_finish_extreme_x}
        {
          max(
            \l__timechart_finish_extreme_x,
            \l__timechart_finish_definite_x
          )
        }
    }
    \pgfmathsetmacro{\l__timechart_start_solid_x}
      { \l__timechart_start_definite_x }
    \pgfmathsetmacro{\l__timechart_finish_solid_x}
      { \l__timechart_finish_definite_x }
```

All the data needed to draw the interval are now ready. Shift to the correct vertical coordinate and open a scope for drawing.

```
    \pgftransformshift{ \pgfpoint{0}{\l__timechart_current_y} }
    \pgfscope
```

First, do the necessary clipping if the interval extends beyond the specified tolerance from the chart, then process the ranges, then draw the solid part of the bar, then do the labelling, then define the node for the bar. Note that ranges wth the ‘slant’ style function by adding a new clipping path and extending the solid part, so that there is only *one* solid rectangle drawn.

```
    \__timechart_interval_beyond_clip:
    \int_case:nn {\l__timechart_start_range_type_int}
      {
        {0}{ \__timechart_interval_start_range_fade: }
        {1}{ \__timechart_interval_start_range_slant: }
      }
    \int_case:nn {\l__timechart_finish_range_type_int}
      {
        {0}{ \__timechart_interval_finish_range_fade: }
        {1}{ \__timechart_interval_finish_range_slant: }
      }
    \__timechart_interval_draw_solid:
    \__timechart_interval_mark:
    \__timechart_interval_define_bar_node:
    \endpgfscope
    \__timechart_interval_label:
  }
```

(*End of definition for* `__timechart_draw_visible_interval:nn`*.*)

14.15.4 Clipping

\__timechart_interval_beyond_clip: If the interval bar extends beyond the specified tolerance from the chart, clip it appropriately. To avoid repeated computation, use `\l__timechart_tmpa_bool` and `\l__timechart_tmpb_bool` to store whether it is necessary to clip on the left and right (respectively). If both are false, there is no need for any clipping.

```
\cs_new:Npn\__timechart_interval_beyond_clip:
  {
    \__timechart_pgfmathsetbool:nn{\l__timechart_tmpa_bool}
      {\l__timechart_start_extreme_x<=\l__timechart_start_tolerance_x}
    \__timechart_pgfmathsetbool:nn{\l__timechart_tmpb_bool}
      {\l__timechart_finish_extreme_x>=\l__timechart_finish_tolerance_x}
    \bool_if:nT{ \l__timechart_tmpa_bool || \l__timechart_tmpb_bool }
      { \__timechart_interval_beyond_clip_aux: }
  }
```

(*End of definition for* `\__timechart_interval_beyond_clip:`*.*)

\__timechart_interval_beyond_clip_aux: This macro does the actual clipping. The clipping path starts at the north-east and proceeds clockwise.

```
\cs_new:Npn\__timechart_interval_beyond_clip_aux:
  {
    \pgfinterruptboundingbox
    \bool_if:NTF\l__timechart_tmpb_bool
      {
        \pgfpathmoveto{
          \pgfpoint
            {\l__timechart_finish_beyond_x}
            {\l__timechart_bar_thickness_pgf}
        }
        \pgfpathlineto{
          \pgfpoint
            {\l__timechart_finish_beyond_x}
            {\l__timechart_bar_half_thickness_pgf}
        }
        \pgfpatharc
          {90}
        {270}
        {\l__timechart_beyond_x_radius_pgf
          ~and~\l__timechart_bar_half_thickness_pgf}
      \pgfpathlineto{
        \pgfpoint
          {\l__timechart_finish_beyond_x}
          {-\l__timechart_bar_thickness_pgf}
      }
    }
    {
      \pgfpathmoveto{
        \pgfpoint
          {\l__timechart_right_nonclip_x}
          {\l__timechart_bar_thickness_pgf}
      }
      \pgfpathlineto{
        \pgfpoint
          {\l__timechart_right_nonclip_x}
          {-\l__timechart_bar_thickness_pgf}
      }
    }
  \bool_if:NTF\l__timechart_tmpa_bool
    {
      \pgfpathlineto{
        \pgfpoint
          {\l__timechart_start_beyond_x}
          {-\l__timechart_bar_thickness_pgf}
      }
      \pgfpathlineto{
        \pgfpoint
          {\l__timechart_start_beyond_x}
          {-\l__timechart_bar_half_thickness_pgf}
      }
      \pgfpatharc
        {-90}
        {90}
        {\l__timechart_beyond_x_radius_pgf
          ~and~\l__timechart_bar_half_thickness_pgf}
      \pgfpathlineto{
        \pgfpoint
          {\l__timechart_start_beyond_x}
          {\l__timechart_bar_thickness_pgf}
      }
    }
    {
      \pgfpathlineto{
        \pgfpoint
          {\l__timechart_left_nonclip_x}
          {-\l__timechart_bar_thickness_pgf}
      }
      \pgfpathlineto{
        \pgfpoint
          {\l__timechart_left_nonclip_x}
          {\l__timechart_bar_thickness_pgf}
        }
      }
    \pgfpathclose
    \pgfusepath{clip}
    \endpgfinterruptboundingbox
  }
```

(*End of definition for* `\__timechart_interval_beyond_clip_aux:`*.*)

14.15.5 Fading in/out

Set up the PGF 'in' fading.

```
\pgfdeclarehorizontalshading{__timechart_fade_in_shading}{4bp}{
   color(0bp)=(transparent!100);
   color(1bp)=(transparent!100);
   color(3bp)=(transparent!0);
   color(4bp)=(transparent!0)
}
\pgfdeclarefading
  {__timechart_fade_in}
  {\pgfuseshading{__timechart_fade_in_shading}}
```

__timechart_interval_start_range_fade:

Draw fading to indicate a start range. Because some PDF renderers may otherwise leave a gap between the fade and solid part, compute `\l__timechart_fade_extra_pgf` and draw this amount of overlap with the solid part. Because of the definition of the 'in' fading, `\l__timechart_fade_extra_pgf` must not exceed half of the fade width. And the overlap should not exceed the actual length of the solid part.

```
\cs_new:Npn\__timechart_interval_start_range_fade:
  {
    \pgfmathsetlengthmacro{\l__timechart_fade_width_pgf}
      { \l__timechart_start_solid_x-\l__timechart_start_extreme_x }
    \__timechart_pgfmathsetbool:nn{\l__timechart_tmpa_bool}
      { \l__timechart_fade_width_pgf
        > \pgfkeysvalueof{/timechart/fade~minimum~width} }
    \bool_if:NT \l__timechart_tmpa_bool
      {
        \__timechart_if_x_range_intersect_tolerance_bounds:nnT
          {\l__timechart_start_extreme_x}{\l__timechart_start_solid_x}
          {
            \pgfmathsetmacro{\l__timechart_fade_extra_pgf}
            {
              min(
                .49*\l__timechart_fade_width_pgf,
                \l__timechart_finish_solid_x-\l__timechart_start_solid_x
              )
            }
            \pgfscope
            \pgfpathrectanglecorners{
              \pgfpoint
                {\l__timechart_start_extreme_x}
                {-\l__timechart_bar_half_thickness_pgf}
            }{
              \pgfpoint
                {\l__timechart_start_solid_x+\l__timechart_fade_extra_pgf}
                {\l__timechart_bar_half_thickness_pgf}
            }
            \pgfgettransform{\__timechart_transform_current}
            \pgfsetfading{__timechart_fade_in}{
              \pgfsettransform{\__timechart_transform_current}
              \pgftransformshift{
                \pgfpoint
                  {
                    .5*\l__timechart_start_extreme_x
                    +.5*\l__timechart_start_solid_x
                  }
                  {0}
              }
              \pgftransformxscale{
                (\l__timechart_start_solid_x-\l__timechart_start_extreme_x)
                /2bp
              }
              \pgftransformyscale{\l__timechart_bar_thickness_pgf/4bp}
            }
            \pgfsetfillcolor{\l__timechart_bar_color}
            \pgfusepath{fill}
            \endpgfscope
          }
    }
  }
```

(*End of definition for* `\__timechart_interval_start_range_fade:`*.*)

Set up the PGF 'out' fading.

```
\pgfdeclarehorizontalshading{__timechart_fade_out_shading}{4bp}{
    color(0bp)=(transparent!0);
   color(1bp)=(transparent!0);
   color(3bp)=(transparent!100);
  color(4bp)=(transparent!100)
}
\pgfdeclarefading
  {__timechart_fade_out}
  {\pgfuseshading{__timechart_fade_out_shading}}
```

`\__timechart_interval_finish_range_fade:`

Draw fading to indicate a finish range. As in `\__timechart_interval_start_range_fade:`, a small overlap is computed.

```
\cs_new:Npn\__timechart_interval_finish_range_fade:
  {
    \pgfmathsetlengthmacro{\l__timechart_fade_width_pgf}
      { \l__timechart_finish_extreme_x-\l__timechart_finish_solid_x }
      \__timechart_pgfmathsetbool:nn{\l__timechart_tmpa_bool}
      { \l__timechart_fade_width_pgf
        > \pgfkeysvalueof{/timechart/fade~minimum~width} }
    \bool_if:NT \l__timechart_tmpa_bool
      {
        \__timechart_if_x_range_intersect_tolerance_bounds:nnT
          {\l__timechart_finish_solid_x}{\l__timechart_finish_extreme_x}
          {
            \pgfmathsetmacro{\l__timechart_fade_extra_pgf}
            {
              min(
                .49*\l__timechart_fade_width_pgf,
                \l__timechart_finish_solid_x-\l__timechart_start_solid_x
              )
            }
            \pgfscope
            \pgfpathrectanglecorners{
              \pgfpoint
                {\l__timechart_finish_solid_x-\l__timechart_fade_extra_pgf}
                {-\l__timechart_bar_half_thickness_pgf}
            }{
              \pgfpoint
                {\l__timechart_finish_extreme_x}
                {\l__timechart_bar_half_thickness_pgf}
            }
            \pgfgettransform{\__timechart_transform_current}
            \pgfsetfading{__timechart_fade_out}{
              \pgfsettransform{\__timechart_transform_current}
              \pgftransformshift{
                \pgfpoint{
                  .5*\l__timechart_finish_solid_x
                  +.5*\l__timechart_finish_extreme_x
                }
                {0}
              }
              \pgftransformxscale{
                (\l__timechart_finish_extreme_x-\l__timechart_finish_solid_x)
                /2bp
              }
              \pgftransformyscale{\l__timechart_bar_thickness_pgf/4bp}
            }
            \pgfsetfillcolor{\l__timechart_bar_color}
            \pgfusepath{fill}
            \endpgfscope
          }
    }
  }
```

(*End of definition for* `\__timechart_interval_finish_range_fade:`.)

14.15.6 Slant in/out

__timechart_interval_start_range_slant:

Clip and modify `\l__timechart_start_solid_x` to draw a slanting shape to indicate the start range.

```
\cs_new:Npn\__timechart_interval_start_range_slant:
  {
    \__timechart_if_equal:nnF
      {\l__timechart_start_extreme_x}{\l__timechart_start_solid_x}
      {
        \__timechart_if_x_range_intersect_tolerance_bounds:nnT
          {\l__timechart_start_extreme_x}{\l__timechart_start_solid_x}
          {
            \pgfinterruptboundingbox
            \pgfpathmoveto{
              \pgfpoint
                {\l__timechart_start_extreme_x}
                {-\l__timechart_bar_half_thickness_pgf}
            }
            \pgfpathlineto{
              \pgfpoint
                {\l__timechart_start_solid_x}
                {\l__timechart_bar_half_thickness_pgf}
            }
            \pgfpathlineto{
              \pgfpoint
                {\l__timechart_start_solid_x}
                {\l__timechart_bar_thickness_pgf}
            }
            \pgfpathlineto{
              \pgfpoint
                {\l__timechart_right_nonclip_x}
                {\l__timechart_bar_thickness_pgf}
            }
            \pgfpathlineto{
              \pgfpoint
                {\l__timechart_right_nonclip_x}
                {-\l__timechart_bar_thickness_pgf}
            }
            \pgfpathlineto{
              \pgfpoint
                {\l__timechart_start_extreme_x}
                {-\l__timechart_bar_thickness_pgf}
            }
            \pgfpathclose
            \pgfusepath{clip}
            \endpgfinterruptboundingbox
            \pgfmathsetmacro{\l__timechart_start_solid_x}
              {\l__timechart_start_extreme_x}
          }
      }
  }
```

(*End of definition for* `\__timechart_interval_start_range_slant:`.)

__timechart_interval_finish_range_slant:

Clip and modify `\l__timechart_finish_solid_x` to draw a slanting shape to indicate the finish range.

```
\cs_new:Npn\__timechart_interval_finish_range_slant:
  {
    \__timechart_if_equal:nnF
      {\l__timechart_finish_solid_x}{\l__timechart_finish_extreme_x}
      {
        \__timechart_if_x_range_intersect_tolerance_bounds:nnT
          {\l__timechart_finish_solid_x}{\l__timechart_finish_extreme_x}
          {
            \pgfinterruptboundingbox
            \pgfpathmoveto{
              \pgfpoint
                {\l__timechart_finish_solid_x}
                {-\l__timechart_bar_half_thickness_pgf}
            }
            \pgfpathlineto{
              \pgfpoint
                {\l__timechart_finish_extreme_x}
                {\l__timechart_bar_half_thickness_pgf}
            }
            \pgfpathlineto{
              \pgfpoint
                {\l__timechart_finish_extreme_x}
                {\l__timechart_bar_thickness_pgf}
            }
            \pgfpathlineto{
              \pgfpoint
                {\l__timechart_left_nonclip_x}
                {\l__timechart_bar_thickness_pgf}
            }
            \pgfpathlineto{
              \pgfpoint
                {\l__timechart_left_nonclip_x}
                {-\l__timechart_bar_thickness_pgf}
            }
            \pgfpathlineto{
              \pgfpoint
                {\l__timechart_finish_solid_x}
                {-\l__timechart_bar_thickness_pgf}
            }
            \pgfpathclose
            \pgfusepath{clip}
            \endpgfinterruptboundingbox
            \pgfmathsetmacro{\l__timechart_finish_solid_x}
              {\l__timechart_finish_extreme_x}
          }
      }
  }
```

(*End of definition for* `\__timechart_interval_finish_range_slant:`.)

14.15.7 Solid bar

__timechart_interval_draw_solid:

Draw the solid part of an interval.

```
\cs_new:Npn\__timechart_interval_draw_solid:
  {
    \__timechart_if_x_range_intersect_tolerance_bounds:nnT
      {\l__timechart_start_solid_x}
      {\l__timechart_finish_solid_x}
      {
        \pgfpathrectanglecorners{
          \pgfpoint
            {\l__timechart_start_solid_x}
            {-\l__timechart_bar_half_thickness_pgf}
        }{
          \pgfpoint
            {\l__timechart_finish_solid_x}
            {\l__timechart_bar_half_thickness_pgf}
        }
        \pgfsetfillcolor{\l__timechart_bar_color}
        \pgfusepath{fill}
      }
  }
```

(*End of definition for* `\__timechart_interval_draw_solid:`.)

14.15.8 Marks

```
\msg_new:nnn{timechart}{interval_mark_outside}
  { Attempt~to~mark~outside~interval~at~date~#1 }
```

`\__timechart_interval_mark:` Draw marks on an interval.

```
\cs_new:Npn\__timechart_interval_mark:
  {
    \pgfscope
    \pgfsetstrokecolor{\l__timechart_mark_color}
    \foreach \year in \l__timechart_mark_text
      {
        \pgfmathsetmacro{\l__timechart_mark_x}{yeartox(\year)}
        \__timechart_pgfmathsetbool:nn{\l__timechart_tmpa_bool}
          {
            or (
              \l__timechart_mark_x < \l__timechart_start_extreme_x,
              \l__timechart_mark_x > \l__timechart_finish_extreme_x
            )
          }
        \bool_if:NT\l__timechart_tmpa_bool
          { \msg_error:nne{timechart}{interval_mark_outside}{\year} }
        \pgfpathmoveto{
          \pgfpoint
            {\l__timechart_mark_x}
            {-\l__timechart_bar_half_thickness_pgf}
        }
        \pgfpathlineto{
          \pgfpoint
            {\l__timechart_mark_x}
            {\l__timechart_bar_half_thickness_pgf}
        }
        \pgfusepath{draw}
      }
    \endpgfscope
  }
```

(*End of definition for* `\__timechart_interval_mark:`.)

14.15.9 Bar node

`\__timechart_interval_define_bar_node:` Define a node with into which the bar fits exactly.

```
\cs_new:Npn\__timechart_interval_define_bar_node:
  {
    \__timechart_make_rectangle_node:nnnn
      {
        \pgfpoint
          {\l__timechart_start_extreme_x}
          {-\l__timechart_bar_half_thickness_pgf}
      }{
        \pgfpoint
          {\l__timechart_finish_extreme_x}
          {\l__timechart_bar_half_thickness_pgf}
      }
      {\pgfkeysvalueof{/timechart/interval~bar~node~name}}
      {\c_false_bool}
  }
```

(*End of definition for* `\__timechart_interval_define_bar_node:`.)

14.15.10 Label

__timechart_interval_label:

Place the label for the item.

```
\cs_new:Npn\__timechart_interval_label:
  {
    \str_if_empty:NF \l__timechart_label_text
      {
        \pgfinterruptboundingbox
        \int_case:nn {\l__timechart_label_pos_int}
        {
          {0}{ \__timechart_interval_label_left: }
          {1}{ \__timechart_interval_label_center: }
          {2}{ \__timechart_interval_label_right: }
        }
        \endpgfinterruptboundingbox
      }
  }
```

(*End of definition for* `\__timechart_interval_label:`.)

__timechart_interval_label_left:

Place the label on the left of the interval.

```
\cs_new:Npn\__timechart_interval_label_left:
  {
    \__timechart_if_x_in_bounds:nT{\l__timechart_start_extreme_x}
      {
        \group_begin:
        \pgftransformshift{
          \pgfpoint
            {\l__timechart_start_extreme_x}
            {\l__timechart_text_baseline_pgf}
        }
        \node[/timechart/interval~label,anchor=base~east]
          (\l__timechart_interval_label_node_name)
          at (0,0)
          {
            \__timechart_make_ref:NN
              \l__timechart_ref_text
              \l__timechart_label_text
          };
        \group_end:
      }
  }
```

(*End of definition for* `\__timechart_interval_label_left:`.)

__timechart_interval_label_right:

Place the label on the right of the interval.

```
\cs_new:Npn\__timechart_interval_label_right:
  {
    \__timechart_if_x_in_bounds:nT{\l__timechart_finish_extreme_x}
      {
        \group_begin:
        \pgftransformshift{
          \pgfpoint
            {\l__timechart_finish_extreme_x}
            {\l__timechart_text_baseline_pgf}
        }
        \node[/timechart/interval~label,anchor=base~west]
          (\l__timechart_interval_label_node_name)
          at (0,0)
          {
            \__timechart_make_ref:NN
              \l__timechart_ref_text
              \l__timechart_label_text
          };
        \group_end:
      }
  }
```

(*End of definition for* `\__timechart_interval_label_right:`.)

__timechart_interval_label_center:

Place the label at the center of the interval.

```
\cs_new:Npn\__timechart_interval_label_center:
  {
    \pgfmathsetlengthmacro{\l__timechart_label_anchor_x}
      {
        .5*max(\l__timechart_start_definite_x,\l__timechart_start_x)
        + .5*min(\l__timechart_finish_definite_x,\l__timechart_finish_x)
      }
```

First draw the ‘background label’ on the layer below the bar.

```
    \group_begin:
    \pgftransformshift{
      \pgfpoint
        {\l__timechart_label_anchor_x}
        {\l__timechart_text_baseline_pgf}
    }
    \pgfonlayer{labelbg}
    \node[
      /timechart/interval~label~centered~background,
      anchor=base
    ]
      at (0,0)
      { \l__timechart_label_text };
    \endpgfonlayer
    \group_end:
```

Then draw the label on top of the bar, clipping it to the bar outline.

```
    \pgfscope
    \pgfpathrectanglecorners{
      \pgfpoint
        {\l__timechart_start_extreme_x}
        {-\l__timechart_bar_half_thickness_pgf}
    }{
      \pgfpoint
        {\l__timechart_finish_extreme_x}
        {\l__timechart_bar_half_thickness_pgf}
    }
    \pgfusepath{clip}
    \group_begin:
    \pgftransformshift{
      \pgfpoint
        {\l__timechart_label_anchor_x}
        {\l__timechart_text_baseline_pgf}
    }
    \node[
    /timechart/interval~label~centered,
    anchor=base
    ]
      (\l__timechart_interval_label_node_name)
      at (0,0)
      {
        \__timechart_make_ref:NN
        \l__timechart_ref_text
        \l__timechart_label_text
      };
    \group_end:
    \endpgfscope
  }
```

(*End of definition for* `\__timechart_interval_label_center:`.)

14.16 Text

`\__timechart_text_user:Omm` Place text. This macro will be made available as `\timecharttext` inside the `timechart` environment.

#1 : PGF keys under `/timechart/` to apply.
#2 : Year at which to place text.
#3 : Text.

```
\NewDocumentCommand{\__timechart_text_user:Omm}{ O{} m m }
  {
    \str_if_empty:nF{#3}{
      \group_begin:
      \pgfmathsetmacro{\l__timechart_text_x}{yeartox(#2)}
      \__timechart_if_x_in_tolerance_bounds:nT{\l__timechart_text_x}
        {
```

Process keys supplied locally and retrieve needed keys.

```
          \pgfqkeys{/timechart}{
            #1,
            ref/.get=\l__timechart_ref_text,
            text~node~name/.get=\l__timechart_node_name_text,
            text~baseline/.get=\l__timechart_text_baseline_pgf,
          }
```

Shift to the correct vertical coordinate and place the text.

```
          \pgftransformshift{
            \pgfpoint{0}{\l__timechart_current_y}
          }
          \pgfinterruptboundingbox
          \group_begin:
          \pgftransformshift{
            \pgfpoint{\l__timechart_text_x}{\l__timechart_text_baseline_pgf}
          }
          \cs_set:Npn\l__timechart_text{#3}
          \int_case:nn {\l__timechart_text_pos_int}
          {
            {0}{ \cs_set:Npn\l__timechart_node_anchor_text{base~east} }
            {1}{ \cs_set:Npn\l__timechart_node_anchor_text{base} }
            {2}{ \cs_set:Npn\l__timechart_node_anchor_text{base~west} }
          }
          \node[/timechart/text,anchor=\l__timechart_node_anchor_text]
            (\l__timechart_node_name_text)
            at (0,0)
            {
              \__timechart_make_ref:NN
              \l__timechart_ref_text
              \l__timechart_text
            };
          \group_end:
          \endpgfinterruptboundingbox
        }
      \group_end:
    }
```

Since the text itself does not affect the bounding box, create a space (which will handle the automatic step).

```
    \__timechart_space_user:O[#1]
  }
```

(*End of definition for* `\__timechart_text_user:Omm`.)

14.17 Space

`\__timechart_space_user:O`

Create a vertical space as if there were an interval at the current coordinate. This macro will be made available as `\timechartspace` inside the `timechart` environment.

```
\NewDocumentCommand{\__timechart_space_user:O}{ O{} }
  {
    \group_begin:
```

Process keys supplied locally and retreive the one needed value.

```
    \pgfqkeys{/timechart}{
      #1,
      interval~bar~thickness/.get=\l__timechart_bar_thickness_pgf,
    }
    \pgfmathsetlengthmacro{\l__timechart_bar_half_thickness_pgf}
      { .5*\l__timechart_bar_thickness_pgf }
```

Shift to the correct vertical coordinate and create the space.

```
    \pgftransformshift{
      \pgfpoint{0}{\l__timechart_current_y}
    }
    \pgfpathmoveto{
      \pgfpoint{0}{-\l__timechart_bar_half_thickness_pgf}
    }
    \pgfpathmoveto{
      \pgfpoint{0}{\l__timechart_bar_half_thickness_pgf}
    }
    \pgfusepath{discard}
    \group_end:
    \bool_if:NT\l__timechart_autostep_bool{
      \__timechart_step_y_user:0
    }
  }
```

(*End of definition for* `\__timechart_space_user:0`.)

14.18 Legends

`\timechartlegenditem` Draw a bar suitable for use in a legend, applying style in `#1`.

```
\NewDocumentCommand{\timechartlegenditem}{ O{} }
  {
    \__timechart_legend_aux:nn{#1}{
      \pgfmathsetlengthmacro{\l__timechart_start_solid_x}{0}
      \pgfmathsetlengthmacro{\l__timechart_finish_solid_x}
        {\l__timechart_legenditem_width_pgf}
      \__timechart_interval_draw_solid:
    }
  }
```

(*End of definition for* `\timechartlegenditem`. *This function is documented on page 14.*)

`\timechartlegendstartrange`
`\timechartlegendfinishrange`

Draw a bar with start/finish range, suitable for use in a legend, applying style in `#1`.

```
\NewDocumentCommand{\timechartlegendstartrange}{ O{} }
  {
    \__timechart_legend_aux:nn{#1}{
      \pgfmathsetlengthmacro{\l__timechart_start_extreme_x}{0}
      \pgfmathsetlengthmacro{\l__timechart_start_solid_x}
        {\pgfkeysvalueof{/timechart/legend~item~range~width}}
      \pgfmathsetlengthmacro{\l__timechart_finish_solid_x}
        {\l__timechart_legenditem_width_pgf}
      \int_case:nn {\l__timechart_start_range_type_int}
        {
          {0}{ \__timechart_interval_start_range_fade: }
          {1}{ \__timechart_interval_start_range_slant: }
        }
      \__timechart_interval_draw_solid:
    }
  }
```

```
\NewDocumentCommand{\timechartlegendfinishrange}{ O{} }
  {
    \__timechart_legend_aux:nn{#1}{
      \pgfmathsetlengthmacro{\l__timechart_start_solid_x}{0}
      \pgfmathsetlengthmacro{\l__timechart_finish_solid_x}
        {
          \l__timechart_legenditem_width_pgf
          -\pgfkeysvalueof{/timechart/legend~item~range~width}
        }
      \pgfmathsetlengthmacro{\l__timechart_finish_extreme_x}
        {\l__timechart_legenditem_width_pgf}
      \int_case:nn {\l__timechart_finish_range_type_int}
        {
          {0}{ \__timechart_interval_finish_range_fade: }
          {1}{ \__timechart_interval_finish_range_slant: }
        }
      \__timechart_interval_draw_solid:
    }
  }
```

(*End of definition for* `\timechartlegendstartrange` *and* `\timechartlegendfinishrange`*. These functions are documented on page 14.*)

`\__timechart_legend_aux:nn`

Auxiliary command for legend items. Draw a Ti*k*Z picture, applying PGF keys `#1` under `/timechart/` and using Ti*k*Z code `#2`.

```
\cs_new:Npn\__timechart_legend_aux:nn #1#2
  {
    \tikzpicture
```

Process supplied PGF keys and retrieve only those necessary.

```
    \pgfkeys{
      /timechart/.cd,
      #1,
      interval~bar~thickness/.get=\l__timechart_bar_thickness_pgf,
      interval~bar~color/.get=\l__timechart_bar_color,
      interval~minimum~width/.get=\l__timechart_minimum_width_pgf,
      beyond~length/.get=\l__timechart_beyond_length_pgf,
      legend~item~width/.get=\l__timechart_legenditem_width_pgf,
    }
    \pgfmathsetlengthmacro{\l__timechart_bar_half_thickness_pgf}
      {.5*\l__timechart_bar_thickness_pgf}
    \pgfmathsetmacro{\l__timechart_start_x}{0}
    \pgfmathsetmacro{\l__timechart_finish_x}
      {\l__timechart_legenditem_width_pgf}
    \pgfmathsetlengthmacro{\l__timechart_start_beyond_x}
      {\l__timechart_start_x-\l__timechart_beyond_length_pgf}
    \pgfmathsetlengthmacro{\l__timechart_finish_beyond_x}
      {\l__timechart_finish_x+\l__timechart_beyond_length_pgf}
    \pgfmathsetmacro{\l__timechart_current_y}{0}
```

Ensure that the legend is ‘visible’ from the perspective of the drawing macros.

```
    \pgfmathsetmacro{\l__timechart_start_tolerance_x}
      {\l__timechart_start_x-10mm}
    \pgfmathsetmacro{\l__timechart_finish_tolerance_x}
      {\l__timechart_finish_x+10mm}
    #2
```

Make sure that the legend has the required bounding box.

```
    \pgfresetboundingbox
    \pgfmoveto{
      \pgfpoint{0}{-\l__timechart_bar_half_thickness_pgf}
    }
    \pgfmoveto{
      \pgfpoint
        {\l__timechart_legenditem_width_pgf}
        {\l__timechart_bar_half_thickness_pgf}
    }
    \endtikzpicture%
  }
```

(*End of definition for* `\__timechart_legend_aux:nn`.)

```
⟨/package⟩
```