Drawing Gantt Charts in LaTeX with TikZ

The pgfgantt package*

Wolfgang Skala†

April 18th, 2011

The pgfgantt package provides the ganttchart environment, which draws a Gantt chart within a TikZ picture. The user may add various elements to the chart, namely titles (\ganttitle, \ganttitlelist), bars (\ganttbar), milestones (\ganttmilestone), groups (\ganttgroup) and different links between these elements (\ganttlink). Furthermore, the appearance of the chart elements is highly customizable, owing to a number of keys.

Contents

*This document describes version v2.0, dated 2011/10/10.

†Division of Structural Biology, Department of Molecular Biology, University of Salzburg, Austria; Wolfgang.Skala@stud.sbg.ac.at

1 Introduction

The `pgfgantt` package allows you to draw Gantt charts in LaTeX. Thus, you can describe simple project schedules without having to include images produced by external programs. Similar to Martin Kumm's `gantt` package[1] (which inspired `pgfgantt`'s fundamental aspects), `pgfgantt` bases upon the Ti*k*Z frontend of PGF[2]. Besides, it provides a comprehensive (and portable) alternative to `pst-gantt`[3].

`pgfgantt` requires a *current* PGF installation. **Note that the version number must at least be 2.10, dated October 25th, 2010.** If you get a lot of errors and LaTeX complains that `\pgfkeysdefnargs` is undefined, your PGF installation is most likely too old. Furthermore, `pgfgantt` **2.0 is not fully downwards compatible. In particular, the syntax of** `\ganttlink` **has changed.**

Acknowledgements I would like to thank Petr Pošík (Czech Technical University in Prague) and Raphaël Clifford (University of Bristol) for their ideas concerning the new features in version 2.0.

[1] `http://www.martin-kumm.de/tex_gantt_package.php`
[2] `http://ctan.org/tex-archive/graphics/pgf/`
[3] `http://ctan.org/tex-archive/graphics/pstricks/contrib/pst-gantt/`

2 User Guide

2.1 Overview

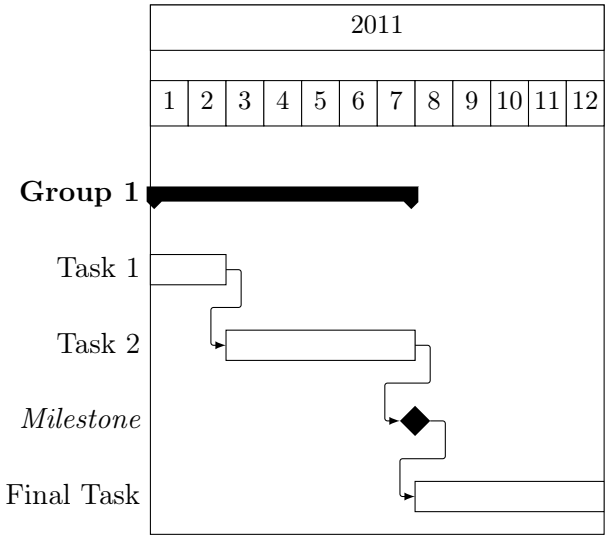
To load the package, simply put

```
\usepackage{pgfgantt}
```

into the document preamble.

Compare the following code, which demonstrates some commands provided by `pgfgantt`, to the output it produces:

```
\begin{tikzpicture}
  \begin{ganttchart}{12}
    \gantttitle{2011}{12} \\
    \gantttitlelist{1,...,12}{1} \\
    \ganttgroup{Group 1}{1}{7} \\
    \ganttbar{Task 1}{1}{2} \\
    \ganttlinkedbar{Task 2}{3}{7} \ganttnewline
    \ganttmilestone{Milestone}{7} \ganttnewline
    \ganttbar{Final Task}{8}{12}
    \ganttlink{elem2}{elem3}
    \ganttlink{elem3}{elem4}
  \end{ganttchart}
\end{tikzpicture}
```



2.2 Specifying Keys

Keys (sometimes called *options*) modify the output from `pgfgantt`'s commands. You may specify a key in two ways: (1) Pass it to the optional argument present in each command, e. g.

```
\ganttbar[bar label font=\bfseries]{Task 1}{1}{2}
```

This locally changes a key for the element(s) drawn by that command. (2) Alternatively, specify a key by the \ganttset{⟨*key=value list*⟩} macro, which sets its keys globally (or rather within the current TEX group): \ganttset

```
\ganttset{bar label font=\bfseries}
```

Since pgfgantt uses the pgfkeys package for key management, all its keys reside in the /pgfgantt/ path. However, if you set your keys by one of the methods explained above, this path is automatically prepended to each key.

2.3 The Canvas

Let us have a look at the basic anatomy of a Gantt chart and define some common terms. Each *chart* consists of several *elements*, such as titles, bars and connections between bars. Commands that start with \gantt... draw these elements. When specifying start and end *coordinates* for these commands, we use the dimensionless *chart coordinate system*, whose origin lies in the top left corner. Along the x-axis, one unit corresponds to one *time slot*; along the y-axis, one unit equals one *line*.

The ganttchart environment groups several of the element-drawing macros into a single chart: ganttchart

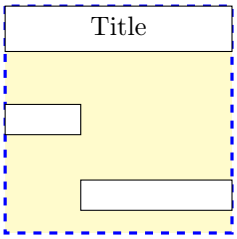
```
\begin{ganttchart}[⟨options⟩]{⟨number of time slots⟩}
  ...
\end{ganttchart}
```

The environment has one optional and one mandatory argument. The former specifies the ⟨*options*⟩ for the chart, the latter indicates the ⟨*number of time slots*⟩. Each ganttchart must be surrounded by a tikzpicture environment.

/pgfgantt/canvas=⟨*style*⟩ initial value: fill=white

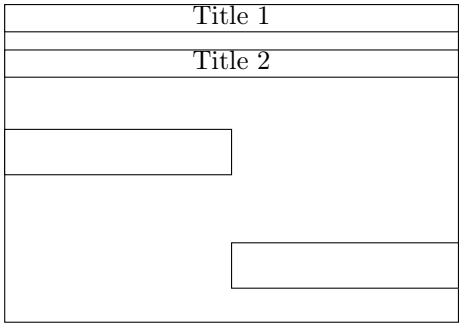
The canvas key changes the appearance of the canvas. ⟨*style*⟩ is a list of TikZ keys such as fill, draw or dashed. By default, the canvas is a white rectangle with a black frame.

```
\begin{tikzpicture}
  \begin{ganttchart}%
      [canvas={fill=yellow!25, draw=blue, dashed, very thick}]{6}
    \ganttitle{Title}{6} \\
    \ganttbar{}{1}{2} \\
    \ganttbar{}{3}{6}
  \end{ganttchart}
\end{tikzpicture}
```



/pgfgantt/x unit=⟨*dimension*⟩ .5cm
/pgfgantt/y unit title=⟨*dimension*⟩ 1cm
/pgfgantt/y unit chart=⟨*dimension*⟩ 1cm
These keys specify the width of a time slot and the height of title or chart lines, respectively. Typically, the x/y-dimension ratio approximates 1 : 2, and the line height is equal over the whole chart. Other dimensions are well possible, but you might have to change several spacing-related keys in order to obtain a pleasing chart.

```
\begin{tikzpicture}
  \begin{ganttchart}[x unit=1cm, y unit title=.6cm, y unit chart=1.5cm]{6}
    \gantttitle{Title 1}{6} \\
    \gantttitle{Title 2}{6} \\
    \ganttbar{}{1}{3} \\
    \ganttbar{}{4}{6}
  \end{ganttchart}
\end{tikzpicture}
```



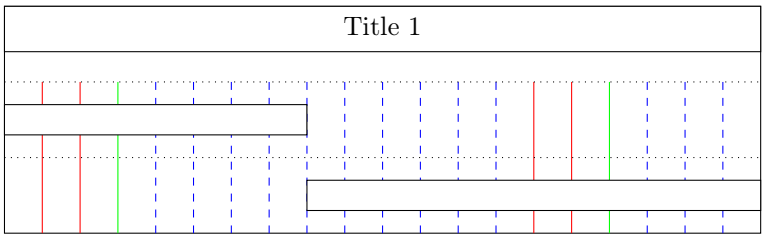
/pgfgantt/hgrid[=false/true/⟨*style list*⟩] false
/pgfgantt/hgrid style=⟨*style*⟩ dotted
/pgfgantt/vgrid[=false/true/⟨*style list*⟩] false
hgrid draws a horizontal grid which starts immediately below the last title element. The key can be specified in four different ways: Firstly, hgrid=false eliminates the horizontal grid. You may omit this declaration, since it is the default. Secondly,

both `hgrid` and `hgrid=true` activate the horizontal grid, which is then drawn in the default style `dotted`. Finally, `hgrid=`⟨*style list*⟩ draws the horizontal grid in the given ⟨*style list*⟩ (see below).

`hgrid style` changes the style of single horizontal grid lines that are drawn with `\ganttnewline[grid]` (see section 2.4). The `vgrid` key governs the vertical grid; otherwise, use it exactly like `hgrid`.

Style lists allow you to draw the grid lines in different styles. Each style list consists of several *style list items* separated by a comma. A style list item has the general syntax `*{`⟨*n*⟩`}{`⟨*style*⟩`}` and orders the package to repeat the ⟨*style*⟩ ⟨*n*⟩-times. (This syntax is reminiscent of column specifications in a `tabular` environment.) Thus, the list `*2{red}, *1{green}, *{10}{blue, dashed}` instructs `pgfgantt` to draw first two red vertical grid lines, then a green one and finally ten dashed blue lines. If any grid lines remain to be drawn at the end of the list, the package starts again at the beginning of the list.

```
\begin{tikzpicture}
  \begin{ganttchart}%
      [hgrid=true,
      vgrid={*2{red}, *1{green}, *{10}{blue, dashed}}]{20}
    \ganttitle{Title 1}{20} \\
    \ganttbar{}{1}{8} \\
    \ganttbar{}{9}{20}
  \end{ganttchart}
\end{tikzpicture}
```



In most situations, you can omit the multiplier `*1`. Hence, the following style lists are equal:

`{*1{red}, *1{blue, dashed}}`
`{{red}, {blue, dashed}}`
`{red, {blue, dashed}}`

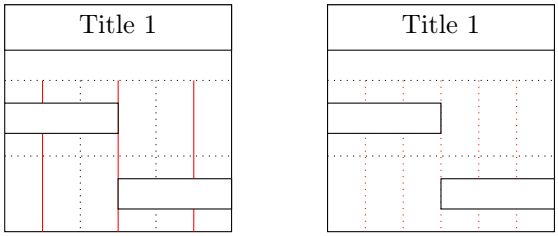
However, if you wish to use a single style comprising two or more keys for all grid lines, e. g. `red, dotted`, you *must* retain the multiplier (i. e., `{*1{red, dotted}}`).

```
% wrong code

\begin{tikzpicture}
  \begin{ganttchart}%
      [hgrid=true,
      vgrid={{red, dotted}}]{6}
    \ganttttitle{Title 1}{6} \\
    \ganttbar{}{1}{3} \\
    \ganttbar{}{4}{6}
  \end{ganttchart}
\end{tikzpicture}
```

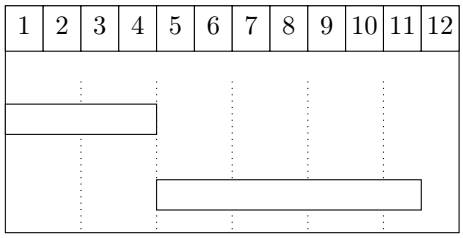
```
% correct code

\begin{tikzpicture}
  \begin{ganttchart}%
      [hgrid=true,
      vgrid={*1{red, dotted}}]{6}
    \ganttttitle{Title 1}{6} \\
    \ganttbar{}{1}{3} \\
    \ganttbar{}{4}{6}
  \end{ganttchart}
\end{tikzpicture}
```



In a chart with many time slots, drawing vertical grid lines between all of them will lead to a confusing appearance. In such a case, you can pass an appropriate ⟨*style list*⟩ to `vgrid` in order to draw every second grid line, for example.

```
\begin{tikzpicture}
  \begin{ganttchart}%
      [vgrid={draw=none, dotted}]{12}
    \ganttttitlelist{1,...,12}{1} \\
    \ganttbar{}{1}{4} \\
    \ganttbar{}{5}{11}
  \end{ganttchart}
\end{tikzpicture}
```

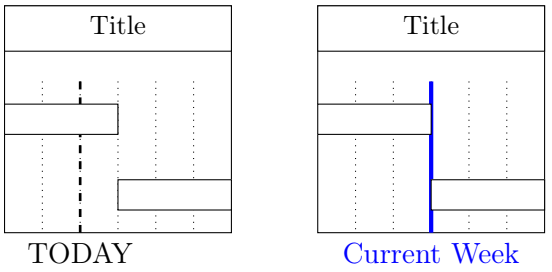


/pgfgantt/today=⟨*time slot*⟩	none
/pgfgantt/today rule=⟨*style*⟩	dashed, line width=1pt
/pgfgantt/today label=⟨*text*⟩	TODAY

Sometimes, you may wish to indicate the current day, month or the like on a Gantt chart. In order to do so, pass an integer value to the `today` key, which draws a vertical rule at the corresponding ⟨*time slot*⟩. This rule appears in the ⟨*style*⟩ denoted by `today rule`, while `today label` contains the ⟨*text*⟩ below the rule.

```
\begin{tikzpicture}%
    [x=.5cm, y=1cm, baseline]
  \begin{ganttchart}%
      [vgrid, today=2]{6}
    \gantttitle{Title}{6} \\
    \ganttbar{}{1}{3} \\
    \ganttbar{}{4}{6}
  \end{ganttchart}
\end{tikzpicture}
```

```
\begin{tikzpicture}%
    [x=.5cm, y=1cm, baseline]
  \begin{ganttchart}%
      [vgrid, today=3,
      today label=\textcolor{blue}%
        {Current Week},
      today rule={blue, ultra thick}]{6}
    \gantttitle{Title}{6} \\
    \ganttbar{}{1}{3} \\
    \ganttbar{}{4}{6}
  \end{ganttchart}
\end{tikzpicture}
```



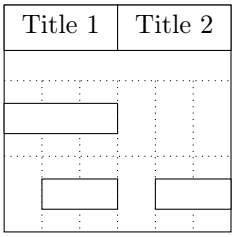
2.4 Line Breaks between Chart Elements

`pgfgantt` does not automatically begin a new line after finishing a chart element. Instead, you must insert an explicit line break with `\ganttnewline`. Within a `ganttchart` environment, `\\` is defined as a shortcut for `\ganttnewline`, so that the syntax is reminiscent of LaTeX's `tabular` enviroment.

`\ganttnewline`

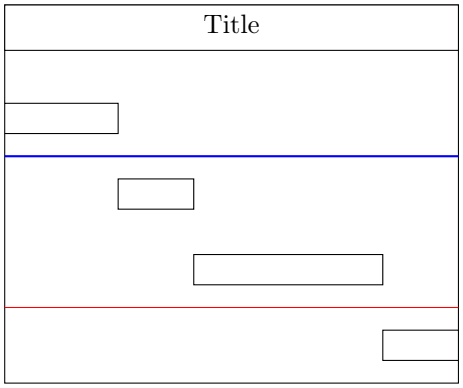
`\\`

```
\begin{tikzpicture}
  \begin{ganttchart}[hgrid, vgrid]{6}
    \gantttitle{Title 1}{3}
    \gantttitle{Title 2}{3} \\
    \ganttbar{}{1}{3} \ganttnewline
    \ganttbar{}{2}{3}
    \ganttbar{}{5}{6}
  \end{ganttchart}
\end{tikzpicture}
```



Even if you prefer a canvas without a horizontal grid, you may nevertheless want to separate certain lines by a grid rule. For this purpose, specify the optional argument `[grid]` for `\ganttnewline` (or `\\`), which draws a grid rule in `hgrid style` between the current and the new line. Alternatively, directly give the desired style as optional argument.

```
\begin{tikzpicture}
  \begin{ganttchart}[hgrid style=red]{12}
    \ganttitle{Title}{12} \\
    \ganttbar{}{1}{3} \ganttnewline[thick, blue]
    \ganttbar{}{4}{5} \\
    \ganttbar{}{6}{10} \\[grid]
    \ganttbar{}{11}{12}
  \end{ganttchart}
\end{tikzpicture}
```



2.5 Titles

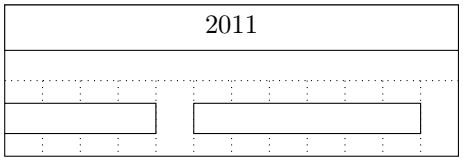
A *title* (comprising one or more lines) at the top of a Gantt chart usually indicates the period of time covered by that chart. For example, the first line could span twelve time slots and display the current year, while the second line could contain twelve elements, each of which corresponds to one month. For these purposes, `pgfgantt` implements two titling commands.

`\ganttitle` draws a single title element: `\ganttitle`

```
\ganttitle[⟨options⟩]{⟨label⟩}{⟨number of time slots⟩}
```

The ⟨*label*⟩ appears in the center of the title element, which covers the ⟨*number of time slots*⟩ starting from the right end of the last title element (or from the beginning of the line, if the title element is the first element in this line). Mostly, you will employ `\ganttitle` for titles that span several time slots.

```
\begin{tikzpicture}
  \begin{ganttchart}[hgrid, vgrid]{12}
    \ganttitle{2011}{12} \\
    \ganttbar{}{1}{4}
    \ganttbar{}{6}{11}
  \end{ganttchart}
\end{tikzpicture}
```



Whenever you want to draw a larger number of title elements that are equal in size and follow a common enumeration scheme, the `\ganttitlelist` macro provides a fast solution:

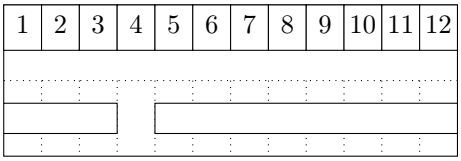
`\ganttitlelist`

```
\ganttitlelist[⟨options⟩]{⟨pgffor list⟩}{⟨length of each element⟩}
```

This macro generates one title element for each member of the ⟨*pgffor list*⟩. The second mandatory argument specifies the ⟨*length of each element*⟩. The Ti*k*Z manual describes the syntax for the ⟨*pgffor list*⟩ in more detail, but we will mention two of the most common applications:

1. In order to draw twelve title elements that contain the numbers from 1 to 12 (indicating the months of a year), enter `1,...,12` as the ⟨*pgffor*⟩ list.

```
\begin{tikzpicture}
  \begin{ganttchart}[hgrid, vgrid]{12}
    \ganttitlelist{1,...,12}{1} \\
    \ganttbar{}{1}{3}
    \ganttbar{}{5}{12}
  \end{ganttchart}
\end{tikzpicture}
```



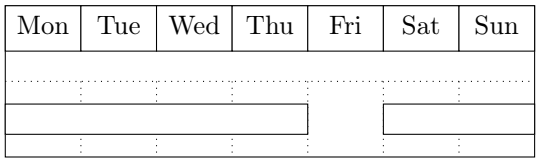
Note that we would have obtained the same result if we had written

```
\ganttttitle{1}{1} \ganttttitle{2}{1} ... \ganttttitle{12}{1} \\
```

2. In order to draw seven title elements containing the names of the weekdays (e. g., “Mon” to “Sun”), we have to change the `title list options` key:

   `/pgfgantt/title list options`=⟨*pgffor options*⟩ `var=\x, evaluate=\x`
   This key changes the ⟨*pgffor options*⟩ of the `\foreach` command called by `\ganttttitlelist`. Again, the Ti*k*Z manual is the definitive reference on possible ⟨*pgffor options*⟩. There is just one thing to keep in mind: The macro that yields the labels to be printed by `\ganttttitlelist` must be called `\x`.
   The following example shows how you can implement a title line enumerating the days of the week:

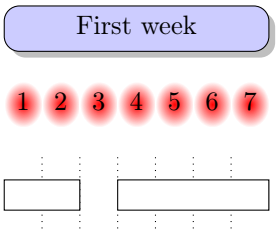
```
\usepackage{pgfcalendar}
  ...
\begin{tikzpicture}
  \begin{ganttchart}[hgrid, vgrid, x unit=1cm]{7}
    \ganttttitlelist[title list options={%
        var=\y, evaluate=\y as \x%
        using "\pgfcalendarweekdayshortname{\y}"%
      }]{0,...,6}{1} \\
    \ganttbar{}{1}{4}
    \ganttbar{}{6}{7}
  \end{ganttchart}
\end{tikzpicture}
```



`/pgfgantt/title`=⟨*style*⟩ `fill=white`
Sets the appearance of a title element.

```
\usetikzlibrary{shadows}
\usetikzlibrary{shadings}
  ...
\begin{tikzpicture}
  \begin{ganttchart}%
      [vgrid, canvas={draw=none},
      title={fill=blue!20, rounded corners=2mm, drop shadow}]{7}
    \ganttttitle{First week}{7} \\
```

```
    \ganttitlelist[title={draw=none, inner color=red}]{1,...,7}{1} \\
    \ganttbar{}{1}{2}
    \ganttbar{}{4}{7}
  \end{ganttchart}
\end{tikzpicture}
```



/pgfgantt/title label font=⟨*font commands*⟩ `\small`

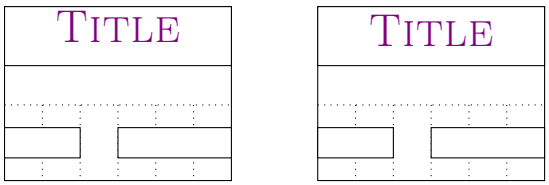
Selects the font of the text inside a title element. In most cases, you can include font format commands directly in the first mandatory argument of `\ganttitle`. However, you *must* use the `title label font` key if you intend to change the font size. Otherwise, the vertical alignment of the title label will be incorrect with the standard anchor.

```
% Wrong alignment

\begin{tikzpicture}
  \begin{ganttchart}%
      [vgrid, hgrid,
      y unit title=1.3cm]{6}
    \ganttitle{%
      \LARGE\color{violet}%
      \scshape Title}{6} \\
    \ganttbar{}{1}{2}
    \ganttbar{}{4}{6}
  \end{ganttchart}
\end{tikzpicture}
```

```
% Correct alignment

\begin{tikzpicture}
  \begin{ganttchart}%
      [vgrid, hgrid,
      y unit title=1.3cm,
      title label font={\LARGE,
      \color{violet},\scshape}]{6}
    \ganttitle{Title}{6} \\
    \ganttbar{}{1}{2}
    \ganttbar{}{4}{6}
  \end{ganttchart}
\end{tikzpicture}
```



/pgfgantt/title label anchor=⟨*anchor*⟩ `anchor=mid`

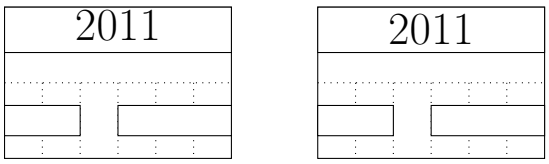
By default, title labels are vertically centered at half their x-height. This yields a good alignment for labels whose letters have equal amounts of ascenders and descenders (e. g., lowercase numbers). However, when the letters contain mostly ascenders (e. g., uppercase numbers), the label position will appear too high. In this case, you should change the anchor:

```
% Badly centered label

\begin{tikzpicture}%
  \begin{ganttchart}%
      [vgrid, hgrid,
      title label font={\LARGE}%
      ]{6}
    \gantttitle{2011}{6} \\
    \ganttbar{}{1}{2}
    \ganttbar{}{4}{6}
  \end{ganttchart}
\end{tikzpicture}
```

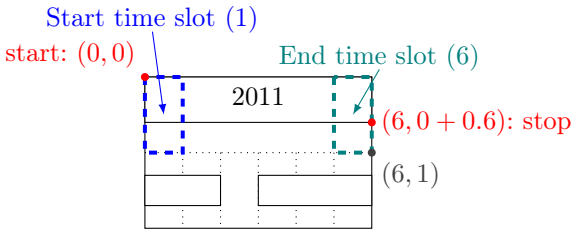
```
% Nicely centered label

\begin{tikzpicture}
  \begin{ganttchart}%
      [vgrid, hgrid,
      title label font={\LARGE},
      title label anchor={below=-1.5ex}%
      ]{6}
    \gantttitle{2011}{6} \\
    \ganttbar{}{1}{2}
    \ganttbar{}{4}{6}
  \end{ganttchart}
\end{tikzpicture}
```

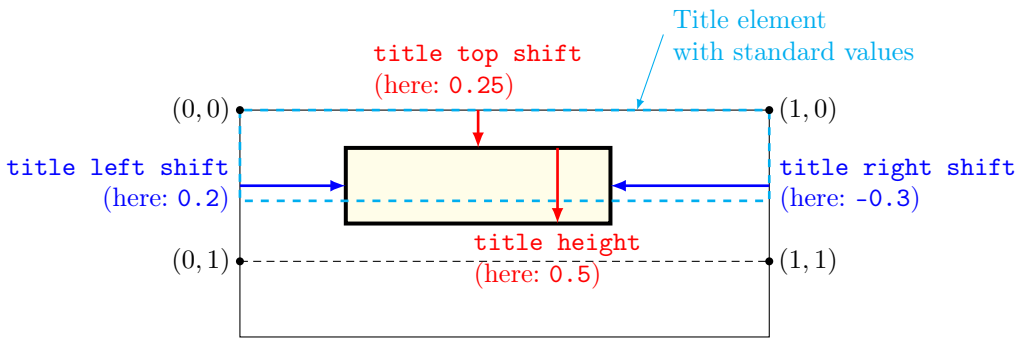


`/pgfgantt/title left shift`=⟨*factor*⟩ 0
`/pgfgantt/title right shift`=⟨*factor*⟩ 0
`/pgfgantt/title top shift`=⟨*factor*⟩ 0
`/pgfgantt/title height`=⟨*factor*⟩ 0.6

The first three keys shift the coordinates of a title element's borders (or rather of its corners), while `title height` changes its height. By default, the left upper corner of a title element coincides with the origin of the start time slot; its right lower corner touches the right border of the end time slot 0.6 units below the upper line border:

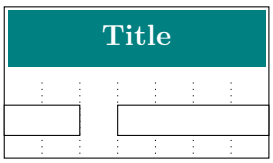


The figure below shows a Gantt chart with two lines and one (large) time slot and indicates the distances modified by these keys.



For example, you might devise a layout where the title element does not touch the borders of the start and end time slot.

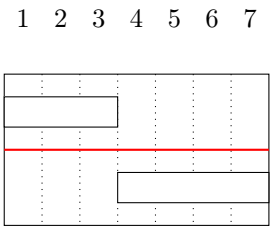
```
\begin{tikzpicture}
  \begin{ganttchart}[vgrid, title={fill=teal, draw=none},
      title label font=\color{white}\bfseries,
      title left shift=.1, title right shift=-.1,
      title top shift=.05, title height=.75]{7}
    \gantttitle{Title}{7} \\
    \ganttbar{}{1}{2}
    \ganttbar{}{4}{7}
  \end{ganttchart}
\end{tikzpicture}
```



/pgfgantt/include title in canvas=false/true true

The canvas normally comprises all lines of the chart. However, you may wish that your title elements only consist of text lacking any frame or background. In this case, the canvas probably should exclude all lines containing title elements, which you achieve by include title in canvas=false.

```
\begin{tikzpicture}
  \begin{ganttchart}%
      [hgrid={*1{draw=red, thick}}, vgrid,
      title={draw=none, fill=none}, include title in canvas=false]{7}
    \gantttitlelist{1,...,7}{1} \\
    \ganttbar{}{1}{3} \\
    \ganttbar{}{4}{7}
  \end{ganttchart}
\end{tikzpicture}
```



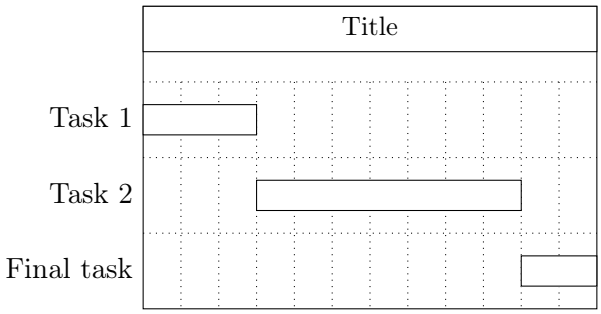
2.6 Bars

On a Gantt chart, a *bar* indicates the duration of a task or one of its parts.

```
\ganttbar[⟨options⟩]{⟨label⟩}{⟨start time slot⟩}{⟨end time slot⟩}
```

The \ganttbar macro draws a bar from the ⟨*start time slot*⟩ to the ⟨*end time slot*⟩ and adds a ⟨*label*⟩ at the left of the chart. \ganttbar

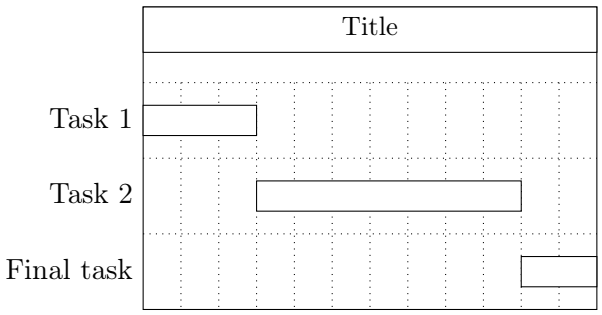
```
\begin{tikzpicture}
  \begin{ganttchart}[vgrid, hgrid]{12}
    \ganttitle{Title}{12} \\
    \ganttbar{Task 1}{1}{3} \\
    \ganttbar{Task 2}{4}{10} \\
    \ganttbar{Final task}{11}{12}
  \end{ganttchart}
\end{tikzpicture}
```



/pgfgantt/time slot modifier=⟨*factor*⟩ -1

Note that a bar usually touches the left border of the ⟨*start time slot*⟩ and not the right, as it would if the ⟨*start time slot*⟩ were strictly interpreted as an x-coordinate. However, you may prefer to work with "real" x-coordinates instead of time slots. In this case, just set the time slot modifier key to zero. This will essentially eliminate the semi-intelligent behavior of pgfgantt with respect to the conversion of x-coordinates. This feature may prove useful if you decide to use real numbers for some time slots.

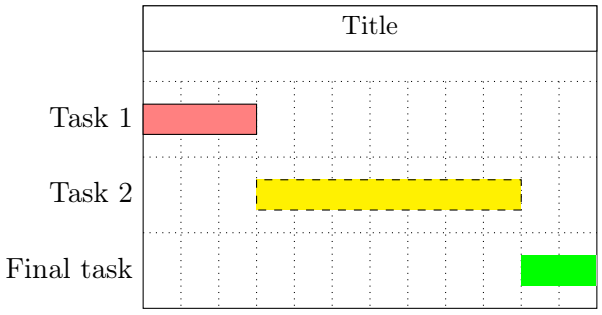
```
\begin{tikzpicture}
  \begin{ganttchart}[vgrid, hgrid, time slot modifier=0]{12}
    \ganttitle{Title}{12} \\
    \ganttbar{Task 1}{0}{3} \\
    \ganttbar{Task 2}{3}{10} \\
    \ganttbar{Final task}{10}{12}
  \end{ganttchart}
\end{tikzpicture}
```



/pgfgantt/bar=⟨*style*⟩ `fill=white`

Determines the appearance of the bar.

```
\begin{tikzpicture}
  \begin{ganttchart}[vgrid, hgrid, bar={fill=red!50}]{12}
    \ganttitle{Title}{12} \\
    \ganttbar{Task 1}{1}{3} \\
    \ganttbar[bar={fill=yellow, dashed}]{Task 2}{4}{10} \\
    \ganttbar[bar={fill=green, draw=none}]{Final task}{11}{12}
  \end{ganttchart}
\end{tikzpicture}
```

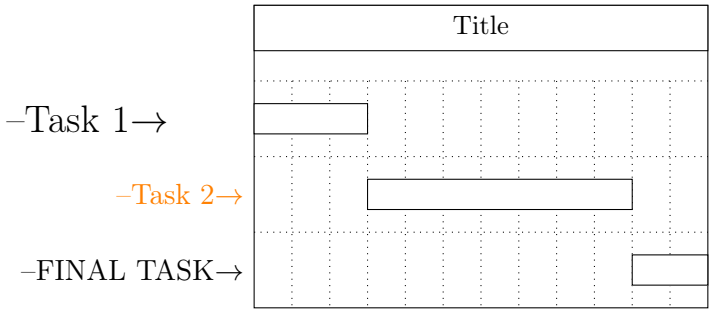


/pgfgantt/bar label text=⟨*text*⟩ `\strut#1`

/pgfgantt/bar label font=⟨*font commands*⟩ \normalsize
/pgfgantt/bar label anchor=⟨*anchor*⟩ anchor=east

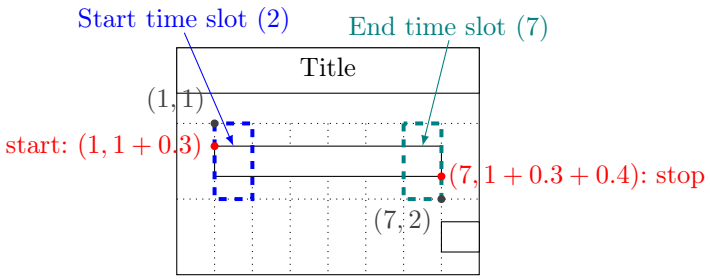
The bar label text key configures the label ⟨*text*⟩ next to each bar. This key should contain a single parameter token (#1), which is replaced by the first mandatory argument of \ganttbar. The \strut in the standard value ensures equal vertical spacing of the labels. bar label font selects the font for the bar label, bar label anchor determines its anchor. The last control sequence in ⟨*font commands*⟩ may take a single argument (like \textit).

```
\begin{tikzpicture}
  \begin{ganttchart}
      [vgrid, hgrid, bar label font=\Large,
      bar label text={--#1$\rightarrow$}]{12}
    \ganttitle{Title}{12} \\
    \ganttbar[bar label anchor={left=1cm}]{Task 1}{1}{3} \\
    \ganttbar[bar label font=\color{orange}]{Task 2}{4}{10} \\
    \ganttbar[bar label font=\MakeUppercase]{Final task}{11}{12}
  \end{ganttchart}
\end{tikzpicture}
```

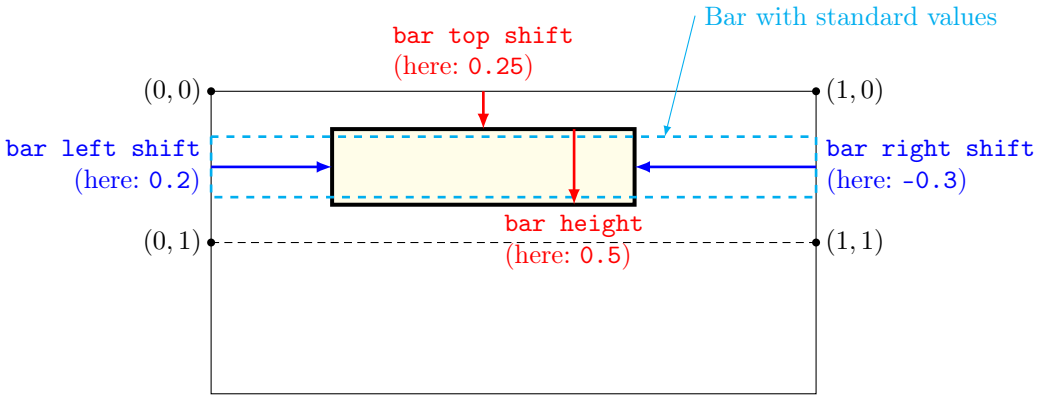


/pgfgantt/bar left shift=⟨*factor*⟩ 0
/pgfgantt/bar right shift=⟨*factor*⟩ 0
/pgfgantt/bar top shift=⟨*factor*⟩ 0.3
/pgfgantt/bar height=⟨*factor*⟩ 0.4

The first three keys shift the coordinates of a bar's borders (or rather of its corners), while bar height changes its height. By default, the left upper corner of a bar is 0.3 units below the origin of the start time slot; its right lower corner touches the right border of the end time slot 0.4 units below the upper line border:

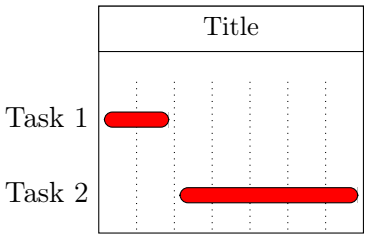


The figure below shows a Gantt chart with two lines and one (large) time slot and indicates the distances modified by these keys.



For example, you might devise a layout with small, rounded bars that do not touch the borders of their start and end time slots.

```
\begin{tikzpicture}
  \begin{ganttchart}[vgrid, bar={fill=red, rounded corners=3pt},
      bar left shift=.15, bar right shift=-.15,
      bar top shift=.4, bar height=.2]{7}
    \ganttitle{Title}{7} \\
    \ganttbar{Task 1}{1}{2} \\
    \ganttbar{Task 2}{3}{7}
  \end{ganttchart}
\end{tikzpicture}
```



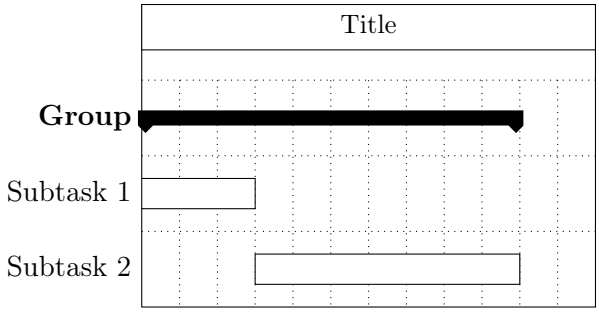
2.7 Groups

Groups subsume several subtasks (represented by bars) into a single task.

```
\ganttgroup[⟨options⟩]{⟨label⟩}{⟨start time slot⟩}{⟨end time slot⟩}
```

The \ganttgroup macro draws a group from the ⟨*start time slot*⟩ to the ⟨*end time slot*⟩ and adds a ⟨*label*⟩ at the left of the chart. Note that a group will start at the left border of the ⟨*start time slot*⟩ (and not at the right, as it would if the ⟨*start time slot*⟩ were strictly interpreted as an x-coordinate). \ganttgroup

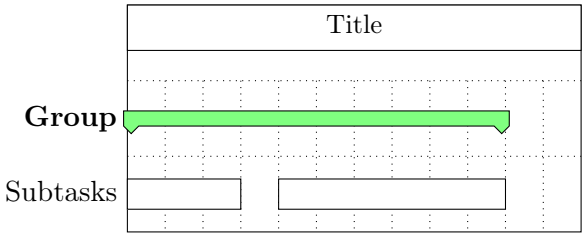
```
\begin{tikzpicture}
  \begin{ganttchart}[vgrid, hgrid]{12}
    \ganttitle{Title}{12} \\
    \ganttgroup{Group}{1}{10} \\
    \ganttbar{Subtask 1}{1}{3} \\
    \ganttbar{Subtask 2}{4}{10}
  \end{ganttchart}
\end{tikzpicture}
```



/pgfgantt/group=⟨*style*⟩ `fill=black`

Changes the appearance of a group.

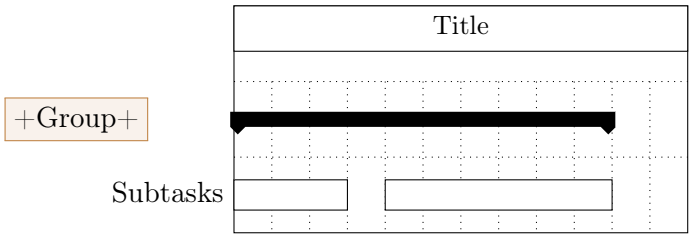
```
\begin{tikzpicture}
  \begin{ganttchart}[vgrid, hgrid, group={draw=black, fill=green!50}]{12}
    \ganttitle{Title}{12} \\
    \ganttgroup{Group}{1}{10} \\
    \ganttbar{Subtasks}{1}{3}
    \ganttbar{}{5}{10}
  \end{ganttchart}
\end{tikzpicture}
```



/pgfgantt/group label text=⟨*text*⟩ `\strut#1`
/pgfgantt/group label font=⟨*font commands*⟩ `\normalsize\bfseries`
/pgfgantt/group label anchor=⟨*anchor*⟩ `anchor=east`

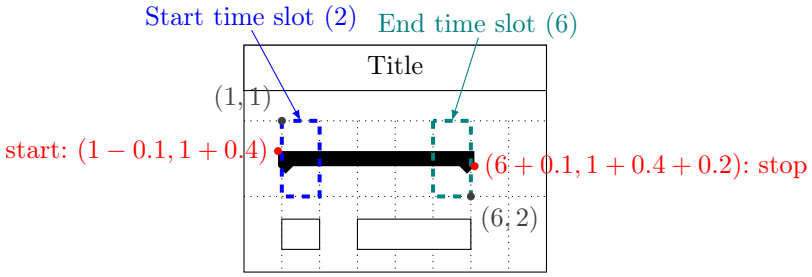
The `group label text` key configures the label ⟨*text*⟩ next to each group. This key should contain a single parameter token (`#1`), which is replaced by the first mandatory argument of `\ganttgroup`. The `\strut` in the standard value ensures equal vertical spacing of the labels. `group label font` selects the font of the group label, `group label anchor` determines its anchor. The last control sequence in ⟨*font commands*⟩ may take a single argument (like `\textit`).

```
\begin{tikzpicture}
  \begin{ganttchart}%
      [vgrid, hgrid,
      group label font={\fcolorbox{brown}{brown!10}},
      group label anchor={left=1cm},
      group label text={+#1+}]{12}
    \ganttitle{Title}{12} \\
    \ganttgroup{Group}{1}{10} \\
    \ganttbar{Subtasks}{1}{3}
    \ganttbar{}{5}{10}
  \end{ganttchart}
\end{tikzpicture}
```

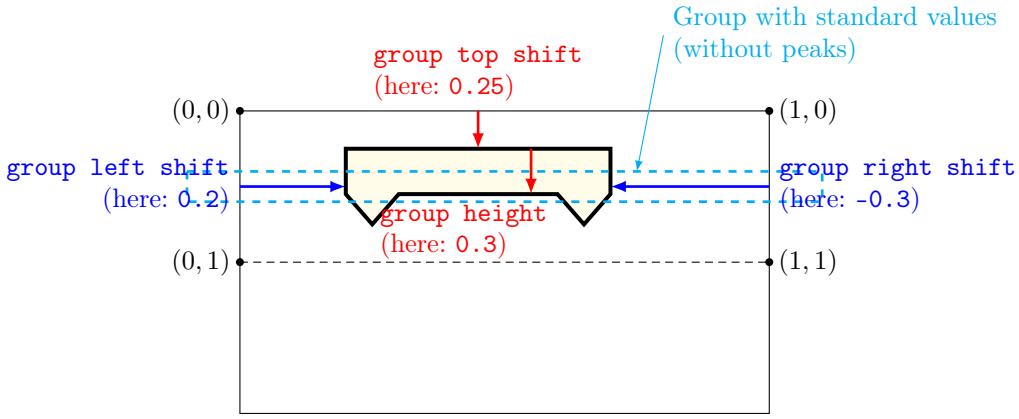


/pgfgantt/group left shift=⟨*factor*⟩ `-0.1`
/pgfgantt/group right shift=⟨*factor*⟩ `0.1`
/pgfgantt/group top shift=⟨*factor*⟩ `0.4`
/pgfgantt/group height=⟨*factor*⟩ `0.2`

The first three keys shift the coordinates of a group's borders (or rather of its corners), while `group height` changes its height. By default, the left upper corner of a group is 0.1 units left of and 0.4 units below the start time slot origin; its right lower corner (not counting the peak) lies 0.1 units right of and 0.3 units below the right border of the end time slot:



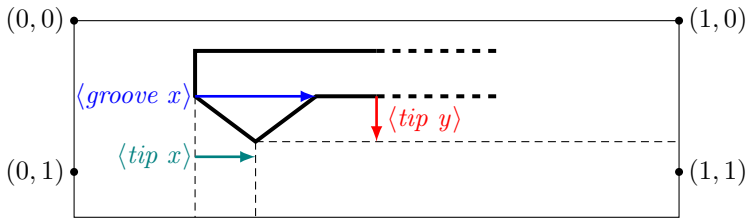
The figure below shows a Gantt chart with two lines and one (large) time slot and indicates the distances modified by these keys.



`/pgfgantt/group left peak`={⟨*tip x*⟩}{⟨*groove x*⟩}{⟨*tip y*⟩}
`/pgfgantt/group right peak`={⟨*tip x*⟩}{⟨*groove x*⟩}{⟨*tip y*⟩}
`/pgfgantt/group peaks`={⟨*tip x*⟩}{⟨*groove x*⟩}{⟨*tip y*⟩} `0.2 0.4 0.1`

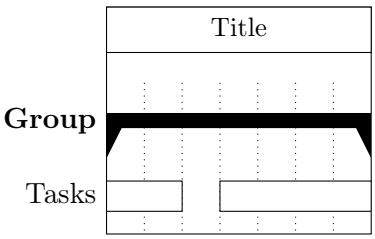
These keys govern the appearance of the peaks at both ends of a group. By default, the tip of each peak lies 0.2 units inward from a group's bottom corner and 0.1 units beneath, while the groove lies 0.4 units inward. While `group left peak` applies only to the left peak and `group right peak` affects only the right peak, `group peaks` sets the dimensions for both peaks simultaneously. You always have to specify three arguments for these keys. However, if you leave one of them blank, the corresponding space parameter retains its current value.

The figure below exemplifies the space parameters as they apply to the left peak.



For example, you might prefer that your groups stay within the start and end time slot, and that the peaks are more acute:

```
\begin{tikzpicture}
  \begin{ganttchart}%
      [vgrid, group left shift=0, group right shift=0,
      group peaks={0}{}{.4}]{7}
    \gantttitle{Title}{7} \\
    \ganttgroup{Group}{1}{7} \\
    \ganttbar{Tasks}{1}{2}
    \ganttbar{}{4}{7}
  \end{ganttchart}
\end{tikzpicture}
```



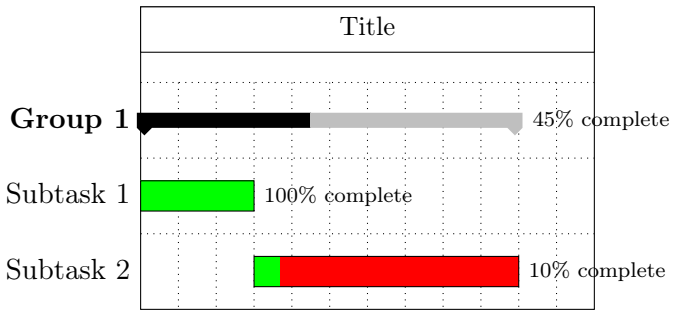
2.8 Progress Bars and Progress Groups

Progress bars and *progress groups* illustrate the extent to which a (sub-)task has been completed. In order to draw a progress element, you simply specify the `progress` key in the optional argument to the respective standard macro.

`/pgfgantt/progress`=`none`/⟨*number*⟩ `none`
`/pgfgantt/bar incomplete`=⟨*style*⟩
`/pgfgantt/group incomplete`=⟨*style*⟩
`/pgfgantt/incomplete`=⟨*style*⟩ `fill=black!25`

The `progress` key specifies that a task (represented by a bar) or a group thereof is ⟨*number*⟩ percent complete. Starting from the left, ⟨*number*⟩ percent of the element's area appear in the basic style (i. e., `bar` or `group`), while the `bar incomplete` and `group incomplete` keys, respectively, determine the appearance of the remainder. For convenience, the `incomplete` key simultaneously sets the incomplete style for bars and groups.

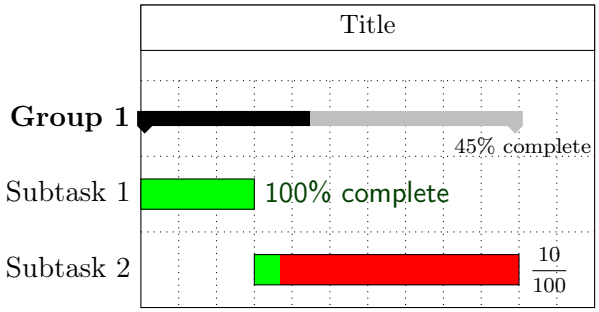
```
\begin{tikzpicture}
  \begin{ganttchart}[vgrid, hgrid, bar={fill=green}]{12}
    \ganttitle{Title}{12} \\
    \ganttgroup[progress=45]{Group 1}{1}{10} \\
    \ganttbar[progress=100]{Subtask 1}{1}{3} \\
    \ganttbar[progress=10, bar incomplete={fill=red}]{Subtask 2}{4}{10}
  \end{ganttchart}
\end{tikzpicture}
```



/pgfgantt/progress label text=⟨*text*⟩ #1\% complete
/pgfgantt/progress label font=⟨*font commands*⟩ \scriptsize
/pgfgantt/progress label anchor=⟨*anchor*⟩ anchor=west

The progress label text key sets the ⟨*text*⟩ that appears beside each progress element in order to indicate its completeness. This key may contain a single parameter token (#1), which is replaced by the value of progress. The label is typeset in the progress label font. In addition, progress label anchor governs its placement. By changing the default value, you may prevent the label from overlapping with other elements of your chart.

```
\begin{tikzpicture}
  \begin{ganttchart}[vgrid, hgrid, bar={fill=green}]{12}
    \ganttitle{Title}{12} \\
    \ganttgroup%
        [progress=45, progress label anchor={below=3pt}]%
      {Group 1}{1}{10} \\
    \ganttbar%
        [progress=100, progress label font=\color{green!25!black}\textsf]%
      {Subtask 1}{1}{3} \\
    \ganttbar%
        [progress=10, incomplete={fill=red},
        progress label text={$\displaystyle\frac{#1}{100}$}]%
      {Subtask 2}{4}{10}
  \end{ganttchart}
\end{tikzpicture}
```



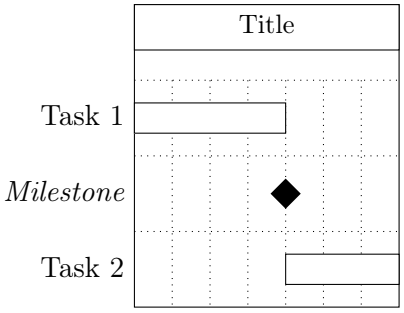
2.9 Milestones

A *milestone* signifies that an important task has been completed or that a crucial goal has been reached.

```
\ganttmilestone[⟨options⟩]{⟨label⟩}{⟨time slot⟩}
```

The \ganttmilestone macro draws a milestone at the given ⟨*time slot*⟩ and adds a ⟨*label*⟩ at the left of the chart. \ganttmilestone

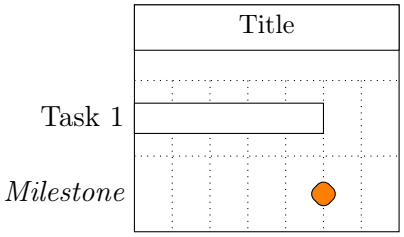
```
\begin{tikzpicture}
  \begin{ganttchart}[vgrid, hgrid]{7}
    \ganttitle{Title}{7} \\
    \ganttbar{Task 1}{1}{4} \\
    \ganttmilestone{Milestone}{4} \\
    \ganttbar{Task 2}{5}{7}
  \end{ganttchart}
\end{tikzpicture}
```



Note that the milestone is usually centered on the vertical grid line between its ⟨*time slot*⟩ and the following one.

/pgfgantt/milestone=⟨*style*⟩ `fill=black`
Determines the appearance of the milestone.

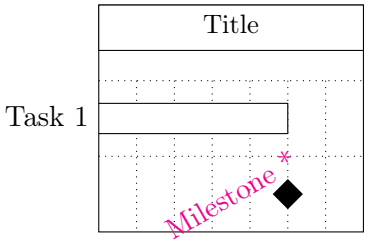
```
\begin{tikzpicture}
  \begin{ganttchart}%
      [vgrid, hgrid,
      milestone={fill=orange, draw=black, rounded corners=3pt}]{7}
    \ganttitle{Title}{7} \\
    \ganttbar{Task 1}{1}{5} \\
    \ganttmilestone{Milestone}{5}
  \end{ganttchart}
\end{tikzpicture}
```



/pgfgantt/milestone label text=⟨*text*⟩ `\strut#1`
/pgfgantt/milestone label font=⟨*font commands*⟩ `\normalsize\itshape`
/pgfgantt/milestone label anchor=⟨*anchor*⟩ `anchor=east`

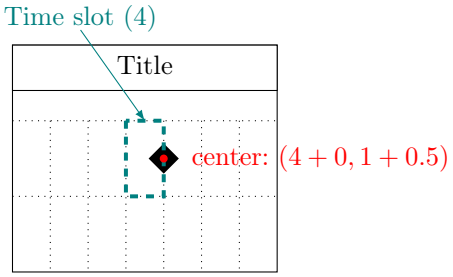
The `milestone label text` key configures the label ⟨*text*⟩ next to each milestone. This key should contain a single parameter token (`#1`), which is replaced by the first mandatory argument of `\ganttmilestone`. The `\strut` in the standard value ensures equal vertical spacing of the labels. `milestone label font` sets the font of the milestone label, while `milestone label anchor` determines its placement. The last macro in ⟨*font commands*⟩ may take a single argument, as we show in the following (somewhat silly) example.

```
\begin{tikzpicture}
  \begin{ganttchart}%
      [vgrid, hgrid,
      milestone label font=\color{magenta}\rotatebox{30},
      milestone label anchor={right=.7cm},
      milestone label text={#1 *}]{7}
    \ganttitle{Title}{7} \\
    \ganttbar{Task 1}{1}{5} \\
    \ganttmilestone{Milestone}{5}
  \end{ganttchart}
\end{tikzpicture}
```

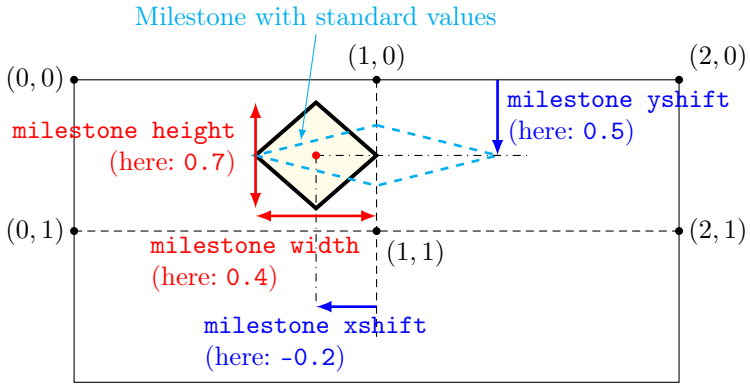


/pgfgantt/milestone width=⟨*factor*⟩ 0.8
/pgfgantt/milestone height=⟨*factor*⟩ 0.4
/pgfgantt/milestone xshift=⟨*factor*⟩ 0
/pgfgantt/milestone yshift=⟨*factor*⟩ 0.5

These keys set the width and height of a milestone and shift the coordinates of its center. By default, a milestone is 0.8 units wide and 0.4 units high. Since the ideal x-vector/y-vector ratio is $1:2$, the milestone appears square with these settings. Its center lies on the right border and 0.5 units below the top border of its time slot.



The figure below shows a Gantt chart with a single milestone and two (large) time slots; it indicates the distances modified by the four keys explained above.



2.10 Links

So far, we have drawn charts whose elements were quite independent of each other. However, relations or *links* between these elements frequently appear on real Gantt

charts. For example, a task may only start if a previous one has been completed, or finishing a task may constitute a milestone.

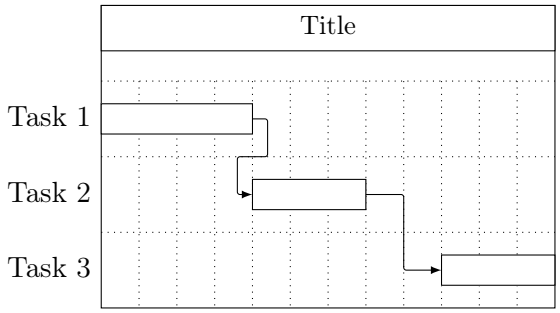
```
\ganttlink[⟨options⟩]{⟨start element name⟩}{⟨end element name⟩}
```

/pgfgantt/name=⟨*name*⟩
The \ganttlink macro connects two elements, which are specified by their ⟨*name*⟩s. By default, chart elements are named automatically: The first one receives the name `elem0`, the second one is called `elem1` and so on. However, the `name` key allows you to assign a name to each chart element.

\ganttlink

```
\begin{tikzpicture}
  \begin{ganttchart}%
      [vgrid, hgrid]{12}
    \ganttitle{Title}{12} \\
    \ganttbar{Task 1}{1}{4} \\
    \ganttbar{Task 2}{5}{7} \\
    \ganttbar{Task 3}{10}{12}
    \ganttlink{elem0}{elem1}
    \ganttlink{elem1}{elem2}
  \end{ganttchart}
\end{tikzpicture}
```

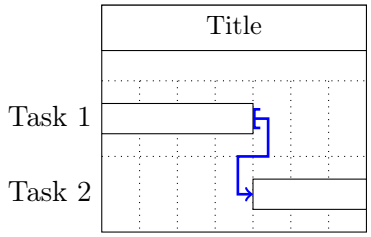
```
\begin{tikzpicture}
  \begin{ganttchart}%
      [vgrid, hgrid]{12}
    \ganttitle{Title}{12} \\
    \ganttbar[name=b1]%
      {Task 1}{1}{4} \\
    \ganttbar[name=b2]%
      {Task 2}{5}{7} \\
    \ganttbar[name=xyz]%
      {Task 3}{10}{12}
    \ganttlink{b1}{b2}
    \ganttlink{b2}{xyz}
  \end{ganttchart}
\end{tikzpicture}
```



/pgfgantt/link=⟨*style*⟩ `-latex, rounded corners=1pt`
Sets the appearance of the link.

```
\begin{tikzpicture}
  \begin{ganttchart}[vgrid, hgrid, link={[-to, line width=1pt, blue}]{7}
    \ganttitle{Title}{7} \\
    \ganttbar{Task 1}{1}{4} \\
    \ganttbar{Task 2}{5}{7}
```

```
    \ganttlink{elem0}{elem1}
  \end{ganttchart}
\end{tikzpicture}
```



/pgfgantt/link type=⟨*type*⟩ arrow

Link types fall into two categories:

1. *Arrow-like links* are the default. As you can see from the examples above, they consist of three segments (two horizontal, one vertical) if their start and end time slots are sufficiently separated. Otherwise, they comprise five segments (three horizontal, two vertical). Three keys further modify the appearance of arrow-like links:

   /pgfgantt/link mid=⟨*factor*⟩ 0.5

   The link mid key changes the position of the single vertical segment (in three-part links) or of the middle horizontal segment (in five-part links). By default, these segments are horizontally centered between the left and the right vertical segment, or vertically centered between the upper and the lower horizontal segment, respectively.

   /pgfgantt/link bulge=⟨*factor*⟩ 0.4

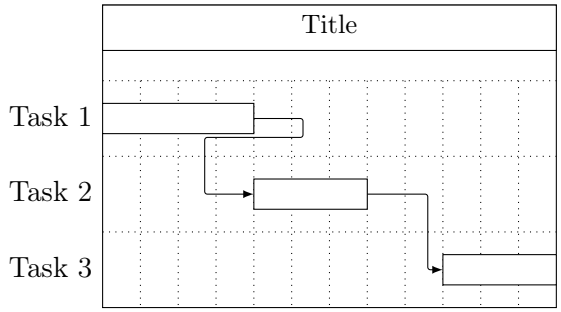
   In five-part links, the upper and lower vertical segments are shifted along the x-axis by +link bulge and −link bulge, respectively.

   /pgfgantt/link tolerance=⟨*factor*⟩ 0.6

   This key governs whether pgfgantt draws a five- or a three-part link. If the true x-coordinates of the link start and end differ by at least link tolerance (this is the case for the second link in the example below), the package draws a five-part link.

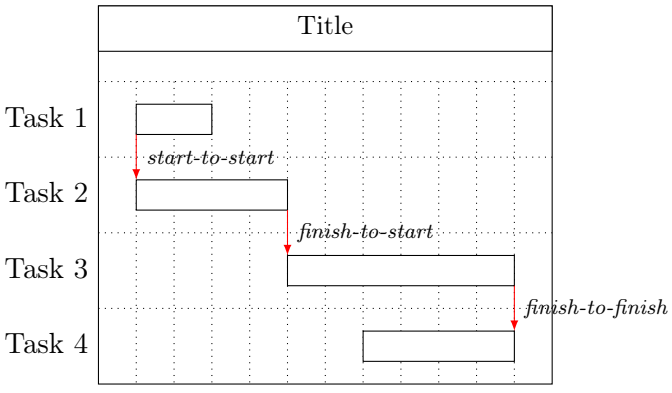
```
\begin{tikzpicture}
  \begin{ganttchart}[vgrid, hgrid, link mid=.25, link bulge=1.3]{12}
    \ganttitle{Title}{12} \\
    \ganttbar{Task 1}{1}{4} \\
    \ganttbar{Task 2}{5}{7} \\
    \ganttbar{Task 3}{10}{12}
```

```
    \ganttlink{elem0}{elem1}
    \ganttlink[link mid=.8]{elem1}{elem2}
  \end{ganttchart}
\end{tikzpicture}
```



2. *Straight links* are only meant for connecting two bars in order to establish start-to-finish relations (`s-f`), start-to-start relations (`s-s`) etc. Their ⟨*type*⟩ identifiers commemorate the syntax for specifying arrow tips in Ti*k*Z: Each identifier is composed of two letters separated by a hyphen.

```
\begin{tikzpicture}
  \begin{ganttchart}[vgrid, hgrid, link={-latex, red}]{12}
    \ganttitle{Title}{12} \\
    \ganttbar{Task 1}{2}{3} \\
    \ganttbar{Task 2}{2}{5} \\
    \ganttbar{Task 3}{6}{11} \\
    \ganttbar{Task 4}{8}{11}
    \ganttlink[link type=s-s]{elem0}{elem1}
    \ganttlink[link type=f-s]{elem1}{elem2}
    \ganttlink[link type=f-f]{elem2}{elem3}
  \end{ganttchart}
\end{tikzpicture}
```

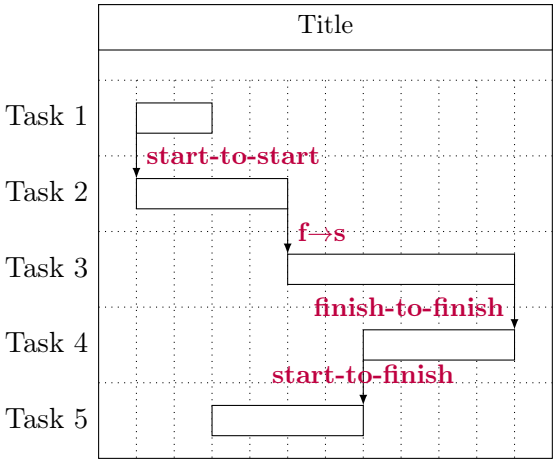


`/pgfgantt/link label=`{⟨*link type key*⟩}{⟨*label text*⟩} (miscellaneous)
`/pgfgantt/link label font=`⟨*font*⟩ `\scriptsize\itshape\normalcolor`
`/pgfgantt/link label anchor=`⟨*anchor*⟩ `anchor=west`

Since straight links all look the same, a label indicates the respective relationship. You can redefine these labels with the `link label` key, which changes the ⟨*label text*⟩ for a ⟨*link type key*⟩.

The `link label font` key specifies the font for the label, `link label anchor` determines its placement (by default, the label appears to the right of the straight link's center).

```
\begin{tikzpicture}
  \begin{ganttchart}[vgrid, hgrid,
      link label font=\small\color{purple}\textbf,
      link label={f-s}{f$\to$s}]{12}
    \ganttitle{Title}{12} \\
    \ganttbar{Task 1}{2}{3} \\
    \ganttbar{Task 2}{2}{5} \\
    \ganttbar{Task 3}{6}{11} \\
    \ganttbar{Task 4}{8}{11} \\
    \ganttbar{Task 5}{4}{7}
    \ganttlink[link type=s-s]{elem0}{elem1}
    \ganttlink[link type=f-s]{elem1}{elem2}
    \ganttlink[link type=f-f, link label anchor={anchor=east}]%
      {elem2}{elem3}
    \ganttlink[link type=s-f, link label anchor={anchor=base}]%
      {elem3}{elem4}
  \end{ganttchart}
\end{tikzpicture}
```



2.11 Linked Bars and Linked Milestones

Since you'll most likely draw a lot of arrow-like links between bars and milestones, `pgfgantt` provides two convenient shortcuts for these tasks:

```
\ganttlinkedbar[⟨options⟩]{⟨label⟩}{⟨start time slot⟩}{⟨end time slot⟩}
\ganttlinkedmilestone[⟨options⟩]{⟨label⟩}{⟨time slot⟩}
```

These macros work exactly like the standard versions, but they additionally draw a link from the previous element to the bar or milestone. In the following example, the code on the left is equivalent to the code on the right.

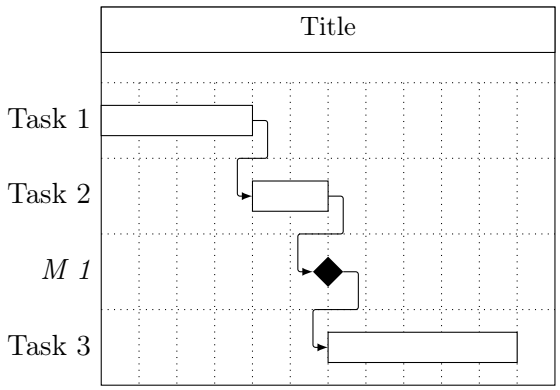
\ganttlinkedbar
\ganttlinkedmilestone

```
% Short version

\begin{tikzpicture}
  \begin{ganttchart}%
      [vgrid, hgrid]{12}
    \ganttitle{Title}{12} \\
    \ganttbar{Task 1}{1}{4} \\
    \ganttlinkedbar{Task 2}{5}{6} \\
    \ganttlinkedmilestone{M 1}{6} \\
    \ganttlinkedbar{Task 3}{7}{11}
  \end{ganttchart}
\end{tikzpicture}
```

```
% Long version

\begin{tikzpicture}
  \begin{ganttchart}%
      [vgrid, hgrid]{12}
    \ganttitle{Title}{12} \\
    \ganttbar{Task 1}{1}{4} \\
    \ganttbar{Task 2}{5}{6} \\
    \ganttmilestone{M 1}{6} \\
    \ganttbar{Task 3}{7}{11}
    \ganttlink{elem0}{elem1}
    \ganttlink{elem1}{elem2}
    \ganttlink{elem2}{elem3}
  \end{ganttchart}
\end{tikzpicture}
```

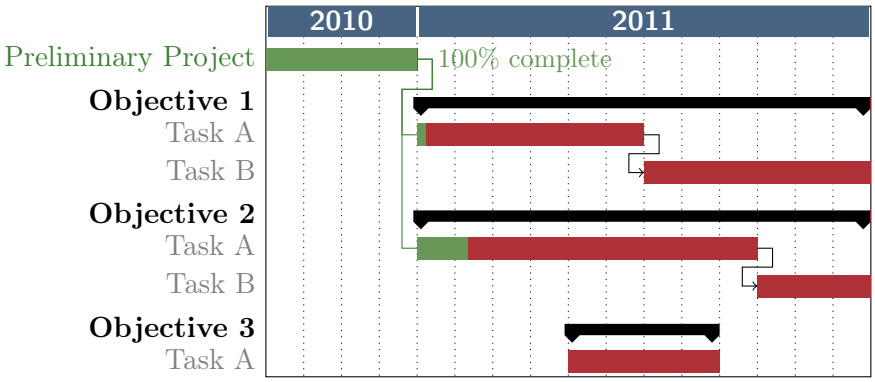


2.12 Style Examples

The first example plays around with colors and notably uses equal x- and y-vectors.

```
\begin{tikzpicture}
  \begin{ganttchart}%
      [y unit title=0.4cm,
```

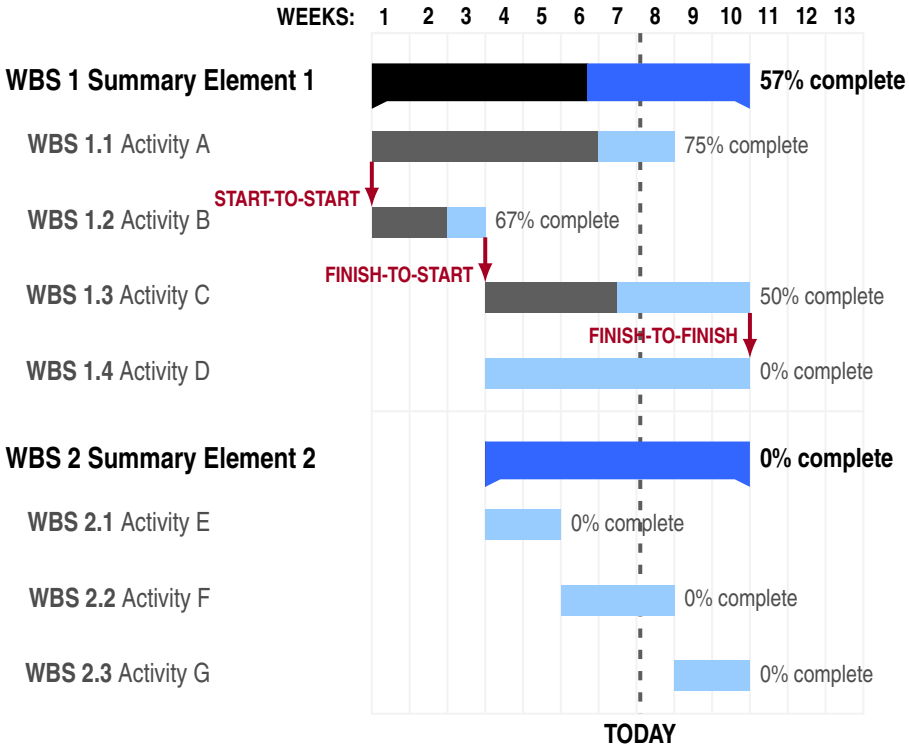
```
      y unit chart=0.5cm,
      vgrid,
      title={draw=none, fill=RoyalBlue!50!black},
      title label font=\sffamily\bfseries\color{white},
      title label anchor={below=-1.6ex},
      title left shift=.05,
      title right shift=-.05,
      title height=1,
      bar={draw=none, fill=OliveGreen!75},
      bar height=.6,
      bar label font=\normalsize\color{black!50},
      group right shift=0,
      group top shift=.6,
      group height=.3,
      group peaks={}{}{.2},
      incomplete={fill=Maroon}]{16}
    \gantttitle{2010}{4}
    \gantttitle{2011}{12} \\
    \ganttbar%
        [progress=100, progress label font=\small\color{OliveGreen!75},
        progress label anchor={right=4pt},
        bar label font=\normalsize\color{OliveGreen},
        name=pp]%
      {Preliminary Project}{1}{4} \\
    \ganttset{progress label text={}, link={black, -to}}
    \ganttgroup{Objective 1}{5}{16} \\
    \ganttbar[progress=4, name=T1A]{Task A}{5}{10} \\
    \ganttlinkedbar[progress=0]{Task B}{11}{16} \\
    \ganttgroup{Objective 2}{5}{16} \\
    \ganttbar[progress=15, name=T2A]{Task A}{5}{13} \\
    \ganttlinkedbar[progress=0]{Task B}{14}{16} \\
    \ganttgroup{Objective 3}{9}{12} \\
    \ganttbar[progress=0]{Task A}{9}{12}
    \ganttset{link={OliveGreen}}
    \ganttlink[link mid=.4]{pp}{T1A}
    \ganttlink[link mid=.159]{pp}{T2A}
  \end{ganttchart}
\end{tikzpicture}
```



The second example demonstrates that `pgfgantt` is really flexible: Even an appearance quite different from the standard layout is possible. (More precisely, the code below tries to reproduce the Gantt chart from the English Wikipedia site, see `http://en.wikipedia.org/wiki/Gantt_chart`.)

```
\definecolor{barblue}{RGB}{153,204,254}
\definecolor{groupblue}{RGB}{51,102,254}
\definecolor{linkred}{RGB}{165,0,33}
\begin{tikzpicture}
  \renewcommand\sfdefault{phv}
  \renewcommand\mddefault{mc}
  \renewcommand\bfdefault{bc}
  \sffamily
  \begin{ganttchart}%
      [canvas={fill=none, draw=black!5, line width=.75pt},
      hgrid style={draw=black!5, line width=.75pt},
      vgrid={*1{draw=black!5, line width=.75pt}},
      today=7.1,
      today rule={draw=black!64,
        dash pattern=on 3.5pt off 4.5pt, line width=1.5pt},
      today label={\small\bfseries TODAY},
      title={draw=none, fill=none},
      title label font=\bfseries\footnotesize,
      title label anchor={below=7pt},
      include title in canvas=false,
      bar label font=\mdseries\small\color{black!70},
      bar label anchor={left=2cm},
      bar={draw=none, fill=black!63},
      bar incomplete={fill=barblue},
      progress label font=\mdseries\footnotesize\color{black!70},
      group incomplete={fill=groupblue},
      group left shift=0,
      group right shift=0,
      group height=.5,
```

```
      group peaks={0}{}{},
      group label anchor={left=.6cm},
      link={-latex, line width=1.5pt, linkred},
      link label font=\scriptsize\bfseries\color{linkred}\MakeUppercase,
      link label anchor={below left=-2pt and 0pt}
      ]{13}
    \gantttitle[title label anchor={below left=7pt and -3pt}]%
      {WEEKS:\quad1}{1}
    \gantttitlelist{2,...,13}{1} \\
    \ganttgroup[progress=57, progress label font=\bfseries\small]%
      {WBS 1 Summary Element 1}{1}{10} \\
    \ganttbar[progress=75, name=WBS1A]%
      {\textbf{WBS 1.1} Activity A}{1}{8} \\
    \ganttbar[progress=67, name=WBS1B]%
      {\textbf{WBS 1.2} Activity B}{1}{3} \\
    \ganttbar[progress=50, name=WBS1C]%
      {\textbf{WBS 1.3} Activity C}{4}{10} \\
    \ganttbar[progress=0, name=WBS1D]%
      {\textbf{WBS 1.4} Activity D}{4}{10} \\[grid]
    \ganttgroup[progress=0, progress label font=\bfseries\small]%
      {WBS 2 Summary Element 2}{4}{10} \\
    \ganttbar[progress=0]{\textbf{WBS 2.1} Activity E}{4}{5} \\
    \ganttbar[progress=0]{\textbf{WBS 2.2} Activity F}{6}{8} \\
    \ganttbar[progress=0]{\textbf{WBS 2.3} Activity G}{9}{10}
    \ganttlink[link type=s-s]{WBS1A}{WBS1B}
    \ganttlink[link type=f-s]{WBS1B}{WBS1C}
    \ganttlink[link type=f-f, link label anchor={left}]{WBS1C}{WBS1D}
  \end{ganttchart}
\end{tikzpicture}
```


WEEKS: 1 2 3 4 5 6 7 8 9 10 11 12 13
WBS 1 Summary Element 1
57% complete
WBS 1.1 Activity A
75% complete
START-TO-START
WBS 1.2 Activity B
67% complete
FINISH-TO-START
WBS 1.3 Activity C
50% complete
FINISH-TO-FINISH
WBS 1.4 Activity D
0% complete
WBS 2 Summary Element 2
0% complete
WBS 2.1 Activity E
0% complete
WBS 2.2 Activity F
0% complete
WBS 2.3 Activity G
0% complete
TODAY

3 Implementation

3.1 Packages

pgfgantt is modest in terms of dependencies: It only requires the Ti*k*Z package and some of its libraries.

```
\RequirePackage{tikz}
  \usetikzlibrary{arrows,backgrounds,calc,patterns,positioning}

```

3.2 Global Counters and Booleans

We define a number of global counters: gtt@width equals the number of time slots. gtt@currentline holds the current line; it starts from 0 and decreases. gtt@lasttitleline equals the line of the title element drawn last. Furthermore, gtt@lasttitleslot corresponds to the x-coordinate of its right border. gtt@elementid enumerates the automatic names of chart elements. gtt@currgrid is the index of the current grid line drawn.

```
\newcounter{gtt@width}
\newcounter{gtt@currentline}
\newcounter{gtt@lasttitleline}
\newcounter{gtt@lasttitleslot}
\newcounter{gtt@elementid}
\newcounter{gtt@currgrid}
```

The macros \gtt@lastelement and \gtt@currentelement save the name of the current and last chart element drawn. Thereby, the \ganttlinked... macros can add a link connecting them.

\gtt@lastelement
\gtt@currentelement
\ifgtt@intitle

The boolean \ifgtt@intitle is true at the start of a ganttchart environment and set to false as soon as the first non-title element is encountered.

```
\def\gtt@lastelement{}
\def\gtt@currentelement{}
\newif\ifgtt@intitle

```

3.3 Macros for Key Management

\ganttset changes the current key path to /pgfgantt/ and then executes the keys in its mandatory argument.

\ganttset

```
\def\ganttset#1{\pgfqkeys{/pgfgantt}{#1}}

```

The following four auxiliary macros save us some code when we devise keys later on. Firstly, \@gtt@keydef{⟨*key*⟩} declares the key /pgfgantt/⟨*key*⟩, which saves its value in the macro gtt@⟨*key*⟩.

\@gtt@keydef

```
\def\@gtt@keydef#1{%
  \pgfkeysdef{/pgfgantt/#1}{%
    \expandafter\def\csname gtt@#1\endcsname{##1}%
  }%
}
```

Secondly, `\@gtt@get{`⟨*key*⟩`}` retrieves the value stored by a ⟨*key*⟩. `\@gtt@get`

```
\def\@gtt@get#1{\csname gtt@#1\endcsname}
```

Thirdly, `\@gtt@stylekeydef{`⟨*key*⟩`}` declares a ⟨*key*⟩ which saves its value as the internal style key `/pgfgantt/`⟨*key*⟩`@style`. `\@gtt@stylekeydef`

```
\def\@gtt@stylekeydef#1{%
  \pgfkeysdef{/pgfgantt/#1}{%
    \pgfkeys{/pgfgantt/#1@style/.style={##1}}%
  }%
}
```

Finally, `\@gtt@linkkeydef{`⟨*link type*⟩`}{`⟨*label*⟩`}{`⟨*start anchor*⟩`}{`⟨*end anchor*⟩`}` declares the subkey ⟨*link type*⟩ of the choice key `/pgfgantt/link type`. Such keys store the ⟨*label*⟩ in `\gtt@link@type`, the ⟨*start anchor*⟩ in `\gtt@link@startanchor` and the ⟨*end anchor*⟩ in `\gtt@link@endanchor`. `\@gtt@linkkeydef` `\gtt@link@type` `\gtt@link@startanchor` `\gtt@link@endanchor`

```
\def\@gtt@linkkeydef#1#2#3#4{%
  \pgfkeysdef{/pgfgantt/link type/#1}{%
    \def\gtt@link@type{#1}%
    \def\gtt@link@startanchor{#3}%
    \def\gtt@link@endanchor{#4}%
  }%
  \ganttset{link label={#1}{#2}}%
}

```

3.4 Option Declarations

`hgrid` checks whether its value is `false` and sets the boolean `\ifgtt@hgrid` accordingly. If the value is `true` or missing, horizontal grid lines appear `dotted`. `hgrid` `hgrid style` `\ifgtt@hgrid` `\gtt@hgridstyle`

```
\@gtt@stylekeydef{hgrid style}
\newif\ifgtt@hgrid
\pgfkeysdef{/pgfgantt/hgrid}{%
  \def\@tempa{#1}%
  \def\@tempb{false}%
  \ifx\@tempa\@tempb%
    \gtt@hgridfalse%
  \else%
    \gtt@hgridtrue%
    \def\@tempb{true}%
    \ifx\@tempa\@tempb%
      \def\gtt@hgridstyle{dotted}%
    \else%
```

```
      \def\gtt@hgridstyle{#1}%
    \fi%
  \fi%
}
\pgfkeys{/pgfgantt/hgrid/.default=dotted}
```

Analogously, we declare `vgrid`.

vgrid
\ifgtt@vgrid
\gtt@vgridstyle

```
\newif\ifgtt@vgrid
\pgfkeysdef{/pgfgantt/vgrid}{%
  \def\@tempa{#1}%
  \def\@tempb{false}%
  \ifx\@tempa\@tempb%
    \gtt@vgridfalse%
  \else%
    \gtt@vgridtrue%
    \def\@tempb{true}%
    \ifx\@tempa\@tempb%
      \def\gtt@vgridstyle{dotted}%
    \else%
      \def\gtt@vgridstyle{#1}%
    \fi%
  \fi%
}
\pgfkeys{/pgfgantt/vgrid/.default=dotted}
```

The following three keys store the basis vectors for the chart.

x unit
y unit title
y unit chart

```
\@gtt@keydef{x unit}
\@gtt@keydef{y unit title}
\@gtt@keydef{y unit chart}
```

Here is a set of keys related to the canvas ...

canvas
today
today rule
today label

```
\@gtt@stylekeydef{canvas}
\@gtt@keydef{today}
\@gtt@stylekeydef{today rule}
\@gtt@keydef{today label}
```

... and of keys that influence the title. Note that `\@gtt@keydef` cannot define `title list options`, since `\@gtt@titlelistoptions` is expanded after a `\foreach` statement, where `\@gtt@get` will not work.

title
title label font
title label anchor
title list options
title left shift
title right shift
title top shift
title height
\gtt@titlelistoptions

```
\@gtt@stylekeydef{title}
\@gtt@keydef{title label font}
\@gtt@stylekeydef{title label anchor}
\pgfkeysdef{/pgfgantt/title list options}{%
  \def\gtt@titlelistoptions{[#1]}%
```

```
}
\@gtt@keydef{title left shift}
\@gtt@keydef{title right shift}
\@gtt@keydef{title top shift}
\@gtt@keydef{title height}

```

`include title in canvas` is the only true boolean key in the package.

include title in canvas
\ifgtt@includetitle

```
\newif\ifgtt@includetitle
\pgfkeys{/pgfgantt/include title in canvas/.is if=gtt@includetitle}

```

The `name` key saves unique names for chart elements. The `time slot modifier` option controls the semi-intelligent behaviour of the package regarding the conversion of title slots to x-coordinates. A value of `0` essentially means "interpret all end time slots as x-coordinates".

name
time slot modifier

```
\@gtt@keydef{name}
\@gtt@keydef{time slot modifier}

```

Some standard key declarations for bars ...

bar
bar label text
bar label font
bar label anchor
bar left shift
bar right shift
bar top shift
bar height
\gtt@barlabeltext

```
\@gtt@stylekeydef{bar}
\pgfkeysdef{/pgfgantt/bar label text}{%
  \def\gtt@barlabeltext##1{#1}%
}
\@gtt@keydef{bar label font}
\@gtt@stylekeydef{bar label anchor}
\@gtt@keydef{bar left shift}
\@gtt@keydef{bar right shift}
\@gtt@keydef{bar top shift}
\@gtt@keydef{bar height}

```

... and groups.

group
group label text
group label font
group label anchor
group left shift
group right shift
group top shift
group height
\gtt@grouplabeltext

```
\@gtt@stylekeydef{group}
\pgfkeysdef{/pgfgantt/group label text}{%
  \def\gtt@grouplabeltext##1{#1}%
}
\@gtt@keydef{group label font}
\@gtt@stylekeydef{group label anchor}
\@gtt@keydef{group left shift}
\@gtt@keydef{group right shift}
\@gtt@keydef{group top shift}
\@gtt@keydef{group height}
```

`gantt left peak` checks for each of its three values whether it is non-empty and only then changes the corresponding length macro.

group left peak
\gtt@groupleftpeakmidx
\gtt@groupleftpeakinnerx
\gtt@groupleftpeaky

```
\pgfkeysdefnargs{/pgfgantt/group left peak}{3}{%
```

```
  \def\@tempa{#1}%
  \ifx\@tempa\@empty\else\def\gtt@groupleftpeakmidx{#1}\fi%
  \def\@tempa{#2}%
  \ifx\@tempa\@empty\else\def\gtt@groupleftpeakinnerx{#2}\fi%
  \def\@tempa{#3}%
  \ifx\@tempa\@empty\else\def\gtt@groupleftpeaky{#3}\fi%
}
```

`group right peak` works similar, but a `-` also counts as an empty value (the reason for this will soon become apparent).

group right peak
\gtt@grouprightpeakmidx
\gtt@grouprightpeakinnerx
\gtt@grouprightpeaky

```
\pgfkeysdefnargs{/pgfgantt/group right peak}{3}{%
  \def\@tempa{#1}%
  \def\@tempb{-}%
  \ifx\@tempa\@empty\else%
    \ifx\@tempa\@tempb\else\def\gtt@grouprightpeakmidx{#1}\fi%
  \fi%
  \def\@tempa{#2}%
  \ifx\@tempa\@empty\else%
    \ifx\@tempa\@tempb\else\def\gtt@grouprightpeakinnerx{#2}\fi%
  \fi%
  \def\@tempa{#3}%
  \ifx\@tempa\@empty\else\def\gtt@grouprightpeaky{#3}\fi%
}
```

`group peaks` simultaneously sets `group left peak` and `group right peak`. In order to preserve the symmetry of the peaks, the key adds a negative sign (i. e., a hyphen in the source code) to ⟨*groove x*⟩ and ⟨*inner x*⟩ of `group right peak`. Therefore, the latter key must interpret its first and second value as "empty" even if they contain a single hyphen.

group peaks

```
\pgfkeysdefnargs{/pgfgantt/group peaks}{3}{%
  \ganttset{group left peak={#1}{#2}{#3}, group right peak={-#1}{-#2}{#3}}%
}

```

The keys below manage the progress elements. Note the way in which we declare `progress label text`, so that a `#1` in its value is replaced by the argument of `\gtt@progresslabeltext`.

progress
bar incomplete
group incomplete
incomplete
progress label text
progress label font
progress label anchor
\gtt@progresslabeltext

```
\@gtt@keydef{progress}
\@gtt@stylekeydef{bar incomplete}
\@gtt@stylekeydef{group incomplete}
\pgfkeysdef{/pgfgantt/incomplete}{%
  \ganttset{bar incomplete={#1}, group incomplete={#1}}%
}
\pgfkeysdef{/pgfgantt/progress label text}{%
  \def\gtt@progresslabeltext##1{#1}%
}
\@gtt@keydef{progress label font}
\@gtt@stylekeydef{progress label anchor}
```

Here are the declarations of the milestone-related keys.

milestone
milestone label text
milestone label font
milestone label anchor
milestone width
milestone height
milestone xshift
milestone yshift
\gtt@milestonelabeltext

```
\@gtt@stylekeydef{milestone}
\pgfkeysdef{/pgfgantt/milestone label text}{%
  \def\gtt@milestonelabeltext##1{#1}%
}
\@gtt@keydef{milestone label font}
\@gtt@stylekeydef{milestone label anchor}
\@gtt@keydef{milestone width}
\@gtt@keydef{milestone height}
\@gtt@keydef{milestone xshift}
\@gtt@keydef{milestone yshift}
```

Next, we declare the keys that modify links and the five link types.

link
link type
link label
link label font
link label anchor
link mid
link bulge
link tolerance

```
\@gtt@stylekeydef{link}
\pgfkeysdefnargs{/pgfgantt/link label}{2}{%
  \expandafter\def\csname gtt@link@#1@labeltext\endcsname{#2}%
}
\pgfkeys{/pgfgantt/link type/.is choice}
\@gtt@linkkeydef{arrow}{\relax}{}{}
\@gtt@linkkeydef{s-s}{start-to-start}{lower left}{upper left}
\@gtt@linkkeydef{s-f}{start-to-finish}{lower left}{upper right}
\@gtt@linkkeydef{f-s}{finish-to-start}{lower right}{upper left}
\@gtt@linkkeydef{f-f}{finish-to-finish}{lower right}{upper right}

\@gtt@keydef{link label font}
\@gtt@stylekeydef{link label anchor}
\@gtt@keydef{link mid}
\@gtt@keydef{link bulge}
\@gtt@keydef{link tolerance}
```

Finally, we initialize all keys in order to define the commands that save their values.

```
\ganttset{
  canvas={fill=white},
  x unit=.5cm,
  y unit title=1cm,
  y unit chart=1cm,
  hgrid style=dotted,
  today=none,
  today rule={dashed, line width=1pt},
  today label=TODAY,
  title={fill=white},
  title label font=\small,
  title label anchor={anchor=mid},
  title left shift=0,
  title right shift=0,
```

```
  title top shift=0,
  title height=.6,
  title list options={var=\x, evaluate=\x},
  include title in canvas,
  name={},
  time slot modifier=-1,
  bar={fill=white},
  bar label text={\strut#1},
  bar label font=\normalsize,
  bar label anchor={anchor=east},
  bar left shift=0,
  bar right shift=0,
  bar top shift=.3,
  bar height=.4,
  group={fill=black},
  group label text={\strut#1},
  group label font=\normalsize\bfseries,
  group label anchor={anchor=east},
  group left shift=-.1,
  group right shift=.1,
  group top shift=.4,
  group height=.2,
  group peaks={.2}{.4}{.1},
  progress=none,
  incomplete={fill=black!25},
  progress label text={#1\% complete},
  progress label font=\scriptsize,
  progress label anchor={anchor=west},
  milestone={fill=black},
  milestone label text={\strut#1},
  milestone label font=\normalsize\itshape,
  milestone label anchor={anchor=east},
  milestone width=.8,
  milestone height=.4,
  milestone xshift=0,
  milestone yshift=.5,
  link={-latex, rounded corners=1pt},
  link type=arrow,
  link label font=\scriptsize\itshape\normalcolor,
  link label anchor={anchor=west},
  link mid=.5,
  link bulge=.4,
  link tolerance=0.6
}
```

3.5 The Horizontal and Vertical Grid

The `\gtt@vgrid@do` macro decomposes the style list for the vertical grid into its comma-separated items. The item is analyzed (see below) only if some grid lines are still left to draw. Note the "elegant" quadruple `\expandafter` construction, which enables tail recursion. `\gtt@vgrid@do`

```
\def\gtt@vgrid@do#1,{%
  \ifx\relax#1\else%
    \ifnum\value{gtt@currgrid}>\value{gtt@width}\else%
      \gtt@vgrid@analyze#1\relax%
      \expandafter\expandafter\expandafter\gtt@vgrid@do%
    \expandafter\fi%
  \fi%
}
```

In the absence of a star as the first token in a style list item, `\gtt@vgrid@analyze` adds the multiplier `1` to the input stream. `\gtt@vgrid@analyze`

```
\def\gtt@vgrid@analyze{%
  \@ifstar{\gtt@vgrid@draw}{\gtt@vgrid@draw1}%
}
```

`\gtt@vgrid@draw` draws as many grid lines as required by the multiplier. It increases `gtt@currgrid` after each line drawn and breaks the loop as soon as all grid rules have been drawn. `\gtt@vgrid@draw`

```
\def\gtt@vgrid@draw#1#2\relax{%
  \foreach \i in {1,...,#1} {%
    \draw[#2] (\value{gtt@currgrid} * \@gtt@get{x unit}, \y@upper pt) --%
      (\value{gtt@currgrid} * \@gtt@get{x unit}, \y@lower pt);%
    \stepcounter{gtt@currgrid}%
    \ifnum\value{gtt@currgrid}>\value{gtt@width}\breakforeach\fi%
  }%
}
```

The corresponding macros for the horizontal grid work like their vertical grid analogues. `\gtt@hgrid@do` `\gtt@hgrid@analyze` `\gtt@hgrid@draw`

```
\def\gtt@hgrid@do#1,{%
  \ifx\relax#1\else
    \ifnum\value{gtt@currgrid}<\value{gtt@currentline}\else%
      \gtt@hgrid@analyze#1\relax%
      \expandafter\expandafter\expandafter\gtt@hgrid@do%
    \expandafter\fi%
  \fi%
}

\def\gtt@hgrid@analyze{%
```

```
  \@ifstar{\gtt@hgrid@draw}{\gtt@hgrid@draw1}%
}

\def\gtt@hgrid@draw#1#2\relax{%
  \foreach \i in {1,...,#1} {%
    \pgfmathsetmacro\y@upper{%
      \value{gtt@lasttitleline} * \@gtt@get{y unit title} +%
      (\value{gtt@currgrid} - \value{gtt@lasttitleline})%
      * \@gtt@get{y unit chart}%
    }%
    \draw[#2] (0pt, \y@upper pt) --
      (\value{gtt@width} * \@gtt@get{x unit}, \y@upper pt);%
    \addtocounter{gtt@currgrid}{-1}%
    \ifnum\value{gtt@currgrid}<\value{gtt@currentline}\breakforeach\fi%
  }%
}

```

3.6 The Main Environment

At the beginning of a `ganttchart` environment, the keys in its optional argument are executed. `gtt@width` saves the environment's mandatory argument (i. e., the number of time slots). All counters are set to `0`. Since we expect a chart to start with at least one title element, `\ifgtt@intitle` is true. Within the environment, the control symbol `\\` is equivalent to `\ganttnewline` (similar to the syntax of a LaTeX table).

ganttchart
\\

```
\newenvironment{ganttchart}[2][]{%
  \ganttset{#1}%
  \setcounter{gtt@width}{#2}%
  \setcounter{gtt@currentline}{0}%
  \setcounter{gtt@lasttitleline}{0}%
  \setcounter{gtt@elementid}{0}%
  \setcounter{gtt@currgrid}{1}%
  \gtt@intitletrue%
  \let\\\ganttnewline%
}{%
```

After the contents of the environment have been drawn, we add the canvas to the background layer. The `ganttchart` environment and all `\gantt...` macros save their x- and y-coordinates in local internal macros called `\x@left`, `\x@right`, `\y@upper` and `\y@lower`. The upper y-coordinate of the canvas is either zero or excludes the title lines if `include title in canvas` is false. The lower y-coordinate must take into account different y-units in the title and the rest of the chart.

\x@left
\x@right
\y@upper
\y@lower

```
  \begin{scope}[on background layer]%
    \ifgtt@includetitle%
      \def\y@upper{0}%
    \else%
```

```
      \pgfmathsetmacro\y@upper{%
        \value{gtt@lasttitleline} * \@gtt@get{y unit title}%
      }%
    \fi%
    \pgfmathsetmacro\y@lower{%
      \value{gtt@lasttitleline} * \@gtt@get{y unit title}%
      + (\value{gtt@currentline} - \value{gtt@lasttitleline} - 1)%
      * \@gtt@get{y unit chart}%
    }%
    \draw[/pgfgantt/canvas@style]
      (0pt, \y@upper pt) rectangle
      (\value{gtt@width} * \@gtt@get{x unit}, \y@lower pt);%
    \pgfmathsetmacro\y@upper{%
      \value{gtt@lasttitleline} * \@gtt@get{y unit title}%
    }%
```

The contents of the vertical grid style list are evaluated at most `gtt@width`-times, but the loop breaks as soon as all grid lines have been drawn.

```
    \ifgtt@vgrid
      \addtocounter{gtt@width}{-1}%
      \foreach \x in {1,...,\value{gtt@width}} {%
        \expandafter\gtt@vgrid@do\gtt@vgridstyle,\relax,%
        \ifnum\value{gtt@currgrid}>\value{gtt@width}\breakforeach\fi%
      }%
      \stepcounter{gtt@width}%
    \fi%
```

Now, we draw the horizontal grid. If we exclude the title from the canvas, we omit the uppermost horizontal grid line since it would coincide with the canvas border.

`\hgrid@upper`

```
    \ifgtt@hgrid%
      \ifgtt@includetitle%
        \setcounter{gtt@currgrid}{\value{gtt@lasttitleline}}%
      \else%
        \pgfmathsetcounter{gtt@currgrid}{\value{gtt@lasttitleline}-1}%
      \fi%
      \edef\hgrid@upper{\thegtt@currgrid}%
      \foreach \t in {\hgrid@upper,...,\value{gtt@currentline}} {%
        \expandafter\gtt@hgrid@do\gtt@hgridstyle,\relax,%
        \ifnum\value{gtt@currgrid}<\value{gtt@currentline}\breakforeach\fi%
      }%
    \fi%
```

The last task of `ganttchart` is to apply the `today` key if its value differs from `none`.

```
    \def\@tempa{none}%
    \ifx\gtt@today\@tempa\else%
      \draw[/pgfgantt/today rule@style]
        (\@gtt@get{today} * \@gtt@get{x unit}, \y@upper pt) --
```

```
        (\@gtt@get{today} * \@gtt@get{x unit}, \y@lower pt);%
      \node at (\@gtt@get{today} * \@gtt@get{x unit}, \y@lower pt)
        [anchor=north] {\@gtt@get{today label}};%
    \fi%
  \end{scope}%
}
```

3.7 Starting a New Line

\ganttnewline

Unless the optional argument of `\ganttnewline` is empty, this macro adds a horizontal grid rule between the current and the new line. The style of this line is either `hgrid style` or the style specified in the optional argument. Anyway, `\ganttnewline` decreases `gtt@currentline` and, if we are still in the title, `gtt@lasttitleline`. Since the new line starts at time slot zero, `gtt@lasttitleslot` is reset.

```
\newcommand\ganttnewline[1][]{%
  \def\@tempa{#1}%
  \def\@tempb{grid}%
  \ifx\@tempa\@empty\else
    \ifx\@tempa\@tempb%
      \def\@tempa{/pgfgantt/hgrid style@style}%
    \fi%
    \pgfmathsetmacro\y@upper{%
      \value{gtt@lasttitleline} * \@gtt@get{y unit title}%
      + (\value{gtt@currentline} - \value{gtt@lasttitleline} - 1)%
      * \@gtt@get{y unit chart}%
    }
    \expandafter\draw\expandafter[\@tempa]
      (0pt, \y@upper pt) --
      (\value{gtt@width} * \@gtt@get{x unit}, \y@upper pt);%
  \fi%
  \addtocounter{gtt@currentline}{-1}%
  \ifgtt@intitle\addtocounter{gtt@lasttitleline}{-1}\fi%
  \setcounter{gtt@lasttitleslot}{0}%
}
```

3.8 Title Elements

\gantttitle

`\gantttitle` draws a title element (i. e., a rectangle with a single node at its center). For reasons that will become clear below, the rectangle essentially starts at the x-coordinate stored in `gtt@lasttitleslot`. This counter is updated at the end of the macro.

Note that in order to keep key changes local, all macros that draw chart elements set the keys specified as their optional argument within a group.

```
\newcommand\gantttitle[3][]{%
  \begingroup%
```

```
  \ganttset{#1}%
  \pgfmathsetmacro\x@left{%
    (\value{gtt@lasttitleslot} + \@gtt@get{title left shift})%
    * \@gtt@get{x unit}%
  }%
  \pgfmathsetmacro\x@right{%
    (\value{gtt@lasttitleslot} + #3 + \@gtt@get{title right shift})%
    * \@gtt@get{x unit}%
  }%
  \pgfmathsetmacro\y@upper{%
    (\value{gtt@currentline} - \@gtt@get{title top shift})%
    * \@gtt@get{y unit title}%
  }%
  \pgfmathsetmacro\y@lower{%
    (\value{gtt@currentline} - \@gtt@get{title top shift}%
    - \@gtt@get{title height}) * \@gtt@get{y unit title}%
  }%
  \draw[/pgfgantt/title@style]
    (\x@left pt, \y@upper pt) rectangle
    (\x@right pt, \y@lower pt);%
  \@gtt@get{title label font}%
  \node at ($(\x@left pt,\y@upper pt)!.5!(\x@right pt,\y@lower pt)$)
    [/pgfgantt/title label anchor@style] {#2};%
  \addtocounter{gtt@lasttitleslot}{#3}%
  \endgroup%
}
```

`\ganttitlelist` generates title elements by repeatedly calling `\ganttitle`. Since the latter always starts after the last time slot occupied by the previous element, `\ganttitlelist` does not have to calculate the respective x-coordinates explicitly. `\ganttitlelist`

```
\newcommand\ganttitlelist[3][]{%
  \begingroup%
  \ganttset{#1}%
  \expandafter\foreach\gtt@titlelistoptions in {#2} {\ganttitle{\x}{#3}}%
  \endgroup%
}
```

3.9 Chart Elements

All chart elements that can be linked (i. e. bars, groups and milestones) add a node of shape `chart element`, whose name equals the value of the `name` key (or "`elem`⟨*number*⟩" if `name` is empty). A `chart element` node has an anchor in the center of the chart element (`center`) and six anchors at the top, middle and bottom of the element's sides. The `\ganttlink` macro relies on these anchors for calculating the link coordinates.

```
\pgfdeclareshape{chart element}{%
  \savedanchor\lowerleft{%
    \pgfpoint{\x@left pt}{\y@lower pt}%
  }%
  \savedanchor\leftcenter{%
    \pgfpoint{\x@left pt}{\y@upper pt / 2 + \y@lower pt / 2}%
  }%
  \savedanchor\upperleft{%
    \pgfpoint{\x@left pt}{\y@upper pt}%
  }%
  \savedanchor\lowerright{%
    \pgfpoint{\x@right pt}{\y@lower pt}%
  }%
  \savedanchor\rightcenter{%
    \pgfpoint{\x@right pt}{\y@upper pt / 2 + \y@lower pt / 2}%
  }%
  \savedanchor\upperright{%
    \pgfpoint{\x@right pt}{\y@upper pt}%
  }%
  \savedanchor\centerpoint{%
    \pgfpoint{\x@right pt / 2 + \x@left pt / 2}%
      {\y@upper pt / 2 + \y@lower pt / 2}%
  }%
  \anchor{lower left}{\lowerleft}%
  \anchor{left}{\leftcenter}%
  \anchor{upper left}{\upperleft}%
  \anchor{lower right}{\lowerright}%
  \anchor{right}{\rightcenter}%
  \anchor{upper right}{\upperright}%
  \anchor{center}{\centerpoint}%
}
```

3.10 Bars

`\ganttbar` first defines the usual coordinate macros and adds a `chart element` node. This node is called `elem`⟨*number*⟩ if the `name` key is empty. `\ganttbar` `\gtt@name`

```
\newcommand\ganttbar[4][]{%
  \begingroup%
  \ganttset{#1}%
  \pgfmathsetmacro\x@left{%
    (#3 + \@gtt@get{time slot modifier} + \@gtt@get{bar left shift})%
    * \@gtt@get{x unit}%
  }%
  \pgfmathsetmacro\x@right{%
    (#4 + \@gtt@get{bar right shift}) * \@gtt@get{x unit}%
  }%
  \pgfmathsetmacro\y@upper{%
```

```
    \value{gtt@lasttitleline} * \@gtt@get{y unit title}
    + (\value{gtt@currentline} - \value{gtt@lasttitleline}
    - \@gtt@get{bar top shift}) * \@gtt@get{y unit chart}%
  }%
  \pgfmathsetmacro\y@lower{%
    \y@upper - \@gtt@get{bar height} * \@gtt@get{y unit chart}%
  }%
  \ifx\gtt@name\@empty\edef\gtt@name{elem\thegtt@elementid}\fi%
  \node[shape=chart element] (\@gtt@get{name})
    at ($(\x@left pt, \y@upper pt)!.5!(\x@right pt, \y@lower pt)$) {};
```

If the first mandatory argument of `\ganttbar` is not empty, we print a label with its anchor at the left canvas border halfway between the upper and lower y-coordinate of the bar.

```
  \def\@tempa{#2}%
  \ifx\@tempa\@empty\else%
    \node at ($(0pt, \y@upper pt)!.5!(0pt, \y@lower pt)$)
      [/pgfgantt/bar label anchor@style]
      {\@gtt@get{bar label font}{\gtt@barlabeltext{#2}}};%
  \fi%
```

`\gtt@pl@draw` saves the commands that will produce the progress label. This macro does nothing unless (a) the `progress` key differs from `none` and (b) `progress label text` differs from `\relax`. Otherwise, it creates a vertically centered node to the right of the bar.

`\gtt@pl@draw`

```
  \def\@tempa{none}%
  \ifx\gtt@progress\@tempa%
    \def\gtt@progress{100}%
    \let\gtt@pl@draw\relax%
  \else
    \expandafter\ifx\gtt@progresslabeltext\relax\relax%
      \let\gtt@pl@draw\relax%
    \else%
      \def\gtt@pl@draw{%
        \node at ($(\x@right pt, \y@upper pt)!.5!
          (\x@right pt, \y@lower pt)$)
          [/pgfgantt/progress label anchor@style] {%
            \@gtt@get{progress label font}{%
              \@gtt@get{progresslabeltext}{\@gtt@get{progress}}%
            }%
          };%
      }%
    \fi%
  \fi%
```

In order to draw the left (complete) and right (incomplete) part of a progress bar, we clip the corresponding rectangles depending on the value of `progress`. Note that we turn off the border of these rectangles and draw it with an additional, third

command.

```
  \begin{scope}%
    \clip (\x@left pt, \y@upper pt) rectangle
      ($(\x@left pt, \y@lower pt)!\gtt@progress/100!
        (\x@right pt, \y@lower pt)$);%
    \draw[/pgfgantt/bar@style, draw=none] (\x@left pt, \y@upper pt)
      rectangle (\x@right pt, \y@lower pt);%
  \end{scope}%
  \begin{scope}%
    \clip ($(\x@left pt, \y@upper pt)!\gtt@progress/100!
      (\x@right pt, \y@upper pt)$)
      rectangle (\x@right pt, \y@lower pt);%
    \draw[/pgfgantt/bar incomplete@style, draw=none]
      (\x@left pt, \y@upper pt) rectangle (\x@right pt, \y@lower pt);%
  \end{scope}%
  \draw[/pgfgantt/bar@style, fill=none]
    (\x@left pt, \y@upper pt) rectangle (\x@right pt, \y@lower pt);%
  \gtt@pl@draw%
```

Since the first bar clearly appears after the last line containing a title element, we set the boolean `\ifgtt@intitle` to false.

```
  \xdef\gtt@lastelement{\gtt@currentelement}%
  \xdef\gtt@currentelement{\@gtt@get{name}}%
  \stepcounter{gtt@elementid}%
  \global\gtt@intitlefalse%
  \endgroup%
}
```

The shortcut version `\ganttlinkedbar` calls both `\ganttbar` and `\ganttlink`. `\ganttlinkedbar`

```
\newcommand\ganttlinkedbar[4][]{%
  \begingroup%
  \ganttset{#1}%
  \ganttbar{#2}{#3}{#4}%
  \ganttlink{\gtt@lastelement}{\gtt@currentelement}
  \endgroup%
}
```

3.11 Links

`\ganttlink` takes two completely different approaches to drawing links, depending on the link type. First, we recall the coordinates of the chart elements to be linked. `\ganttlink`

```
\newcommand\ganttlink[3][]{%
  \begingroup%
  \ganttset{#1}%
  \pgfpointanchor{#2}{right}
  \edef\x@left{\the\pgf@x}
```

```
\edef\y@upper{\the\pgf@y}
\pgfpointanchor{#3}{left}
\edef\x@right{\the\pgf@x}
\edef\y@lower{\the\pgf@y}
```

Arrow-like links: The first and last coordinate of the link should touch the preceding or following element at the center of its right or left border, respectively. We check if the connected elements lie in the same row or not (i. e., their y-coordinates differ at most 1 pt). In the latter case, `\pgfmathparse` yields `0`.

```
\def\@tempa{arrow}
\ifx\@tempa\gtt@link@type%
  \pgfmathparse{abs(\y@upper - \y@lower) <= 1}
  \ifcase\pgfmathresult%
```

Once again, two possibilities arise: Either the elements to be connected are at least separated by `link tolerance` time slots, in which case we draw a three-part arrow. Alternatively, the elements lie in adjacent time slots or even overlap, in which case we draw a five-part arrow.

```
    \pgfmathparse{%
      (\x@right - \x@left)
      >= \@gtt@get{link tolerance} * \@gtt@get{x unit}%
    }%
    \ifcase\pgfmathresult%
      \draw[/pgfgantt/link@style]
        (#2.right) --
        (\x@left + \@gtt@get{link bulge} * \@gtt@get{x unit},
          \y@upper) --
        ($(\x@left + \@gtt@get{link bulge} * \@gtt@get{x unit},
          \y@upper)!%
          \@gtt@get{link mid}!%
          (\x@left + \@gtt@get{link bulge} * \@gtt@get{x unit},
          \y@lower)$) --
        ($(\x@right - \@gtt@get{link bulge} * \@gtt@get{x unit},
          \y@upper)!%
          \@gtt@get{link mid}!%
          (\x@right - \@gtt@get{link bulge} * \@gtt@get{x unit},
          \y@lower)$) --
        (\x@right - \@gtt@get{link bulge} * \@gtt@get{x unit},
          \y@lower) --
        (#3.left);%
    \else%
      \draw[/pgfgantt/link@style]
        (#2.right) --
        ($(\x@left, \y@upper)!\@gtt@get{link mid}!
          (\x@right, \y@upper)$) --
        ($(\x@left, \y@lower)!\@gtt@get{link mid}!
          (\x@right, \y@lower)$) --
        (#3.left);%
```

```
      \fi%
```

For elements that lie in the same row, we draw a simple arrow.

```
    \else
      \draw[/pgfgantt/link@style] (#2.right) -- (#3.left);
    \fi
```

Straight links: A simple arrow connects the two elements. The label is anchored halfway between the start and end coordinates.

```
  \else%
    \draw[/pgfgantt/link@style]
      (#2.\gtt@link@startanchor) -- (#3.\gtt@link@endanchor)
      node [pos=.5, /pgfgantt/link label anchor@style] {%
        \@gtt@get{link label font}{%
          \csname gtt@link@\gtt@link@type @labeltext\endcsname%
        }%
      };%
  \fi%
  \endgroup%
}
```

3.12 Groups

Groups and bars are quite similar. First, we define the usual coordinate macros and add a `chart element` node.

\ganttgroup

```
\newcommand\ganttgroup[4][]{%
  \begingroup%
  \ganttset{#1}%
  \pgfmathsetmacro\x@left{%
    (#3 + \@gtt@get{time slot modifier} + \@gtt@get{group left shift})%
    * \@gtt@get{x unit}%
  }%
  \pgfmathsetmacro\x@right{%
    (#4 + \@gtt@get{group right shift}) * \@gtt@get{x unit}%
  }%
  \pgfmathsetmacro\y@upper{%
    \value{gtt@lasttitleline} * \@gtt@get{y unit title}
    + (\value{gtt@currentline} - \value{gtt@lasttitleline}
    - \@gtt@get{group top shift}) * \@gtt@get{y unit chart}%
  }%
  \pgfmathsetmacro\y@lower{%
    \y@upper - \@gtt@get{group height} * \@gtt@get{y unit chart}%
  }%
  \ifx\gtt@name\@empty\edef\gtt@name{elem\thegtt@elementid}\fi%
  \node[shape=chart element] (\@gtt@get{name})
    at ($(\x@left pt, \y@upper pt)!.5!(\x@right pt, \y@lower pt)$) {};
```

If the first mandatory argument of `\ganttgroup` is not empty, we print a label with its anchor at the left canvas border halfway between the upper and lower y-coordinate of the group.

```
\def\@tempa{#2}%
\ifx\@tempa\@empty\else%
  \node at ($(0pt, \y@upper pt)!.5!(0pt, \y@lower pt)$)
    [/pgfgantt/group label anchor@style]
    {\@gtt@get{group label font}{\gtt@grouplabeltext{#2}}};%
\fi%
```

`\gtt@pl@draw` saves the commands that will produce the progress label. This macro does nothing unless (a) the `progress` key differs from none and (b) `progress label text` differs from `\relax`. Otherwise, it creates a vertically centered node to the right of the group.

```
\def\@tempa{none}%
\ifx\gtt@progress\@tempa%
  \def\gtt@progress{100}%
  \let\gtt@pl@draw\relax%
\else
  \expandafter\ifx\gtt@progresslabeltext\relax\relax%
    \let\gtt@pl@draw\relax%
  \else%
    \def\gtt@pl@draw{%
      \node at ($(\x@right pt, \y@upper pt)!.5!
        (\x@right pt, \y@lower pt)$)
        [/pgfgantt/progress label anchor@style] {%
          \@gtt@get{progress label font}{%
            \@gtt@get{progresslabeltext}{\@gtt@get{progress}}%
          }%
        };%
    }%
  \fi%
\fi%
```

In order to draw the left (complete) and right (incomplete) part of a progress group, we clip the corresponding polygons depending on the value of `progress`. Note that we turn off the border of these polygons and draw it with an additional, third command. The clipped area must include the highest peak, so we determine its height and store it in `\@maxpeak`.

`\@maxpeak`

```
\pgfmathsetmacro\@maxpeak{%
  \gtt@grouprightpeaky > \gtt@groupleftpeaky ?%
  \gtt@grouprightpeaky * \@gtt@get{y unit chart} :%
  \gtt@groupleftpeaky * \@gtt@get{y unit chart}%
}%
\begin{scope}%
  \clip (\x@left pt, \y@upper pt) rectangle
    ($(\x@left pt, \y@lower pt - \@maxpeak pt)!%
```

```
      \gtt@progress/100!%
      (\x@right pt, \y@lower pt - \@maxpeak pt)$);%
  \path[/pgfgantt/group@style,draw=none]
    (\x@left pt, \y@upper pt) --
    (\x@right pt, \y@upper pt) --
    (\x@right pt, \y@lower pt) --
    (\x@right pt + \gtt@grouprightpeakmidx * \@gtt@get{x unit},
      \y@lower pt - \gtt@grouprightpeaky * \@gtt@get{y unit chart}) --
    (\x@right pt + \gtt@grouprightpeakinnerx * \@gtt@get{x unit},
      \y@lower pt) --
    (\x@left pt + \gtt@groupleftpeakinnerx * \@gtt@get{x unit},
      \y@lower pt) --
    (\x@left pt + \gtt@groupleftpeakmidx * \@gtt@get{x unit},
      \y@lower pt - \gtt@groupleftpeaky * \@gtt@get{y unit chart}) --
    (\x@left pt, \y@lower pt) --
    cycle;%
\end{scope}%
\begin{scope}%
  \clip ($(\x@left pt, \y@upper pt)!%
      \gtt@progress/100!%
      (\x@right pt, \y@upper pt)$)
    rectangle (\x@right pt, \y@lower pt - \@maxpeak pt);
  \path[/pgfgantt/group incomplete@style]
    (\x@left pt, \y@upper pt) --
    (\x@right pt, \y@upper pt) --
    (\x@right pt, \y@lower pt) --
    (\x@right pt + \gtt@grouprightpeakmidx * \@gtt@get{x unit},
      \y@lower pt - \gtt@grouprightpeaky * \@gtt@get{y unit chart}) --
    (\x@right pt + \gtt@grouprightpeakinnerx * \@gtt@get{x unit},
      \y@lower pt) --
    (\x@left pt + \gtt@groupleftpeakinnerx * \@gtt@get{x unit},
      \y@lower pt) --
    (\x@left pt + \gtt@groupleftpeakmidx * \@gtt@get{x unit},
      \y@lower pt - \gtt@groupleftpeaky * \@gtt@get{y unit chart}) --
    (\x@left pt, \y@lower pt) --
    cycle;%
\end{scope}%
\path[/pgfgantt/group@style,fill=none]
  (\x@left pt, \y@upper pt) --
  (\x@right pt, \y@upper pt) --
  (\x@right pt, \y@lower pt) --
  (\x@right pt + \gtt@grouprightpeakmidx * \@gtt@get{x unit},
    \y@lower pt - \gtt@grouprightpeaky * \@gtt@get{y unit chart}) --
  (\x@right pt + \gtt@grouprightpeakinnerx * \@gtt@get{x unit},
    \y@lower pt) --
  (\x@left pt + \gtt@groupleftpeakinnerx * \@gtt@get{x unit},
    \y@lower pt) --
  (\x@left pt + \gtt@groupleftpeakmidx * \@gtt@get{x unit},
    \y@lower pt - \gtt@groupleftpeaky * \@gtt@get{y unit chart}) --
```

```
    (\x@left pt, \y@lower pt) --
    cycle;%
  \gtt@pl@draw%
  \xdef\gtt@lastelement{\gtt@currentelement}%
  \xdef\gtt@currentelement{\@gtt@get{name}}%
  \stepcounter{gtt@elementid}%
  \global\gtt@intitlefalse%
  \endgroup%
}
```

The shortcut version `\ganttlinkedgroup` calls both `\ganttgroup` and `\ganttlink`. `\ganttlinkedgroup`

```
\newcommand\ganttlinkedgroup[4][]{%
  \begingroup%
  \ganttset{#1}%
  \ganttgroup{#2}{#3}{#4}%
  \ganttlink{\gtt@lastelement}{\gtt@currentelement}
  \endgroup%
}
```

3.13 Milestones

`\ganttmilestone` calculates some coordinates and adds a `chart element` node. We also need the coordinates of the center, which are saved in `\x@mid` and `\y@mid`. `\ganttmilestone` `\x@mid` `\y@mid`

```
\newcommand\ganttmilestone[3][]{%
  \begingroup%
  \ganttset{#1}%
  \pgfmathsetmacro\x@mid{%
    (#3 + \@gtt@get{milestone xshift}) * \@gtt@get{x unit}%
  }%
  \pgfmathsetmacro\x@left{%
    \x@mid - \@gtt@get{milestone width} / 2 * \@gtt@get{x unit}%
  }
  \pgfmathsetmacro\x@right{%
    \x@mid + \@gtt@get{milestone width} / 2 * \@gtt@get{x unit}%
  }
  \pgfmathsetmacro\y@mid{%
    \value{gtt@lasttitleline} * \@gtt@get{y unit title}%
    + (\value{gtt@currentline} - \value{gtt@lasttitleline}%
    - \@gtt@get{milestone yshift}) * \@gtt@get{y unit chart}%
  }%
  \pgfmathsetmacro\y@upper{%
    \y@mid + \@gtt@get{milestone height} / 2 * \@gtt@get{y unit chart}%
  }%
  \pgfmathsetmacro\y@lower{%
    \y@mid - \@gtt@get{milestone height} / 2 * \@gtt@get{y unit chart}%
  }%
```

```
  \ifx\gtt@name\@empty\edef\gtt@name{elem\thegtt@elementid}\fi%
  \node[shape=chart element] (\@gtt@get{name})
    at ($(\x@left pt, \y@upper pt)!.5!(\x@right pt, \y@lower pt)$) {};
```

If the first mandatory argument of `\ganttmilestone` is not empty, we print a label whose anchor lies on the left canvas border at the height of the milestone's center.

```
  \def\@tempa{#2}%
  \ifx\@tempa\@empty\else%
    \node at (0pt, \y@mid pt)
      [/pgfgantt/milestone label anchor@style]
      {\@gtt@get{milestone label font}{\gtt@milestonelabeltext{#2}}};%
  \fi%
```

Drawing the milestone itself is quite simple, since the `progress` key is irrelevant.

```
  \path[/pgfgantt/milestone@style]
    (\x@left pt, \y@mid pt) --
    (\x@mid pt, \y@lower pt) --
    (\x@right pt, \y@mid pt) --
    (\x@mid pt, \y@upper pt) --
    cycle;%
  \xdef\gtt@lastelement{\gtt@currentelement}%
  \xdef\gtt@currentelement{\@gtt@get{name}}%
  \stepcounter{gtt@elementid}%
  \global\gtt@intitlefalse%
  \endgroup%
}

```

The shortcut version `\ganttlinkedmilestone` calls both `\ganttmilestone` and `\ganttlink`.

```
\newcommand\ganttlinkedmilestone[3][]{%
  \begingroup%
  \ganttset{#1}%
  \ganttmilestone{#2}{#3}%
  \ganttlink{\gtt@lastelement}{\gtt@currentelement}
  \endgroup%
}
```

4 Index

Numbers written in bold refer to the page where the corresponding entry is described; numbers in italic refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

H

I

L

M

N

O

5 Change History

v1.0

General: Initial release 1

v1.1

General: `bar label text` configures the text of a bar label. 17

`group label text` configures the text of a group label. 20

`link tolerance` decides whether a five- or a three-part link is drawn. 28

`milestone label text` configures the text of a milestone label. . . . 25

The `time slot modifier` key has been added. If set to zero, all x-coordinates are interpreted as given, without regarding them as time slots. 15

The `vgrid lines list` key now determines the number of vertical grid lines drawn. 7

The introduction now clarifies what I mean by "a current PGF installation". 2

v2.0

General: Added style lists for the horizontal and vertical grid. 6

Completely rewrote the calculation of coordinates. 5

Removed the `hgrid shift` and `last line height` keys. 7

Removed the `vgrid lines list` key, as its behaviour can be simulated by an appropriate ⟨*style list*⟩ for `vgrid`. 7

Removed the `vgrid style` key. . . 6

The `x unit`, `y unit title` and `y unit chart` keys now specify the width of time slots and the height of title or chart lines, respectively. Thus, one can draw titles whose height differs from the rest of the chart. Furthermore, the x- and y-dimensions of the chart are independent of the dimensions of the surrounding `tikzpicture`. 5

The optional argument of `\ganttnewline` now also takes a style. 9

The syntax of `\ganttlink` was completely changed. The command now takes one optional and *two* mandatory arguments. The latter specify the name of the chart elements to be linked. Consequently, the keys `b-b`, `b-m`, `m-b` and `m-m` were removed. The keys `s-s`, `s-f`, `f-s` and `f-f` are now values for the `link type` key. . . . 26