The talk Document Class*

Martin Wiebusch

September 1, 2025

Abstract

The talk document class allows you to create slides for screen presentations or printing on transparencies. It also allows you to print personal notes for your talk. You can create overlays and display structure information (current section / subsection, table of contents) on your slides. The main feature that distinguishes talk from other presentation classes like beamer, prosper or powerdot is that it allows the user to define an arbitrary number of *layouts* and switch between these layouts from slide to slide. For examle, the default layout of the talk-sidebars style shows a (navigatable) table of contents in a bar on the right side of each slide, but it also provides a `nosidebar` layout where the sidebar is removed to make space for more content. This way you can present information in different formats but still maintain a consistent design throughout your presentation.

The talk class makes no restrictions on the slide design whatsoever. The entire look and feel of the presentation can be defined by the user. The style definitions should be put in a separate sty file. Currently the package comes with two pre-defined slide styles: `talk-simple.sty` and `talk-sidebars.sty`. The former is a simple design for short talks while the latter is better for longer seminars with content structured into sections and subsections. Customising your presentation style is easy and `talk-simple.sty` contains lots of comments to guide you through the process.

*This file has version number 2.0, last revised 31 August 2025

Contents

1 Installation and Requirements **2**

2 Using talk **3**

2.1 Class Options 3

2.2 Style Packages and Layouts 5

2.3 Global Specifications 7

2.4 Environments 7

2.5 Positioning Tools 9

2.6 Title and Contents Pages 10

3 The talk Class for Package Writers **11**

3.1 Mode Conditionals 11

3.2 Slide Dimensions 12

3.3 Global Specifications 12

3.4 Counters 12

3.5 Layouts 13

3.6 Typesetting Slides 14

3.7 The Table of Contents 16

3.8 Paragraph Spacing 18

4 Contact **19**

1 Installation and Requirements

The talk class requires the packages environ, pgf version 1.18 or above, graphicx, xstring and multido. The `talk-sidebars` style additionally requires hyperref. They can all be obtained from

`http://www.ctan.org`.

To install the talk class, you have to copy the files `talk.cls`, `talk-simple.sty` and `talk-sidebars.sty` to a place where LaTeX can find them.

2 Using talk

When using talk the visual 'look and feel' of your presentation is determined by *style package* which is separate from the core `talk` document class. Currently talk comes with two such packages called `talk-simple.sty` and `talk-sidebars.sty`. The former is a simple design for short talks while the latter is better for longer seminars with content structured into sections and subsections.

To see the two built-in style packages in action take a look at the example files `simple-example.tex` and `sidebars-example.tex`. This will give you a good idea of how to use talk in practice. You may notice that the two tex files are actually quite similar. This is not accidental. The talk package defines a common interface so that, when writing your presentation, you can focus on content and do not have to worry about presentation. The visual aspects of the presentation are defined in the style file. One of the distinguishing features of talk is that style packages can define an arbitrary number of *layouts*. The idea is that the layouts provided by a style package have consistent design but cater for slightly different types of content. For example, you may want a different layout for the slide where you show an outline of your talk, but it should still have the same colour scheme and fonts as the regular slides.

As a talk user you are strongly encouraged to create your own styles by modifying one of the built-in style files. The file `talk-simple.sty` is a good starting point. It contains lots of comments and tips on how to customise it. A more in-depth guide to writing your own style file can be found in *section 3*. In this section we will discuss the common interface defined by the talkclass, i.e. we will see how to create a presentation using the talk class and some ready-to-use style package like `talk-sidebars.sty`.

The general structure of a presentation `tex` file is shown in *figure 1*. Note that you can structure your talk in the usual way with `\section` and `\subsection` commands. How these commands are handled depends on the loaded style package.

2.1 Class Options

The talk class is loaded in the first line of the listing in *figure 1*:

```
\documentclass[⟨options⟩]{talk}
\usepackage{⟨style-def⟩}
⋮
(more package inclusions)
⋮
\title{⟨title⟩}
\author{⟨author⟩}
\date{⟨date⟩}
⋮
(global specifications required by ⟨style-def⟩)
⋮
\begin{document}
  \begin{slide}[⟨slide-style⟩]{⟨slide-title⟩}
    (body of first slide)
  \end{slide}
  \begin{notes}
    (notes on first slide)
  \end{notes}
  ⋮
  (more slides and notes)
  ⋮
  \section[⟨short title⟩]{⟨long title⟩}
  ⋮
  (more slides and notes)
  ⋮
  \subsection[⟨short title⟩]{⟨long title⟩}
  ⋮
  (more slides and notes)
  ⋮
  (more sections and subsections)
  ⋮
\end{document}
```

Figure 1: The general structure of a presentation `tex` file.

`\documentclass[`⟨*options*⟩`]{talk}`

The available options are

`screen`, `slides`, `notes`, `rotate` and `norotate`.

The talk class is built upon the `article` class, so it will pass all unknown options to `article`. Thus, in principle, all options of the `article` class can be used with the talk class as well. However, some options like `twocolumn` etc. may lead to undesired results.

The options `screen`, `slides` and `notes` determine the *mode* in which your presentation is compiled. The options `rotate` and `norotate` only take effect in the `slides` mode.

`screen` Use the `screen` mode to create a screen presentation. With this option the paper size is set to the slide size, so that your slides can be displayed without white margins using the fullscreen mode of your favorite document viewer.

`slides` `rotate` `norotate` If you use the `slides` mode your presentation is prepared for printout on transparencies. The slides are centered horizontally and vertically on normal paper. The default paper size is A4, but you can change it by using any of the paper size options of the `article` class. The options `rotate` and `norotate` determine whether or not the slides are rotated by 90 degrees in counterclockwise direction. By default rotation is enabled. In `slides` mode the slides can also be magnified. You can set the magnification factor with

`\slidesmag` `\slidesmag{`⟨*factor*⟩`}`.

`notes` The `notes` mode allows you to print personal notes for your presentation. In this mode the slides are inserted in the flowing text, between your annotations.

2.2 Style Packages and Layouts

Style definitions for the talk class are most conveniently included through a separate *style package*. Style packages can be included with the usual `\usepackage` command:

`\usepackage[`⟨*package-options*⟩`]{`⟨*style-package*⟩`}`.

By convention, talk style package names should start with `talk-`, e.g. `talk-simple` or `talk-sidebars`. The available package options depend on the chosen style package. The principal task of a talk style package is to set the width and height of the slides and define a number of *layouts*. To switch between different slide styles you can use the command

`\layout`

`\layout{`⟨*layout-name*⟩`}`.

To change the layout for only one slide you can pass a layout name as an optional argument to the `slide` or `multislide` environment.

`talk-simple`

The `talk-simple` style package provides a simple but appealing design for short presentations. It defines a `default` layout for regular slides, a `notitle` layout where the slide title is removed to make more space for content and a `onlytitle` layout where the slide body is not shown and only the slide title is printed centered on the slide. (The latter is useful for emphasising the start of a new topic.) You can customise the colour of the title with the `\titlecolor` command. The syntax is

`\titlecolor`

`\titlecolor{`⟨*red-value*⟩`,`⟨*green-value*⟩`,`⟨*blue-value*⟩`}`

where ⟨*red-value*⟩, ⟨*green-value*⟩ and ⟨*blue-value*⟩ are numbers between 0 and 1.

For further customisations (like adding background images or defining new layouts) you can simply copy the file `talk-simple.sty` and modify it. It's easier than you might think. The file contains lots of comments which guide you through the process. More information about style packages can be found in *section 3*.

`talk-sidebars`

The `talk-sidebars` style package is intended for longer presentations like seminar talks which consist of multiple sections and subsections. It displays the table of contents in a sidebar on the right of each slide and highlights the (sub-)section that you are currently in. It uses the hyperref package to make the sidebar *navigatable*: you can jump to a different section or sub-section by clicking on the title in the sidebar. The style package has one option, `compress`, which affects the way in which sections and sub-sections are displayed in the sidebar. It defines defines a `default` layout for regular slides, an `outline` layout for showing the outline of the talk (i.e. the sections

Command	Effect
`\backgroundcolor`	sets the colour of the slide background
`\sidebarcolor`	sets the colour of the sidebar
`\titlecolour`	sets the colour of the slide title
`\sidebartitlecolor`	sets the colour of the sidebar title
`\highlightcolor`	sets the colour of highlighted sections and subsections in the sidebar

Table 1: The colour commands of the sidebars package

and sub-sections) at the start of the presentation, a `nosidebar` layout which removes the side bar to make more space for content and a `notitle` layout which also removes the title of the slide.

`\backgroundcolor` `\sidebarcolor` `\titlecolor` `\sidebartitlecolor` `\highlightcolor`

The colours of the `talk-sidebars` style can be customised with the commands listed in *table 1*. Their syntax is the same as for the `\titlecolor` command in `talk-simple`.

2.3 Global Specifications

`\title` `\author` `\date`

Like the `article` class, talk allows you to specify the title, author and date of your talk with the commands

```
\title[⟨short-title⟩]{⟨title⟩}
\author[⟨short-author⟩]{⟨author⟩}
\date{⟨date⟩}
```

`\maketitleslide`

Style packages should define a `\maketitleslide` command which generates a title slide showing this information. In what context the short versions ⟨*short-title*⟩ and ⟨*short-author*⟩ are used depends on the style package. The talk-sidebars package uses the long versions on the title slide but displays ⟨*short-title*⟩ and ⟨*short-author*⟩ in the side bar. Other style packages may also define addtitional commands, that allow you to specify additional information like institute, logo, place where the talk was given etc.

2.4 Environments

The talk class defines three environments: `slide`, `multislide` and `notes`. All typeset material in your talk should be enclosed in one of these environments.

`slide` The `slide` environment is the most important environment in the talk class. It allows you to typeset the contents of your slides. Its syntax is:

```
\begin{slide}[⟨layout-name⟩]{⟨slide-title⟩}
  (slide body)
\end{slide}
```

The ⟨*style-name*⟩ argument is optional. It must be the name of one of the layouts defined in style package you have loaded. If no ⟨*layout-name*⟩ argument is given, talk uses the layout specified in the last call of the `\layout` command.

`notes` The `notes` environment allows you to include annotations to your slides in the `tex` file. The contents of the `notes` environment are ignored if you compile your presentation in the `screen` or `slides` mode.

`multislide` The `multislide` environment can be used to create overlays. Its syntax is:

```
\begin{multislide}[⟨layout-name⟩]{⟨sub-slides⟩}{⟨slide-title⟩}
  (slide body)
\end{multislide}
```

As for the `slide` environment the optional ⟨*layout-name*⟩ argument and the ⟨*slide-title*⟩ argument specify the layout and the slide title. The ⟨*sub-slides*⟩ argument has to be an integer number greater than zero. It specifies the number of sub-slides that the `multislide` environment will generate. In the body of the `multislide` environment, you can use the commands `\fromslide`, `\toslide` and `\onlyslide` to specify which material goes on which sub-slide.

`\fromslide` `\toslide` `\onlyslide` The syntax of the commands `\fromslide`, `\toslide` and `\onlyslide` is:

```
\fromslide*{n}{⟨material⟩}
\toslide*{n}{⟨material⟩}
\onlyslide*{n}{⟨material⟩}
```

The `\fromslide*` command ignores ⟨*material*⟩ on the first $n-1$ sub-slides. The `\toslide*` command ignores ⟨*material*⟩ on all sub-slides

after the n-th. The `\onlyslide*` command ignores ⟨*material*⟩ on all sub-slides exept the n-th. If you use the unstarred commands `\fromslide`, `\toslide` and `\onlyslide` the ⟨*material*⟩ is not ignored but made invisible, so that it still uses up the space (pretty much like the `\phantom` command).

2.5 Positioning Tools

Unlike ordinary text documents slides often contain many graphical elements or graphics and text in 'non-standard' arrangements. Positioning these elements with standard LaTeXcommands can be challenging. The talk package provides a couple of positioning commands which make this task a bit easier.

`\shiftbox` The `\shiftbox` command lets you freely position text and graphics relative to the *current point* (i.e. the point on the current baseline where the next character would be inserted). The syntax is

```
\shiftbox[⟨anchor⟩]{⟨x⟩}{⟨y⟩}{⟨material⟩}
```

The ⟨*material*⟩ will be set in a horizontal box (like `\mbox`) and placed a horizontal distance ⟨x⟩ and a vertical distance ⟨y⟩ away from the current point. With the optional argument ⟨*anchor*⟩ you can specify which corner of the box containing ⟨*material*⟩ should be placed at that location. Use `tl` for the top-left corner, `br` for the bottom-right corner and `tr` or `bl` for the other two corners. Instead of a lowercase `b` you can also use an uppercase `B` to anchor at the baseline of the box containing ⟨*material*⟩. The default for ⟨*anchor*⟩ is `Bl`.

Note that the `\shiftbox` does not use up any space on the current line, i.e. it does not move the current point. Thus, if you call `\shiftbox` multiple times in a row the relative positions of the boxes will be as you expect them to be. For example,

```
\shiftbox[bl]{1cm}{1mm}{foo}
\shiftbox[tl]{1cm}{-1mm}{bar}
```

will print 'foo' and 'bar' one centimetre to the right of the current point. The words will be left-aligned on top of each other with a 2 mm vertical gap between them.

The `\shiftbox` command gives you maximal freedom in the way you position things on the slide. But quite often you just want to show two things (e.g. a picture and some text) side by side. In this case it is simpler to use the `\twocolumn` command. The syntax is

`\twocolumn`

`\twocolumn[`⟨*fraction*⟩`]{`⟨*valign*⟩`}{`⟨*left-material*⟩`}{`⟨*right-material*⟩`}`

This will show ⟨*left-material*⟩ in the left column and ⟨*right-material*⟩ in the right column. The command uses up the entire available horizontal space. The ⟨*valign*⟩ argument determines how the material in the two columns is vertically aligned. The possible values are `t` (top-aligned), `c` (centered) and `b` (bottom-aligned). The optional ⟨*fraction*⟩ argument lets you control the relative width of the columns. It defaults to 0.5, in which case the two columns have the same width.

2.6 Title and Contents Pages

Most talks begin with a title page showing the title of the talk, the name of the speaker and possibly additional information like the date and place where the talk is given, the institute of the speaker etc. For long talks you'll also want to show an 'outline' or 'table of contents' of your talk at the beginning.

`\maketitleslide`

Style packages for the talk class should define a `\maketitleslide` command which generates the title slide using the information specified with the `\title`, `\author` and `\date` (and possibly some other) commands.

`\tableofcontents`

Style packages may also redefine the standard LaTeX command `\tableofcontents` in such a way that it produces a suitable table of contents. This command should be used in the body of a slide, e.g.

```
\begin{slide}[outline]{Contents}
  \tableofcontents
\end{slide}
```

For some talks the table of contents may not fit on one slide. The talk class cannot break material into multiple slides automatically, but it allows you to split the table of contents manually. To do this you can pass an optional argument to the `\tableofcontents` command, which has the following form:

`\tableofcontents[`⟨*fromsec*⟩`.`⟨*fromsubsec*⟩`-`⟨*tosec*⟩`.`⟨*tosubsec*⟩`]`

where ⟨*fromsec*⟩, ⟨*fromsubsec*⟩, ⟨*tosec*⟩ and ⟨*tosubsec*⟩ are integer numbers. Their names are self-explaining. Note that the argument of `\tableofcontents` must always have the form given above. If you wan to display the sections 3 to 5 with all their subsections on one slide, you have to write

```
\tableofcontents[3.0-5.99]
```

(assuming that section 5 does not have more than 99 subsections).

3 The talk Class for Package Writers

The entire look-and-feel of a talk presentation is determined by external style packages. The macros provided by the talk class itself only take care of more technical issues like

- magnifying and positioning the slides on the paper,
- creating overlays,
- keeping a table of contents that is accessible on every slide,
- keeping a catalog of layouts and allowing the user to switch between them.

To create your own talk style the best starting point is the file `talk-simple.sty`. It contains lots of comments which explain what is happening and give you tips for how to customise it. If you want a more structured explanation just continue reading.

3.1 Mode Conditionals

As we have seen in *section 2.1* a talk presentation can be compiled in three different modes: `slides`, `screen` and `notes`. To implement mode-specific behaviour the talk class provides the following `if` commands:

`\@ifslides`
`\@ifscreen`
`\@ifnotes`

```
\@ifslides{⟨if-code⟩}{⟨else-code⟩}
\@ifscreen{⟨if-code⟩}{⟨else-code⟩}
\@ifnotes{⟨if-code⟩}{⟨else-code⟩}
```

The ⟨*if-code*⟩ is executed if the talk is compiled in `slides`, `screen` or `notes` mode, respectively, and the ⟨*else-code*⟩ is executed otherwise.

3.2 Slide Dimensions

`\slidewidth` `\slideheight` `\@slidesize`

The width and height of the slides can be accessed through the `\slidewidth` and `\slideheight` commands. However, to set the slide dimensions you should always use the command

```
\@slidesize{⟨width⟩}{⟨height⟩}
```

as it also adjusts the papersize and slide positioning.

3.3 Global Specifications

`\@title` `\@author` `\@date`

The title, author and date set by the user with the `\title`, `\author` and `\date` commands are stored in the macros `\@title`, `\@author` and `\@date`. If you intend to display additional information like the speakers institute on your slides you should define an `\institute` command in analogy to the commands above:

```
\gdef\@institute{}
\newcommand{\institute}[1]{\gdef\@institute{#1}}
```

3.4 Counters

`slide` `subslide`

In addition to the standard counters of the `article` class, talk defines the counters `slide` and `subslide`. The number of the current slide is stored in `slide`. The slides of a `multislide` environment have the same `slide` number and different `subslide` numbers.

`\theslide` `\thesubslide`

The commands `\theslide` and `\thesubslide` can be used to print the slide or subslide number, respectively. They are defined as

```
\newcommand{\theslide}{\arabic{slide}}
\newcommand{\thesubslide}{\theslide.\arabic{subslide}}
```

You can redefine them to change the way the slide and subslide numbers are displayed

\theslidelabel For printing labels on your slides you should use the `\theslidelabel` command. It calls `\theslide` when you are in a `slide` environment and `\thesubslide` when you are in a `multislide` environment.

3.5 Layouts

To change the look-and-feel of your slides you have to redefine some or all of the following commands:

```
\@makeslide
\@makenotesslide
\@maketocsection
\@maketocsubsection
```

These commands will be called by the `slide`, `multislide` and `notes` \@newlayout environments. Their exact meaning will be explained later on. To create a layout you have to wrap a `\@newslidestyle` command around your definitions. A typical layout definition takes the form

```
\@newlayout{⟨layout-name⟩}{
  \renewcommand{\@makeslide}{⟨stuff⟩}
  \renewcommand{\@maketocsection}[3]{⟨stuff⟩}
  \renewcommand{\@maketocsubsection}[4]{⟨stuff⟩}
}
```

The `\@newlayout` command simply dumps its second argument into a macro called `\talk@sty@`⟨*layout-name*⟩. When the layout is loaded with `\layout{`⟨*layout-name*⟩`}` or with the optional argument of the `slide` environment, all the `\@make`... commands are reset to their default definitions. Then the macro `\talk@sty@`⟨*layout-name*⟩ is executed.

Note that the `\renewcommand` calls in the second argument of `\@newslidestyle` appear in the definition of the `\talk@sty@`⟨*style-name*⟩ command. Therefore the arguments of `\@maketocsection` and `\@maketocsubsection` have to be referenced with double hashes, for example

```
\@newslidestyle{⟨style-name⟩}{
  \renewcommand{\@maketocsection}[3]{Section ##1: ##3}
}
```

Single hashes would refer to the arguments of `\talk@sty@`⟨*style-name*⟩.

3.6 Typesetting Slides

`\@slidetitle` `\@slidebody` The `slide` and `multislide` environments store the slide title in the macro `\@slidetitle` and the slide body in the macro `\@slidebody`. When you redefine the various `\@make`... commands you can therefore use `\@slidetitle` and `\@slidebody` to insert the title and body of the current slide.

`\@makeslide` To generate a slide the `slide` environment executes the macro `\@makeslide` inside a `\parbox` with width and height set to the dimensions of the slide. When `\@makeslide` is called the current point will always be in the upper left corner of the parbox. Thus, you can use the `\shiftbox` command to place graphical elements (e.g. a background image) and text elements (e.g. the slide title or body) using coordinates relative to the top-left corner of the slide. This is the method used in the `talk-simple.sty` and `talk-sidebars.sty` style packages.

Before the call to `\@makeslide` it executes several commands that set the bounding box of the picture to a box of width `\slidewidth` and height `\slideheight` and scales and rotates the picture in accordance with the on the compilation mode and class options. The `\@makeslide` macro should therefore expand to a sequence of valid pgf commands which draw the slide inside a box of width `\slidewidth` and height `\slideheight`, with the origin located at the lower-left corner. To obtain an LR box containing the properly scaled and rotated slide you can use the `\@slidebox` macro.

`\@slidebox`

`\@makenotesslide` If you compile in the `notes` mode the `slide` and `multislide` environments call `\@makenotesslide` to insert the current slide in the flowing text. By default the `\@makenotesslide` command simply centers the parbox containing the slide horizontally. You can change this behaviour by redefining the `\@makenotesslide` command. For example, if you only want to print the title of each slide in your notes, you should include something like

```
\renewcommand{\@makenotesslide}{
  \begin{center}\textbf{\@slidetitle}\end{center}
}
```

`\@slidebox` in your style definition. The parbox with the fully assembled slide can be accessed with the `\@slidebox` command. The default definition of `\@makenotesslide` is

```
\newcommand{\@makenotesslide}{
  \par\hspace*{\fill}\@slidebox\hspace*{\fill}\par
}
```

Note how the talk gives you complete artistic freedom in the design of your slides: It lets you define the macros that generate the slides while contents like the slide title and body are previously stored in macros like `\@slidetitle` and `\@slidebody`, so that you can insert them where you like. For completeness we now summarise all commands yielding user defined contents:

`\@slidetitle` Title of the current slide.

`\@slidebody` Body of the current slide.

`\theslidelabel` Label of the current slide. Shows the slide number in a `slide` environment and the slide and subslide number in a `multislide` environment.

`\@title` Title of the presentation.

`\@shorttitle` Short version of the title.

`\@author` Author of the presentation.

`\@shortauthor` Short version of the author.

`\@date` Date specified with the `\date` command (`\today` by default).

`\@tableofcontents` Prints the table of contents. See the next subsection for more information.

3.7 The Table of Contents

`\@tableofcontents` The table of contents of your talk can be created with the `\@tableofcontents` macro. Its name is slightly misleading because, in fact, it allows you to display any kind of structure information on your slides. For example, you can use it to print only the title of the current section.

`\@maketocsection` `\@maketocsubsection` The `\@tableofcontents` macro expands to a series of `\@maketocsection` and `\@maketocsubsection` commands. By default these commands do nothing. You can control the appearence of the table of contents by redefining them. Their syntax is

`\@maketocsection{`⟨*section*⟩`}{`⟨*short-title*⟩`}{`⟨*long-title*⟩`}`

`\@maketocsubsection{`⟨*section*⟩`}{`⟨*subsection*⟩`}%`
`{`⟨*short-title*⟩`}{`⟨*long-title*⟩`}`

where ⟨*section*⟩ and ⟨*subsection*⟩ are integer numbers.

`\@ifcurrentsection` `\@ifcurrentsubsection` Note that the `\@tableofcontents` macro always expands to the full list of sections and subsections. To implement a special treatment for the *current* section or subsection you can use the `\@ifcurrentsection` and `\@ifcurrentsubsection` commands. Their syntax is

`\@ifcurrentsection{`⟨*number*⟩`}{`⟨*if-code*⟩`}{`⟨*else-code*⟩`}`
`\@ifcurrentsubsection{`⟨*number*⟩`}{`⟨*if-code*⟩`}{`⟨*else-code*⟩`}`

The ⟨*if-code*⟩ is executed if ⟨*number*⟩ matches the current section or subsection, respectively, and ⟨*else-code*⟩ is executed otherwise. For example, if you want to display the current section in the top left corner of each slide, your style definition should look somewhat like

```
\newslidestyle{⟨style-name⟩}{
  \renewcommand{\@maketocsection}[3]{
    \@ifcurrentsection{##1}{##3}{}
  }
  \renewcommand{\@makeslidecontent}{
    \@tableofcontents
    ⋮
  }
  ⋮
}
```

\tableofcontents

Most talks begin with an outline of the talk's contents. As a package writer you should therefore provide a `\tableofcontents` command that allows the user to print the full table of contents. (talk already defines the `\tableofcontents` command, but it does nothing by default.) You can achieve this, too, by redefining `\@maketocsection` and `\@maketocsubsection` and then calling `\@tableofcontents`.

\@ifinrange

However, if the table of contents does not fit on one slide, the user should be able to split it, using an optional range argument of the form shown in *section 2.6*. It is the package writers task to implement this feature, but the parsing of the range argument is done by the `\@ifinrange` macro. Its syntax is

`\@ifinrange{`⟨*sec*⟩`}{`⟨*subsec*⟩`}{`⟨*range*⟩`}{`⟨*if-code*⟩`}{`⟨*else-code*⟩`}`

⟨*sec*⟩ and ⟨*subsec*⟩ are section and subsection numbers and ⟨*range*⟩ is a string of the form

⟨*fromsec*⟩`.`⟨*fromsubsec*⟩`-`⟨*tosec*⟩`.`⟨*tosubsec*⟩

The ⟨*if-code*⟩ is executed if the subsection specified by ⟨*sec*⟩ and ⟨*subsec*⟩ lies in the range specified by ⟨*range*⟩, the ⟨*else-code*⟩ is executed otherwise.

A typical definition of the `\tableofcontents` command will therefore look as follows:

```
\renewcommand{\tableofcontents}[1][0.0-99.99]{
  \bgroup
  \def\@maketocsection##1##2##3{
    \@ifinrange{##1}{0}{#1}{
      ##1.\space ##3\par
    }{}
  }
  \def\@maketocsubsection##1##2##3##4{
    \@ifinrange{##1}{##2}{#1}{
      ##1.##2.\space ##3\par
    }{}
  }
  \@tableofcontents
  \egroup
}
```

If you use grouping (`\bgroup` and `\egroup` or curly brackets) and plain TEX definitions (`\def` instead of `\renewcommand`), as shown above, your definitions remain local to the goup, so you don't have to worry about restoring the original definitions of `\@maketocsection` and `\@maketocsubsection`. A more sophisticated definition of the `\tableofcontents` command can be found in `sidebars.sty`.

3.8 Paragraph Spacing

The talk class and the accompanying style packages `talk-simple.sty` and `talk-sidebars.sty` make extensive use of the `\parbox` command and the `minipage` environment to position text elements. The standard definitions of these commands have the unfortunate property that they override the global setting for `\parskip` which controls the space inserted between paragraphs. You can set `\parskip` globally to something non-zero, but inside parboxes and minipages the separation between paragraphs will still be zero. (Try it out!) This behaviour may be acceptable in document classes like `article` where parboxes and minipages are only uses in exceptional circumstances, but becomes problematic when virtually all visible material lives inside a parbox or minipage. To mitigate this talk provides the `\setparskip` command which sets the value of `\parskip` for all following paragraphs, including those inside parboxes or minipages. The syntax is

`\setparskip`

```
\setparskip{⟨glue⟩}
```

where ⟨*glue*⟩ is a glue item to be used to separate paragraphs. For example, the `talk-simple.sty` and `talk-sidebars.sty` packages use

```
\setparskip{1.5ex plus0.5ex minus0.5ex}
```

4 Contact

For comments, bug reports, feature requests or submitting style packages please raise an issue at

`https://github.com/mwiebusch78/talk`

Martin Wiebusch, 31 August 2025