The beamer class

User Guide for version 3.62.

```
\begin{frame}
  \frametitle{There Is No Largest Prime Number}
  \framesubtitle{The proof uses \textit{reductio ad absurdum}.}
  \begin{theorem}
    There is no largest prime number.
  \end{theorem}
  \begin{proof}
    \begin{enumerate}
    \item<1-| alert@1> Suppose $p$ were the largest prime number.
    \item<2-> Let $q$ be the product of the first $p$ numbers.
    \item<3-> Then $q+1$ is not divisible by any of them.
    \item<1-> But $q + 1$ is greater than $1$, thus divisible by some prime
      number not in the first $p$ numbers.\qedhere
    \end{enumerate}
  \end{proof}
\end{frame}
```

7.4.9 How to Use This License

To use this license, place in each of the components of your work both an explicit copyright notice including your name and the year the work was authored and/or last substantially modi ed. Include also a statement that the distribution and/or modi cation of that component is constrained by the conditions in this license.

Here is an example of such a notice and statement:

```
%% pig.dtx
%% Copyright 2005 M. Y. Name
%
% This work may be distributed and/or modified under the
% conditions of the LaTeX Project Public License, either version 1.3
% of this license or (at your option) any later version.
% The latest version of this license is in
%    https://www.latex-project.org/lppl.txt
% and version 1.3 or later is part of all distributions of LaTeX
% version 2005/12/01 or later.
%
% This work has the LPPL maintenance status `maintained'.
%
% The Current Maintainer of this work is M. Y. Name.
%
% This work consists of the files pig.dtx and pig.ins
% and the derived file pig.sty.
```

Given such a notice and statement in a le, the conditions given in this license document would apply, with the `Work' referring to the three les ` pig.dtx ', `pig.ins ', and `pig.sty ' (the last being generated from `pig.dtx ' using `pig.ins '), the `Base Interpreter' referring to any `LaTeX-Format', and both `Copyright Holder' and `Current Maintainer' referring to the person `M. Y. Name'.

If you do not want the Maintenance section of lppl to apply to your Work, change `maintained' above into `author-maintained'. However, we recommend that you use `maintained' as the Maintenance section was added in order to ensure that your Work remains useful to the community even when you can no longer maintain and support it yourself.

7.4.10 Derived Works That Are Not Replacements

Several clauses of the lppl specify means to provide reliability and stability for the user community. They therefore concern themselves with the case that a Derived Work is intended to be used as a (compatible or incompatible) replacement of the original Work. If this is not the case (e.g., if a few lines of code are reused for a completely di erent task), then clauses 6b and 6d shall not apply.

7.4.11 Important Recommendations

De ning What Constitutes the Work The lppl requires that distributions of the Work contain all the les of the Work. It is therefore important that you provide a way for the licensee to determine which les constitute the Work. This could, for example, be achieved by explicitly listing all the les of the Work near the copyright notice of each le or by using a line such as:

```
% This work consists of all files listed in manifest.txt.
```

in that place. In the absence of an unequivocal list it might be impossible for the licensee to determine what is considered by you to comprise the Work and, in such a case, the licensee would be entitled to make reasonable conjectures as to which les comprise the Work.

Part II

Building a Presentation

This part contains an explanation of all the commands that are used to create presentations. It starts with a section treating the commands and environments used to create frames, the basic building blocks of presentations. Next, the creation of overlays is explained.

The following three sections concern commands and methods of structuring a presentation. In order, the static global structure, the interactive global structure, and the local structure are treated.

Two further sections treat graphics and animations. Much of the material in these sections applies to other packages as well, not just to beamer .

8 Creating Frames

8.1 The Frame Environment

A presentation consists of a series of frames. Each frame consists of a series of slides. You create a frame using frame environment.[1] All of the text that is not tagged by overlay specifications is shown on all slides of the frame. (Overlay specifications are explained in more detail in later sections. For the moment, let's just say that an overlay specification is a list of numbers or number ranges in pointed brackets that is put after certain commands as in `\uncover<1,2>{Text}`.) If a frame contains commands that have an overlay specification, the frame will contain multiple slides; otherwise it contains only one slide.

```
\begin{frame}<⟨overlay specification⟩>[<⟨default overlay specification⟩>][⟨options⟩]{⟨title⟩}{⟨subtitle⟩}
  ⟨environment contents⟩
\end{frame}
```

The ⟨overlay specification⟩ dictates which slides of a frame are to be shown. If left out, the number is calculated automatically. The ⟨environment contents⟩ can be normal LaTeX text, but may not contain `\verb` commands or `verbatim` environments or any environment that changes the character codes, unless the `fragile` option is given.

The optional ⟨title⟩ is detected by an opening brace, that is, if the first thing in the frame is an opening brace then it is assumed that a frame title follows. Likewise, the optional ⟨subtitle⟩ is detected the same way, that is, by an opening brace following the ⟨title⟩. The title and subtitle can also be given using the `\frametitle` and `\framesubtitle` commands.

The normal LaTeX command `\frame` is available inside frames with its usual meaning. Both outside and inside frames it is always available as `\framelatex`.

Example:

```
\begin{frame}{A title}
  Some content.
\end{frame}
% Same effect:
\begin{frame}
  \frametitle{A title}
  Some content.
\end{frame}
```

Example:

```
\begin{frame}<beamer>{Outline}  % frame is only shown in beamer mode
  \tabelofcontent[current]
\end{frame}
```

Normally, the complete ⟨environment contents⟩ is put on a slide. If the text does not fit on a slide, being too high, it will be squeezed as much as possible, a warning will be issued, and the text just extends unpleasantly over the bottom. You can use the option `allowframebreaks` to cause the ⟨frame text⟩ to be split among several slides, though you cannot use overlays then. See the explanation of the `allowframebreaks` option for details.

The ⟨default overlay specification⟩ is an optional argument that is detected according to the following rule: If the first optional argument in square brackets starts with a `<`, then this argument is a ⟨default overlay specification⟩, otherwise it is a normal ⟨options⟩ argument. Thus `\begin{frame}[<+->][plain]` would be legal, but also `\begin{frame}[plain]`.

The effect of the ⟨default overlay specification⟩ is the following: Every command or environment inside the frame that accepts an action specification, see Section 9.6.3, (this includes the `\item` command, the `actionenv` environment, `\action`, and all block environments) and that is not followed by an overlay specification gets the ⟨default overlay specification⟩ as its specification. By providing an incremental specification like `<+->`, see Section 9.6.4, this will essentially cause all blocks and all enumerations to be uncovered piece-wise (blocks internally employ action specifications).

[1] The command `\frame` is supported for legacy documents.

Example: In this frame, the theorem is shown from the first slide on, the proof from the second slide on, with the first two itemize points shown one after the other; the last itemize point is shown together with the first one. In total, this frame will contain four slides.

```
\begin{frame}[<+->]
  \begin{theorem}
    $A = B$.
  \end{theorem}
  \begin{proof}
    \begin{itemize}
    \item Clearly, $A = C$.
    \item As shown earlier, $C = B$.
    \item<3-> Thus $A = B$.
    \end{itemize}
  \end{proof}
\end{frame}
```

The following ⟨options⟩ may be given:

- allowdisplaybreaks=⟨break desirability⟩ causes the AMSTEX command \allowdisplaybreaks[⟨break desirability⟩] to be issued for the current frame. The ⟨break desirability⟩ can be a value between 0 (meaning formulas may never be broken) and 4 (the default, meaning that formulas can be broken anywhere without any penalty). The option is just a convenience and makes sense only together with the allowsframebreaks option.
- allowframebreaks=⟨fraction⟩. When this option is given, the frame will be automatically broken up into several frames if the text does not fit on a single slide. In detail, when this option is given, the following things happen:
  1. Overlays are not supported.
  2. Any notes for the frame created using the \note command will be inserted after the first page of the frame.
  3. Any footnotes for the frame will be inserted on the last page of the frame.
  4. If there is a frame title, each of the pages will have this frame title, with a special note added indicating which page of the frame that page is. By default, this special note is a Roman number. However, this can be changed using the following template.

     Beamer-Template/-Color/-Font frametitle continuation

     The text of this template is inserted at the end of every title of a frame with the allowframebreaks option set.

     The following template options are predefined:

     [default] Installs a Roman number as the template. The number indicates the current page of the frame.

     [roman] Alias for the default.

     [from second][⟨text⟩] Installs a template that inserts ⟨text⟩ from the second page of a frame on. By default, the text inserted is \insertcontinuationtext, which in turn is (cont.) by default.

     The following inserts are available:

     \insertcontinuationcount inserts the current page of the frame as an arabic number.

     \insertcontinuationcountroman inserts the current page of the frame as an (uppercase) Roman number.

     \insertcontinuationtext just inserts the text (cont.) or, possibly, a translation thereof (like (Forts.) in German).

  If a frame needs to be broken into several pages, the material on all but the last page fills only 95% of each page by default. Thus, there will be some space left at the top and/or bottom, depending on the vertical placement option for the frame. This yields a better visual result than a 100% filling, which typically looks crowded. However, you can change this percentage using the optional argument

⟨fraction⟩, where 1 means 100% and 0.5 means 50%. This percentage includes the frame title. Thus, in order to split a frame roughly in half, you should give 0.6 as ⟨fraction⟩.

Most of the fine details of normal TEX page breaking also apply to this option. For example, when you wish equations to be broken automatically, be sure to use the \allowdisplaybreaks command. You can insert \break, \nobreak, and \penalty commands to control where breaks should occur. The commands \pagebreak and \nopagebreak also work, including their options. Since you typically do not want page breaks for the frame to apply also to the article mode, you can add a mode specification like <presentation> to make these commands apply only to the presentation modes. The command \framebreak is a shorthand for \pagebreak<presentation> and \noframebreak is a shorthand for \nopagebreak<presentation>.

The use of this option is evil. In a (good) presentation you prepare each slide carefully and think twice before putting something on a certain slide rather than on some different slide. Using the allowframebreaks option invites the creation of horrible, endless presentations that resemble more a paper projected on the wall than a presentation. Nevertheless, the option does have its uses. Most noticeably, it can be convenient for automatically splitting bibliographies or long equations.

Example:

```
\begin{frame}[allowframebreaks]{References}
  \begin{thebibliography}{XX}

  \bibitem...
  \bibitem...
    ...
  \bibitem...
  \end{thebibliography}
\end{frame}
```

Example:

```
\begin{frame}[allowframebreaks,allowdisplaybreaks]{A Long Equation}
  \begin{align}
    \zeta(2) &= 1 + 1/4 + 1/9 + \cdots \\
    &= ... \\
    ...
    &= \pi^2/6.
  \end{align}
\end{frame}
```

- b, c, t will cause the frame to be vertically aligned at the bottom/center/top. This overrides the global placement policy, which is governed by the class options b and c.
- nofranumbering tells beamer not to step the framenumber counter for this frame.
- fragile=singleslide tells beamer that the frame contents is fragile. This means that the frame contains text that is not interpreted as usual. For example, this applies to verbatim text, which is, obviously, interpreted somewhat differently from normal text.

  If a frame contains fragile text, different internal mechanisms are used to typeset the frame to ensure that inside the frame the character codes can be reset. The price of switching to another internal mechanism is that either you cannot use overlays or an external file needs to be written and read back (which is not always desirable).

  In detail, the following happens when this option is given for normal (pdf)LATEX: The contents of the frame is scanned and then written to a special file named ⟨jobname⟩.vrb or, if a label has been assigned to the frame, ⟨jobname⟩.⟨current frame number⟩.vrb. Then, the frame is started anew and the content of this file is read back. Since, upon reading of a file, the character codes can be modified, this allows you to use both verbatim text and overlays.

  To determine the end of the frame, the following rule is used: The first occurrence of a single line containing exactly \end{⟨frame environment name⟩} ends the frame. The ⟨environment name⟩ is

normally frame, but it can be changed using the environment option. This special rule is needed since the frame contents is, after all, not interpreted when it is gathered.

You can also add the optional information =singleslide . This tells beamer that the frame contains only a single slide. In this case, the frame contents is not written to a special le, but interpreted directly, which is faster and cleaner.

- environment= hframe environment namei. This option is useful only in conjunction with the fragile option (but it is not used for fragile=singleslide , only for the plain fragile). The hframe environment namei is used to determine the end of the scanning when gathering the frame contents. Normally, the frame ends when a line reading \end{frame} is reached. However, if you use \begin{frame} inside another environment, you need to use this option:

  Example:

```
\newenvironment{slide}[1]
  {\begin{frame}[fragile,environment=slide]
      \frametitle{#1}}
  {\end{frame}}

\begin{slide}{My title}
  Text.
\end{slide}
```

  If you did not specify the option environment=slide in the above example, TEX would miss the end of the slide since it does not interpret text while gathering the frame contents.

- label= hnamei causes the frame's contents to be stored under the name hnamei for later resumption using the command \againframe . Furthermore, on each slide of the frame a label with the name hnamei<hslide numberi> is created. On the rst slide, furthermore, a label with the name hnamei is created (so the labels hnamei and hnamei<1> point to the same slide). Note that labels in general, and these labels in particular, can be used as targets for hyperlinks.

  You can use this option together with fragile .

- plain causes the headlines, footlines, and sidebars to be suppressed. This is useful for creating single frames with di erent head- and footlines or for creating frames showing big pictures that completely ll the frame.

  Example: A frame with a picture completely lling the frame:

```
\begin{frame}[plain]
  \begin{centering}%
    \pgfimage[height=\paperheight]{somebigimagefile}%
    \par%
  \end{centering}%
\end{frame}
```

  Example: A title page, in which the head- and footlines are replaced by two graphics.

```
\setbeamertemplate{title page}
{
  \pgfuseimage{toptitle}
  \vskip0pt plus 1filll

  \begin{centering}
    {\usebeamerfont{title}\usebeamercolor[fg]{title}\inserttitle}

    \insertdate
  \end{centering}

  \vskip0pt plus 1filll
  \pgfuseimage{bottomtitle}
```

```
}
\begin{frame}[plain]
  \titlepage
\end{frame}
```

- shrink=⟨minimum shrink percentage⟩. This option will cause the text of the frame to be shrunk if it is too large to fit on the frame. beamer will first normally typeset the whole frame. Then it has a look at vertical size of the frame text (excluding the frame title). If this vertical size is larger than the text height minus the frame title height, beamer computes a shrink factor and scales down the frame text by this factor such that the frame text then fills the frame completely. Using this option will automatically cause the squeeze option to be used, also.

  Since the shrinking takes place only after everything has been typeset, shrunk frame text will not fill the frame completely horizontally. For this reason, you can specify a ⟨minimum shrink percentage⟩ like 20. If this percentage is specified, the frame will be shrunk at least by this percentage. Since beamer knows this, it can increase the horizontal width proportionally such that the shrunk text once more fills the entire frame. If, however, the percentage is not enough, the text will be shrunk as needed and you will be punished with a warning message.

  The best way to use this option is to identify frames that are overly full, but in which all text absolutely has to be fit on a single frame. Then start specifying first shrink=5 , then shrink=10 , and so on, until no warning is issued any more (or just ignore the warning when things look satisfactory).

  Using this option is very evil. It will result in changes of the font size from slide to slide, which is a typographic nightmare. Its usage can always be avoided by restructuring and simplifying frames, which will result in a better presentation.

  Example:

```
\begin{frame}[shrink=5]
  Some evil endless slide that is 5\% too large.
\end{frame}
```

- squeeze causes all vertical spaces in the text to be squeezed together as much as possible. Currently, this just causes the vertical space in enumerations or itemizations to be reduced to zero.

  Using this option is not good, but also not evil.

article In article mode, the frame environment does not create any visual reference to the original frame (no frame is drawn). Rather, the frame text is inserted into the normal text. To change this, you can modify the templates frame begin and frame end, see below. To suppress a frame in article mode, you can, for example, specify<presentation> as overlay specification.

Beamer-Template frame begin

The text of this template is inserted at the beginning of each frame in article mode (and only there). You can use it, say, to start a minipage environment at the beginning of a frame or to insert a horizontal bar or whatever.

Beamer-Template frame end

The text of this template is inserted at the end of each frame in article mode.

You can use the frame environment inside other environments like this

```
\newenvironment{slide}{\begin{frame}}{\end{frame}}
```

or like this

```
\newenvironment{myframe}[1]
  {\begin{frame}[fragile,environment=myframe]\frametitle{#1}}
  {\end{frame}}
```

However, the actual mechanics are somewhat sensitive since the collecting of the frame contents is not easy, so do not attempt anything too fancy. As a rule, the beginning of the environment can be pretty arbitrary, but the ending must end with \end{frame} and should not contain any \end{xxx} . Anything really complex is likely to fail. If you need some \end{xxx} there, de ne a new command that contains this stu as in the following example:

```
\newenvironment{itemizeframe}
  {\begin{frame}\startitemizeframe}
  {\stopitemizeframe\end{frame}}

\newcommand\startitemizeframe{\begin{bfseries}\begin{itemize}}
\newcommand\stopitemizeframe{\end{itemize}\end{bfseries}}

\begin{itemizeframe}
\item First item
\end{itemizeframe}
```

8.2 Components of a Frame

Each frame consists of several components:

1. a headline and a footline,
2. a left and a right sidebar,
3. navigation bars,
4. navigation symbols,
5. a logo,
6. a frame title,
7. a background, and
8. some frame contents.

A frame need not have all of these components. Usually, the rst three components are automatically setup by the theme you are using.

8.2.1 The Headline and Footline

The headline of a frame is the area at the top of the frame. If it is not empty, it should show some information that helps the audience orientate itself during your talk. Likewise, the footline is the area at the bottom of the frame.

beamer does not use the standard LaTeX mechanisms for typesetting the headline and the footline. Instead, the special headline and footline templates are used to typeset them.

The size of the headline and the footline is determined as follows: Their width is always the paper width. Their height is determined by tentatively typesetting the headline and the footline right after the \begin{document} command. The head of the headline and the footline at that point is frozen and will be used throughout the whole document, even if the headline and footline vary in height later on (which they should not).

The appearance of the headline and footline is determined by the following templates:

Beamer-Template/-Color/-Font headline

This template is used to typeset the headline. Thebeamer -color and -font headline are installed at the beginning. The background of thebeamer -color is not used by default, that is, no background rectangle is drawn behind the headline and footline (this may change in the future with the introduction of a headline and a footline canvas).

The width of the headline is the whole paper width. The height is determined automatically as described above. The headline is typeset in vertical mode with interline skip turned o and the paragraph skip set to zero.

Inside this template, the \\ command is changed such that it inserts a comma instead.

Example:

```
\setbeamertemplate{headline}
{%
  \begin{beamercolorbox}{section in head/foot}
    \vskip2pt\insertnavigation{\paperwidth}\vskip2pt
  \end{beamercolorbox}%
}
```

The following template options are prede ned:

- [default] The default is just an empty headline. To get the default headline of earlier versions of the beamer class, use thecompatibility theme.
- [infolines theme] This option becomes available (and is used) if theinfolines outer theme is loaded. The headline shows current section and subsection.
- [miniframes theme] This option becomes available (and is used) if theminiframes outer theme is loaded. The headline shows the sections with small clickable mini frames below them.
- [sidebar theme] This option becomes available (and is used) if thesidebar outer theme is loaded and if the head height (and option of the sidebar theme) is not zero. In this case, the headline is an empty bar of the background colorframetitle with the logo to the left or right of this bar.
- [smoothtree theme] This option becomes available (and is used) if thesmoothtree outer theme is loaded. A smoothed navigation tree is shown in the headline.
- [smoothbars theme] This option becomes available (and is used) if thesmoothbars outer theme is loaded. A smoothed version of the miniframes headline is shown.
- [tree] This option becomes available (and is used) if thetree outer theme is loaded. A navigational tree is shown in the headline.
- [split theme] This option becomes available (and is used) if thesplit outer theme is loaded. The headline is split into a left part showing the sections and a right part showing the subsections.
- [text line]{ htexti} The headline is typeset more or less as if it were a normal text line with the htexti as contents. The left and right margin are setup such that they are the same as the margins of normal text. The htexti is typeset inside an\hbox , while the headline is normally typeset in vertical mode.

Inside the template numerous inserts can be used:

- \insertnavigation{ hwidth i} Inserts a horizontal navigation bar of the given hwidthi into a template. The bar lists the sections and below them mini frames for each frame in that section.
- \insertpagenumber Inserts the current page number into a template.
- \insertsection Inserts the current section into a template.
- \insertsectionnavigation{ hwidthi} Inserts a vertical navigation bar containing all sections, with the current section highlighted.
- \insertsectionnavigationhorizontal{ hwidthi}{ hleft inserti}{ hright inserti} Inserts a horizontal navigation bar containing all sections, with the current section highlighted. The hleft inserti will be inserted to the left of the sections, the{ hright inserti} to the right. By inserting a triple ll (a filll) you can ush the bar to the left or right.

  Example:

  ```
  \insertsectionnavigationhorizontal{.5\textwidth}{\hskip0pt plus1filll}{}
  ```

- \insertshortauthor[hoptionsi] Inserts the short version of the author into a template. The text will be printed in one long line, line breaks introduced using the\\ command are suppressed. The following hoptionsi may be given:

  width= hwidthi causes the text to be put into a multi-line minipage of the given size. Line breaks are still suppressed by default.

center centers the text inside the minipage created using the width option, rather than having it left aligned.

respectlinebreaks causes line breaks introduced by the \\ command to be honored.

Example: \insertauthor[width={3cm},center,respectlinebreaks]

- \insertshortdate[⟨options⟩] Inserts the short version of the date into a template. The same options as for \insertshortauthor may be given.
- \insertshortinstitute[⟨options⟩] Inserts the short version of the institute into a template. The same options as for \insertshortauthor may be given.
- \insertshortpart[⟨options⟩] Inserts the short version of the part name into a template. The same options as for \insertshortauthor may be given.
- \insertshorttitle[⟨options⟩] Inserts the short version of the document title into a template. Same options as for \insertshortauthor may be given.
- \insertshortsubtitle[⟨options⟩] Inserts the short version of the document subtitle. Same options as for \insertshortauthor may be given.
- \insertsubsection Inserts the current subsection into a template.
- \insertsubsubsection Inserts the current subsection into a template.
- \insertsubsectionnavigation{⟨width⟩} Inserts a vertical navigation bar containing all subsections of the current section, with the current subsection highlighted.
- \insertsubsectionnavigationhorizontal{⟨width⟩}{⟨left insert⟩}{⟨right insert⟩} See \insertsectionnavigationhorizontal.
- \insertverticalnavigation{⟨width⟩} Inserts a vertical navigation bar of the given ⟨width⟩ into a template. The bar shows a little table of contents. The individual lines are typeset using the templates section in head/foot and subsection in head/foot.
- \insertframenumber Inserts the number of the current frame (not slide) into a template.
- \insertslidenumber Inserts the number of the current slide in frame into a template.
- \insertoverlaynumber Inserts the number of current overlay counter into a template. This is generally equal to \insertslidenumber, except when an overlay specification is used in a frame environment.
- \inserttotalframenumber Inserts the total number of the frames (not slides) into a template. The number is only correct on the second run of TEX on your document.
- \insertmainframenumber Inserts the number of the frames in the main part (before \appendix command) into a template. The number is only correct on the second run of TEX on your document.
- \insertappendixframenumber Inserts the number of the frames in the appendix part (after \appendix command) into a template. The number is only correct on the second run of TEX on your document.
- \insertframestartpage Inserts the page number of the first page of the current frame.
- \insertframeendpage Inserts the page number of the last page of the current frame.
- \insertsubsectionstartpage Inserts the page number of the first page of the current subsection.
- \insertsubsectionendpage Inserts the page number of the last page of the current subsection.
- \insertsectionstartpage Inserts the page number of the first page of the current section.
- \insertsectionendpage Inserts the page number of the last page of the current section.
- \insertpartstartpage Inserts the page number of the first page of the current part.
- \insertpartendpage Inserts the page number of the last page of the current part.
- \insertpresentationstartpage Inserts the page number of the first page of the presentation.
- \insertpresentationendpage Inserts the page number of the last page of the presentation (excluding the appendix).
- \insertappendixstartpage Inserts the page number of the first page of the appendix. If there is no appendix, this number is the last page of the document.

- \insertappendixendpage Inserts the page number of the last page of the appendix. If there is no appendix, this number is the last page of the document.
- \insertdocumentstartpage Inserts 1.
- \insertdocumentendpage Inserts the page number of the last page of the document (including the appendix).
- \usebeamertemplate*{page number in head/foot} inserts a customisable template which e.g. inserts the current and total number of frames.

Beamer-Template/-Color/-Font footline

This template behaves exactly the same way as the headline. Note that, sometimes quite annoyingly, beamer currently adds a space of 4pt between the bottom of the frame's text and the top of the footline.

The following template options are prede ned:

- [default] The default is an empty footline. Note that the navigational symbols are not part of the footline by default. Rather, they are part of an (invisible) right sidebar.
- [infolines theme] This option becomes available (and is used) if the infolines outer theme is loaded. The footline shows things like the author's name and the title of the talk.
- [miniframes theme] This option becomes available (and is used) if the miniframes outer theme is loaded. Depending on the exact options that are used when the miniframes theme is loaded, di erent things can be shown in the footline.
- [page number] Shows the current page number in the footline.
- [frame number] Shows the current frame number in the footline.
- [split] This option becomes available (and is used) if the split outer theme is loaded. The footline (just like the headline) is split into a left part showing the author's name and a right part showing the talk's title.
- [text line]{ htexti } The footline is typeset more or less as if it were a normal text line with the htexti as contents. The left and right margin are setup such that they are the same as the margins of normal text. The htexti is typeset inside an \hbox , while the headline is normally typeset in vertical mode. Using the \strut command somewhere in such a line might be a good idea.

The same inserts as for headlines can be used.

Beamer-Template/-Color/-Font page number in head/foot

These beamer -color and -font are used to typeset the page number or frame number in the footline.

The beamer -template provides a convenient way to format the page or frame number in the footline. It is used by the infolines outer theme and the page numberand frame number footline themes. It can also be used with the miniframes and split outer themes, but for them it is set to empty as default.

The following template options are prede ned:

- [default] The default option is empty.
- [framenumber] This option inserts the current frame number.
- [totalframenumber] In addition to the current frame number, this option also shows the total number of frames.
- [appendixframenumber] This options replicates the behaviour of theappendixnumberbeamerpackage. In the main part before the \appendix command the current frame number and the total number of frames in the main part is displayed. In the appendix only the frame number within the appendix and the total number of frames in the appendix are shown.
- [pagenumber] Shows the current page number.
- [totalpagenumber] In addition to the current page number also the total page number is displayed.

8.2.2 The Sidebars

Sidebars are vertical areas that stretch from the lower end of the headline to the top of the footline. There can be a sidebar at the left and another one at the right (or even both). Sidebars can show a table of contents, but they could also be added for purely aesthetic reasons.

When you install a sidebar template, you must explicitly specify the horizontal size of the sidebar using the command \setbeamersize with the option sidebar width left or sidebar width right. The vertical size is determined automatically. Each sidebar has its own background canvas, which can be setup using the sidebar canvas templates.

Adding a sidebar of a certain size, say 1 cm, will make the main text 1 cm narrower. The distance between the inner side of a side bar and the outer side of the text, as specified by the command \setbeamersize with the option text margin left and its counterpart for the right margin, is not changed when a sidebar is installed.

Internally, the sidebars are typeset by showing them as part of the headline. The beamer class keeps track of six dimensions, three for each side: the variables \beamer@leftsidebar and \beamer@rightsidebar store the (horizontal) sizes of the side bars, the variables \beamer@leftmargin and \beamer@rightmargin store the distance between sidebar and text, and the macros \Gm@lmargin and \Gm@rmargin store the distance from the edge of the paper to the edge of the text. Thus the sum \beamer@leftsidebar and \beamer@leftmargin is exactly \Gm@lmargin. Thus, if you wish to put some text right next to the left sidebar, you might write \hskip-\beamer@leftmargin to get there.

Beamer-Template/-Color/-Font sidebar left

Color/font parents: sidebar

The template is used to typeset the left sidebar. As mentioned above, the size of the left sidebar is set using the command

```
\setbeamersize{sidebar width left=2cm}
```

beamer will not clip sidebars automatically if they are too large.

When the sidebar is typeset, it is put inside a \vbox. You should currently setup things like the \hsize or the \parskip yourself.

The following template options are predefined:

- [default] installs an empty template.
- [sidebar theme] This option is available if the outer theme sidebar is loaded with the left option. In this case, this options is selected automatically. It shows a mini table of contents in the sidebar.

Beamer-Template/-Color/-Font sidebar right

Color/font parents: sidebar

This template works the same way as the template for the left.

The following template options are predefined:

- [default] The default right sidebar has zero width. Nevertheless, it shows navigational symbols and, if installed, a logo at the bottom of the sidebar, protruding to the left into the text.
- [sidebar theme] This option is available, if the outer theme sidebar is loaded with the right option. In this case, this option is selected automatically. It shows a mini table of contents in the sidebar.

Beamer-Template sidebar canvas left

Like the overall background canvas, this canvas is drawn behind the actual text of the sidebar. This template should normally insert a rectangle of the size of the sidebar, though a too large height will not lead to an error or warning. When this template is called, the beamer -color sidebar left will have been installed.

The following template options are predefined:

- [default] uses a large rectangle colored with sidebar.bg as the sidebar canvas. However, if the background of sidebar is empty, nothing is drawn and the canvas is transparent.

- [vertical shading][⟨color options⟩] installs a vertically shaded background. The following ⟨color options⟩ may be given:

  top=⟨color⟩ specifies the color at the top of the sidebar. By default, 25% of the foreground of the beamer-color palette primary is used.

  bottom=⟨color⟩ specifies the color at the bottom of the sidebar (more precisely, at a distance of the page height below the top of the sidebar). By default, the background of normal text at the moment of invocation of this command is used.

  middle=⟨color⟩ specifies the color for the middle of the sidebar. Thus, if this option is given, the shading changes from the bottom color to this color and then to the top color.

  midpoint=⟨factor⟩ specifies at which point of the page the middle color is used. A factor of 0 is the bottom of the page, a factor of 1 is the top. The default, which is 0.5, is in the middle.

  Note that you must give real LaTeX colors here. This often makes it necessary to invoke the command \usebeamercolor before this command can be used.

  Also note, that the width of the sidebar should be setup before this option is used.

  Example: A stylish, but not very useful shading:

```
{\usebeamercolor{palette primary}}
\setbeamertemplate{sidebar canvas}[vertical shading]
[top=palette primary.bg,middle=white,bottom=palette primary.bg]
```

- [horizontal shading][⟨color options⟩] installs a horizontally shaded background. The following ⟨color options⟩ may be given:

  left=⟨color⟩ specifies the color at the left of the sidebar.

  right=⟨color⟩ specifies the color at the right of the sidebar.

  middle=⟨color⟩ specifies the color in the middle of the sidebar.

  midpoint=⟨factor⟩ specifies at which point of the sidebar the middle color is used. A factor of 0 is the left of the sidebar, a factor of 1 is the right. The default, which is 0.5, is in the middle.

  Example: Adds two pillars

```
\setbeamersize{sidebar width left=0.5cm,sidebar width right=0.5cm}

{\usebeamercolor{sidebar}}

\setbeamertemplate{sidebar canvas left}[horizontal shading]
[left=white,middle=sidebar.bg,right=white]
\setbeamertemplate{sidebar canvas right}[horizontal shading]
[left=white,middle=sidebar.bg,right=white]
```

Beamer-Template sidebar canvas right

Works exactly as for the left side.

8.2.3 Navigation Bars

Many themes install a headline or a sidebar that shows a navigation bar. Although these navigation bars take up quite a bit of space, they are often useful for two reasons:

- They provide the audience with a visual feedback of how much of your talk you have covered and what is yet to come. Without such feedback, an audience will often puzzle whether something you are currently introducing will be explained in more detail later on or not.
- You can click on all parts of the navigation bar. This will directly jump you to the part you have clicked on. This is particularly useful to skip certain parts of your talk and during a question session, when you wish to jump back to a particular frame someone has asked about.

Some navigation bars can be compressed using the following option:

\documentclass[compress]{beamer}

Tries to make all navigation bars as small as possible. For example, all small frame representations in the navigation bars for a single section are shown alongside each other. Normally, the representations for different subsections are shown in different lines. Furthermore, section and subsection navigations are compressed into one line.

Some themes use the \insertnavigation to insert a navigation bar into the headline. Inside this bar, small icons are shown (called "mini frames") that represent the frames of a presentation. When you click on such an icon, the following happens:

- If you click on (the icon of) any frame other than the current frame, the presentation will jump to the first slide of the frame you clicked on.
- If you click on the current frame and you are not on the last slide of this frame, you will jump to the last slide of the frame.
- If you click on the current frame and you are on the last slide, you will jump to the first slide of the frame.

By the above rules you can:

- Jump to the beginning of a frame from somewhere else by clicking on it once.
- Jump to the end of a frame from somewhere else by clicking on it twice.
- Skip the rest of the current frame by clicking on it once.

We also tried making a jump to an already-visited frame jump automatically to the last slide of this frame. However, this turned out to be more confusing than helpful. With the current implementation a double-click always brings you to the end of a slide, regardless from where you come from.

Parent Beamer-Template mini frames

This parent template has the children mini frame and mini frame in current subsection.

Example: \setbeamertemplate{mini frames}[box]

The following template options are predefined:

- [default] shows small circles as mini frames.
- [box] shows small rectangles as mini frames.
- [tick] shows small vertical bars as mini frames.

Beamer-Template/-Color/-Font mini frame

The template is used to render the mini frame of the current frame in a navigation bar.

The width of the template is ignored. Instead, when multiple mini frames are shown, their position is calculated based on the beamer-sizes mini frame size and mini frame offset. See the command \setbeamersize for a description of how to change them.

Beamer-Template mini frame in current subsection

This template is used to render the mini frame of frames in the current subsection that are not the current frame. The beamer-color/-font mini frame installed prior to the usage of this template is invoked.

Beamer-Template mini frame in other subsection

This template is used to render mini frames of frames from subsections other than the current one.

The following template options are predefined:

12.2 Highlighting

The beamer class prede nes commands and environments for highlighting text. Using these commands makes it easy to change the appearance of a document by changing the theme.

\structure<⟨overlay speci cation⟩>{⟨text⟩}

The given text is marked as part of the structure, that is, it is supposed to help the audience see the structure of your presentation. If the ⟨overlay speci cation⟩ is present, the command only has an e ect on the speci ed slides.

Example: \structure{Paragraph Heading.}

Internally, this command just puts the text inside a structureenv environment.

article Structure text is typeset as bold text. This can be changed by modifying the templates.

Beamer-Color/-Font structure

This color/font is used when structured text is typeset, but it is also widely used as a base for many other colors including the headings of blocks, item buttons, and titles. In most color themes, the colors for navigational elements in the headline or the footline are derived from the foreground color of structure . By changing the structure color you can easily change the basic color of your presentation, other than the color of normal text. See also the related colorlocal structure and the related font tiny structure .

Inside the \structure command, the background of the color is ignored, but this is not necessarily true for elements that inherit their color from structure . There is no template structure , use structure begin and structure end instead.

Beamer-Color local structure

This color should be used to typeset structural elements that change their color according to the local environment. For example, an item button in an itemize environment changes its color according to circumstances. If it is used inside an example block, it should have the example text color; if it is currently alerted it should have the alerted text color. This color is setup by certain environments to have the color that should be used to typeset things like item buttons. Since the color used for items, item , inherits from this color by default, items automatically change their color according to the current situation.

If you write your own environment in which the item buttons and similar structural elements should have a di erent color, you should change the colorlocal structure inside these environments.

Beamer-Font tiny structure

This special font is used for tiny structural text. Basically, this font should be used whenever a structural element uses a tiny font. The idea is that the tiny versions of the structure font often are not suitable. For example, it is often necessary to use a boldface version for them. Also, one might wish to have serif smallcaps structural text, but still retain normal sans-serif tiny structural text.

\begin{structureenv}<⟨overlay speci cation⟩>
⟨environment contents⟩
\end{structureenv}

Environment version of the \structure command.

Beamer-Template structure begin

This text is inserted at the beginning of a structureenv environment.

The following template options are prede ned:

^ [default]

article The text is typeset in boldface.

Beamer-Template structure end

This text is inserted at the end of a structureenv environment.

\alert< hoverlay speci cationi >{ hhighlighted texti }

The given text is highlighted, typically by coloring the text red. If the hoverlay speci cationi is present, the command only has an e ect on the speci ed slides.

Example: This is \alert{important}.

Internally, this command just puts the highlighted text inside an alertenv .

article Alerted text is typeset as emphasized text. This can be changed by modifying the templates, see below.

Beamer-Color/-Font alerted text

This color/font is used when alerted text is typeset. The background is currently ignored. There is no template alerted text , rather there are templates alerted text begin and alerted text end that are inserted before and after alerted text.

\begin{alertenv}< hoverlay speci cationi >
henvironment contentsi
\end{alertenv}

Environment version of the \alert command.

Beamer-Template alerted text begin

This text is inserted at the beginning of a an alertenv environment.

The following template options are prede ned:

- [default]

presentation This changes the colorlocal structure to alerted text . This causes things like buttons or items to be colored in the same color as the alerted text, which is often visually pleasing. See also the \structure command.

article The text is emphasized.

Beamer-Template alerted text end

This text is inserted at the end of an alertenv environment.

12.3 Block Environments

The beamer class prede nes an environment for typesetting a block of text that has a heading. The appearance of blocks can easily be changed using the following template:

Parent Beamer-Template blocks

Changing this parent template changes the templates of normal blocks, alerted blocks, and example blocks.

Example: \setbeamertemplate{blocks}[default]

Example: \setbeamertemplate{blocks}[rounded][shadow=true]

The following template options are prede ned:

- [default] The default setting typesets the block title on its own line. If a background is speci ed either for the block title or for the block body , this background color is used as background of the title or body, respectively. For alerted and example blocks, the corresponding beamer -colors and -fonts are used, instead.
- [rounded][hshadow=truei] Makes the blocks rounded. This means that the corners of the backgrounds of the blocks are rounded o . If the shadow=true option is given, a shadow is drawn behind the block.

\begin{block}<⟨action specification⟩>{⟨block title⟩}<⟨action specification⟩>
  ⟨environment contents⟩
\end{block}

Only one ⟨action specification⟩ may be given. Inserts a block, like a definition or a theorem, with the title ⟨block title⟩. If the ⟨action specification⟩ is present, the given actions are taken on the specified slides, see Section 9.6.3. In the example, the definition is shown only from slide 3 onward.

Example:

```
\begin{block}<3->{Definition}
  A \alert{set} consists of elements.
\end{block}
```

article The block name is typeset in bold.

Beamer-Template block begin

This template is inserted at the beginning of a block before the ⟨environment contents⟩. Inside this template, the block title can be accessed via the following insert:

^ \insertblocktitle Inserts the ⟨block title⟩ into the template.

When the template starts, no special color or font is installed (for somewhat complicated reasons). Thus, this template should install the correct colors and fonts for the title and the body itself.

Beamer-Template block end

This template is inserted at the end of a block.

Beamer-Color/-Font block title

This beamer-color/-font should be used to typeset the title of the block. Since neither the color nor the font are setup automatically, the template block begin must do so itself.

The default block template and also the rounded version honor the background of this color.

Beamer-Color/-Font block body

This beamer-color/-font should be used to typeset the body of the block, that is, the ⟨environment contents⟩. As for block title, the color and font must be setup by the template block begin.

\begin{alertblock}<⟨action specification⟩>{⟨block title⟩}<⟨action specification⟩>
  ⟨environment contents⟩
\end{alertblock}

Inserts a block whose title is highlighting. Behaves like the block environment otherwise.

Example:

```
\begin{alertblock}{Wrong Theorem}
  $1=2$.
\end{alertblock}
```

article The block name is typeset in bold and is emphasized.

Beamer-Template block alerted begin

Same applies as for normal blocks.

Beamer-Template block alerted end

Same applies as for normal blocks.

Beamer-Color/-Font block title alerted

Same applies as for normal blocks.

Beamer-Color/-Font block body alerted

Same applies as for normal blocks.

The appearance of the theorem is governed by the templates theorem begin and theorem end, see their description later on for details on how to change these. Every theorem is put into a block environment, thus the templates for blocks also apply.

The theorem style (a concept from amsthm) used for this environment is plain . In this style, the body of a theorem should be typeset in italics. The head of the theorem should be typeset in a bold font, but this is usually overruled by the templates.

If the option envcountsect is given either as class option in one of the presentation modes or as an option to the package beamerarticle in article mode, then the numbering of the theorems is local to each section with the section number pre xing the theorem number; otherwise they are numbered consecutively throughout the presentation or article. We recommend using this option in article mode.

By default, no theorem numbers are shown in the presentation modes.

Example:

```
\begin{theorem}[Kummer, 1992]
  If $\#^_A^n$ is $n$-enumerable, then $A$ is recursive.
\end{theorem}

\begin{theorem}<2->[Tantau, 2002]
  If $\#_A^2$ is $2$-fa-enumerable, then $A$ is regular.
\end{theorem}
```

The environments corollary , fact , and lemma behave exactly the same way.

\documentclass[envcountsect]{beamer}

Causes theorems, de nitions, and so on to be numbered locally to each section. Thus, the rst theorem of the second section would be Theorem 2.1 (assuming that there are no de nitions, lemmas, or corollaries earlier in the section).

\begin{definition}<haction speci cation i>[hadditional texti]<haction speci cation i>
  henvironment contentsi
\end{definition}

Behaves like the theorem environment, except that the theorem style definition is used. In this style, the body of a theorem is typeset in an upright font.

The environment definitions behaves exactly the same way.

\begin{example}<haction speci cation i>[hadditional texti]<haction speci cation i>
  henvironment contentsi
\end{example}

Behaves like the theorem environment, except that the theorem style example is used. A side-e ect of using this theorem style is that the henvironment contentsi is put in an exampleblock instead of a block .

The environment examples behaves exactly the same way.

presentation — The default template for typesetting theorems suppresses the theorem number, even if this number is available for typesetting (which it is by default in all prede ned environments; but if you de ne your own environment using \newtheorem* no number will be available).

article — In article mode, theorems are automatically numbered. By specifying the class option envcountsect , theorems will be numbered locally to each section, which is usually a good idea, except for very short articles.

\begin{proof}<haction speci cation i>[hproof namei]<haction speci cation i>
  henvironment contentsi
\end{proof}

Typesets a proof. If the optional hproof namei is given, it completely replaces the word Proof. This is di erent from normal theorems, where the optional argument is shown in brackets.

At the end of the theorem, a \qed symbol is shown, except if you say \qed here earlier in the proof (this is exactly as in amsthm). The default \qed symbol is an open circle. To completely suppress the symbol, write \def\qedsymbol{} in your preamble. To get a closed square, say

\setbeamertemplate{qed symbol}{\vrule width1.5ex height1.5ex depth0pt}

If you use babel and a different language, the text Proof is replaced by whatever is appropriate in the selected language.

Example:

```
\begin{proof}<2->[Sketch of proof]
  Suppose ...
\end{proof}
```

Beamer-Template/-Color/-Font qed symbol

The symbol is shown at the end of every proof.

You can define new environments using the following command:

\newtheorem*{⟨environment name⟩}[⟨numbered same as⟩]{⟨head text⟩}[⟨number within⟩]

This command is used exactly the same way as in the amsthm package (as a matter of fact, it is the command from that package). For example, the two optional arguments, ⟨numbered same as⟩ and ⟨number within⟩, are mutually exclusive; see the documentation of amsthm for details. The only difference is that environments declared using this command are overlay specification-aware in beamer and that, when typeset, are typeset according to beamer 's templates.

article Environments declared using this command are also overlay specification-aware in article mode.

Example: \newtheorem{observation}[theorem]{Observation}

You can also use amsthm's command \newtheoremstyle to define new theorem styles. Note that the default template for theorems will ignore any head font setting, but will honor the body font setting.

If you wish to define the environments like theorem differently (for example, have it numbered within each subsection), you can use the following class option to disable the definition of the predefined environments:

\documentclass[notheorems]{beamer}

Switches off the definition of default blocks like theorem, but still loads amsthm and makes theorems overlay specification-aware.

The option is also available as a package option for beamerarticle and has the same effect.

article In the article version, the package amsthm sometimes clashes with the document class. In this case you can use the following option, which is once more available as a class option for beamer and as a package option for beamerarticle, to switch off the loading of amsthm altogether.

\documentclass[noamsthm]{beamer}

Does not load amsthm and also not amsmath. Environments like theorem or proof will not be available.

\documentclass[noamssymb]{beamer}

Does not load amssymb. This option is mainly intended for users who are loading specialist font packages. Note that \blacktriangleright needs to be defined if itemize environments are in use.

Parent Beamer-Template theorems

This template is a parent of theorem begin and theorem end, see the first for a detailed discussion of how the theorem templates are set.

Example: \setbeamertemplate{theorems}[numbered]

The following template options are predefined:

- [default] By default, theorems are typeset as follows: The font specification for the body is honored, the font specification for the head is ignored. No theorem number is printed.
- [normal font] Like the default, except all font specifications for the body are ignored. Thus, the fonts are used that are normally used for blocks.
- [numbered] This option is like the default, except that the theorem number is printed for environments that are numbered.
- [ams style] This causes theorems to be put in a block or example block, but to be otherwise typeset as is normally done in amsthm. Thus the head font and body font depend on the setting for the theorem to be typeset and theorems are numbered.

Beamer-Template theorem begin

Whenever an environment declared using the command \newtheorem is to be typeset, this template is inserted at the beginning and the template theorem end at the end. If there is an overlay specification when an environment like theorem is used, this overlay specification will directly follow the \block beginning template\i upon invocation. This is even true if there was an optional argument to the theorem environment. This optional argument is available via the insert \inserttheoremaddition.

Numerous inserts are available in this template, see below.

Before the template starts, the font is set to the body font prescribed by the environment to be typeset.

Example: The following typesets theorems like amsthm:

```
\setbeamertemplate{theorem begin}
{%
  \begin{\inserttheoremblockenv}
  {%
    \inserttheoremheadfont
    \inserttheoremname
    \inserttheoremnumber
    \ifx\inserttheoremaddition\@empty\else\ (\inserttheoremaddition)\fi%
    \inserttheorempunctuation
  }%
}
\setbeamertemplate{theorem end}{\end{\inserttheoremblockenv}}
```

Example: In the following example, all font suggestions for the environment are suppressed or ignored; and the theorem number is suppressed.

```
\setbeamertemplate{theorem begin}
{%
  \normalfont% ignore body font
  \begin{\inserttheoremblockenv}
  {%
    \inserttheoremname
    \ifx\inserttheoremaddition\@empty\else\ (\inserttheoremaddition)\fi%
  }%
}
\setbeamertemplate{theorem end}{\end{\inserttheoremblockenv}}
```

The following inserts are available inside this template:

- \inserttheoremblockenv This will normally expand to block, but if a theorem that has theorem style example is typeset, it will expand to example block. Thus you can use this insert to decide which environment should be used when typesetting the theorem.
- \inserttheoremheadfont This will expand to a font changing command that switches to the font to be used in the head of the theorem. By not inserting it, you can ignore the head font.
- \inserttheoremname This will expand to the name of the environment to be typeset (like Theorem or Corollary).

- \inserttheoremnumber This will expand to the number of the current theorem preceded by a space or to nothing, if the current theorem does not have a number.
- \inserttheoremaddition This will expand to the optional argument given to the environment or will be empty, if there was no optional argument.
- \inserttheorempunctuation This will expand to the punctuation character for the current environment. This is usually a period.

Beamer-Template theorem end

Inserted at the end of a theorem.

Beamer-Template proof begin

Inserted at the beginning of a proof environment. This template behaves like a normal block begin template by default.

- \insertproofname This will expand to the proof name, followed by a period most of the time.

Beamer-Template proof end

Inserted at the end of a proof environment.

12.5 Framed and Boxed Text

In order to draw a frame (a rectangle) around some text, you can use LaTeXs standard command \fbox and also \frame (inside a beamer frame, the \frame command changes its meaning to the normal LaTeX \frame command). More frame types are offered by the package fancybox, which defines the following commands: \shadowbox, \doublebox, \ovalbox, and \Ovalbox. Please consult the LaTeX Companion for details on how to use these commands.

The beamer class also defines two environments for creating colored boxes.

\begin{beamercolorbox}[⟨options⟩]{⟨beamer color⟩}
⟨environment contents⟩
\end{beamercolorbox}

This environment can be used to conveniently typeset some text using some beamer-color. Basically, the following two command blocks do the same:

```
\begin{beamercolorbox}{beamer color}
  Text
\end{beamercolorbox}

{
  \usebeamercolor{beamer color}
  \colorbox{bg}{
    \color{fg}
    Text
  }
}
```

In other words, the environment installs the ⟨beamer color⟩ and uses the background for the background of the box and the foreground for the text inside the box. However, in reality, numerous ⟨options⟩ can be given to specify in much greater detail how the box is rendered.

If the background color of ⟨beamer color⟩ is empty, no background is drawn behind the text, that is, the background is transparent.

This command is used extensively by the default inner and outer themes for typesetting the headlines and footlines. It is not really intended to be used in normal frames (for example, it is not available inside article mode). You should prefer using structuring elements like blocks or theorems that automatically insert colored boxes as needed.

Example: The following example could be used to typeset a headline with two lines, the first showing the document title, the second showing the author's name:

```
\begin{beamercolorbox}[ht=2.5ex,dp=1ex,center]{title in head/foot}
  \usebeamerfont{title in head/foot}
  \insertshorttitle
\end{beamercolorbox}%
\begin{beamercolorbox}[ht=2.5ex,dp=1ex,center]{author in head/foot}
  \usebeamerfont{author in head/foot}
  \insertshortauthor
\end{beamercolorbox}
```

Example: Typesetting a postit:

```
\setbeamercolor{postit}{fg=black,bg=yellow}
\begin{beamercolorbox}[sep=1em,wd=5cm]{postit}
  Place me somewhere!
\end{beamercolorbox}
```

The following ⟨options⟩ can be given:

- wd=⟨width⟩ sets the width of the box. This command has two effects: First, TEX's \hsize is set to ⟨width⟩. Second, after the box has been typeset, its width is set to ⟨width⟩ (no matter what it actually turned out to be). Since setting the \hsize does not automatically change some of LATEX's linewidth dimensions, you should consider using a minipage inside this environment if you fool around with the width.

  If the width is larger than the normal text width, as specified by the value of \textwidth, the width of the resulting box is reset to the width \textwidth, but intelligent negative skips are inserted at the left and right end of the box. The net effect of this is that you can use a width larger than the text width for a box and can insert the resulting box directly into normal text without getting annoying warnings and having the box positioned sensibly.
- dp=⟨depth⟩ sets the depth of the box, overriding the real depth of the box. The box is first typeset normally, then the depth is changed afterwards. This option is useful for creating boxes that have guaranteed size.

  If the option is not given, the box has its natural depth, which results from the typesetting. For example, a box containing only the letter a will have a different depth from a box containing only the letter g.
- ht=⟨height⟩ sets the height of the box, overriding the real height. Note that the height does not include the depth (see, for example, the TEX-book for details). If you want a one-line box that always has the same size, setting the height to 2.25ex and the depth to 1ex is usually a good option.
- left causes the text inside the box to be typeset left-aligned and with a (radically) ragged right border. This is the default. To get a better ragged right border, use the rightskip option. Note that this will override any leftskip or rightskip setting.
- right causes the text to be right-aligned with a (radically) ragged left border. Note that this will override any leftskip or rightskip setting.
- center centers the text inside the box. Note that this will override any leftskip or rightskip setting.
- leftskip=⟨left skip⟩ installs the ⟨left skip⟩ inside the box as the left skip. TEX's left skip is a glue that is inserted at the left end of every line. See the TEX-book for details. Note that this will override any left, center or right setting.
- rightskip=⟨right skip⟩ install the ⟨right skip⟩. To get a good ragged right border, try, say, \rightskip=0pt plus 4em. Note that this will override any left, center or right setting.
- sep=⟨dimension⟩ sets the size of extra space around the text. This space is added inside the box, which means that if you specify a sep of 1cm and insert the box normally into the vertical list, then the left border of the box will be aligned with the left border of the slide text, while the left border of the text inside the box will be 1cm to the right of this left border. Likewise, the text inside the box will stop 1cm before the right border of the normal text.

- colsep=⟨dimension⟩ sets the extra color separation space around the text. This space behaves the same way as the space added by sep, only this space is only inserted if the box has a colored background, that is, if the background of the ⟨beamer color⟩ is not empty. This command can be used together with sep, the effects accumulate.
- colsep*=⟨dimension⟩ sets an extra color separation space around the text that is horizontally outside the box. This means that if the box has a background, this background will protrude by ⟨dimension⟩ to the left and right of the text, but this protruding background will not be taken into consideration by TEX for typesetting purposes.

  A typical example usage of this option arises when you insert a box with a colored background in the middle of normal text. In this case, if the background color is set, you would like a background to be drawn behind the text and you would like a certain extra space around this text (the background should not stop immediately at the borders of the text, this looks silly) and you would like the normal text always to be at the same horizontal position, independently of whether a background is present or not. In this case, using colsep*=4pt is a good option.
- shadow=⟨true or false⟩ draws a shadow behind the box. Currently, this option only has an effect if used together with the rounded option, but that may change.
- rounded=⟨true or false⟩ causes the borders of the box to be rounded or if there is a background installed. This command internally calls beamerboxesrounded. In this case, colsep* option will have no effect.
- ignorebg causes the background color of the ⟨beamer color⟩ to be ignored, that is, to be treated as if it were set to transparent or empty.
- vmode causes TEX to be in vertical mode when the box starts. Normally, TEX will be in horizontal mode at the start of the box (a \leavevmode is inserted automatically at the beginning of the box unless this option is given). Only TEXperts need this option, so, if you use it, you will probably know what you are doing anyway.

\begin{beamerboxesrounded}[⟨options⟩]{⟨head⟩}
  ⟨environment contents⟩
\end{beamerboxesrounded}

The text inside the environment is framed by a rectangular area with rounded corners. For the large rectangular area, the beamer -color specified with the lower option is used. Its background is used for the background, its foreground for the foreground. If the ⟨head⟩ is not empty, ⟨head⟩ is drawn in the upper part of the box using the beamer -color specified with the upper option for the fore- and background. The following options can be given:

- lower=⟨beamer color⟩ sets the beamer -color to be used for the lower (main) part of the box. Its background is used for the background, its foreground for the foreground of the main part of the box. If either is empty, the current background or foreground is used. The box will never be transparent.
- upper=⟨beamer color⟩ sets the beamer -color used for the upper (head) part of the box. It is only used if the ⟨head⟩ is not empty.
- width=⟨dimension⟩ causes the width of the text inside the box to be the specified ⟨dimension⟩. By default, the \textwidth is used. Note that the box will protrude 4pt to the left and right.
- shadow=⟨true or false⟩. If set to true, a shadow will be drawn.

If no ⟨head⟩ is given, the head part is completely suppressed.

Example:

```
\begin{beamerboxesrounded}[upper=block head,lower=block body,shadow=true]{Theorem}
  $A = B$.
\end{beamerboxesrounded}
```

article This environment is not available in article mode.

12.6 Figures and Tables

You can use the standard LaTeX environments figure and table much the same way you would normally use them. However, any placement specification will be ignored. Figures and tables are immediately inserted where the environments start. If there are too many of them to fit on the frame, you must manually split them among additional frames or use the allowframebreaks option.

Example:

```
\begin{frame}
  \begin{figure}
    \pgfuseimage{myfigure}
    \caption{This caption is placed below the figure.}
  \end{figure}

  \begin{figure}
    \caption{This caption is placed above the figure.}
    \pgfuseimage{myotherfigure}
  \end{figure}
\end{frame}
```

Beamer-Template/-Color/-Font caption

This template is used to render the caption.

The following template options are predefined:

- [default] typesets the caption name (a word like Figure or Abbildung or Table, depending on whether a table or figure is typeset and depending on the currently installed language) before the caption text. No number is printed, since these make little sense in a normal presentation.
- [numbered] adds the figure or table number to the caption. Use this option only if your audience has a printed handout or printed lecture notes that follow the same numbering.
- [caption name own line] As the name suggests, this options puts the caption name (like Figure) on its own line.

Inside the template, you can use the following inserts:

- \insertcaption Inserts the text of the current caption.
- \insertcaptionname Inserts the name of the current caption. This word is either Table or Figure or, if the babel package is used, some translation thereof.
- \insertcaptionnumber Inserts the number of the current figure or table.

Beamer-Color/-Font caption name

These beamer-color and -font are used to typeset the caption name (a word like Figure). The caption template must directly use them, they are not installed automatically by the \insertcaptionname command.

Beamer-Template caption label separator

This template is inserted between caption name and caption text.

The following template options are predefined:

- [default] Typesets the colon followed by the space.
- [none] Typesets no separator.
- [colon] Alias for the default.
- [period] Typesets the period followed by the space.
- [space] Typesets the space.
- [quad] Typesets the \quad followed by the space.
- [endash] Typesets the en-dash surrounded by spaces (-).

12.7 Splitting a Frame into Multiple Columns

The beamer class offers several commands and environments for splitting (perhaps only part of) a frame into multiple columns. These commands have nothing to do with LaTeX's commands for creating columns. Columns are especially useful for placing a graphic next to a description/explanation.

The main environment for creating columns is called `columns`. Inside this environment, you can either place several `column` environments, each of which creates a new column, or use the `\column` command to create new columns.

`\begin{columns}<`*overlay specification*`>[`*options*`]`
environment contents
`\end{columns}`

A multi-column area. If *overlay specification* is present, the environment will only be shown on the specified slides. Inside the environment you should place only `column` environments or `\column` commands (see below). The following *options* may be given:

- `b` will cause the bottom lines of the columns to be vertically aligned.
- `c` will cause the columns to be centered vertically relative to each other. Default, unless the global option `t` is used.
- `onlytextwidth` is the same as `totalwidth=\textwidth`.
- `t` will cause the first lines of the columns to be aligned. Default if global option `t` is used.
- `T` is similar to the `t` option, but `T` aligns the tops of the first lines while `t` aligns the so-called baselines of the first lines. If strange things seem to happen in conjunction with the `t` option (for example if a graphic suddenly drops down with the `t` option instead of going up,), try using this option instead.
- `totalwidth=`*width* will cause the columns to occupy not the whole page width, but only *width*, all told. Note that this means that any margins are ignored.

Example:

```
\begin{columns}[t]
  \begin{column}{5cm}
    Two\\lines.
  \end{column}
  \begin{column}{5cm}
    One line (but aligned).
  \end{column}
\end{columns}
```

Example:

```
\begin{columns}[t]
  \column{5cm}
    Two\\lines.

  \column[T]{5cm}
    \includegraphics[height=3cm]{mygraphic.jpg}
\end{columns}
```

article This environment is ignored in article mode.

To create a column, you can either use the `column` environment or the `\column` command.

`\begin{column}<`*overlay specification*`>[`*placement*`]{`*column width*`}`
environment contents
`\end{column}`

Creates a single column of width *column width*. If *overlay specification* is present, the column will only be shown on the specified slides. The vertical placement of the enclosing `columns` environment can be

overruled by specifying a specific ⟨placement⟩ (t and T for the two top modes, c for centered, and b for bottom).

Example: The following code has the same effect as the above examples:

```
\begin{columns}
  \begin{column}[t]{5cm}
    Two\\lines.
  \end{column}
  \begin{column}[t]{5cm}
    One line (but aligned).
  \end{column}
\end{columns}
```

article This command is ignored in article mode.

\column<⟨overlay specification⟩>[⟨placement⟩]{⟨column width⟩}

Starts a single column. The parameters and options are the same as for the column environment. The column automatically ends with the next occurrence of \column or of a column environment or of the end of the current columns environment.

Example:

```
\begin{columns}
  \column[t]{5cm}
    Two\\lines.
  \column[t]{5cm}
    One line (but aligned).
\end{columns}
```

article This command is ignored in article mode.

12.8 Positioning Text and Graphics Absolutely

Normally, beamer uses TEX's normal typesetting mechanism to position text and graphics on the page. In certain situation you may instead wish a certain text or graphic to appear at a page position that is specified absolutely. This means that the position is specified relative to the upper left corner of the slide.

The package textpos provides several commands for positioning text absolutely and it works together with beamer. When using this package, you will typically have to specify the option overlay and perhaps absolute. For details on how to use the package, please see its documentation.

12.9 Verbatim and Fragile Text

If you wish to use a {verbatim} environment in a frame, you have to add the option [fragile] to the {frame} environment. The \end{frame} must be alone on a single line (except for any leading whitespace). Using this option will cause the frame contents to be written to an external file and then read back. See the description of the {frame} environment for more details.

You must also use the [fragile] option for frames that include any fragile text, which is any text that is not interpreted the way text is usually interpreted by TEX. For example, if you use a package that (locally) redefined the meaning of, say, the character &, you must use this option.

Inside {verbatim} environments you obviously cannot use commands like \alert<2> to highlight part of code since the text is written in, well, verbatim. There are several good packages like alltt or listings that allow you to circumvent this problem. For simple cases, the following environment can be used, which is defined by beamer:

```
\begin{semiverbatim}
  ⟨environment contents⟩
```

\end{semiverbatim}

The text inside this environment is typeset like verbatim text. However, the characters \ , { , and } retain their meaning. Thus, you can say things like

\alert<1->{std::cout << "AT&T likes 100% performance";}

To typeset the three characters \ , { , and } you can use the commands \\ (which is redefined inside this environment you do not need it anyway), \{ , and \} . Thus in order to get typeset \alert<1>{X} you can write \\alert<1>\{X\} .

12.10 Abstract

The {abstract} environment is overlay specification-aware in beamer :

\begin{abstract}<⟨action specification⟩>
⟨environment contents⟩
\end{abstract}

You can use this environment to typeset an abstract.

Beamer-Color/-Font abstract

These beamer -color and -font are used to typeset the abstract. If a background color is set, this background color is used as background for the whole abstract by default.

Beamer-Template/-Color/-Font abstract title

Color parents: titlelike

This template is used for the title. By default, this inserts the word \abstractname , centered. The background color is ignored.

Beamer-Template abstract begin

This template is inserted at the very beginning of the abstract, before the abstract title and the ⟨environment contents⟩ is inserted.

Beamer-Template abstract end

This template is inserted at the end of the abstract, after the ⟨environment contents⟩ .

12.11 Verse, Quotations, Quotes

LaTeX defines three environments for typesetting quotations and verses: verse , quotation , and quote . These environments are also available in the beamer class, where they are overlay specification-aware. If an overlay specification is given, the verse or quotation is shown only on the specified slides and is covered otherwise. The difference between a quotation and a quote is that the first has paragraph indentation, whereas the second hasn't.

You can change the font and color used for these by changing the beamer -colors and -fonts listed below. Unlike the standard LaTeX environments, the default font theme typesets a verse in an italic serif font, quotations and quotes are typeset using an italic font (whether serif or sans-serif depends on the standard document font).

\begin{verse}<⟨action specification⟩>
⟨environment contents⟩
\end{verse}

You can use this environment to typeset a verse.

Beamer-Color/-Font verse

These beamer -color and -font are used to typeset the verse. If a background color is set, this background color is used as background for the whole abstract. The default font is italic serif.

Beamer-Template verse begin

This template is inserted at the beginning of the verse.

Beamer-Template verse end

This template is inserted at the end of the verse.

\begin{quotation}<⟨action specification⟩>
⟨environment contents⟩
\end{quotation}

Use this environment to typeset multi-paragraph quotations. Think again, before presenting multi-paragraph quotations.

Beamer-Color/-Font quotation

These beamer -color and -font are used to typeset the quotation.

Beamer-Template quotation begin

This template is inserted at the beginning of the quotation.

Beamer-Template quotation end

This template is inserted at the end of the quotation.

\begin{quote}<⟨action specification⟩>
⟨environment contents⟩
\end{quote}

Use this environment to typeset a single-paragraph quotation.

Beamer-Color/-Font quote

These beamer -color and -font are used to typeset the quote.

Beamer-Template quote begin

This template is inserted at the beginning of the quote.

Beamer-Template quote end

This template is inserted at the end of the quote.

12.12 Footnotes

First a word of warning: Using footnotes is usually not a good idea. They disrupt the flow of reading.

You can use the usual \footnote command. It has been augmented to take an additional option, for placing footnotes at the frame bottom instead of at the bottom of the current minipage.

\footnote<⟨overlay specification⟩>[⟨options⟩]{⟨text⟩}

Inserts a footnote into the current frame. Footnotes will always be shown at the bottom of the current frame; they will never be moved to other frames. As usual, one can give a number as ⟨options⟩, which will cause the footnote to use that number. The beamer class adds one additional option:

- frame causes the footnote to be shown at the bottom of the frame. This is normally the default behavior anyway, but in minipages and certain blocks it makes a difference. In a minipage, the footnote is usually shown as part of the minipage rather than as part of the frame.

If an ⟨overlay specification⟩ is given, this causes the footnote ⟨text⟩ to be shown only on the specified slides. The footnote symbol in the text is shown on all slides.

Example: \footnote{On a fast machine.}

Example: \footnote[frame,2]{Not proved.}

Example: \footnote<.->{Der Spiegel, 4/04, S.~90.}

article In article mode, footnotes are typeset as usual. The frame option has no effect, which means in a minipage the footnote is shown as part of it.

Beamer-Template/-Color/-Font footnote

This template will be used to render the footnote. Inside this template, the following two inserts can be used:

- \insertfootnotetext Inserts the current footnote text.
- \insertfootnotemark Inserts the current footnote mark (like a raised number). This mark is computed automatically.

Beamer-Color/-Font footnote mark

This beamer -color/-font is used when rendering the footnote mark, both in the text and at the beginning of the footnote itself.

13 Graphics

In the following, the advantages and disadvantages of different possible ways of creating graphics for beamer presentations are discussed. Much of the information presented in the following is not really special to beamer, but applies to any other document class as well.

13.1 Including External Graphic Files Versus Inlines Graphics

There are two principal ways of creating TEX-documents that include graphics: Either the graphic resides in an external file that is included or the graphic is inlined, which means that TEX-file contains a bunch of commands like draw a red line from here to there. In the following, the advantages and disadvantages of these two approaches are discussed.

You can use an external program, like xfig, GIMP or Inkscape, to create a graphic. These programs have an option to export graphic files in a format that can then be inserted into the presentation.

The main advantage is:

- You can use a powerful program to create a high-quality graphic.

The main disadvantages are:

- You have to worry about many files. Typically there are at least two for each presentation, namely the program's graphic data file and the exported graphic file in a format that can be read by TEX.
- Changing the graphic using the program does not automatically change the graphic in the presentation. Rather, you must reexport the graphic and rerun LATEX.
- It may be difficult to get the line width, fonts, and font sizes right.
- Creating formulas as part of graphics is often difficult or impossible.

You can use all the standard LATEX commands for inserting graphics, like \includegraphics. Also, the pgf package offers commands for including graphics. Either will work fine in most situations, so choose whichever you like. Like \pgfdeclareimage, \includegraphics also includes an image only once in a pdf file, even if it used several times (as a matter of fact, the graphics package is even a bit smarter about this than pgf). However, currently only pgf offers the ability to include images that are partly transparent.

At the end of this section you will find notes on how to include specific graphic formats like .eps or .jpg.

The commands \includegraphics, \pgfuseimage, and \pgfimage are overlay-specification-aware in beamer. If the overlay specification does not apply, the command has no effect. This is useful for creating a simple animation where each picture of the animation resides in a different file:

```
\begin{frame}
  \includegraphics<1>[height=2cm]{step1.pdf}%
  \includegraphics<2>[height=2cm]{step2.pdf}%
  \includegraphics<3>[height=2cm]{step3.pdf}%
\end{frame}
```

A different way of creating graphics is to insert graphic drawing commands directly into your LATEX file. There are numerous packages that help you do this. They have various degrees of sophistication. Inlining graphics suffers from none of the disadvantages mentioned above for including external graphic files, but the main disadvantage is that it is often hard to use these packages. In some sense, you program your graphics, which requires a bit of practice.

When choosing a graphic package, there are a few things to keep in mind:

- Many packages produce poor quality graphics. This is especially true of the standard picture environment of LATEX.
- Powerful packages that produce high-quality graphics often do not work together with pdflatex, lualatex or xelatex.

- The most powerful and easiest-to-use package around, namely pstricks, does not work together with pdflatex, lualatex or xelatex and this is a fundamental problem. Due to the fundamental differences between pdf and PostScript, it is not possible to write a pdflatex back-end for pstricks. (Situation with lualatex and xelatex is very similar.) Regardless, pst-pdf, xetex-pstricks and pdftricks package can help here and simplify things from user's perspective.

There are three possible solutions to the above problem, each with it's own advantages and disadvantages.

- Use the pgf package. It produces high-quality graphics and works together with pdflatex, lualatex, xelatex and also with normal latex. It is not as powerful as pstricks (as pointed out above, this is because of rather fundamental reasons) and not as easy to use, but it should be sufficient in most cases.
- Use luamplib package and lualatex. It provides you with an environment using which you can type MetaPost code directly in your document.
- Use pstricks and stick to latex and dvips or use some of the workarounds mentioned above.

13.2 Including Graphic Files Ending .eps or .ps

External graphic files ending with the extension .eps (Encapsulated PostScript) or .ps (PostScript) can be included if you use latex and dvips, but not when using pdflatex. This is true both for the normal graphics package and for pgf. When using pgf, do not add the extension .eps. When using graphics, do add the extension.

If you have a .eps graphic and wish to use pdflatex, you can use the program ps2pdf to convert the graphic to a .pdf file. Some modern distributions enable write18 which allows pdflatex to do that automatically. Note, however, that it is often a better idea to directly generate a .pdf if the program that produced the .eps supports this.

13.3 Including Graphic Files Ending .pdf, .jpg, .jpeg or .png

The four formats .pdf, .jpg, .jpeg, and .png can only be included by pdflatex. As before, do not add these extension when using pgf, but do add them when using graphics. If your graphic file has any of these formats and you wish/must use latex and dvips, you have to convert your graphic to .eps first.

13.4 Including Graphic Files Ending .mps

A graphic file ending .mps (MetaPost PostScript) is a special kind of Encapsulated PostScript file. Files in this format are produced by the MetaPost program. As you know, TEX is a program that converts simple plain text into beautifully typeset documents. The MetaPost program is similar, only it converts simple plain text into beautiful graphics.

The MetaPost program converts a plain text file ending .mp into an .mps file (although for some unfathomable reason the extension is not added). The .mp file must contain text written in the MetaFont programming language. Since .mps files are actually also .eps files, you can use the normal \includegraphics command to include them.

However, as a special bonus, you can also include such a file when using pdflatex. Normally, pdflatex cannot handle .eps files, but the .mps files produced by MetaPost have such a simple and special structure that this is possible. The graphics package implements some filters to convert such PostScript output to pdf on-the-fly. For this to work, the file should end .mps instead of .eps. The following command can be used to make the graphics package just assume the extension .mps for any file it knows nothing about (like files ending with .1, which is what MetaPost loves to produce):

```
\DeclareGraphicsRule{*}{mps}{*}{}
```

This special feature currently only works with the graphics package, not with pgf.

13.5 Including Graphic Files Ending .mmp

The format .mmp (Multi-MetaPost) is actually not a format that can be included directly in a TEX-file. Rather, like a .mp file, it first has to be converted using the MetaPost program. The crucial difference between .mp and .mmp is that in the latter multiple graphics can reside in a single .mmp file (actually, multiple graphics can also reside in a .mp file, but by convention such a file is called .mmp). When running MetaPost on a .mmp file, it will create not a single encapsulated PostScript file, but several, ending .0, .1, .2, and so on. The idea is that .0 might contain a main graphic and the following pictures contain overlay material that should be incrementally added to this graphic.

To include the series of resulting files, you can use the command \multiinclude from the mpmulti or from the xmpmulti package. How this program works is explained in Section 14.1.3.

14 Animations, Sounds, and Slide Transitions

14.1 Animations

14.1.1 Including External Animation Files

If you have created an animation using some external program (like a renderer), you can use the capabilities of the presentation program (like the Acrobat Reader) to show the animation. Unfortunately, currently there is no portable way of doing this and even the Acrobat Reader does not support this feature on all platforms.

To include an animation in a presentation, you can use, for example, the package `multimedia.sty` which is part of the beamer package. You have to include this package explicitly. Despite being distributed as part of the beamer distribution, this package is perfectly self-su cient and can be used independently of beamer .

\usepackage{multimedia}

A stand-alone package that implements several commands for including external animation and sound les in a pdf document. The package can be used together with both dvips plus ps2pdf and pdflatex , though the special sound support is available only in pdflatex .

When including this package, you must also include the hyperref package. The multimedia is one of the few packages that needs to be loaded after hyperref . Since beamer includes hyperref automatically, you need not worry about this when creating a presentation using beamer .

For including an animation in a pdf le, you can use the command \movie , which is explained below. Depending on the used options, this command will either setup the pdf le such that the viewer application (like the Acrobat Reader) itself will try to play the movie or that an external program will be called. The latter approach, though much less exible, must be taken if the viewer application is unable to display the movie itself.

\movie[hoptionsi]{ hposter textiv }{ hmovie lenamei }

This command will insert the movie with the lename hmovie lenamei into the pdf le. The movie le must reside at some place where the viewer application will be able to nd it, which is typically only the directory in which the nal pdf le resides. The movie le will not be embedded into the pdf le in the sense that the actual movie data is part of the main.pdf le. The movie le must hence be copied and passed along with the pdf le. (Nevertheless, one often says that the movie is embedded in the document, but that just means that one can click on the movie when viewing the document and the movie will start to play.)

The movie will use a rectangular area whose size is determined either by the width= and height= options or by the size of the hposter textiv . The hposter textiv can be any TEX text; for example, it might be a \pgfuseimage command or an \includegraphics command or a pgfpicture environment or just plain text. The hposter textiv is typeset in a box, the box is inserted into the normal text, and the movie rectangle is put exactly over this box. Thus, if the hposter textiv is an image from the movie, this image will be shown until the movie is started, when it will be exactly replaced by the movie itself. However, there is also a di erent, sometimes better, way of creating a poster image, namely by using the poster option as explained later on.

The aspect ratio of the movie will not be corrected automatically if the dimension of the hposter textiv box does not have the same aspect ratio. Most movies have an aspect ratio of 4:3 or 16:9.

Despite the name, a movie may consist only of sound with no images. In this case, the hposter textiv might be a symbol representing the sound. There is also a di erent, dedicated command for including sounds in a pdf le, see the \sound command in Section 14.2.

Unless further options are given, the movie will start only when the user clicks on it. Whether the viewer application can actually display the movie depends on the application and the version. For example, the Acrobat Reader up to version 5 does not seem to be able to display any movies or sounds on Linux. On the other hand, the Acrobat Reader Version 6 on MacOS is able to display anything that QuickTime can display, which is just about everything. Embedding movies in a pdf document is provided for by the pdf standard and is not a peculiarity of the Acrobat Reader. In particular, one might expect other viewers like xpdf and poppler -based viewers (Okular, Evince) to support embedded movies in the future.

Example: \movie{\pgfuseimage{myposterimage}}{mymovie.avi}

Example: \movie[width=3cm,height=2cm,poster]{}{mymovie.mpg}

If your viewer application is not able to render your movie, but some external application is, you must use the externalviewer option. This will ask the viewer application to launch an application for showing the movie instead of displaying it itself. Since this application is started in a new window, this is not nearly as nice as having the movie displayed directly by the viewer (unless you use evil trickery to suppress the frame of the viewer application). Which application is chosen is left to the discretion of the viewer application, which tries to make its choice according to the extension of the ⟨movie filename⟩ and according to some mapping table for mapping extensions to viewer applications. How this mapping table can be modified depends on the viewer application, please see the release notes of your viewer.

The following ⟨options⟩ may be given:

- autostart. Causes the movie to start playing immediately when the page is shown. At most one movie can be started in this way. The viewer application will typically be able to show at most one movie at the same time anyway. When the page is no longer shown, the movie immediately stops. This can be a problem if you use the \movie command to include a sound that should be played on after the page has been closed. In this case, the \sound command must be used.
- borderwidth=⟨TEX dimension⟩. Causes a border of thickness ⟨TEX dimension⟩ to be drawn around the movie. Some versions of the Acrobat Reader seem to have a bug and do not display this border if is smaller than 0.5bp (about 0.51pt).
- depth=⟨TEX dimension⟩. Overrides the depth of the ⟨poster text⟩ box and sets it to the given dimension.
- duration=⟨time⟩s. Specifies in seconds how long the movie should be shown. The ⟨time⟩ may be a fractional value and must be followed by the letter s. For example, duration=1.5s will show the movie for one and a half seconds. In conjunction with the start option, you can cut out a part of a movie for display.
- externalviewer. As explained above, this causes an external application to be launched for displaying the movie in a separate window. Most options, like duration or loop, have no effect since they are not passed along to the viewer application.
- height=⟨TEX dimension⟩. Overrides the height of the ⟨poster text⟩ box and sets it to the given dimension.
- label=⟨movie label⟩. Assigns a label to the movie such that it can later be referenced by the command \hyperlinkmovie, which can be used to stop the movie or to show a different part of it. The ⟨movie label⟩ is not a normal label. It should not be too fancy, since it is inserted literally into the pdf code. In particular, it should not contain closing parentheses.
- loop. Causes the movie to start again when the end has been reached. Normally, the movie just stops at the end.
- once. Causes the movie to just stop at the end. This is the default.
- open. Causes the player to stay open when the movie has finished.
- palindrome. Causes the movie to start playing backwards when the end has been reached, and to start playing forward once more when the beginning is reached, and so on.
- poster. Asks the viewer application to show the first image of the movie when the movie is not playing. Normally, nothing is shown when the movie is not playing (and thus the box containing the ⟨poster text⟩ is shown). For a movie that does not have any images (but sound) or for movies with an uninformative first image this option is not so useful.
- repeat is the same as loop.
- showcontrols=⟨true or false⟩. Causes a control bar to be displayed below the movie while it is playing. Instead of showcontrols=true you can also just say showcontrols. By default, no control bar is shown.
- start=⟨time⟩s. Causes the first ⟨time⟩ seconds of the movie to be skipped. For example, start=10s,duration=5s will show seconds 10 to 15 of the movie, when you play the movie.

- width= ⟨TEXdimension⟩ works like the height option, only for the width of the poster box.

Example: The following example creates a background sound for the slide.

```
\movie[autostart]{}{test.wav}
```

Example: A movie with two extra buttons for showing different parts of the movie.

```
\movie[label=cells,width=4cm,height=3cm,poster,showcontrols,duration=5s]{}{cells.avi}

\hyperlinkmovie[start=5s,duration=7s]{cells}{\beamerbutton{Show the middle stage}}

\hyperlinkmovie[start=12s,duration=5s]{cells}{\beamerbutton{Show the late stage}}
```

A movie can serve as the destination of a special kind of hyperlink, namely a hyperlink introduced using the following command:

\hyperlinkmovie[⟨options⟩]{⟨movie label⟩}{⟨text⟩}

Causes the ⟨text⟩ to become a movie hyperlink. When you click on the ⟨text⟩, the movie with the label ⟨movie label⟩ will start to play (or stop or pause or resume, depending on the ⟨options⟩). The movie must be on the same page as the hyperlink.

The following ⟨options⟩ may be given, many of which are the same as for the \movie command; if a different option is given for the link than for the movie itself, the option for the link takes precedence:

- duration= ⟨time⟩s. As for \movie, this causes the movie to be played only for the given number of seconds.
- loop and repeat. As for \movie, this causes the movie to loop.
- once. As for \movie, this causes the movie to played only once.
- palindrome. As for \movie, this causes the movie to be played forth and back.
- pause. Causes the playback of the movie to be paused, if the movie was currently playing. If not, nothing happens.
- play. Causes the movie to be played from whatever start position is specified. If the movie is already playing, it will be stopped and restarted at the starting position. This is the default.
- resume. Resumes playback of the movie, if it has previously been paused. If has not been paused, but not started or is already playing, nothing happens.
- showcontrols= ⟨true or false⟩. As for \movie, this causes a control bar to be shown or not shown during playback.
- start= ⟨time⟩s. As for \movie, this causes the given number of seconds to be skipped at the beginning of the movie if play is used to start the movie.
- stop. Causes the playback of the movie to be stopped.

14.1.2 Animations Created by Showing Slides in Rapid Succession

You can create an animation in a portable way by using the overlay commands of the beamer package to create a series of slides that, when shown in rapid succession, present an animation. This is a flexible approach, but such animations will typically be rather static since it will take some time to advance from one slide to the next. This approach is mostly useful for animations where you want to explain each picture of the animation. When you advance slides by hand, that is, by pressing a forward button, it typically takes at least a second for the next slide to show.

More lively animations can be created by relying on a capability of the viewer program. Some programs support showing slides only for a certain number of seconds during a presentation (for the Acrobat Reader this works only in full-screen mode). By setting the number of seconds to zero, you can create a rapid succession of slides.

To facilitate the creation of animations using this feature, the following commands can be used\animate and \animatevalue.

\animate<*overlay specification*>

The slides specified by *overlay specification* will be shown as quickly as possible.

Example:

```
\begin{frame}
  \frametitle{A Five Slide Animation}
  \animate<2-4>

  The first slide is shown normally. When the second slide is shown
  (presumably after pressing a forward key), the second, third, and
  fourth slides ``flash by.'' At the end, the content of the fifth
  slide is shown.

  ... code for creating an animation with five slides ...
\end{frame}
```

article This command is ignored in article mode.

\animatevalue<*start slidei-end slidei*>{*namei*}{*start valuei*}{*end valuei*}

The *namei* must be the name of a counter or a dimension. It will be varied between two values. For the slides in the specified range, the counter or dimension is set to an interpolated value that depends on the current slide number. On slides before the *start slidei*, the counter or dimension is set to *start valuei*; on the slides after the *end slidei* it is set to *end valuei*.

Example:

```
\newcount\opaqueness
\begin{frame}
  \animate<2-10>
  \animatevalue<1-10>{\opaqueness}{100}{0}
  \begin{colormixin}{\the\opaqueness!averagebackgroundcolor}
    \frametitle{Fadeout Frame}

    This text (and all other frame content) will fade out when the
    second slide is shown. This even works with
    {\color{green!90!black}colored} \alert{text}.
  \end{colormixin}
\end{frame}

\newcount\opaqueness
\newdimen\offset
\begin{frame}
  \frametitle{Flying Theorems (You Really Shouldn't!)}

  \animate<2-14>

  \animatevalue<1-15>{\opaqueness}{100}{0}
  \animatevalue<1-15>{\offset}{0cm}{-5cm}
  \begin{colormixin}{\the\opaqueness!averagebackgroundcolor}
  \hskip\offset
    \begin{minipage}{\textwidth}
      \begin{theorem}
        This theorem flies out.
      \end{theorem}
    \end{minipage}
  \end{colormixin}

  \animatevalue<1-15>{\opaqueness}{0}{100}
  \animatevalue<1-15>{\offset}{-5cm}{0cm}
```

```
\begin{colormixin}{\the\opaqueness!averagebackgroundcolor}
\hskip\offset
  \begin{minipage}{\textwidth}
    \begin{theorem}
      This theorem flies in.
    \end{theorem}
  \end{minipage}
\end{colormixin}
\end{frame}
```

article This command is ignored in article mode.

If your animation graphics reside in individual external graphic files, you might also consider using the \multiinclude command, which is explained in Section 14.1.3, together with \animate. For example, you might create an animation like this, assuming you have created graphic files named animation.1 through to animation.10:

```
\begin{frame}
  \animate<2-9>
  \multiinclude[start=1]{animation}
\end{frame}
```

14.1.3 Including External Animations Residing in Multiple Image Files

Some animations reside in external files in the following way: For each stage of the animation there is an image file containing an image for this stage. You can include such a series of images conveniently by using the style mpmulti.sty from the ppower4 package. This style, written by Klaus Guntermann, introduces a command called \multiinclude that takes the base name of a graphic file like mygraphic and will then search for files called mygraphic.0, mygraphic.1, and so on, till no more files are found. It will then include these graphics files using the \includegraphics command, but will put these graphics on top of each other. Furthermore, and this is the important part, it inserts a \pause command after each graphic. This command is defined in the ppower4 package and has the same effect as the \pause command of beamer. For this reason, both ppower4 and also beamer will first display the basic graphic and will then additionally show the next graphic on each slide.

If you try to use mpmulti.sty directly, you will run into the problem that it includes a file called pause.sty, which is part of the ppower4 package.

You might also consider using the style xmpmulti.sty that comes with beamer. This file is mainly identical to mpmulti, except for two differences: First, it does not include pause.sty, a style that conceptually clashes with beamer, although beamer contains a workaround that sidesteps the problem. Second, it extends the \multiinclude command by allowing a special default overlay specification to be given. The effect of this is explained below.

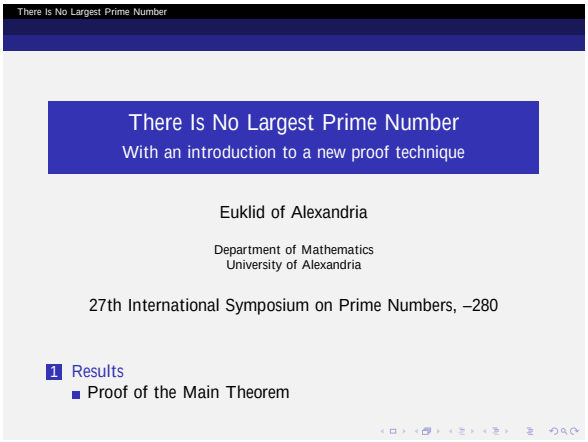
\usepackage{xmpmulti}

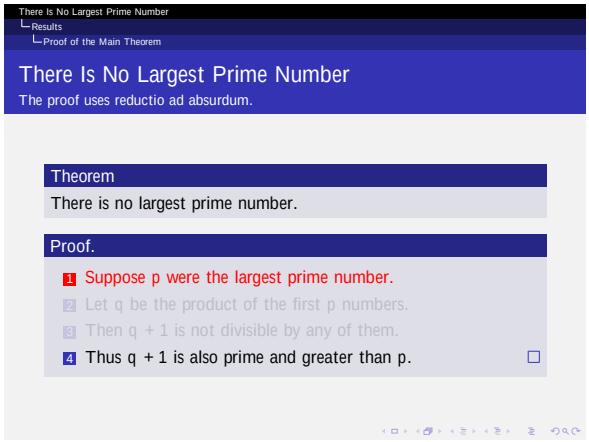
De nes the command \multiinclude. The code of this package is due to Klaus Guntermann with some additions of mine. It can be used together with beamer and with ppower4, i. e., it can be used as a replacement for mpmulti if the pause package is also included in a ppower4-presentation.

\multiinclude[<*hdefault overlay speci cationi*>][*hoptionsi*]{*hbase file namei*}

Except for the possibility of specifying a *hdefault overlay speci cationi*, this command is identical to the \multiinclude command from the ppower4 package.

If no overlay speci cation is given, the command will search for files called *hbase file namei*.*hnumberi* for increasing numbers *hnumberi*, starting with zero. As long as it finds these files, it issues an \includegraphics command on them. The files following the first one are put on top of the first one. Between any two invocations of \includegraphics, a \pause command is inserted. You can modify this behavior in different ways by given suitable *hoptionsi*, see below.



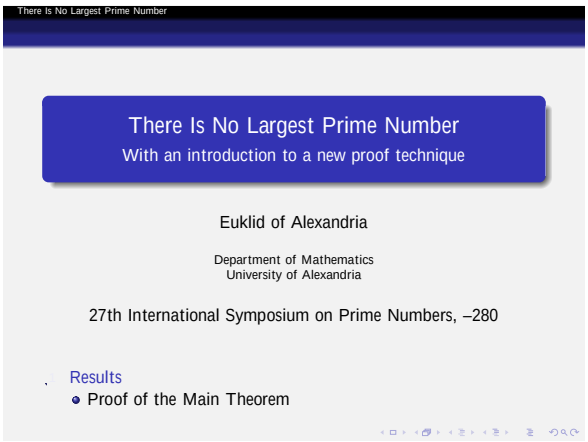


A dominant theme with a tree-like navigation at the top. The rectangular elements mirror the rectangular navigation at the top. The theme can be made less dominant by using a di erent color theme.

Antibes is a town in the south of France. It hosted stacs 2002.

\usetheme{JuanLesPins}

Example:

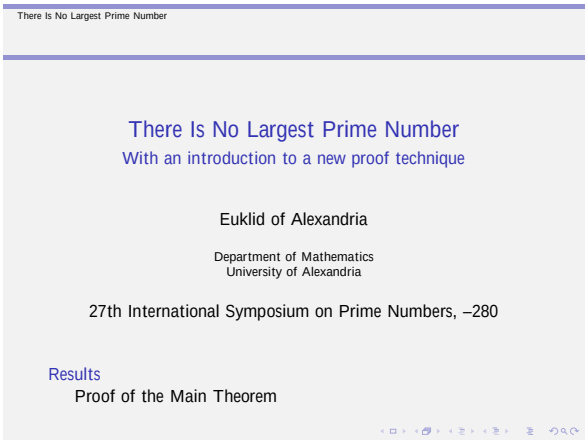


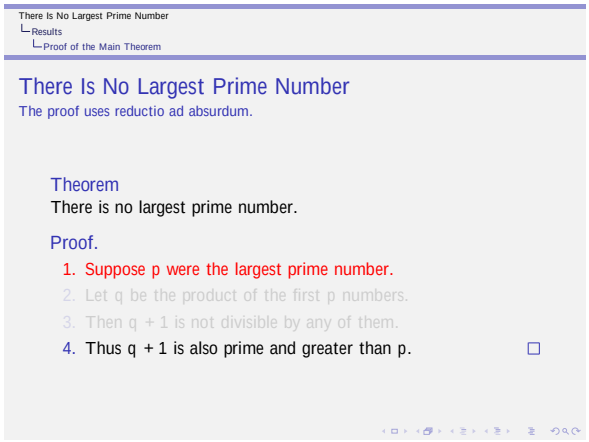
A variation on the Antibes theme that has a much smoother appearance. It can be made less dominant by choosing a di erent color theme.

Juan Les Pins is a cozy village near Antibes. It hosted stacs 2002.

\usetheme{Montpellier}

Example:





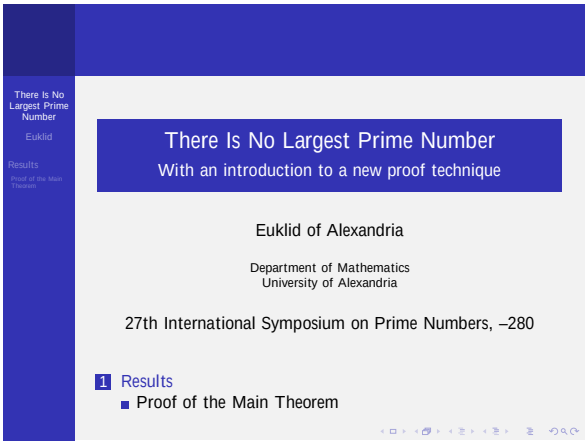
A sober theme giving basic navigational hints. The headline can be made more dominant by using a di erent color theme.

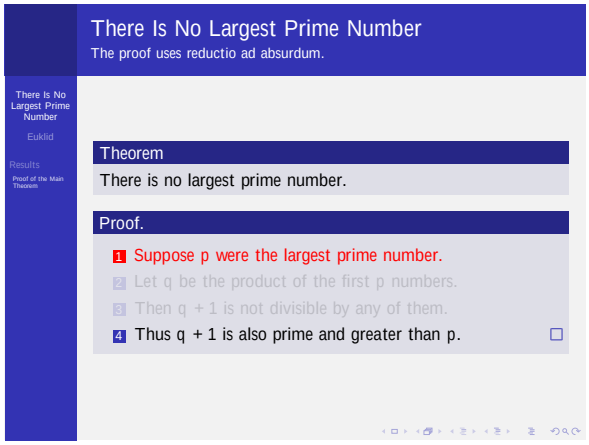
Montpellier is in the south of France. It hosted stacs 2004.

15.4 Presentation Themes with a Table of Contents Sidebar

\usetheme[hoptionsi]{Berkeley}

Example:





A dominant theme. If the navigation bar is on the left, it dominates since it is seen rst. The height of the frame title is xed to two and a half lines, thus you should be careful with overly long titles. A logo will be put in the corner area. Rectangular areas dominate the layout. The theme can be made less dominant by using a di erent color theme.

By default, the current entry of the table of contents in the sidebar will be highlighted by using a more vibrant color. A good alternative is to highlight the current entry by using a di erent color for the background of the current point. The color theme sidebartab installs the appropriate colors, so you just have to say

\usecolortheme{sidebartab}

This color theme works with all themes that show a table of contents in the sidebar.

This theme is useful for long talks like lectures that require a table of contents to be visible all the time.

The following hoptionsi may be given:

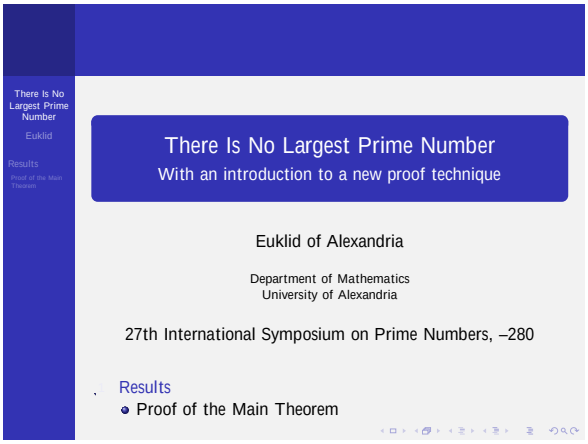
- hideallsubsections causes only sections to be shown in the sidebar. This is useful, if you need to save space.

- hideothersubsections causes only the subsections of the current section to be shown. This is useful, if you need to save space.
- left puts the sidebar on the left (default).
- right puts the sidebar on the right.
- width=⟨dimension⟩ sets the width of the sidebar. If set to zero, no sidebar is created.

Berkeley is on the western coast of the USA, near San Francisco. Till visited Berkeley for a year in 2004.

\usetheme[⟨options⟩]{PaloAlto}

Example:

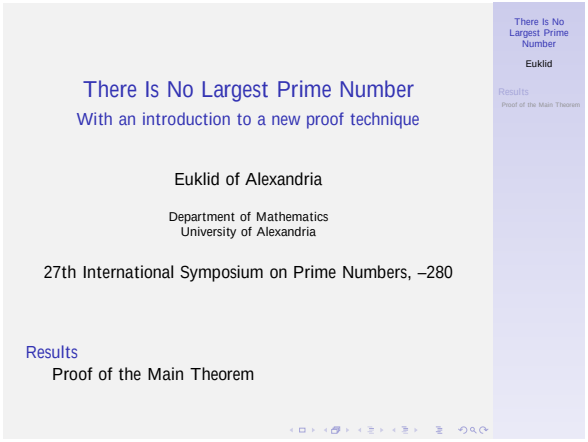


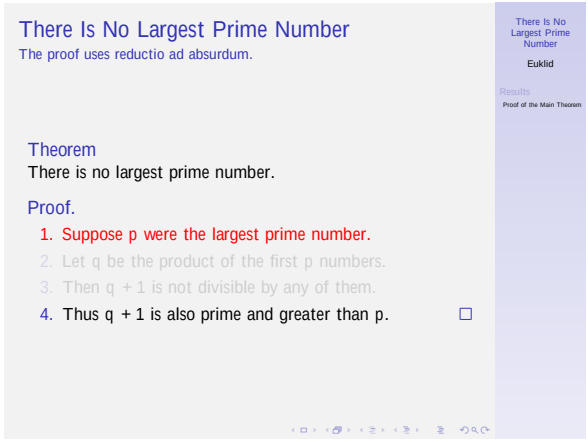
A variation on the Berkeley theme with less dominance of rectangular areas. The same ⟨options⟩ as for the Berkeley theme can be given.

Palo Alto is also near San Francisco. It hosted the Bay Area Theory Workshop 2004.

\usetheme[⟨options⟩]{Goettingen}

Example:



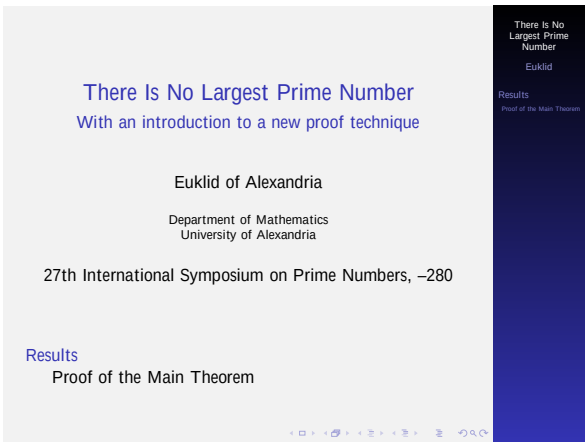


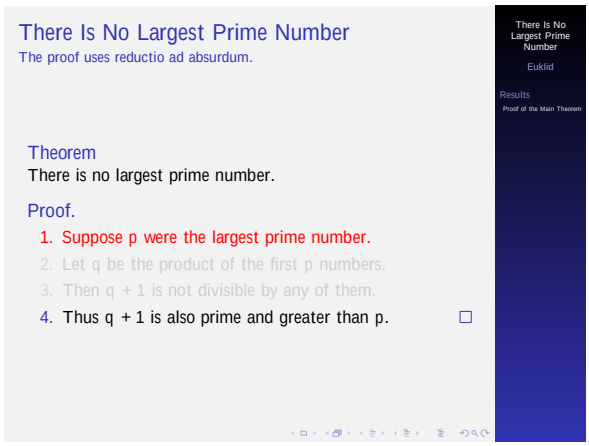
A relatively sober theme useful for a longer talk that demands a sidebar with a full table of contents. The same ⟨options⟩ as for the Berkeley theme can be given.

Göttingen is a town in Germany. It hosted the 42nd Theorietag.

\usetheme[⟨options⟩]{Marburg}

Example:



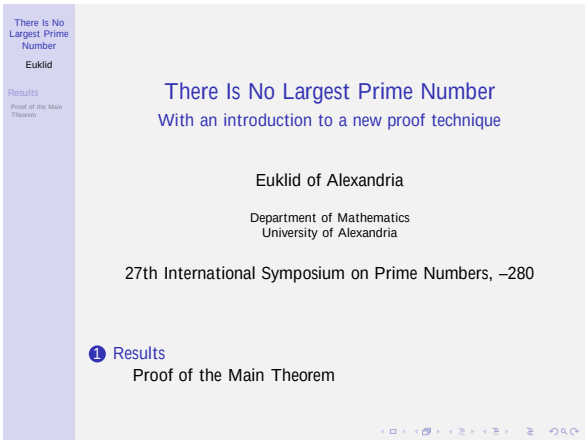


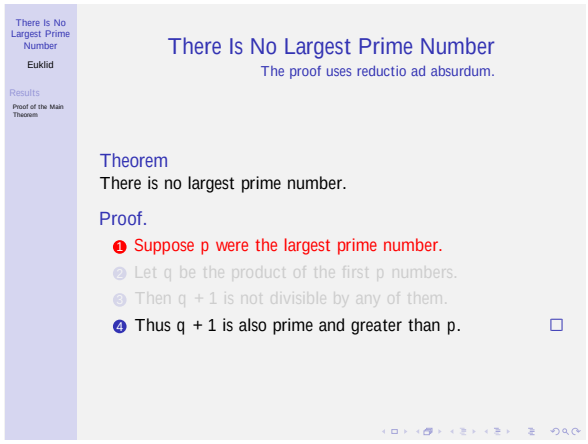
A very dominant variation of the Goettingen theme. The same hoptionsi may be given.

Marburg is a town in Germany. It hosted the 46th Theorietag.

\usetheme[hoptionsi]{Hannover}

Example:





In this theme, the sidebar on the left is balanced by right- ushed frame titles.

The following hoptionsi may be given:

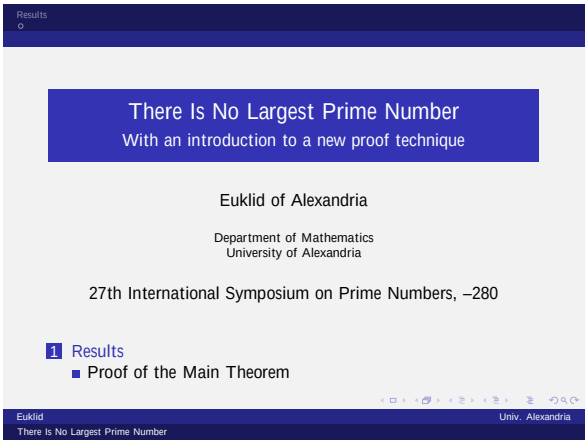
- hideallsubsections causes only sections to be shown in the sidebar. This is useful, if you need to save space.
- hideothersubsections causes only the subsections of the current section to be shown. This is useful, if you need to save space.
- width= hdimensioni sets the width of the sidebar.

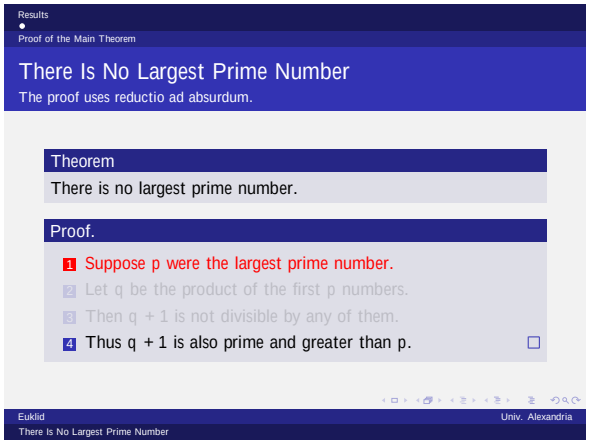
Hannover is a town in Germany. It hosted the 48th Theorietag.

15.5 Presentation Themes with a Mini Frame Navigation

\usetheme[hoptionsi]{Berlin}

Example:





A dominant theme with strong colors and dominating rectangular areas. The head- and footlines give lots of information and leave little space for the actual slide contents. This theme is useful for conferences where the audience is not likely to know the title of the talk or who is presenting it. The theme can be made less dominant by using a di erent color theme.

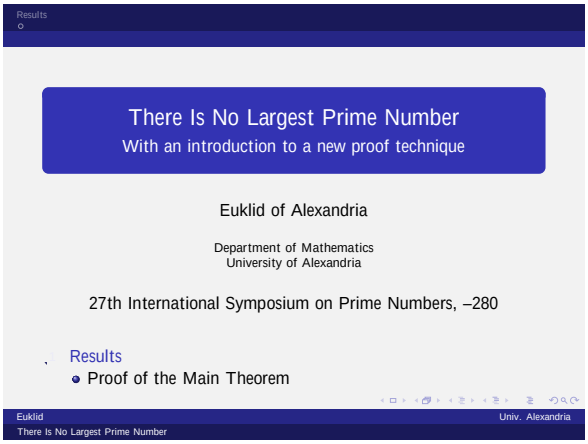
The following hoptionsi may be given:

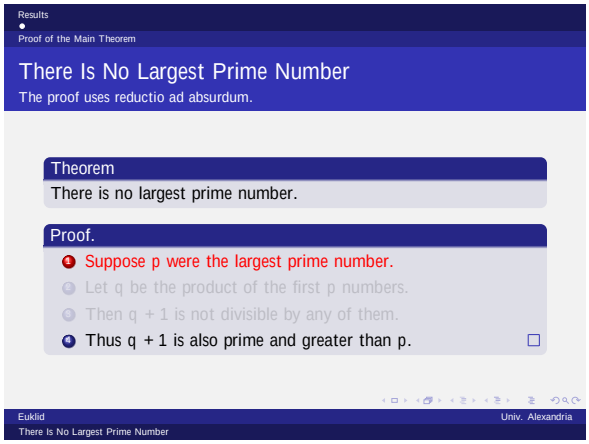
- compress causes the mini frames in the headline to use only a single line. This is useful for saving space.

Berlin is the capital of Germany.

\usetheme[hoptionsi]{Ilmenau}

Example:



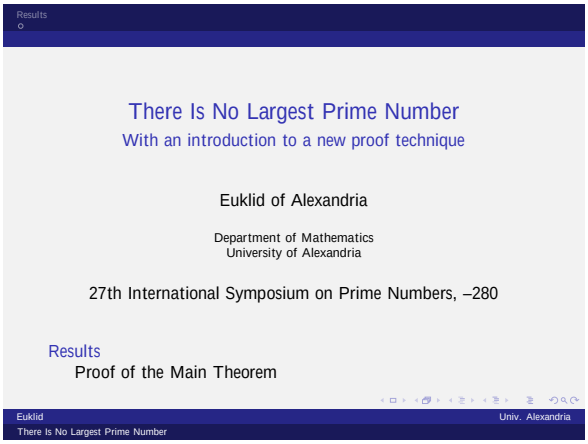


A variation on the Berlin theme. The samehoptionsi may be given.

Ilmenau is a town in Germany. It hosted the 40th Theorietag.

\usetheme{Dresden}

Example:

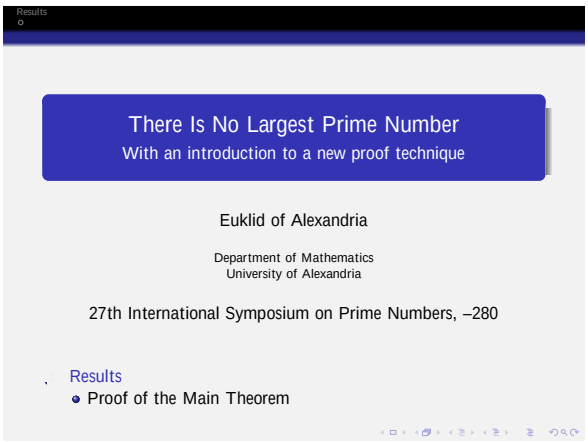


A variation on the Berlin theme with a strong separation into navigational stu at the top/bottom and a sober main text. The samehoptionsi may be given.

Dresden is a town in Germany. It hosted STACS 2001.

\usetheme{Darmstadt}

Example:

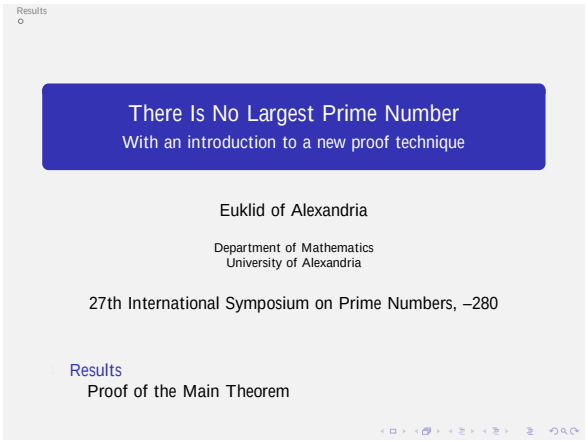


A theme with a strong separation into a navigational upper part and an informational main part. By using a di erent color theme, this separation can be lessened.

Darmstadt is a town in Germany.

\usetheme{Frankfurt}

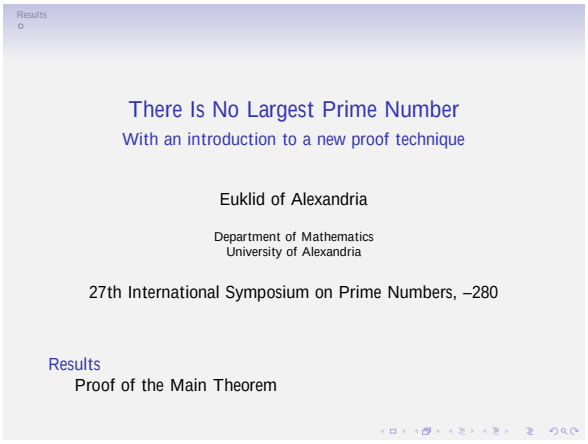
Example:



A variation on the Darmstadt theme that is slightly less cluttered by leaving out the subsection information.
Frankfurt is a town in Germany.

\usetheme{Singapore}

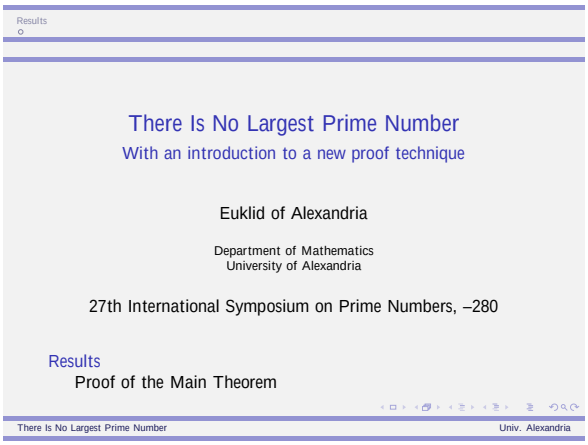
Example:

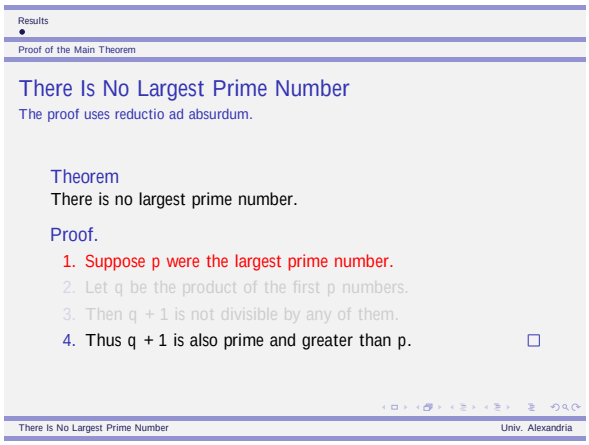


A not-too-sober theme with navigation that does not dominate.
Singapore is located in south-eastern Asia. It hosted cocoon 2002.

\usetheme{Szeged}

Example:





A sober theme with a strong dominance of horizontal lines.

Szeged is on the south border of Hungary. It hosted dlt 2003.

15.6 Presentation Themes with Section and Subsection Tables

\usetheme{Copenhagen}

Example:

A not-quite-too-dominant theme. This theme gives compressed information about the current section and subsection at the top and about the title and the author at the bottom. No shadows are used, giving the presentation a at look. The theme can be made less dominant by using a di erent color theme.

Copenhagen is the capital of Denmark. It is connected to Malmö by the Øresund bridge.

\usetheme{Luebeck}

Example:

A variation on the Copenhagentheme.

Lübeck is a town in northern Germany. It hosted the 41st Theorietag.

\usetheme{Malmoe}

Example:

16.2 Outer Themes

An outer theme dictates (roughly) the overall layout of frames. It specifies where any navigational elements should go (like a mini table of contents or navigational mini frames) and what they should look like. Typically, an outer theme specifies how the following elements are rendered:

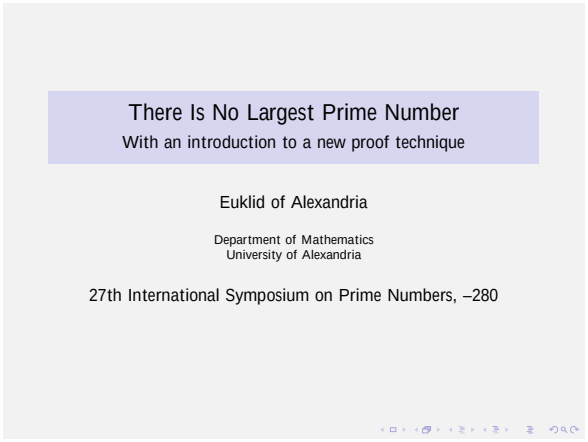
- The head- and footline.
- The sidebars.
- The logo.
- The frame title.

An outer theme will not specify how things like itemize environments should be rendered that is the job of an inner theme.

In the following examples the color theme seahorse is used. Since the default color theme leaves most backgrounds empty, most of the outer themes look too unstructured with the default color theme.

\useoutertheme{default}

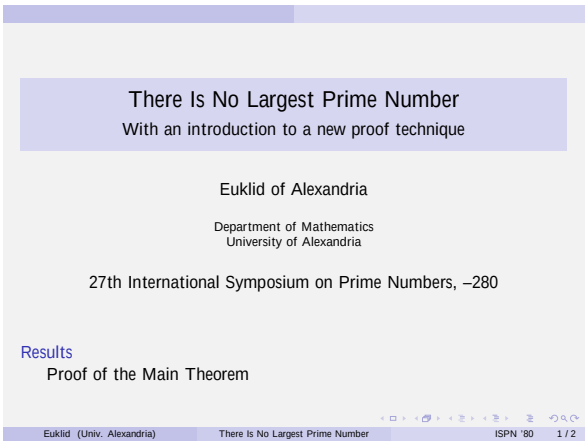
Example:

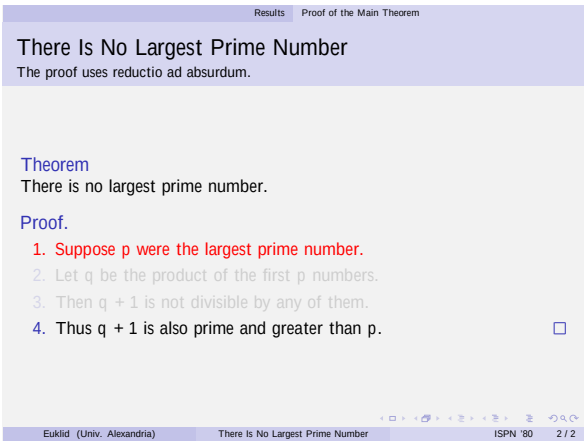


The default layout theme is the most sober and minimalistic theme around. It will push left the frame title and it will not install any head- or footlines. However, even this theme honors the background color specified for the frame title. If a color is specified, a bar occupying the whole page width is put behind the frame title. A background color of the frame subtitle is ignored.

\useoutertheme{infolines}

Example:



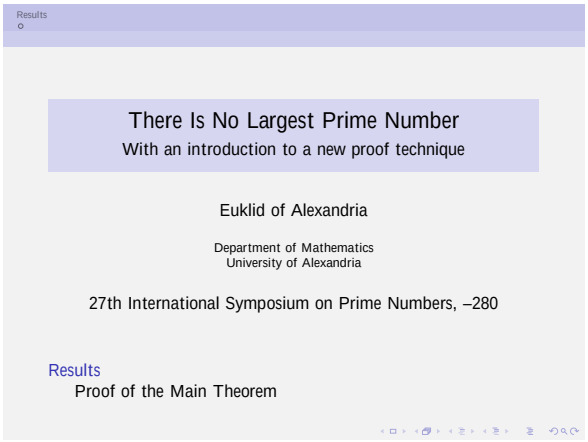


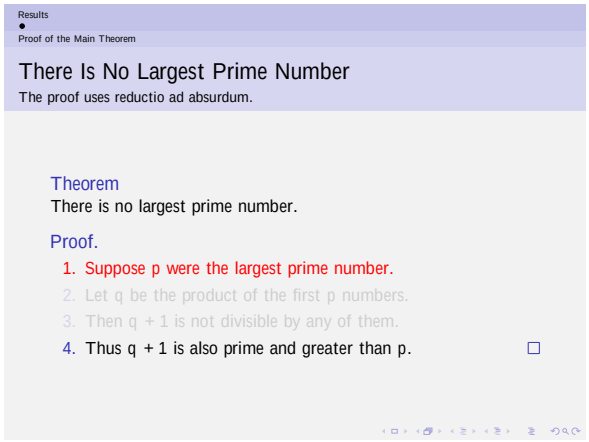
This theme installs a headline showing the current section and the current subsection. It installs a footline showing the author's name, the institution, the presentation's title, the current date, and a frame count. This theme uses only little space.

The colors used in the headline and footline are drawn from palette primary , palette secondary , and palette tertiary (see Section 17 for details on how to change these).

\useoutertheme[hoptionsi]{miniframes}

Example:





This theme installs a headline in which a horizontal navigational bar is shown. This bar contains one entry for each section of the presentation. Below each section entry, small circles are shown that represent the di erent frames in the section. The frames are arranged subsection-wise, that is, there is a line of frames for each subsection. If the class optioncompress is given, the frames will instead be arranged in a single row for each section. The navigation bars draws its color fromsection in head/foot .

Below the navigation bar, a line is put showing the title of the current subsection. The color is drawn from subsection in head/foot .

At the bottom, two lines are put that contain information such as the author's name, the institution, or the paper's title. What is shown exactly is in uenced by the hoptionsi given. The colors are drawn from the appropriate beamer -colors like author in head/foot .

At the top and bottom of both the head- and footline and between the navigation bar and the subsection name, separation lines are drawnif the background color of separation line is set. This separation line will have a height of 3pt. You can get even more ne-grained control over the colors of the separation lines by setting appropriate colors like lower separation line head .

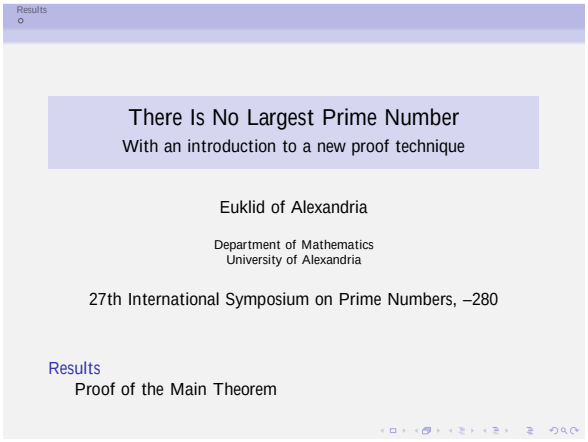
Note: Make sure the document is organized in the section-subsection-frame structure when using miniframes and smoothbars theme. Any frame without a \section or \subsection will bring unpredictable e ects in the navigation bar.

The following hoptionsi can be given:

- footline=empty suppresses the footline (default).
- footline=authorinstitute shows the author's name and the institute in the footline.
- footline=authortitle shows the author's name and the title in the footline.
- footline=institutetitle shows the institute and the title in the footline.
- footline=authorinstitutetitle shows the author's name, the institute, and the title in the footline.
- subsection= htrue or falsei shows or suppresses line showing the subsection in the headline. It is shown by default. If the document does not use subsections, this option should be setfalse .

\useoutertheme[hoptionsi]{smoothbars}

Example:



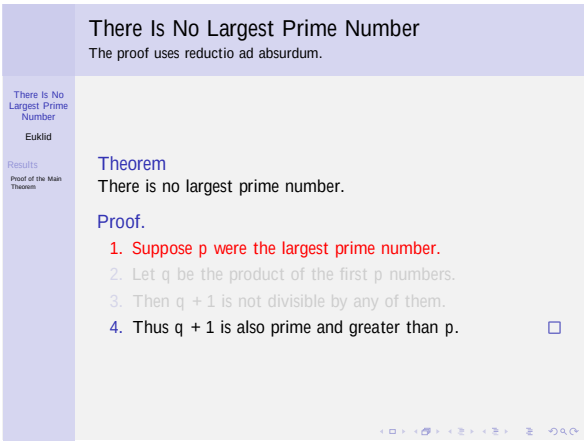
This theme behaves very much like the miniframes theme, at least with respect to the headline. The only differences are that smooth transitions are installed between the background colors of the navigation bar, the (optional) bar for the subsection name, and the background of the frame title. No footline is created. You can get the footlines of the miniframes theme by first loading that theme and then loading the smoothbars theme.

The following hoptionsi can be given:

- subsection= htrue or falsei shows or suppresses line showing the subsection in the headline. It is shown by default.

\useoutertheme[hoptionsi]{sidebar}

Example:



In this layout, a sidebar is shown that contains a small table of contents with the current section, subsection, or subsubsection highlighted. The frame title is vertically centered in a rectangular area at the top that always occupies the same amount of space in all frames. Finally, the logo is shown in the corner resulting from the sidebar and the frame title rectangle.

There are several ways of modifying the layout using the hoptionsi. If you set the width of the sidebar to 0pt, it is not shown, giving you a layout in which the frame title does not wobble since it always occupies the same amount of space on all slides. Conversely, if you set the height of the frame title rectangle to 0pt, the rectangular area is not used and the frame title is inserted normally (occupying as much space as needed on each slide).

The background color of the sidebar is taken from sidebar , the background color of the frame title from frametitle , and the background color of the logo corner from logo .

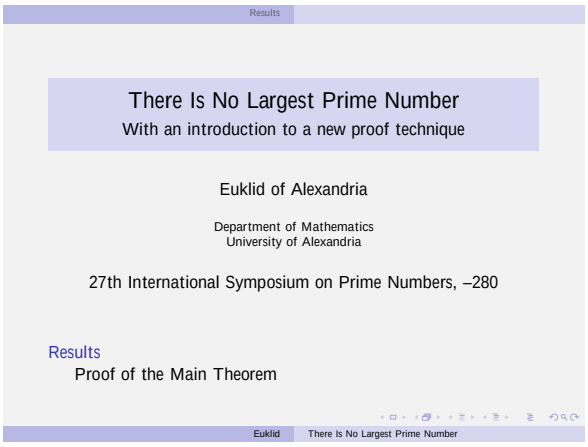
The colors of the entries in the table of contents are drawn from the beamer -color section in sidebar and section in sidebar current as well as the corresponding beamer -colors for subsections. If an entry does not t on a single line it is automatically linebroken.

The following hoptionsi may be given:

- height= hdimensioni speci es the height of the frame title rectangle. If it is set to 0pt, no frame title rectangle is created. Instead, the frame title is inserted normally into the frame. The default is 2.5 base line heights of the frame title font. Thus, there is about enough space for a two-line frame title plus a one-line subtitle.
- hideothersubsections causes all subsections except those of the current section to be suppressed in the table of contents. This is useful if you have lots of subsections.
- hideallsubsections causes all subsections to be suppressed in the table of contents.
- left puts the sidebar on the left side. Note that in a left-to-right reading culture this is the side people look rst. Note also that this table of contents is usually not the most important part of the frame, so you do not necessarily want people to look at it rst. Nevertheless, it is the default.
- right puts the sidebar of the right side.
- width= hdimensioni speci es the width of the sidebar. If it is set to 0pt, it is completely suppressed. The default is 2.5 base line heights of the frame title font.

\useoutertheme{split}

Example:



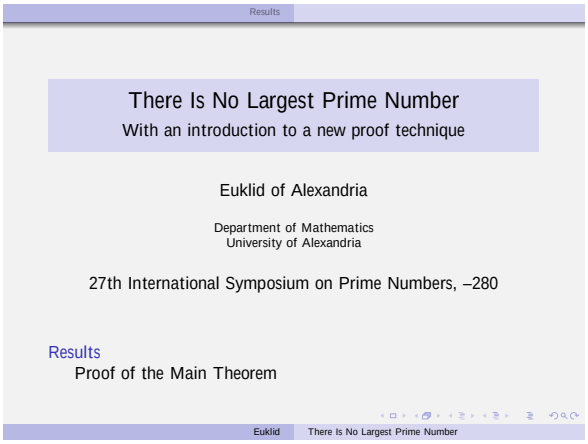
This theme installs a headline in which, on the left, the sections of the talk are shown and, on the right, the subsections of the current section. If the class option compress has been given, the sections and subsections will be put in one line; normally there is one line per section or subsection.

The footline shows the author on the left and the talk's title on the right.

The colors are taken from palette primary and palette quaternary .

\useoutertheme{shadow}

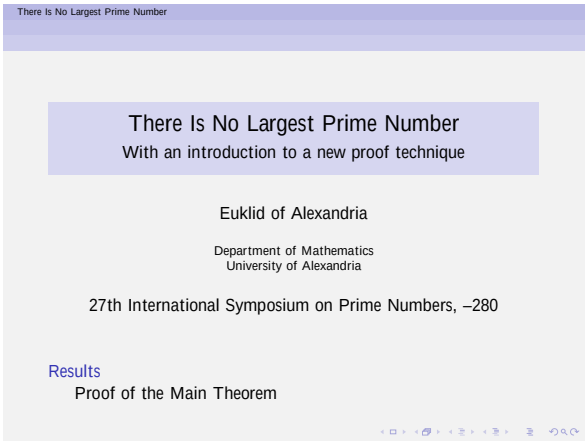
Example:

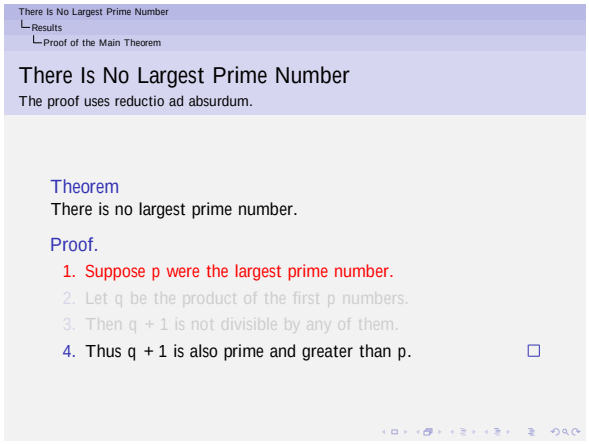


This layout theme extends the split theme by putting a horizontal shading behind the frame title and adding a little shadow at the bottom of the headline.

\useoutertheme[⟨options⟩]{tree}

Example:





In this layout, the headline contains three lines that show the title of the current talk, the current section in this talk, and the current subsection in the section. The colors are drawn from title in head/foot, section in head/foot, and subsection in head/foot.

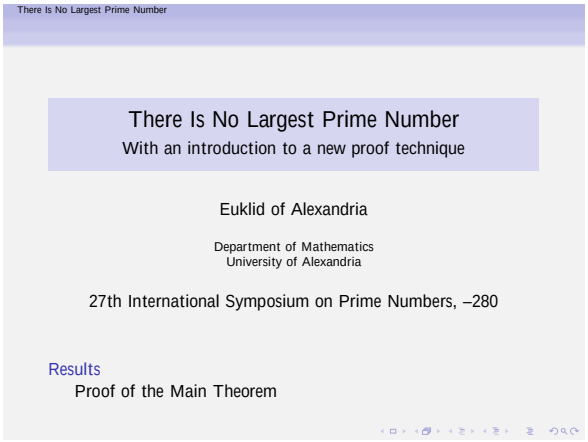
In addition, separation lines of height 3pt are shown above and below the three lines if the background of separation line is set. More fine-grained control of the colors of these lines can be gained by setting upper separation line head and lower separation line head.

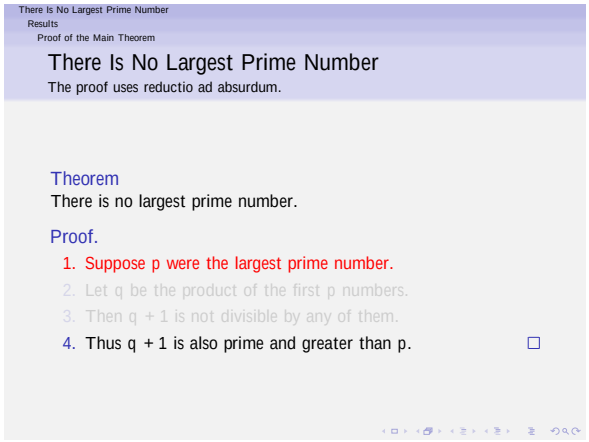
The following ⟨options⟩ may be given:

- hooks causes little hooks to be drawn in front of the section and subsection entries. These are supposed to increase the tree-like appearance.

\useoutertheme{smoothtree}

Example:





This layout is similar to the tree layout. The main difference is that the background colors change smoothly.

16.3 Changing the Templates Used for Different Elements of a Presentation

This section explains how beamer 's template management works.

16.3.1 Overview of Beamer's Template Management

If you only wish to modify the appearance of a single or few elements, you do not need to create a whole new inner or outer theme. Instead, you can modify the appropriate template.

A template specifies how an element of a presentation is typeset. For example, the frametitle template dictates where the frame title is put, which font is used, and so on.

As the name suggests, you specify a template by writing the exact LaTeX code you would also use when typesetting a single frame title by hand. Only, instead of the actual title, you use the command \insertframetitle.

Example: Suppose we would like to have the frame title typeset in red, centered, and boldface. If we were to typeset a single frame title by hand, it might be done like this:

```
\begin{frame}
  \begin{centering}
    \color{red}
    \textbf{The Title of This Frame.}
    \par
  \end{centering}

  Blah, blah.
\end{frame}
```

In order to typeset the frame title in this way on all slides, in the simplest case we can change the frame title template as follows:

```
\setbeamertemplate{frametitle}
{
  \begin{centering}
    \color{red}
    \textbf{\insertframetitle}
    \par
  \end{centering}
}
```

We can then use the following code to get the desired effect:

```
\begin{frame}
  \frametitle{The Title of This Frame.}
```

```
  Blah, blah.
\end{frame}
```

When rendering the frame, beamer will use the code of the frame title template to typeset the frame title and it will replace every occurrence of \insertframetitle by the current frame title.

We can take this example a step further. It would be nicer if we did not have to hardwire the color of the frametitle, but if this color could be specified independently of the code for the template. This way, a color theme could change this color. Since this is a problem that is common to most templates, beamer will automatically setup the beamer-color frametitle when the template frametitle is used. Thus, we can remove the \color{red} command if we set the beamer-color frametitle to red at some point.

```
\setbeamercolor{frametitle}{fg=red}
\setbeamertemplate{frametitle}
{
  \begin{centering}
    \textbf{\insertframetitle}
    \par
  \end{centering}
}
```

Next, we can also make the font themable. Just like the color, the beamer-font frametitle is installed before the frametitle template is typeset. Thus, we should rewrite the code as follows:

```
\setbeamercolor{frametitle}{fg=red}
\setbeamerfont{frametitle}{series=\bfseries}
\setbeamertemplate{frametitle}
{
  \begin{centering}
    \insertframetitle\par
  \end{centering}
}
```

Users, themes, or whoever can now easily change the color or font of the frametitle without having to mess with the code used to typeset it.

article In article mode, most of the template mechanism is switched off and has no effect. However, a few templates are also available. If this is the case, it is specially indicated.

Here are a few hints that might be helpful when you wish to set a template:

- Usually, you might wish to copy code from an existing template. The code often takes care of some things that you may not yet have thought about. The default inner and outer themes might be useful starting points. Also, the file beamerbaseauxtemplates.sty contains interesting auxiliary templates.
- When copying code from another template and when inserting this code in the preamble of your document (not in another style file), you may have to switch on the at-character (@). To do so, add the command \makeatletter before the \setbeamertemplate command and the command \makeatother afterward.
- Most templates having to do with the frame components (headlines, sidebars, etc.) can only be changed in the preamble. Other templates can be changed during the document.
- The height of the headline and footline templates is calculated automatically. This is done by typesetting the templates and then having a look at their heights. This recalculation is done right at the beginning of the document, after all packages have been loaded and even after these have executed their \AtBeginDocument initialization.
- Getting the boxes right inside any template is often a bit of a hassle. You may wish to consult the TEX book for the glorious details on Making Boxes. If your headline is simple, you might also try putting everything into a pgfpicture environment, which makes the placement easier.

16.3.2 Using Beamer's Templates

As a user of the beamer class you typically do not use or invoke templates yourself, directly. For example, the frame title template is automatically invoked by beamer somewhere deep inside the frame typesetting process. The same is true of most other templates. However, if, for whatever reason, you wish to invoke a template yourself, you can use the following command.

\usebeamertemplate***{⟨element name⟩}

If none of the stars is given, the text of the ⟨element name⟩ is directly inserted at the current position. This text should previously have been specified using the \setbeamertemplate command. No text is inserted if this command has not been called before.

Example:

```
\setbeamertemplate{my template}{correct}
...
Your answer is \usebeamertemplate{my template}.
```

If you add one star, three things happen. First, the template is put inside a TEX-group, thereby limiting most side effects of commands used inside the template. Second, inside this group the beamer -color named ⟨element name⟩ is used and the foreground color is selected. Third, the beamer -font ⟨element name⟩ is also used. This one-starred version is usually the best version to use.

If you add a second star, nearly the same happens as with only one star. However, in addition, the color is used with the command \setbeamercolor* . This causes the colors to be reset to the normal text color if no special foreground or background is specified by the beamer -color ⟨element name⟩. Thus, in this twice-starred version, the color used for the template is guaranteed to be independent of the color that was currently in use when the template is used.

Finally, adding a third star will also cause a star to be added to the \setbeamerfont* command. This causes the font used for the template also to be reset to normal text, unless the beamer -font ⟨element name⟩ specifies things differently. This three-star version is the most protected version available.

\ifbeamertemplateempty{⟨beamer template name⟩}{⟨executed if empty⟩}{⟨executed otherwise⟩}

This command checks whether a template is defined and set to a non-empty text. If the text is empty or the template is not defined at all, ⟨executed if empty⟩ is executed. Otherwise, ⟨executed otherwise⟩ is executed.

\expandbeamertemplate{⟨beamer template name⟩}

This command does the same as \usebeamertemplate{⟨beamer template name⟩}. The difference is that this command performs a direct expansion and does not scan for a star. This is important inside, for example, an \edef . If you don't know the difference between \def and \edef , you won't need this command.

16.3.3 Setting Beamer's Templates

To set a beamer -template, you can use the following command:

\setbeamertemplate{⟨element name⟩}[⟨predefined option⟩]⟨args⟩

In the simplest case, if no ⟨predefined option⟩ is given, the ⟨args⟩ must be a single argument and the text of the template ⟨element name⟩ is setup to be this text. Upon later invocation of the template by the command \usebeamertemplate this text is used.

Example:

```
\setbeamertemplate{answer}{correct}
...
Your answer is \usebeamertemplate*{answer}.
```

If you specify a ⟨predefined option⟩, this command behaves slightly differently. In this case, someone has used the command \defbeamertemplate to predefine a template for you. By giving the name of this

prede ned template as the optional parameter ⟨prede ned option⟩, you cause the template ⟨element name⟩ to be set to this template.

Example: \setbeamertemplate{bibliography item}[book] causes the bibliography items to become little book icons. This command causes a subsequent call of \usebeamertemplate{bibliography item} to insert the prede ned code for inserting a book.

Some prede ned template options take parameters themselves. In such a case, the parameters are given as ⟨args⟩.

Example: The ⟨prede ned option⟩ grid for the template background takes an optional argument:

```
\setbeamertemplate{background}[grid][step=1cm]
```

In the example, the second argument in square brackets is the optional argument.

In the descriptions of elements, if there are possible ⟨prede ned option⟩, the description shows how the ⟨prede ned option⟩ can be used together with its arguments, but the \setbeamertemplate{xxxx} is omitted. Thus, the above example would be documented in the description of the background element like this:

^ [grid][⟨step options⟩] causes a light grid to be ...

\addtobeamertemplate{⟨element name⟩}{⟨pre-text⟩}{⟨post-text⟩}

This command adds the ⟨pre-text⟩ before the text that is currently installed as the template ⟨element name⟩ and the ⟨post-text⟩ after it. This allows you a limited form of modi cation of existing templates.

Example: The following commands have the same e ect:

```
\setbeamertemplate{my template}{Hello world!}

\setbeamertemplate{my template}{world}
\addtobeamertemplate{my template}{Hello }{!}
```

If a new template is installed, any additions will be deleted. On the other hand, you can repeatedly use this command to add multiple things.

\defbeamertemplate<⟨mode speci cation⟩>*{⟨element name⟩}{⟨prede ned option⟩}
[⟨argument number⟩][⟨default optional argument⟩]{⟨prede ned text⟩}
[action]{⟨action command⟩}

This command installs a prede ned option for the template ⟨element name⟩. Once this command has been used, users can access the prede ned template using the \setbeamertemplate command.

Example: \defbeamertemplate{itemize item}{double arrow}{$\Rightarrow$}

After the above command has been invoked, the following two commands will have the same e ect:

```
\setbeamertemplate{itemize item}{$\Rightarrow$}
\setbeamertemplate{itemize item}[double arrow]
```

Sometimes, a prede ned template needs to get an argument when it is installed. Suppose, for example, we want to de ne a prede ned template that draws a square as the itemize item and we want to make this size of this square con gurable. In this case, we can specify the ⟨argument number⟩ of the prede ned option the same way one does for the \newcommand command:

```
\defbeamertemplate{itemize item}{square}[1]{\hrule width #1 height #1}

% The following have the same effect:
\setbeamertemplate{itemize item}[square]{3pt}
\setbeamertemplate{itemize item}{\hrule width 3pt height 3pt}
```

As for the \newcommand command, you can also specify a ⟨default optional argument⟩:

```
\defbeamertemplate{itemize item}{square}[1][1ex]{\hrule width #1 height #1}

% The following have the same effect:
```

```
\setbeamertemplate{itemize item}[square][3pt]
\setbeamertemplate{itemize item}{\hrule width 3pt height 3pt}

% So do the following:
\setbeamertemplate{itemize item}[square]
\setbeamertemplate{itemize item}{\hrule width 1ex height 1ex}
```

The starred version of the command installs the predefined template option, but then immediately calls \setbeamertemplate for this option. This is useful for the default templates. If there are any arguments necessary, these are set to \relax.

In certain cases, if a predefined template option is chosen, you do not only wish the template text to be installed, but certain extra actions must also be taken once. For example, a shading must be defined that should not be redefined every time the shading is used later on. To implement such actions, you can use the optional argument ⟨action⟩ following the keyword [action]. Thus, after the normal use of the \defbeamertemplate you add the text [action] and then any commands that should be executed once when the ⟨predefined option⟩ is selected by the \setbeamertemplate command.

Example:

```
\defbeamertemplate{background canvas}{my shading}[2]
{
  \pgfuseshading{myshading}% simple enough
}
[action]
{
  \pgfdeclareverticalshading{myshading}{\the\paperwidth}
  {color(0cm)=(#1); color(\the\paperheight)=(#2)}
}
...

\setbeamertemplate{background canvas}{myshading}{red!10}{blue!10}
% Defines the shading myshading right here. Subsequent calls to
% \usebeamertemplate{background canvas} will yield
% ``\pgfuseshading{myshading}''.
```

article Normally, this command has no effect in article mode. However, if a ⟨mode specification⟩ is given, this command is applied for the specified modes. Thus, this command behaves like the \def command, which also gets the implicit mode specification <presentation> if no other specification is given.

Example: \defbeamertemplate{my template}{default}{something} has no effect in article mode.

Example: \defbeamertemplate<article>{my template}{default}{something} has no effect in presentation modes, but has an effect in article mode.

Example: \defbeamertemplate<all>{my template}{default}{something} applies to all modes.

It is often useful to have access to the same template option via different names. For this, you can use the following command to create aliases:

\defbeamertemplatealias{⟨element name⟩}{⟨new predefined option name⟩}{⟨existing predefined option name⟩}

Causes the two predefined options to have the same effect.

There is no inheritance relation among templates as there is for colors and fonts. This is due to the fact the templates for one element seldom make sense for another. However, sometimes certain elements behave similarly and one would like a \setbeamertemplate to apply to a whole set of templates via inheritance. For example, one might want that \setbeamertemplate{items}[circle] causes all items to use the circle option, though the effects for the itemize item as opposed to the itemize subsubitem as opposed to enumerate item must be slightly different.

The beamer-template mechanism implements a simple form of inheritance via parent templates. In element descriptions, parent templates are indicated via a check mark in parentheses.

\defbeamertemplateparent{ hparent template namei}[hprede ned option namei]{ hchild template listi }
[hargument numberi][hdefault optional argumenti]{ harguments for childreni }

The e ect of this command is that whenever someone calls\setbeamertemplate{ hparent template namei }{ hargsi }, the command \setbeamertemplate{ hchild template namei }{ hargsi } is called for each hchild template namei in the hchild template listi .

The harguments for childreni come into play if the \setbeamertemplate command is called with a prede ned option name (not necessarily the same as thehprede ned option namei , we'll come to that). If \setbeamertemplate is called with some prede ned option name, the children are called with the harguments for childreni instead. Let's look at two examples:

Example: The following is the typical, simple usage:

```
\defbeamertemplateparent{itemize items}{itemize item,itemize subitem,itemize subsubitem}
{}

% The following command has the same effect as the three commands below:
\setbeamertemplate{itemize items}[circle]

\setbeamertemplate{itemize item}[circle] % actually, the ``empty'' argument is added
\setbeamertemplate{itemize subitem}[circle]
\setbeamertemplate{itemize subsubitem}[circle]
```

Example: In the following case, an argument is passed to the children:

```
\defbeamertemplateparent{sections/subsections in toc shaded}
{section in toc shaded,subsection in toc shaded}[1][20]
{[#1]}

% The following command has the same effect as the two commands below:
\setbeamertemplate{sections/subsections in toc shaded}[default][35]

\setbeamertemplate{section in toc shaded}[default][35]
\setbeamertemplate{subsection in toc shaded}[default][35]


% Again:
\setbeamertemplate{sections/subsections in toc shaded}[default]

\setbeamertemplate{section in toc shaded}[default][20]
\setbeamertemplate{subsection in toc shaded}[default][20]
```

In detail, the following happens: When \setbeamertemplate is encountered for a parent template,beamer rst checks whether a prede ned option follows. If not, a single argument is read and\setbeamertemplate is called for all children for this template. If there is a prede ned template option set, beamer evaluates the hargument for childreni . It may contain parameters like #1 or #2. These parameters are lled with the arguments that follow the call of \setbeamertemplate for the parent template. The number of arguments must be the number given ashargument numberi . An optional argument can also be speci ed in the usual way. Once the harguments for the childreni have been computed,\setbeamertemplate is called for all children for the prede ned template and with the computed arguments.

You may wonder what happens when certain prede ned options take a certain number of arguments, but another prede ned option takes a di erent number of arguments. In this case, the above-described mechanism cannot di erentiate between the prede ned options and it is unclear which or even how many arguments should be contained inharguments for childreni . For this reason, you can give the optional argument hprede ned option namei when calling \defbeamertemplateparent . If this optional argument is speci ed, the parenthood of the template applies only to this particular hprede ned option namei . Thus, if someone calls\setbeamertemplate for this hprede ned option namei , the given hargument for childreni is used. For other prede ned option names a possibly di erent de nition is used. You can imaging that leaving

out the optional hprede ned option namei means this hargument for childreni applies to all prede ned option names that have not been specially de ned di erently.

17 Colors

beamer 's color management allows you to specify the color of every element (like, say, the color of the section entries in a table of contents or, say, the color of the subsection entries in a mini table of contents in a sidebar). While the system is quite powerful, it is not trivial to use. To simplify the usage of the color system, you should consider using a prede ned color theme, which takes care of everything for you.

In the following, color themes are explained rst. The rest of the section consists of explanations of how the color management works internally. You will need to read these sections only if you wish to write your own color themes; or if you are quite happy with the prede ned themes but you absolutely insist that displayed mathematical text simply has to be typeset in a lovely pink.

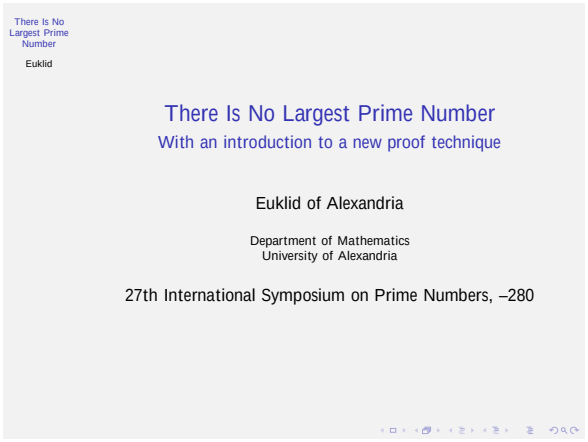
17.1 Color Themes

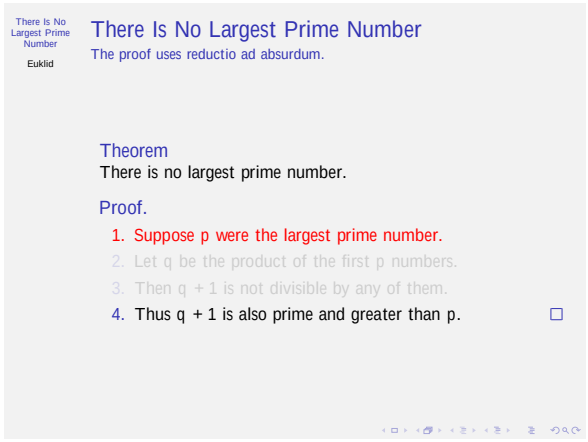
In order to also show the e ect of the di erent color themes on the sidebar, in the following examples the color themes are used together with the outer themesidebar .

17.1.1 Default and Special-Purpose Color Themes

\usecolortheme{default}

Example:





The default color theme is very sober. It installs little special colors and even less backgrounds. The default color theme sets up the default parent relations between the di erentbeamer -colors.

The main colors set in the default color theme are the following:

- normal text is black on white.
- alerted text is red.
- example text is a dark green (green with 50% black).
- structure is set to a light version of MidnightBlue (more precisely, 20% red, 20% green, and 70% blue).

Use this theme for a no-nonsense presentation. Since this theme is loaded by default, you cannot reload it after having loaded another color theme.

\usecolortheme[hoptionsi]{structure}

Example:

The example was created using \usecolortheme[named=SeaGreen]{structure} .

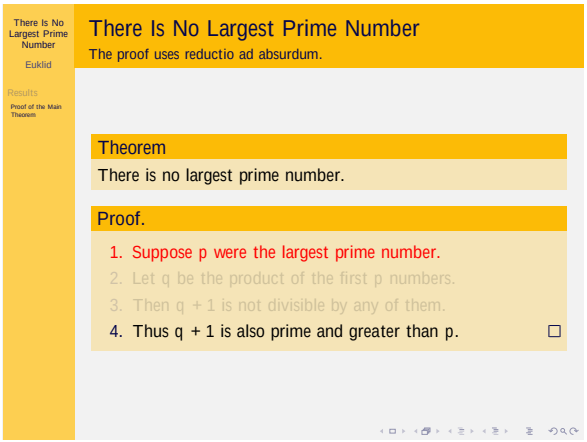
This theme offers a convenient way of changing the color used for structural elements. More precisely, it just changes the foreground of the beamer -color structure . You can also achieve this by directly invoking the function \setbeamercolor , but this color theme makes things a bit easier.

The theme offers several *options*, which can be used to specify the color to be used for structural elements:

- rgb={*rgb tuple*} sets the structure foreground to the specified red-green-blue tuple. The numbers are given as decimals between 0 and 1. For example, rgb={0.5,0,0} yields a dark red.
- RGB={*rgb tuple*} does the same as rgb, except that the numbers range between 0 and 255. For example, RGB={128,0,0} yields a dark red.
- cmyk={*cmyk tuple*} sets the structure foreground to the specified cyan-magenta-yellow-black tuple. The numbers are given as decimals between 0 and 1. For example, cmyk={0,1,1,0.5} yields a dark red.
- cmy={*cmy tuple*} is similar to cmyk, except that the black component is not specified.
- hsb={*hsb tuple*} sets the structure foreground to the specified hue-saturation-brightness tuple. The numbers are given as decimals between 0 and 1. For example, hsb={0,1,.5} yields a dark red.
- named={*color name*} sets the structure foreground to a named color. This color must previously have been defined using the \DefineNamedColor command. Adding the class option xcolor=dvipsnames or xcolor=svgnames will install a long list of standard dvips or SVG color names (respectively). See the file dvipsnam.def for the list.

\usecolortheme{sidebartab}

Example:



This theme uses the colors of Lufthansa, whose logo is a crane. It is not an o cial theme by that company, however.

\usecolortheme{dove}

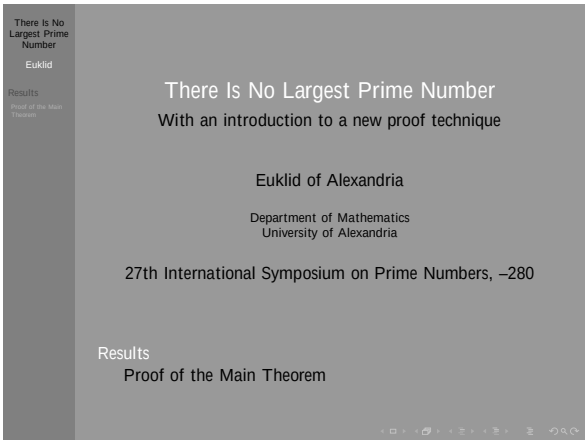
Example:

This theme is nearly a black and white theme and useful for creating presentations that are easy to print on a black-and-white printer. The theme uses grayscale in certain unavoidable cases, but never color. It also changes the font of alerted text to boldface.

When using this theme, you should consider using the class optiongray, which ensures that all colors are converted to grayscale. Also consider using thestructurebold font theme.

\usecolortheme{fly}

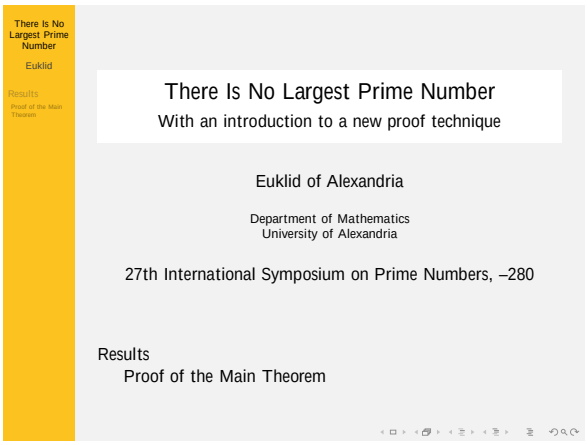
Example:



This theme is the consequent version of beetle and uses white/black/gray throughout. It does not go particularly well with themes that use shadows.

\usecolortheme{monarca}

Example:

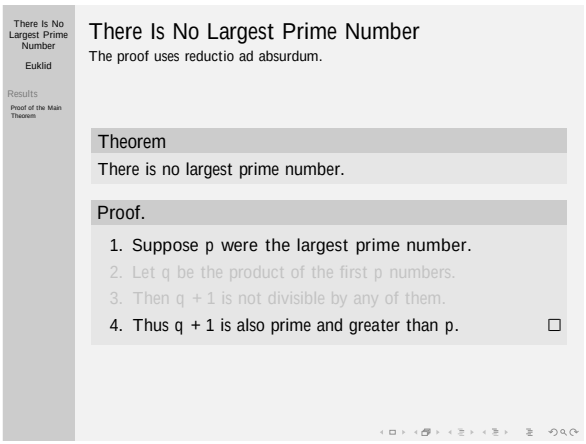


The theme is based on the colors of the Monarch butter y.

Theme author: Max Dohse.

\usecolortheme{seagull}

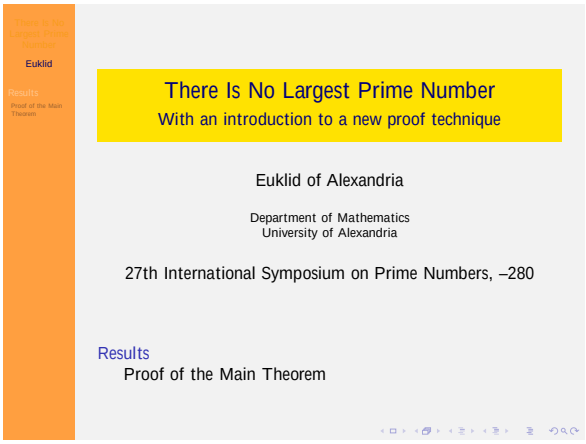
Example:



Like the dove color theme, this theme is useful for printing on a black-and-white printer. However, it uses di erent shades of gray extensively, which may or may not look good on a transparency.

\usecolortheme{wolverine}

Example:

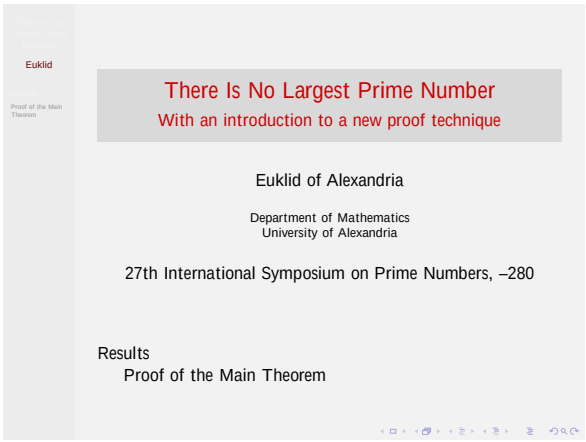


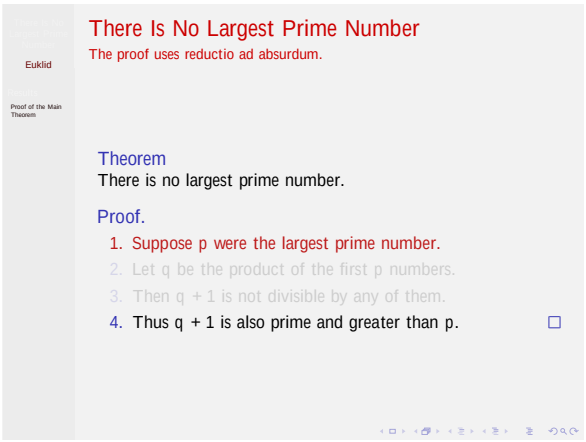
The theme is based on the colors of the University of Michigan's mascot, a wolverine.

Theme author: Madhusudan Singh.

\usecolortheme{beaver}

Example:





The theme is based on the colors of MIT's mascot, a beaver.

Theme author: Madhusudan Singh.

\usecolortheme{spruce}

Example:

The theme is based on the colors of Michigan State University.

Theme author: Alan Munn.

17.1.3 Inner Color Themes

Inner color themes only specify the colors of elements used in inner themes. Most noticeably, they specify the colors used for blocks. They can be used together with other (color) themes. If they are used to change the inner colors installed by a presentation theme or another color theme, they should obviously be specified after the other theme has been loaded. Inner color themes happen to have fewer names.

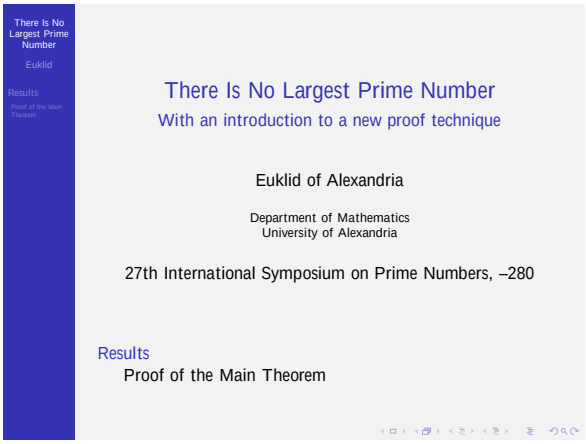
\usecolortheme{lily}

Example:

This theme is mainly used to uninstall any block colors setup by another theme, restoring the colors used in the default theme. In particular, using this theme will remove all background colors for blocks.

\usecolortheme{orchid}

Example:



A color theme somewhere in the middle between the whale and the seahorse. It was graciously donated by Manuel Carro. Like the seahorse, it goes well with the rose and the lily color theme.

17.2 Changing the Colors Used for Different Elements of a Presentation

This section explains how beamer 's color management works.

17.2.1 Overview of Beamer's Color Management

In beamer 's philosophy, every element of a presentation can have a different color. Unfortunately, it turned out that simply assigning a single color to every element of a presentation is not such a good idea. First of all, we sometimes want colors of elements to change during a presentation, like the color of the item indicators when they become alerted or inside an example block. Second, some elements naturally have two colors, namely a foreground and a background, but not always. Third, sometimes elements somehow should not have any special color but should simply run along with the color of their surrounding. Finally, giving a special color to every element makes it very hard to globally change colors (like changing all the different kind-of-blue things into kind-of-red things) and it makes later extensions even harder.

For all these reasons, the color of an element in beamer is a structured object, which we call a beamer -color. Every beamer -color has two parts: a foreground and a background. Either of these may be empty, which means that whatever foreground or background was active before should remain active when the color is used.

beamer -colors can inherit from other beamer -colors and the default themes make extensive use of this feature. For example, there is a beamer -color called structure and all sorts of elements inherit from this color. Thus, if someone changes structure , the color of all these elements automatically change accordingly. When a color inherits from another color, it can nevertheless still override only the foreground or the background.

It is also possible to inherit from another beamer -color in a more sophisticated way, which is more like using the other beamer -color in an indirect way. You can specify that, say, the background of the title should be a 90% of the background of normal text and 10% of the foreground of structure .

Inheritance and using of other beamer -colors is done dynamically. This means that if one of the parent beamer -colors changes during the presentation, the derived colors automatically also change.

The default color theme, which is always loaded, installs numerous beamer -colors and inheritance relations between them. These colors are explained throughout this guide. The color used for, say, frametitles is discussed in the section on frametitles, and so on.

17.2.2 Using Beamer's Colors

A beamer -color is not a normal color as defined by the color and xcolor packages and, accordingly, cannot be used directly as in commands like \color or \colorlet . Instead, in order to use a beamer -color, you should first call the command \usebeamercolor , which is explained below. This command will setup two (normal) colors called fg (for foreground) and bg (for, well, guess what). You can then say \color{fg} to install the foreground color and \color{bg} to install the background color. You can also use the colors fg and bg in any

context in which you normally use a color like, say, red. If a beamer-color does not have a foreground or a background, the colors fg or bg (or both) remain unchanged.

Inside templates, this command will typically have already been called for you with the option [fg].

\usebeamercolor*[⟨fg or bg⟩]{⟨beamer-color name⟩}

This command (possibly) changes the two colors fg and bg to the foreground and background color of the ⟨beamer-color name⟩. If the beamer-color does not specify a foreground, fg is left unchanged; if does not specify a background, bg is left unchanged.

You will often wish to directly use the color fg or bg after using this command. For this common situation, the optional argument ⟨fg or bg⟩ is useful, which may be either fg or bg. Giving this option will cause the foreground fg or the background bg to be immediately installed after they have been setup. Thus, the following command

```
\usebeamercolor[fg]{normal text}
```

is a shortcut for

```
\usebeamercolor{normal text}
\color{fg}
```

If you use the starred version of this command, the beamer-color normal text is used before the command is invoked. This ensures that, barring evil trickery, the colors fg and bg will be setup independently of whatever colors happened to be in use when the command is invoked.

This command has special side-effects. First, the (normal) color parent.bg is set to the value of bg prior to this call. Thus you can access the color that was in use prior to the call of this command via the color parent.bg.

Second, the special color ⟨beamer-color name⟩.fg is globally set to the same value as fg and ⟨beamer-color name⟩.bg is globally set to the value of bg. This allows you to access the foreground or background of a certain ⟨beamer-color name⟩ after another beamer-color has been used. However, referring to these special global colors should be kept to the unavoidable minimum and should be done as locally as possible since a change of the beamer-color will not reflect in a change of the colors ⟨beamer-color name⟩.fg and ⟨beamer-color name⟩.bg until the next invocation of \usebeamercolor. Also, if the ⟨beamer-color name⟩ does not specify a foreground or a background color, then the values of the special colors are whatever happened to be the foreground or background at the time of the last invocation of \usebeamercolor.

So, try not to get into the habit of writing \color{structure.fg} all the time, at least not without a \usebeamercolor{structure} close by.

Example:

```
This text is {\usebeamercolor[fg]{alerted text} alerted}. The
following box uses the fore- and background of frametitles:
{
  \usebeamercolor[fg]{frametitle}
  \colorbox{bg}{Frame Title}
}
```

article This command has no effect in article mode.

\ifbeamercolorempty[⟨fg or bg⟩]{⟨beamer-color name⟩}{⟨if undefined⟩}{⟨if defined⟩}

This command can be used to check whether the foreground or background of some ⟨beamer-color name⟩ is non-empty. If the foreground or background of ⟨beamer-color name⟩ is defined, ⟨if defined⟩ will be executed, otherwise the ⟨if undefined⟩ code.

Example:

```
\ifbeamercolorempty[bg]{frametitle}
{ % ``Transparent background''
  \usebeamercolor[fg]{frametitle}
  \insertframetitle
```

```
}
{ % Opaque background
  \usebeamercolor[fg]{frametitle}
  \colorbox{bg}{\insertframetitle}
}
```

17.2.3 Setting Beamer's Colors

To set or to change a beamer-color, you can use the command \setbeamercolor.

\setbeamercolor*{⟨beamer-color name⟩}{⟨options⟩}

Sets or changes a beamer-color. The ⟨beamer-color name⟩ should be a reasonably simple text (do not try too much trickery and avoid punctuation symbols), but it may contain spaces. Thus, normal text is a valid ⟨beamer-color name⟩ and so is My Color Number 2

In the most simple case, you just specify a foreground by giving the fg= option and, possibly, also a background using the bg= option.

Example: \setbeamercolor{normal text}{fg=black,bg=mylightgrey}

Example: \setbeamercolor{alerted text}{fg=red!80!black}

The effect of this command is accumulative, thus the following two commands

```
\setbeamercolor{section in toc}{fg=blue}
\setbeamercolor{section in toc}{bg=white}
```

have the same effect as

```
\setbeamercolor{section in toc}{fg=blue,bg=white}
```

Naturally, a second call with the same kind of ⟨option⟩ set to a different value overrides a previous call.

The starred version first resets everything, thereby switching off the accumulative effect. Use this starred version to completely reset the definition of some beamer-color.

The following ⟨options⟩ may be given:

- fg=⟨color⟩ sets the foreground color of ⟨beamer-color name⟩ to the given (normal) ⟨color⟩. The ⟨color⟩ may also be a color expression like red!50!black, see the manual of the xcolor package. If ⟨color⟩ is empty, the ⟨beamer-color name⟩ has no special foreground and when the color is used, the foreground currently in force should not be changed.

  Specifying a foreground this way will override any inherited foreground color.
- bg=⟨color⟩ does the same as the fg option, but for the background.
- parent=⟨parent beamer-color(s)⟩ specifies that ⟨beamer-color name⟩ should inherit from the specified ⟨parent beamer-color(s)⟩. Any foreground and/or background color set by the parents will also be used when ⟨beamer-color name⟩ is used. If multiple parents specify a foreground, the last one wins; and likewise for the backgrounds.

  Example:

```
\setbeamercolor{father}{fg=red}
\setbeamercolor{mother}{bg=green}
\setbeamercolor{child}{parent={father,mother}}
\begin{beamercolorbox}{child}
  Terrible red on green text.
\end{beamercolorbox}

\setbeamercolor{father}{fg=blue}
\begin{beamercolorbox}{child}
  Now terrible blue on green text, since parent was changed.
\end{beamercolorbox}
```

Note that a change of the foreground or background of a parent changes the corresponding foreground or background of the child (unless it is overruled).

A beamer -color can not only have parents, but also grandparents and so on.

- use=another beamer-color is used to make sure that another beamer -color is setup correctly before the foreground or background color speci cation are evaluated.

  Suppose you wish the foreground of items to be a mixture of 50% of the foreground of structural elements and 50% of the normal foreground color. You could try

  ```
  \setbeamercolor{item}{fg=structure.fg!50!normal text.fg}
  ```

  However, this will not necessarily give the desired result: If the beamer -color structure changes, the (normal) color structure.fg is not immediately updated. In order to ensure that the normal color structure.fg is correct, use the following:

  ```
  \setbeamercolor{item}{use={structure,normal text},fg=structure.fg!50!normal text.fg}
  ```

  This will guarantee that the colors structure.fg and normal text.fg are setup correctly when the foreground of item is computed.

  To show the di erence, consider the following example:

  ```
  \setbeamercolor{grandfather}{fg=red}
  \setbeamercolor{grandmother}{bg=white}
  \setbeamercolor{father}{parent={grandfather,grandmother}}
  \setbeamercolor{mother}{fg=black}
  {
    \usebeamercolor{father}\usebeamercolor{mother}
    %% Defines father.fg and mother.fg globally
  }
  \setbeamercolor{my color A}{fg=father.fg!50!mother.fg}
  \setbeamercolor{my color B}{use={father,mother},fg=father.fg!50!mother.fg}

  {\usebeamercolor[fg]{my color A} dark red text}
  {\usebeamercolor[fg]{my color B} also dark red text}

  \setbeamercolor{grandfather}{fg=green}

  {\usebeamercolor[fg]{my color A} still dark red text}
  {\usebeamercolor[fg]{my color B} now dark green text}
  ```

17.3 The Color of Mathematical Text

By default, mathematical text does not have any special color it just inherits the surrounding color. Some people prefer mathematical text to have some special color. Though we do not recommend this (we believe mathematical text should not stand out amid the normal text), beamer makes it (reasonably) easy to change the color of mathematical text. Simply change the following colors:

Beamer-Color math text

This color is the parent of math text inlined and math text displayed . It is empty by default. See those colors for details.

Beamer-Color math text inlined

Color parents: math text

If the foreground of this color is set, inlined mathematical text is typeset using this color. This is done via some \everymath hackery and may not work in all cases. If not, you'll have to try to nd a way around the problem. The background is currently ignored.

Beamer-Color math text displayed

Color parents: math text

Like math text inlined, only for so-called displayed mathematical text. This is mathematical text between \[and \] or between $$ and $$ or inside environments like equation or align. The setup of this color is somewhat fragile, use at your own risk. The background is currently ignored.

Beamer-Color normal text in math text

If the foreground of this color is set, normal text inside mathematical text (which is introduced using the \text command) will be typeset using this color. The background is currently ignored.

17.4 The Color Palettes

When one designs a color theme, one faces the following problem: Suppose we want the colors in the headline to gradually change from black to, say, blue. Whatever is at the very top of the headline should be black, what comes right below it should be dark blue, and at the bottom of the headline things should just be blue. Unfortunately, different outer themes will put different things at the top. One theme might put the author at the top, another theme might put the document title there. This makes it impossible to directly assign one of the three colors black, dark blue, and blue to the different elements that are typically rendered in the headline. No matter how we assign them, things will look wrong for certain outer themes.

To circumvent this problem, beamer uses a layer of palette colors. Color themes typically only change these palette colors. For example, a color theme might make the beamer-color palette primary blue, make palette secondary a dark blue, and make palette tertiary black. Outer themes can now setup things such that whatever they show at the top of the headline inherits from palette primary, what comes below inherits from palette secondary, and whatever is at the bottom inherits from palette tertiary. This way, color themes can change the way even complicated outer themes look and they can do so consistently.

Note that the user can still change the color of every element individually, simply by overriding the color(s) of the elements in the headline. In a sense, the palette colors are just a suggestion how things should be colored by an outer theme.

In detail, the following palette colors are used by outer themes.

Beamer-Color palette primary

Outer themes (should) base the color of navigational elements and, possibly, also of other elements, on the four palette colors. The primary palette should be used for the most important navigational elements, which are usually the ones that change most often and hence require the most attention by the audience. The secondary and tertiary are less important, the quaternary one is the least important.

By default, the palette colors do not have a background and the foreground ranges from structure.fg to black.

For the sidebar, there is an extra set of palette colors, see palette sidebar primary.

Beamer-Color palette secondary

See palette primary.

Beamer-Color palette tertiary

See palette primary.

Beamer-Color palette quaternary

See palette primary.

Beamer-Color palette sidebar primary

Similar to palette primary, only outer themes (should) base the colors of elements in the sidebar on the four sidebar palette colors.

Beamer-Color palette sidebar secondary

See palette sidebar primary.

Beamer-Color palette sidebar tertiary

See palette sidebar primary.

Beamer-Color palette sidebar quaternary

See palette sidebar primary.

17.5 Miscellaneous Colors

In this section some basic colors are listed that do not belong to any special commands.

Beamer-Color/-Font normal text

The color is used for normal text. At the beginning of the document the foreground color is installed as \normalcolor. The background of this color is used by the default background canvas for the background of the presentation, see Section 8.2.7. The background is also the default value of the normal color bg.

Since the color is the root of all other beamer-colors, both a foreground and a background must be installed. In particular, to get a transparent background canvas, make the background of the beamer-color background canvas empty, not the background of this color.

The beamer-font currently is not used. In particular, redefining this font will not have any effect. This is likely to change in the future.

Beamer-Color/-Font example text

The color/font is used when text is typeset inside an example block.

Beamer-Color/-Font titlelike

This color/font is a more specialized form of the structure color/font. It is the base for all elements that are like titles. This includes the frame title and subtitle as well as the document title and subtitle.

Beamer-Color separation line

The foreground of this color is used for separating lines. If the foreground is empty, no separation line is drawn.

Beamer-Color upper separation line head

Color parents: separation line

Special case for the uppermost separation line in a headline.

Beamer-Color middle separation line head

Color parents: separation line

Special case for the middle separation line in a headline.

Beamer-Color lower separation line head

Color parents: separation line

Special case for the lower separation line in a headline.

Beamer-Color upper separation line foot

Color parents: separation line

Special case for the uppermost separation line in a footline.

Beamer-Color middle separation line foot

Color parents: separation line

Special case for the middle separation line in a footline.

Beamer-Color lower separation line foot

Color parents: separation line

Special case for the lower separation line in a footline.

17.6 Transparency Effects

By default, covered items are not shown during a presentation. Thus if you write \uncover<2>{Text.}, the text is not shown on any but the second slide. On the other slides, the text is not simply printed using the background color it is not shown at all. This effect is most useful if your background does not have a uniform color.

Sometimes however, you might prefer that covered items are not completely covered. Rather, you would like them to be shown already in a very dim or shaded way. This allows your audience to get a feeling for what is yet to come, without getting distracted by it. Also, you might wish text that is covered once more still to be visible to some degree.

Ideally, there would be an option to make covered text transparent. This would mean that when covered text is shown, it would instead be mixed with the background behind it. Unfortunately, pgf does not support real transparency yet. Instead, transparency is created by mixing the color of the object you want to show with the current background color (the color bg, which has hopefully been setup such that it is the average color of the background on which the object should be placed). To install this effect, you can use:

```
\setbeamercovered{transparent}
```

This command allows you to specify in a quite general way how a covered item should be rendered. You can even specify different ways of rendering the item depending on how long it will take before this item is shown or for how long it has already been covered once more. The transparency effect will automatically apply to all colors, except for the colors in images. For images there is a workaround, see the documentation of the pgf package.

\setbeamercovered{ *⟨options⟩* }

This command offers several different options, the most important of which is transparent. All options are internally mapped to the two options still covered and again covered.

In detail, the following *⟨options⟩* may be given:

- invisible is the default and causes covered text to completely disappear.
- transparent= *⟨opaqueness⟩* causes covered text to be typeset in a transparent way. By default, this means that 85% of the background color is mixed into all colors or that the *⟨opaqueness⟩* of the text is 15%. You can specify a different *⟨percentage⟩*, where 0 means totally transparent and 100 means totally opaque.

  Unfortunately, this value is kind of specific to every projector. What looks good on your screen need not look good during a presentation.
- dynamic Makes all covered text quite transparent, but in a dynamic way. The longer it will take till the text is uncovered, the stronger the transparency.
- highly dynamic Has the same effect as dynamic, but the effect is stronger.
- still covered= *⟨not yet list⟩* specifies how to render covered items that have not yet been uncovered. The *⟨not yet list⟩* should be a list of \opaqueness commands, see the description of that command, below.

  Example:

  ```
  \setbeamercovered{%
    still covered={\opaqueness<1>{15}\opaqueness<2>{10}\opaqueness<3>{5}\opaqueness<4->{2}},
    again covered={\opaqueness<1->{15}}}
  ```

- ^ again covered=⟨once more list⟩ specifies how to render covered items that have once more been covered, that is, that had been shown before but are now covered again.

\opaqueness<⟨overlay specification⟩>{⟨percentage of opaqueness⟩}

The ⟨overlay specification⟩ specifies on which slides covered text should have which ⟨percentage of opaqueness⟩. Unlike other overlay specifications, this ⟨overlay specification⟩ is a relative overlay specification. For example, the specification 3 here means things that will be uncovered three slides ahead, respectively things that have once more been covered for three slides. More precisely, if an item is uncovered for more than one slide and then covered once more, only the first moment of uncovering is used for the calculation of how long the item has been covered once more.

An opaqueness of 100 is fully opaque and 0 is fully transparent. Currently, since real transparency is not yet implemented, this command causes all colors to get a mixing of ⟨percentage of opaqueness⟩% of the current bg. At some future point this command might result in real transparency.

The alternate pgf extension used inside an opaque area is ⟨percentage of opaqueness⟩ opaque. In case of nested calls, only the innermost opaqueness specification is used.

Example:

```
\setbeamercovered{still covered={\opaqueness<1->{15}},again covered={\opaqueness<1->{15}}}
\pgfdeclareimage{book}{book}
\pgfdeclareimage{book.!15opaque}{filenameforbooknearlytransparent}
```

Makes everything that is uncovered in two slides only 15 percent opaque.

18 Fonts

The first subsection introduces the predefined font themes that come with beamer and which make it easy to change the fonts used in a presentation. The next subsection describes further special commands for changing some basic attributes of the fonts used in a presentation. The last subsection explains how you can get a much more fine-grained control over the fonts used for every individual element of a presentation.

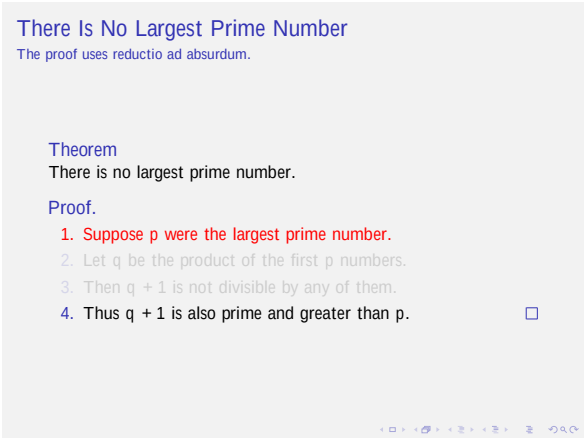
18.1 Font Themes

beamer comes with a set of font themes. When you use such a theme, certain fonts are changed as described below. You can use several font themes in concert. For historical reasons, you cannot change all aspects of the fonts used using font themes in some cases special commands and options are needed, which are described in the next subsection.

The following font themes only change certain font attributes, they do not choose special font families (although that would also be possible and themes doing just that might be added in the future). Currently, to change the font family, you need to load special packages as explained in the next subsection.

\usefonttheme{default}

Example:



The default font theme installs a sans serif font for all text of the presentation. The default theme installs different font sizes for things like titles or head- and footlines, but does not use boldface or italics for highlighting. To change some or all text to a serif font, use the serif theme.

Note: The command \mathrm will always produce upright (not slanted), serif text and the command \mathsf will always produce upright, sans-serif text. The command \mathbf will produce upright, boldface, sans-serif or serif text, depending on whether mathsans or mathserif is used.

To produce an upright, sans-serif or serif text, depending on whether mathsans or mathserif is used, you can use for instance the command \operatorname from the amsmath package. Using this command instead of \mathrm or \mathsf directly will automatically adjust upright mathematical text if you switch from sans-serif to serif or back.

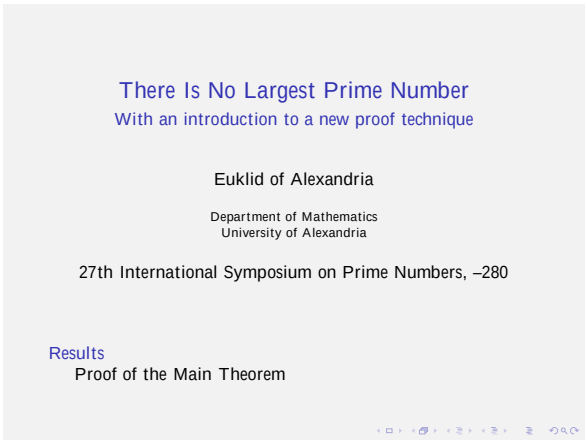
\usefonttheme{professionalfonts}

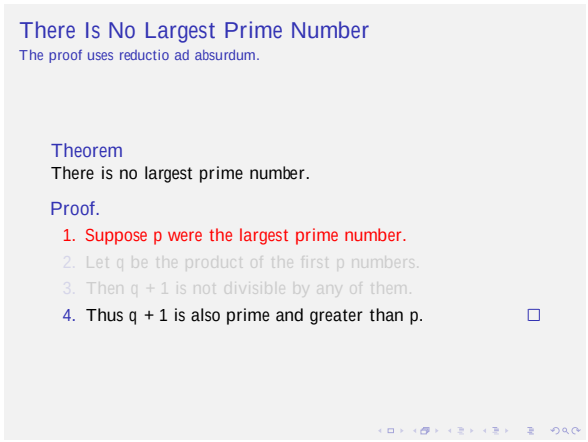
This font theme does not really change any fonts. Rather, it suppresses certain internal replacements performed by beamer. If you use professional fonts (fonts that you buy and that come with a complete set of every symbol in all modes), you do not want beamer to meddle with the fonts you use. beamer normally replaces certain character glyphs in mathematical text by more appropriate versions. For example, beamer will normally replace glyphs such that the italic characters from the main font are used for variables in mathematical text. If your professional font package takes care of this already, beamer 's meddling should be switched off. Note that beamer 's substitution is automatically turned off if one of the following packages

is loaded: arevmath, hvmath, kpfonts , lmodern, lucidabr , lucimatx , mathastext , mathpmnt, mathpple, mathtime, mtpro, and mtpro2. It is also turned o when unicode-math is loaded for the use of Unicode math fonts. If your favorite professional font package is not among these, use the professionalfonts option (and write us an email, so that the package can be added).

\usefonttheme[hoptionsi]{serif}

Example:





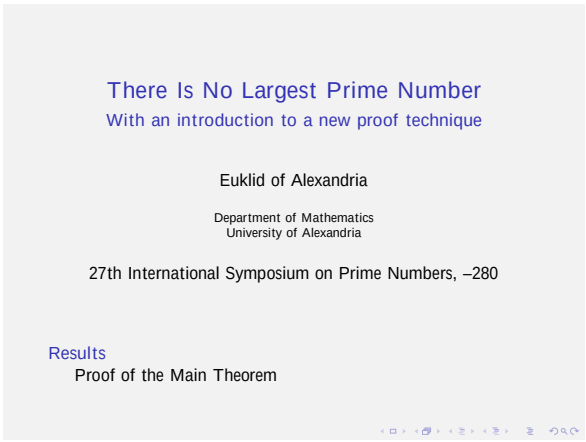
This theme causes all text to be typeset using the default serif font (except if you specify certain hoptionsi). You might wish to consult Section 5.6.2 on whether you should use serif fonts.

The following hoptionsi may be given:

- stillsansserifmath causes mathematical text still to be typeset using sans serif. This option only makes sense if you also use the stillsansseriftext option since sans serif math inside serif text looks silly.
- stillsansserifsmall will cause small text to be still typeset using sans serif. This refers to the text in the headline, footline, and sidebars. Using this options is often advisable since small text is often easier to read in sans serif.
- stillsansseriflarge will cause large text like the presentation title or the frame title to be still typeset using sans serif. Sans serif titles with serif text are a popular combination in typography.
- stillsansseriftext will cause normal text (none of the above three) to be still typeset using sans serif. If you use this option, you should most likely also use the rst two. However, by not using stillsansseriflarge , you get a serif (possibly italic) title over a sans serif text. This can be an interesting visual e ect. Naturally, interesting typographic e ect can mean terrible typographic e ect if you choose the wrong fonts combinations or sizes. You'll need some typographic experience to judge this correctly. If in doubt, try asking someone who should know.
- onlymath is a short-cut for selecting all of the above options except for the rst. Thus, using this option causes only mathematical text to by typeset using a serif font. Recall that, by default, mathematical formulas are also typeset using sans-serif letters. In most cases, this is visually the most pleasing and easily readable way of typesetting mathematical formulas if the surrounding text is typeset using sans serif. However, in mathematical texts the font used to render, say, a variable is sometimes used to di erentiate between di erent meanings of this variable. In such case, it may be necessary to typeset mathematical text using serif letters. Also, if you have a lot of mathematical text, the audience may be quicker to parse it if it is typeset the way people usually read mathematical text: in a serif font.

\usefonttheme[hoptionsi]{structurebold}

Example:



This font theme will cause titles and text in the headlines, footlines, and sidebars to be typeset in a bold font.

The following ⟨options⟩ may be given:

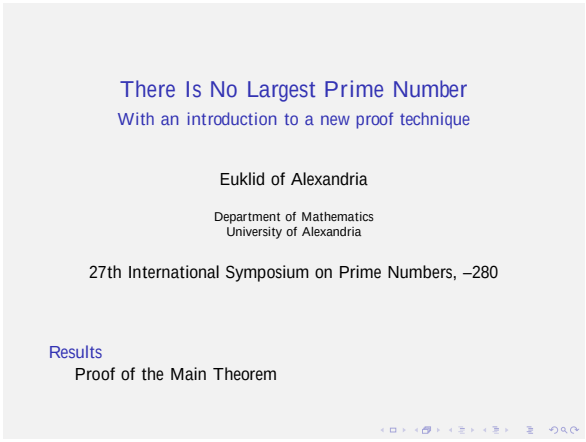
- onlysmall will cause only small text to be typeset in bold. More precisely, only the text in the headline, footline, and sidebars is changed to be typeset in bold. Large titles are not a ected.
- onlylarge will cause only large text to be typeset in bold. These are the main title, frame titles, and section entries in the table of contents.

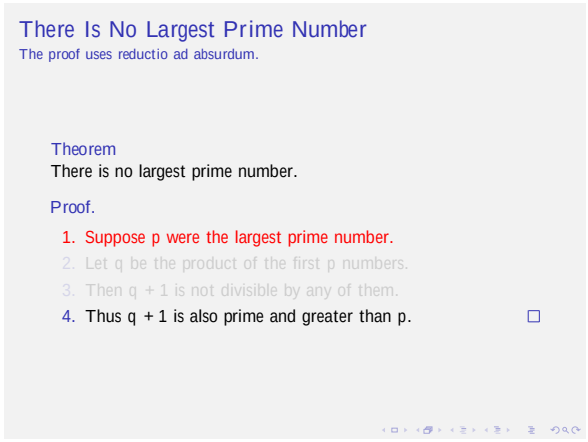
As pointed out in Section 5.6.1, you should use this theme (possibly with the onlysmall option) if your font is not scaled down properly or for light-on-dark text.

The normal themes do not install this theme by default, while the old compatibility themes do. Since you can reload the theme once it has been loaded, you cannot use this theme with the old compatibility themes to set also titles to a bold font.

\usefonttheme[⟨options⟩]{structureitalicserif}

Example:





This theme is similarly as the structurebold font theme, but where structurebold makes text bold, this theme typesets it in italics and in the standard serif font. The same ⟨options⟩ as for the structurebold theme are supported. See Section 5.6.3 for the pros and cons of using italics.

\usefonttheme[⟨options⟩]{structuresmallcapsserif}

Example:

Again, this theme does exactly the same as the structurebold font theme, only this time text is set using small caps and a serif font. The same options as for the structurebold theme are supported. See Section 5.6.3 for the pros and cons of using small caps.

18.2 Font Changes Made Without Using Font Themes

While most font decisions can be made using font themes, for historical reasons some changes can only be made using class options or by loading special packages. These options are explained in the following. Possibly, these options will be replaced by themes in the future.

18.2.1 Choosing a Font Size for Normal Text

As pointed out in Section 5.6.1, measuring the default font size in points is not really a good idea for presentations. Nevertheless, beamer does just that, setting the default font size to 11pt as usual. This may seem ridiculously small, but the actual size of each frame size is by default just 128mm by 96mm and the viewer application enlarges the font. By specifying a default font size smaller than 11pt you can put more onto each slide, by specifying a larger font size you can t on less.

To specify the font size, you can use the following class options:

\documentclass[8pt]{beamer}

This is way too small. Requires that the package extsize is installed.

\documentclass[9pt]{beamer}

This is also too small. Requires that the package extsize is installed.

\documentclass[10pt]{beamer}

If you really need to t more onto each frame, use this option. Works without extsize .

\documentclass[smaller]{beamer}

Same as the 10pt option.

\documentclass[11pt]{beamer}

The default font size. You need not specify this option.

\documentclass[12pt]{beamer}

Makes all fonts a little bigger, which makes the text more readable. The downside is that less ts onto each frame.

\documentclass[bigger]{beamer}

Same as the 12pt option.

subsection title , 99
subsubitem , 117
subsubitem projected , 117
subsubsection in head/foot , 72
subsubsection in sidebar , 72
subsubsection in toc , 97
subsubsection in toc shaded , 97
tiny structure , 118
title page , 94
titlelike , 193
verse , 131
Beamer options
always typeset second mode , 218
hide notes , 206
previous slide on second screen , 218
second mode text on second screen , 217
show notes , 206
show notes on second screen , 206
show only notes , 207
Beamer templates
abstract begin , 131
abstract end , 131
abstract title , 131
alerted text begin , 119
alerted text end , 119
background , 77
background canvas , 77
bibliography entry author , 104
bibliography entry location , 104
bibliography entry note , 105
bibliography entry title , 104
bibliography item , 105
block alerted begin , 120
block alerted end , 120
block begin , 120
block end , 120
block example begin , 121
block example end , 121
button , 108
caption , 128
caption label separator , 128
description body begin , 117
description body end , 117
description item , 117
enumerate item , 115
enumerate mini template , 116
enumerate subitem , 115
enumerate subsubitem , 116
footline , 67
footnote , 133
frame begin , 63
frame end , 63
frametitle , 75
frametitle continuation , 60
headline , 64
itemize item , 114
itemize subitem , 114
itemize subsubitem , 114
itemize/enumerate body begin , 116
itemize/enumerate body end , 116
logo , 74
mini frame , 70
mini frame in current subsection , 70
mini frame in other subsection , 70
navigation symbols , 73
note page , 205
page number in head/foot , 67
part page , 100
proof begin , 125
proof end , 125
qed symbol , 123
quotation begin , 132
quotation end , 132
quote begin , 132
quote end , 132
section in head/foot , 71
section in head/foot shaded , 71
section in sidebar , 71
section in sidebar shaded , 72
section in toc , 96
section in toc shaded , 96
section page , 98
sidebar canvas left , 68
sidebar canvas right , 69
sidebar left , 68
sidebar right , 68
structure begin , 118
structure end , 119
subsection in head/foot , 72
subsection in head/foot shaded , 72
subsection in sidebar , 72
subsection in sidebar shaded , 72
subsection in toc , 96
subsection in toc shaded , 97
subsection page , 99
subsubsection in head/foot , 72
subsubsection in head/foot shaded , 72
subsubsection in sidebar , 72
subsubsection in sidebar shaded , 72
subsubsection in toc , 97
subsubsection in toc shaded , 97
theorem begin , 124
theorem end , 125
title page , 94
verse begin , 131
verse end , 132
beamerarticle package , 209
beamerboxesrounded environment , 127
\beamerbutton , 107
beamercolorbox environment , 125
\beamerdefaultoverlayspecification , 91
beamerfoils package , 234
\beamergotobutton , 108
beamerprosper package , 226
\beamerreturnbutton , 108
beamerseminar package , 231
\beamerskipbutton , 108
beamertexpower package , 236

beaver color theme, 184
beetle color theme, 181
Bergen presentation theme, 150
Berkeley presentation theme, 155
Berlin presentation theme, 157
\bibitem , 105
bibliography entry author template/color/font, 104
bibliography entry location template/color/font, 104
bibliography entry note template/color/font, 105
bibliography entry title template/color/font, 104
bibliography item template/color/font, 105
bigger class option, 199
block environment, 120
block alerted begin template, 120
block alerted end template, 120
block begin template, 120
block body color/font, 120
block body alerted color/font, 120
block body example color/font, 121
block end template, 120
block example begin template, 121
block example end template, 121
block title color/font, 120
block title alerted color/font, 120
block title example color/font, 121
blocks parent template, 119
\blue , 233
\bm, 235
\bmstyle , 235
Boadilla presentation theme, 150
boldequation environment, 235
\boxedsteps , 237
boxes presentation theme, 149
\bstep , 237
button template/color/font, 108
button border color, 108

c class option, 76
CambridgeUSpresentation theme, 152
caption template/color/font, 128
caption label separator template, 128
caption name color/font, 128
circles inner theme, 164
CJKpackage, 16
Class options for beamer
10pt , 199
11pt , 199
12pt , 199
14pt , 200
17pt , 200
20pt , 200
8pt , 199
9pt , 199
aspectratio=141 , 78
aspectratio=149 , 78
aspectratio=1610 , 77
aspectratio=169 , 78
aspectratio=32 , 78
aspectratio=43 , 78
aspectratio=54 , 78
bigger , 199
c, 76
color= hlist of optionsi , 16
compress, 70
draft , 31
envcountsect , 122
handout, 209
hyperref= hlist of optionsi , 17
ignorenonframetext , 210
noamssymb, 123
noamsthm, 123
notheorems, 123
smaller , 199
t , 76
trans , 208
ucs, 17
usepdftitle=false , 95
utf8 , 18
utf8x , 18
xcolor= hlist of optionsi , 20
Classes
foils , 17
prosper , 19
seminar , 19
\code , 237
\codeswitch , 237
color package, 16
Color themes
albatross , 180
beaver, 184
beetle , 181
crane , 181
default , 178
dolphin , 187
dove, 182
fly , 182
lily , 185
monarca, 183
orchid , 185
rose , 186
seagull , 183
seahorse , 187
sidebartab , 179
spruce , 184
structure , 178
whale, 186
wolverine , 184
color= hlist of optionsi class option, 16
Colors, see Beamer colors
colortbl package, 16
\column , 130
column environment, 129
columns environment, 129
compress class option, 70
conference-talks/conference-ornate-20min solution, 43
Copenhagenpresentation theme, 161

crane color theme, 181

Darmstadt presentation theme, 159
\date , 95
default color theme, 178
default font theme, 196
default inner theme, 163
default outer theme, 166
default presentation theme, 149
\defbeamertemplate , 174
\defbeamertemplatealias , 175
\defbeamertemplateparent , 176
definition environment, 122
deluxetable package, 16
description environment, 116
description body begin template, 117
description body end template, 117
description item template/color/font, 117
dolphin color theme, 187
dove color theme, 182
draft class option, 31
DraTex package, 17
Dresden presentation theme, 158
\dstep , 237
\dualslide , 229

EastLansing presentation theme, 152
Elements, see Beamer templates, colors, and fonts
\email , 227
\endfoil , 235
enumerate environment, 114
enumerate package, 17
enumerate item template/color/font, 115
enumerate items parent template, 115
enumerate mini template template/color/font, 116
enumerate subitem template/color/font, 115
enumerate subsubitem template/color/font, 116
enumstep environment, 229
envcountsect class option, 122
Environments
  abstract , 131
  actionenv , 91
  alertblock , 120
  alertenv , 119
  altenv , 85
  beamerboxesrounded, 127
  beamercolorbox, 125
  block , 120
  boldequation , 235
  column, 129
  columns, 129
  definition , 122
  description , 116
  enumerate, 114
  enumstep, 229
  example, 122
  exampleblock, 121
  frame, 59
  Itemize , 229
  itemize , 113
  itemstep , 229
  notes, 230
  onlyenv, 85
  overlayarea , 86
  overprint , 86
  proof , 122
  quotation , 132
  quote, 132
  semiverbatim , 131
  slide , 227, 232
  structureenv , 118
  thebibliography , 104
  theorem, 121
  verse, 131
example environment, 122
example text color/font, 193
exampleblock environment, 121
\expandbeamertemplate , 173

fly color theme, 182
\foilhead , 235
foils class, 17
\FoilTeX , 235
Font themes
  default , 196
  professionalfonts , 196
  serif , 197
  structurebold , 197
  structureitalicserif , 198
  structuresmallcapsserif , 198
fontenc package, 17
Fonts, see Beamer fonts
\fontText , 229
\fontTitle , 229
footline template/color/font, 67
\footnote , 132
footnote template/color/font, 133
footnote mark color/font, 133
fourier package, 17
frame environment, 59
frame begin template, 63
frame end template, 63
\framesubtitle , 76
framesubtitle color/font, 76
\frametitle , 75
frametitle template/color/font, 75
frametitle continuation template/color/font, 60
\framezoom, 111
Frankfurt presentation theme, 159
\FromSlide , 228
\fromSlide , 228

generic-talks/generic-ornate-15min-45min
  solution, 43
Goettingen presentation theme, 156
\green , 233

HA-prosper package, 17

handout class option, 209
Hannover presentation theme, 157
headline template/color/font, 64
\hiddenitem , 229
hide notes beamer option, 206
\hyperlink , 109
\hyperlinkappendixend , 110
\hyperlinkappendixstart , 110
\hyperlinkdocumentend , 110
\hyperlinkdocumentstart , 110
\hyperlinkframeend , 109
\hyperlinkframeendprev , 109
\hyperlinkframestart , 109
\hyperlinkframestartnext , 109
\hyperlinkmovie , 139
\hyperlinkmute , 144
\hyperlinkpresentationend , 109
\hyperlinkpresentationstart , 109
\hyperlinkslidenext , 109
\hyperlinkslideprev , 109
\hyperlinksound , 144
hyperref package, 17
hyperref= hlist of optionsi class option, 17
\hypertarget , 107

\ifarticle , 233
\ifbeamercolorempty , 189
\ifbeamertemplateempty , 173
\ifportrait , 233
\ifslide , 233
\ifslidesonly , 233
ignorenonframetext class option, 210
Ilmenau presentation theme, 158
\includeonlyframes , 31
\includeonlylecture , 101
\includeslide , 212
infolines outer theme, 166
inmargin inner theme, 165
Inner themes
  circles , 164
  default , 163
  inmargin , 165
  rectangles , 164
  rounded, 164
inputenc package, 17
Inserts
  \insertappendixendpage , 67
  \insertappendixframenumber , 66
  \insertappendixstartpage , 66
  \insertauthor , 94
  \insertbackfindforwardnavigationsymbol , 74
  \insertbiblabel , 105
  \insertblocktitle , 120
  \insertbuttontext , 108
  \insertcaption , 128
  \insertcaptionname , 128
  \insertcaptionnumber , 128
  \insertcontinuationcount , 60
  \insertcontinuationcountroman , 60
  \insertcontinuationtext , 60
  \insertdate , 94
  \insertdescriptionitem , 117
  \insertdocnavigationsymbol , 74
  \insertdocumentendpage , 67
  \insertdocumentstartpage , 67
  \insertenumlabel , 115, 116
  \insertfootnotemark , 133
  \insertfootnotetext , 133
  \insertframeendpage , 66
  \insertframenavigationsymbol , 74
  \insertframenumber , 66
  \insertframestartpage , 66
  \insertframesubtitle , 76
  \insertframetitle , 76
  \insertgotosymbol , 108
  \insertinstitute , 94
  \insertlogo , 74
  \insertmainframenumber , 66
  \insertnavigation{ hwidth i}, 65
  \insertnote , 205
  \insertoverlaynumber , 66
  \insertpagenumber , 65
  \insertpart , 100
  \insertpartendpage , 66
  \insertpartheadnumber , 71
  \insertpartnumber , 100
  \insertpartstartpage , 66
  \insertpresentationendpage , 66
  \insertpresentationstartpage , 66
  \insertproofname , 125
  \insertreturnsymbol , 108
  \insertromanpartnumber , 100
  \insertsection , 65, 98
  \insertsectionendpage , 66
  \insertsectionhead , 71
  \insertsectionheadnumber , 71
  \insertsectionnavigation , 65
  \insertsectionnavigationhorizontal , 65
  \insertsectionnavigationsymbol , 74
  \insertsectionnumber , 99
  \insertsectionstartpage , 66
  \insertshortauthor , 65
  \insertshortdate , 66
  \insertshortinstitute , 66
  \insertshortpart , 66
  \insertshortsubtitle , 66
  \insertshorttitle , 66
  \insertskipsymbol , 108
  \insertslideintonotes , 205
  \insertslidenavigationsymbol , 74
  \insertslidenumber , 66
  \insertsubenumlabel , 115
  \insertsubsection , 66, 99
  \insertsubsectionendpage , 66
  \insertsubsectionhead , 72
  \insertsubsectionheadnumber , 72
  \insertsubsectionnavigation , 66
  \insertsubsectionnavigationhorizontal , 66

\reswitch , 237
\revstep , 237
Rochester presentation theme, 153
rose color theme, 186
\rotatefoilhead , 235
rounded inner theme, 164

seagull color theme, 183
seahorse color theme, 187
second mode text on second screen beamer option, 217
\section , 96
section in head/foot template/color/font, 71
section in head/foot shaded template, 71
section in sidebar template/color/font, 71
section in sidebar shaded template/color, 72
section in toc template/color/font, 96
section in toc shaded template/color/font, 96
section name color/font, 98
section page template, 98
section title color/font, 98
\sectionpage , 98
sections/subsections in toc parent template, 103
sections/subsections in toc shaded parent template, 103
seminar class, 19
semiverbatim environment, 131
separation line color, 193
serif font theme, 197
\setbeamercolor , 190
\setbeamercovered , 194
\setbeamerfont , 201
\setbeamersize , 78
\setbeamertemplate , 173
\setjobnamebeamerversion , 212
shadowouter theme, 169
short-talks/speaker_introduction-ornate-2min solution, 43
show notes beamer option, 206
show notes on second screen beamer option, 206
show only notes beamer option, 207
sidebar outer theme, 168
sidebar canvas left template, 68
sidebar canvas right template, 69
sidebar left template/color/font, 68
sidebar right template/color/font, 68
sidebartab color theme, 179
Singapore presentation theme, 160
slide environment, 227, 232
\slideCaption , 229
smaller class option, 199
smoothbars outer theme, 168
smoothtree outer theme, 170
Solutions
  conference-talks/conference-ornate-20min , 43
  generic-talks/generic-ornate-15min-45min , 43
  short-talks/speaker_introduction-ornate-2min , 43
\sound , 143
split outer theme, 169
spruce color theme, 184
\step , 236
\steponce , 236
\stepwise , 236
\structure , 118
structure color theme, 178
structure color/font, 118
structure begin template, 118
structure end template, 119
structurebold font theme, 197
structureenv environment, 118
structureitalicserif font theme, 198
structuresmallcapsserif font theme, 198
subitem color/font, 117
subitem projected color/font, 117
\subject , 95
\subsection , 96, 97
subsection in head/foot template/color/font, 72
subsection in head/foot shaded template, 72
subsection in sidebar template/color/font, 72
subsection in sidebar shaded template, 72
subsection in toc template/color/font, 96
subsection in toc shaded template/color/font, 97
subsection name color/font, 99
subsection page template, 99
subsection title color/font, 99
\subsectionpage , 99
subsubitem color/font, 117
subsubitem projected color/font, 117
\subsubsection , 97
subsubsection in head/foot template/color/font, 72
subsubsection in head/foot shaded template, 72
subsubsection in sidebar template/color/font, 72
subsubsection in sidebar shaded template, 72
subsubsection in toc template/color/font, 97
subsubsection in toc shaded template/color/font, 97
\subtitle , 95
\switch , 236
Szeged presentation theme, 160

t class option, 76
\tableofcontents , 101
Template inserts, see Inserts
Templates, see Beamer templates
\temporal , 83
texpower package, 19
textpos package, 19
thebibliography environment, 104
Themes, see Presentation themes
theorem environment, 121
theorem begin template, 124
theorem end template, 125
theorems parent template, 123
tiny structure font, 118
\title , 94
title page template/color/font, 94
\titlegraphic , 95
titlelike color/font, 193

\titlepage , 94
trans class option, 208
\transblindshorizontal , 145
\transblindsvertical , 145
\transboxin , 145
\transboxout , 145
\transcover , 145
\transdissolve , 145
\transduration , 146
\transfade , 145
\transfly , 146
\transglitter , 146
\transpush , 146
\transreplace , 146
\transsplithorizontalin , 146
\transsplithorizontalout , 146
\transsplitverticalin , 146
\transsplitverticalout , 146
\transwipe , 146
tree outer theme, 170
\tsection , 229
\tsectionandpart , 229

ucs class option, 17
ucs package, 20
\uncover , 83
\UntilSlide , 228
\untilSlide , 228
upper separation line foot color, 193
upper separation line head color, 193
\usebeamercolor , 189
\usebeamerfont , 201
\usebeamertemplate , 173
\usecolortheme , 148
\usefonttheme , 148
\useinnertheme , 148
\useoutertheme , 148
usepdftitle=false class option, 95
\usetheme, 148
utf8 class option, 18
utf8x class option, 18

verse color/font, 131
verse environment, 131
verse begin template, 131
verse end template, 132
\visible , 83
\vstep , 237

Warsawpresentation theme, 162
whale color theme, 186
wolverine color theme, 184

xcolor package, 20
xcolor= hlist of optionsi class option, 20
xmpmulti package, 141