OpTEX slides

Petr Olšák
petr@olsak.net

http://petr.olsak.net/optex
Basics

The the simple document looks like:

```
\slides  % style initialized
%\wideformat  % 16:9
\slideshow  % partially uncovering ideas

\sec First slide
text
\pg;

\sec Second slide
text
\pg.
```

- If `\slideshow` is missing or commented out then “partially uncovering ideas” (see later) are deactivated. It is useful for printing.
- The `\slideshow` must be the last command in the declaration part of the document.
- By default, the slides have A5 landscape format. You can declare `\wideformat`. Then the height is the same but width is 263 mm, i.e. the ratio width:height is 16:9.
Title slide

- Title of the document (used at the first slide) is created by `\tit Title` (terminated by end of line).
- The `\subtit Author name etc.` (terminated by end of line) can be used after `\tit` at the first slide.
- You can use `\n` for a new line in paragraphs or titles.

Default design

- The paragraph texts are ragged right.
- Titles, subtitles, and section titles are centered.
- The `\sec` and `\secc` are printed without numbers.
- No paragraph indentation, a little vertical space between paragraphs.
- The Heros font family (aka Helvetica) is initialized as default. Sans-serif FiraMath font for math typesetting is used.
  If `\fontfam[\langle famname\rangle]` precedes `\slides` then it rewrites default.
- The items in lists are started by a blue square (`\type X` and `\type x`).
One slide (one page)

- Top-level item list is activated by default. The asterisk * opens new item at the top-level list.
- Nested items lists (second and more level) should be created in the \begitems...\enditems environments.
- Each slide (page) must be terminated by \pg; command.
- The last slide must be terminated by \pg. command or by \bye.

\sec My ideas

* First idea
* Second idea
  \begitems
  * First sub-idea
  * Second sub-idea
  \enditems
* Final idea

- Note: each page is processed in a group, so: put your own definitions (if exist) before \slideshow or use \global assignment.
Partially uncovering ideas

- The control sequence `\pg` must be followed by:
Partially uncovering ideas

- The control sequence `\pg` must be followed by:
  - the character `;` – normal next page,
Partially uncovering ideas

- The control sequence \`\textbackslash pg` must be followed by:
  - the character ; – normal next page,
  - the character . – the end of the document,
Partially uncovering ideas

- The control sequence \( \text{\textbackslash pg} \) must be followed by:
  - the character \( ; \) – normal next page,
  - the character \( . \) – the end of the document,
  - the character \( + \) – next page keeps the same text and a next text is added (usable for partially uncovering of ideas).
Partially uncovering ideas

- The control sequence \textbackslash{pg} must be followed by:
  - the character ; – normal next page,
  - the character . – the end of the document,
  - the character + – next page keeps the same text and a next text is added (usable for partially uncovering of ideas).

- Summary:

  \textbackslash{pg}; … next page
  \textbackslash{pg}. … the end of the document
  \textbackslash{pg}+ … uncover next text on the same page
Partially uncovering ideas

- The control sequence \texttt{\textbackslash pg} must be followed by:
  - the character \texttt{;} – normal next page,
  - the character \texttt{.} – the end of the document,
  - the character \texttt{+} – next page keeps the same text and a next text is added (usable for partially uncovering of ideas).

- Summary:

  \texttt{\textbackslash pg;}  … next page
  \texttt{\textbackslash pg.}  … the end of the document
  \texttt{\textbackslash pg+}  … uncover next text on the same page

- When \texttt{\textbackslash slideshow} is not declared then \texttt{\textbackslash pg+} is deactivated.
Partially uncovering ideas

- The control sequence \texttt{\textbackslash pg} must be followed by:
  - the character \texttt{;} – normal next page,
  - the character \texttt{.} – the end of the document,
  - the character \texttt{+} – next page keeps the same text and a next text is added (usable for partially uncovering of ideas).

- Summary:

\texttt{\textbackslash pg; \ldots next page}
\texttt{\textbackslash pg. \ldots the end of the document}
\texttt{\textbackslash pg+ \ldots uncover next text on the same page}

- When \texttt{\slideshow} is not declared then \texttt{\textbackslash pg+} is deactivated.

- The \texttt{\textbackslash pg+} creates a new “virtual page”, so the current paragraph is terminated.
Example with partially uncovering ideas

The previous page was created by:

\sec Partially uncovering ideas

* The control sequence `\pg` must be followed by:`\pg+

\begitems
* the character `.` -- normal next page,`\pg+
* the character `;` -- the end of the document,`\pg+
* the character `+` -- next ... `\pg+
\enditems

* Summary
...
* When `\slideshow` is not declared
  then `\pg+` is deactivated.`\pg+
* The `\pg+` creates a new `"virtual page"`,
  so the current paragraph is terminated.
\pg;
Notes

- When \texttt{\slideshow} is active then you cannot reference sections by hyperlinked \texttt{\ref}. Only \texttt{\pgref} works (and goes to the last layer of the referenced page).

- If the text overfull the page (slide) then it follows to the next page without saying explicitly \texttt{\pg};. But \texttt{\slideshow} cannot work in this case.
Notes

- When `\slideshow` is active then you cannot reference sections by hyperlinked `\ref`. Only `\pgref` works (and goes to the last layer of the referenced page).
- If the text overfull the page (slide) then it follows to the next page without saying explicitly `\pg;`. But `\slideshow` cannot work in this case.

More about design

- You can use `\backgroundpic{<image-file>}` for putting an image to the background.
- You can re-declare `\footline` or re-define internal macros for design as you wish.
- You can put the images or text wherever using `\putimage` or `\puttext` macros...
Putting images and texts wherever

\puttext \langle right \rangle \langle up \rangle \{ \langle text \rangle \} puts a \langle text \rangle to the desired place: It moves the current point \langle right \rangle and \langle up \rangle, puts the \langle text \rangle and returns back, so the typesetting continues from previous position. The parameters \langle right \rangle and \langle up \rangle are dimensions. For example

\puttext 0mm 50mm {\Red HELLO}

prints red HELLO, as shown here.
Putting images and texts wherever

- \puttext \langle right \rangle \langle up \rangle \{ \langle text \rangle \} puts a \langle text \rangle to the desired place: It moves the current point \langle right \rangle and \langle up \rangle, puts the \langle text \rangle and returns back, so the typesetting continues from previous position. The parameters \langle right \rangle and \langle up \rangle are dimensions. For example

  \puttext 0mm 50mm \{\Red HELLO\}

prints red HELLO, as shown here.

- \putpic \langle right \rangle \langle up \rangle \langle width \rangle \langle height \rangle \{ \langle image-file \rangle \} puts the image with desired \langle width \rangle and \langle height \rangle at the position like \puttext puts the text.

- The ring above is the result of

  \putpic .8\hsize 20mm 30mm \nospec \{op-ring.png\}

used at beginning of this paragraph.
Putting images and texts wherever

- \texttt{\puttext \langle right\rangle \langle up\rangle \{\langle text\rangle\}} puts a \langle text\rangle to the desired place: It moves the current point \langle right\rangle and \langle up\rangle, puts the \langle text\rangle and returns back, so the typesetting continues from previous position. The parameters \langle right\rangle and \langle up\rangle are dimensions. For example

  \[
  \texttt{\puttext 0mm 50mm \{\textcolor{red}{HELLO}\}}
  \]

  prints red HELLO, as shown here.

- \texttt{\putpic \langle right\rangle \langle up\rangle \langle width\rangle \langle height\rangle \{\langle image-file\rangle\}} puts the image with desired \langle width\rangle and \langle height\rangle at the position like \texttt{\puttext} puts the text.

- The ring above is the result of

  \[
  \texttt{\putpic .8\hsize 20mm 30mm \nospec \{op-ring.png\}}
  \]

  used at beginning of this paragraph.

- Use \texttt{\nospec} for \langle width\rangle or \langle height\rangle of the image if you don’t want to specify both dimensions (because you don’t want to change the image aspect ratio).
Limits of the \texttt{pg+} sequence

- The \texttt{pg+} sequence (partially uncovering ideas) cannot be used inside a group.
- The exception is the nested environment \texttt{begitems...enditems}.
- The \texttt{pg+} always finalizes the current paragraph. It is impossible to hide only a part of the horizontal mode.
Limits of the `\pg+` sequence

- The `\pg+` sequence (partially uncovering ideas) cannot be used inside a group.
- The exception is the nested environment `\begitems...\enditems`.
- The `\pg+` always finalizes the current paragraph. It is impossible to hide only a part of the horizontal mode.

The `\layers...\endlayers` environment

If you really need something unsupported by `\pg+` then you can use

```
\layers ⟨number⟩
⟨layered text⟩
\endlayers
```

- The `⟨layered text⟩` should include conditions like `\ifnum\layernum...` or `\ifcase\layernum`. It is printed `⟨number⟩` times to more slide pages (layers) with the same surrounding text. See next page...
**Example of \layers environment**

The \slides style provides a shortcut \use and a macro \pshow (means partially show):

\def\use#1#2{\ifnum\layernum#1\relax#2\fi}
\def\pshow#1{\use{=#1}\Red \use{<#1}\Transparent \ignorespaces}

\use{=3}{do something} does something only if \layernum=3.

The {\pshow\langle num\rangle \langle text\rangle} prints \langle text\rangle in Red when current layer is equal to \langle num\rangle or it prints \langle text\rangle normally when current layers is greater than \langle num\rangle. The transparent (invisible) text is used in other cases.

The following dance:

First text.
Example of \texttt{\textbackslash layers} environment

The \texttt{\textbackslash slides} style provides a shortcut \texttt{\textbackslash use} and a macro \texttt{\textbackslash pshow} (means partially show):

\begin{verbatim}
def\use#1#2{\ifnum\layernum#1\relax#2\fi}
def\pshow#1{\use{=#1}\Red \use{<#1}\Transparent \ignorespaces}
\end{verbatim}

\texttt{\use{=3}{do something}} does something only if \texttt{\layernum=3}.

The \{\texttt{\textbackslash pshow\{\textit{num}\} \{\textit{text}\}}\} prints \textit{text} in Red when current layer is equal to \textit{\textit{num}} or it prints \textit{text} normally when current layers is greater than \textit{\textit{num}}. The transparent (invisible) text is used in other cases.

The following dance:

Second text. First text.
Example of \texttt{\textbackslash layers} environment

The \texttt{\textbackslash slides} style provides a shortcut \texttt{\textbackslash use} and a macro \texttt{\textbackslash pshow} (means partially show):

\begin{verbatim}
def\use#1#2{\ifnum\layernum#1\relax#2\fi}
def\pshow#1{\use{=#1}\Red \use{<#1}\Transparent \ignorespaces}
\use{=3}\{\texttt{do something}\} does something only if \texttt{\layernum=3}.
\end{verbatim}

The \texttt{\pshow\langle num\rangle \langle text\rangle} prints \langle text\rangle in Red when current layer is equal to \langle num\rangle or it prints \langle text\rangle normally when current layers is greater than \langle num\rangle. The transparent (invisible) text is used in other cases.

The following dance:
Second text. \texttt{Third text}. First text.
Example of \texttt{\textbackslash layers} environment

The \texttt{\textbackslash slides} style provides a shortcut \texttt{\textbackslash use} and a macro \texttt{\textbackslash pshow} (means partially show):

\begin{verbatim}
def\use#1#2{\ifnum\layernum#1\relax#2\fi}
def\pshow#1{\use{=#1}\textcolor{red}{\use{<#1}}\textcolor{transparent}{\textbackslash ignorespaces}}\use{=3}{\texttt{do something}}\end{verbatim}

\texttt{\use{=3}\{do something\}} does something only if \texttt{\layernum=3}.

The \texttt{\{\textbackslash pshow\langle num\rangle \langle text\rangle\}} prints \texttt{\langle text\rangle} in Red when current layer is equal to \texttt{\langle num\rangle} or it prints \texttt{\langle text\rangle} normally when current layers is greater than \texttt{\langle num\rangle}. The transparent (invisible) text is used in other cases.

The following dance:

Second text. Third text. First text.

was generated by

\begin{verbatim}
\texttt{\textbackslash layers 3 }
\texttt{\{\textbackslash pshow2 Second text.\} \{\textbackslash pshow3 Third text.\} \{\textbackslash pshow1 First text.\} \textbackslash endlayers}
\end{verbatim}
Example of \texttt{\textbackslash layers environment}

The \texttt{\textbackslash slides} style provides a shortcut \texttt{\textbackslash use} and a macro \texttt{\textbackslash pshow} (means partially show):

\begin{verbatim}
\def\use#1#2{\ifnum\layernum#1\relax#2\fi}
\def\pshow#1{\use{=#1}\textcolor{red}{\use{<#1}\textcolor{transparent}{\ignorespaces}}}
\use{=3}{do something} does something only if \texttt{\layernum}=3. The \texttt{\{\textbackslash pshow\langle num\rangle \langle text\rangle\}} prints \texttt{\langle text\rangle} in Red when current layer is equal to \texttt{\langle num\rangle} or it prints \texttt{\langle text\rangle} normally when current layers is greater than \texttt{\langle num\rangle}. The transparent (invisible) text is used in other cases.

The following dance:
Second text. Third text. First text.

was generated by

\begin{verbatim}
\texttt{\textbackslash layers 3}
\{\texttt{\textbackslash pshow2 Second text.}\} \{\texttt{\textbackslash pshow3 Third text.}\} \{\texttt{\textbackslash pshow1 First text.}\}
\texttt{\textbackslash endlayers}
\end{verbatim}

- The \langle layered text \rangle is treated as a macro parameter. So, you cannot use verbatim nor \texttt{\textbackslash sec} titles here. Maximal one \texttt{\textbackslash layers} environment can be per one page (terminated by \texttt{\textbackslash pg+} or \texttt{\textbackslash pg;} or \texttt{\textbackslash pg}.  


Comparison OpTeX slides with Beamer\footnote{http://www.ctan.org/pkg/beamer}

The LaTeX package \texttt{Beamer} gives much more features and many themes are prepared for Beamer, \textit{but}
Comparison \LaTeX\ slides with Beamer¹

The \LaTeX\ package \textbf{Beamer} gives much more features and many themes are prepared for Beamer, \textbf{but}

- the user of Beamer is forced to \textit{program} his/her document using dozens of \verb|\begin{foo}| and \verb|\end{foo}| and many another programming constructions,

¹ \url{http://www.ctan.org/pkg/beamer}
Comparison OpTEX slides with Beamer

The \LaTeX package Beamer gives much more features and many themes are prepared for Beamer, but

- the user of Beamer is forced to program his/her document using dozens of \begin{foo} and \end{foo} and many another programming constructions,
- plain \TeX gives you a possibility to simply write your document with minimal markup. The result is more compact. You can concentrate on the contents of your document, not on the programming syntax.

\footnote{http://www.ctan.org/pkg/beamer}
Comparison OpT\hbox{\TeX} slides with Beamer\footnote{http://www.ctan.org/pkg/beamer}

The \LaTeX{} package \texttt{Beamer} gives much more features and many themes are prepared for Beamer, \textbf{but}

\begin{itemize}
  \item the user of Beamer is forced to \textit{program} his/her document using dozens of \texttt{\begin{foo}} and \texttt{\end{foo}} and many another programming constructions,
  \item plain \TeX gives you a possibility to simply \textit{write} your document with minimal markup. The result is more compact. You can concentrate on the contents of your document, not on the programming syntax.
  \item User needs to read 250 pages of doc for understanding Beamer,
\end{itemize}
Comparison \LaTeX\ slides with Beamer$^1$

The \LaTeX\ package \textbf{Beamer} gives much more features and many themes are prepared for Beamer, but

- the user of Beamer is forced to \textit{program} his/her document using dozens of \texttt{\begin{foo}} and \texttt{\end{foo}} and many another programming constructions,
- \LaTeX\ gives you a possibility to simply \textit{write} your document with minimal markup. The result is more compact. You can concentrate on the contents of your document, not on the programming syntax.
- User needs to read 250 pages of doc for understanding Beamer,
- on the other hand, you need to read only ten slides$^2$ and you are ready to use \textbf{\LaTeX\ slides}.

$^1$ \url{http://www.ctan.org/pkg/beamer}

$^2$ this eleventh slide isn’t counted
Thanks for your attention
Thanks for your attention

Questions?