

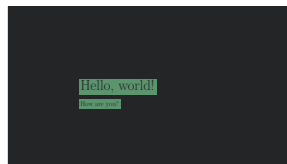
ppt-slides: L^AT_EX Package for Slide Decks à la PowerPoint*

Yegor Bugayenko
yegor256@gmail.com

2026-10-14, 0.8.2

1 Introduction

This package renders slide decks in the PowerPoint™ style:



```
1 \documentclass{article}
2 \usepackage[template,scheme=dark]{ppt-slides}
3 \begin{document}
4 \begin{pptMiddle}
5 \pptTitle{Hello, world!}{How are you?}
6 \end{pptMiddle}
7 \end{document}
```

It is recommended that this package be employed in conjunction with [clicks](#):



```
1 \documentclass{article}
2 \usepackage[template,scheme=dark]{ppt-slides}
3 \usepackage[static]{clicks}
4 \begin{document}
5 Slide decks may be composed in \LaTeX:
6 \click{\pptBanner{Making Slides Is Easy}}
7 \click{Simply employ this package\ldots}
8 \plush{together with the ‘clicks’ package.}
9 The result resembles what PowerPoint
10 produces, yet with the precision of \LaTeX.
11 Use in concert with ‘clicks’ is recommended.
12 \end{document}
```

2 Layout

`\pptToc` First, the narrative is divided into chapters:
`\pptChapter`

```
\documentclass{article}
\usepackage{clicks}
\usepackage[template,scheme=light]{ppt-slides}
\begin{document}
```

*The sources are in GitHub at [yegor256/ppt-slides](https://github.com/yegor256/ppt-slides)

```

\pptToc
\plush{\pptChapter{About Me}}
...
\plush{\pptChapter[Idea]{My Idea Is Novel}}
...
\plush{\pptChapter[FAQ]{Discussion \& Questions}}
...
\end{document}

```

`\pptToc` renders the table of contents in an interactive “clickable” format. Through the use of [crumbs](#), a navigation appears at the upper left corner. An optional argument, as in `\pptToc[Agenda]`, prints a title above the table.

`\pptChapterLabel` Each chapter slide is labelled with the word “Chapter” and its number. The word may be replaced by redefining `\pptChapterLabel`:

```

\renewcommand\pptChapterLabel{Part}

```

`\pptSection` Sections are then placed within chapters:

```

\begin{document}
\pptToc
\plush{\pptChapter{About Me}}
\plush{\pptSection[Student]{I’m a Student}}
...
\plush{\pptSection[Athlete]{Also, I’m an Athlete}}
...
\plush{\pptChapter[Idea]{My Idea Is Novel}}
\plush{\pptSection{Novelty}}
\plush{\pptSection{Impact}}
\end{document}

```

The `\pptChapter` and `\pptSection` commands employed in tandem render an elegant two-level navigation menu at the upper left corner.

`\pptLeft` The title of the presentation and the author’s name may be placed at the foot
`\pptRight` of each slide, on the left and on the right respectively:

```

\documentclass{article}
\usepackage{clicks}
\usepackage[template,scheme=light]{ppt-slides}
\pptLeft{How Did I Spend Last Summer}
\pptRight{Yegor Bugayenko}
\begin{document}
...
\end{document}

```

Should neither the `\pptLeft` nor the `\pptRight` command be used, nothing is printed at the foot.

3 Commands

`\pptPic` An image may be added to a slide (the first argument denotes the width of the image relative to `\textwidth`, while the second supplies its path):



```

1 \documentclass{article}
2 \usepackage[template,scheme=dark]{ppt-slides}
3 \begin{document}
4 \pptPic{0.2}{socrates.jpg} \\
5 This is Socrates, a Greek philosopher.
6 \end{document}

```

`\pptHeader` A header may be printed above a code fragment on a slide (the use of `\ffcode` is recommended):



```

1 \documentclass{article}
2 \usepackage[template,scheme=dark]{ppt-slides}
3 \usepackage{ffcode}
4 \begin{document}
5 \pptHeader{This is How You Print to Console:}
6 \begin{ffcode}
7 void main(char** args) {
8     printf("Hello, world!");
9 }
10 \end{ffcode}
11 \end{document}

```

`\plick` or `\plush` should not be used here, as they do not function with code snippets. The `\pptHeader` command prints a header similar to that produced by the `\pptSection` command, yet without commencing a new section.

`\pptQuote` A quotation may be placed beside an image by means of the `\pptQuote` command:
`\pptBanner`



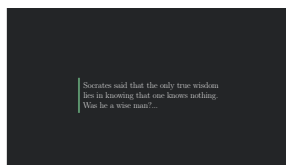
```

1 \documentclass{article}
2 \usepackage[template,scheme=dark]{ppt-slides}
3 \begin{document}
4 \pptBanner{A Cherished Quote}
5 \pptQuote{socrates.jpg}{The only true wisdom
6 lies in knowing that one knows nothing.}{Socrates}
7 \end{document}

```

The background colour of `\pptBanner` is green by default and may be changed by an optional argument, as in `\pptBanner[orange]{Hello}`.

`\pptThought` For a noteworthy thought to be placed at the centre of the slide, `\pptThought` is employed:



```

1 \documentclass{article}
2 \usepackage[template,scheme=dark]{ppt-slides}
3 \begin{document}
4 \pptThought{Socrates said that the only true
5 wisdom lies in knowing that one knows nothing.
6 Was he a wise man?...}
7 \end{document}

```

`\pptMiddle (env.)` Content may be positioned vertically at the centre of the page by means of the `\pptMiddle` environment:

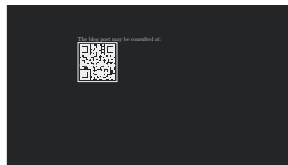


```

1 \documentclass{article}
2 \usepackage[template,scheme=dark]{ppt-slides}
3 \begin{document}
4 \begin{pptMiddle}
5 ‘‘One’s time is limited; it ought not be
6 wasted living someone else’s life.’’ --- Steve Jobs
7 \end{pptMiddle}
8 \end{document}

```

`\pptQR` On occasion, it is more convenient to present a QR code to the audience in lieu of a URL, since the former is more readily consulted—it may simply be scanned:



```

1 \documentclass{article}
2 \usepackage[template,scheme=dark]{ppt-slides}
3 \begin{document}
4 The blog post may be consulted at: \
5 \pptQR{https://www.yegor256.com}
6 \end{document}

```

The code is rendered thus by virtue of the [qr](#) package employed internally. Its height is two inches by default and may be changed by an optional argument, as in `\pptQR[1in]{...}`.

`\pptPin` A small block may be pinned to the upper corner of a slide, outside of the text flow, by means of `\pptPin`; the optional argument is either `right` (the default) or `left`. The `\pptPinQR` command pins a QR code together with its URL in the same way:

```

\pptPin{Remember this!}
\pptPinQR[left]{https://www.yegor256.com}

```

`\pptSnippet` The content of a file may be printed verbatim by means of `\pptSnippet`; the optional argument sets the font size, which is `\small` by default:

```

\pptSnippet[\footnotesize]{src/main.c}

```

`\pptWide` (*env.*) Slide content occasionally needs to occupy the entire visible horizontal width:
`\pptWideOne` (*env.*)



```

1 \documentclass{article}
2 \usepackage[template,scheme=dark]{ppt-slides}
3 \begin{document}
4 \begin{pptWideOne}
5 This paragraph happens to be unusually long,
6 and must therefore occupy the entire visible
7 horizontal width, although the content
8 ordinarily bears a generous left margin.
9 \end{pptWideOne}
10 \end{document}

```

The form `\begin{pptWide}{X}` is likewise admissible, where `X` denotes the number of columns to be rendered.

4 Package Options

`template` Each slide deck begins with a template:

```

\documentclass{article}
\usepackage[template=9x6]{ppt-slides}
\begin{document}
...
\end{document}

```

Only one template is supplied and is used by default: **9x6**. Should the name be omitted, this template is selected. Should the **template** option itself be omitted entirely, the default **article** is rendered, which is rarely the desired outcome.

scheme A colour scheme for the slides may be selected by means of the **scheme** option of the package:

```
\usepackage[template,scheme=light]{ppt-slides}
```

Several schemes are available out of the box: **light**, **dark**, **light-mono**, and **dark-mono**. Should the **scheme** option be omitted while a template is in use, the **light** scheme is selected. A bespoke scheme may also be devised, taking the **ppt-light.tex** file as a model:

```
\usepackage[template,scheme=/usr/local/my-colors.tex]{ppt-slides}
```

directory The location of the templates and schemes may be altered by means of the **directory** package option, with or without a trailing slash:

```
\usepackage[directory=/tmp,template=foo]{ppt-slides}
```

nominutes By default, when the slide deck is rendered in non-static mode (the **static** option of the “clicks” package being absent), a minute tracker appears at the upper right corner of each slide. This may be disabled by means of the **nominutes** option of the package:

```
\usepackage[nominutes]{ppt-slides}
```

nocrumbs By default, the upper left corner of each slide bears so-called crumbs. These may be removed by means of the **nocrumbs** package option:

```
\usepackage[nocrumbs]{ppt-slides}
```

5 Implementation

First, we parse the package options:

```

1 \RequirePackage{pgfopts}
2 \makeatletter
3 \pgfkeys{
4   /ppt-slides/.is family,
5   /ppt-slides,
6   template/.estore in = \ppt@template,
7   template/.default = 9x6,
8   scheme/.estore in = \ppt@scheme,
9   directory/.estore in = \ppt@directory,
10  directory/.default = ,
11  nominutes/.estore in = \ppt@nominutes,
12  nocrumbs/.estore in = \ppt@nocrumbs,
13  directory
14 }
15 \ProcessPgfPackageOptions{/ppt-slides}
16 \makeatother

```

Then, we append a slash to the `directory`, unless it is empty or already ends with one:

```

17 \ExplSyntaxOn
18 \makeatletter
19 \tl_if_empty:NF \ppt@directory {
20   \str_if_eq:eeF { \str_item:Nn \ppt@directory { -1 } } { / } {
21     \tl_put_right:Nn \ppt@directory { / }
22   }
23 }
24 \makeatother
25 \ExplSyntaxOff

```

Then, we ask [crumbs](#), which the template may load first, to increment the section and subsection counters, since chapter numbers and the crumbs bar rely on them:

```

26 \PassOptionsToPackage{increment}{crumbs}

```

Then, we include the template:

```

27 \makeatletter\ifdefined\ppt@template
28   \input{\ppt@directory ppt-templates/ppt-\ppt@template.tex}
29 \else
30   \message{^^Jppt: No template is loaded, because
31     the 'template' option is not specified}
32 \fi\makeatother

```

Then, we include the scheme, either a built-in one by its name or a custom one by the path to its file. The template relies on the colors that schemes define, so the `light` scheme is used when a template is loaded without a scheme:

```

33 \makeatletter\ifdefined\ppt@template\ifdefined\ppt@scheme\else
34   \def\ppt@scheme{light}
35 \fi\fi\makeatother
36 \makeatletter\ifdefined\ppt@scheme
37   \RequirePackage{xcolor}
38   \IfFileExists{\ppt@directory ppt-schemes/ppt-\ppt@scheme.tex}
39     {\input{\ppt@directory ppt-schemes/ppt-\ppt@scheme.tex}}
40     {\input{\ppt@scheme}}
41 \else
42   \message{^^Jppt: No color scheme is loaded, because
43     the 'scheme' option is not specified}
44 \fi\makeatother

```

Then, we include [ifthen](#) in order to enable `if/then/else` commands:

```

45 \RequirePackage{ifthen}

```

Then, we include [href-ul](#) in order to enable underlined hyperlinks:

```

46 \RequirePackage{href-ul}

```

Then, we include [pagecolor](#) in order to make it possible to change the background color of a page:

```

47 \RequirePackage{pagecolor}

```

Then, we include [varwidth](#) in order to change the width of the page, inside `pptWide`:

```

48 \RequirePackage{varwidth}

```

Then, we include [csquotes](#) in order to use double quotes:

```

49 \RequirePackage{csquotes}

```

Then, we include `qrcode` in order to render QR codes:

```
50 \RequirePackage{qrcode}
```

Then, we include `tikz` in order to position elements on the page, especially in the Table of Contents:

```
51 \RequirePackage{tikz}
```

Then, we include `tikzpagenodes` in order to pin a block in `\pptPin`:

```
52 \RequirePackage{tikzpagenodes}
```

Then, we include `enumitem` in order to render inline lists in crumbs:

```
53 \PassOptionsToPackage{inline}{enumitem}
```

```
54 \RequirePackage{enumitem}
```

Then, we include `crumbs` in order to show top-page menus:

```
55 \RequirePackage{crumbs}
```

Then, we include `calc` in order to calculate positions of elements more precisely:

```
56 \RequirePackage{calc}
```

Then, we include `tabularx` in order to render `\pptQuote`:

```
57 \RequirePackage{tabularx}
```

Then, we include `seqsplit` in order to split long links:

```
58 \RequirePackage{seqsplit}
```

Then, we include `changepage` and `multicol` in order to render `pptWideOne` and `pptWide`:

```
59 \RequirePackage{changepage}
```

```
60 \RequirePackage{multicol}
```

Then, we include `verbatim` in order to render `\pptSnippet`:

```
61 \RequirePackage{verbatim}
```

`pptMiddle` (*env.*) Then, we define the `pptMiddle` environment:

```
62 \newenvironment{pptMiddle}
```

```
63 {\vspace*{\fill}}
```

```
64 {\vspace*{\fill}}
```

`\pptBanner` Then, we define the `\pptBanner` command:

```
65 \newcommand{\pptBanner[2][green]{%
```

```
66 \par%
```

```
67 {\setlength{\fboxsep}{6pt}}%
```

```
68 \colorbox{#1}{\color{\thepagecolor}\large #2}}%
```

```
69 \par%
```

```
70 }
```

Then, when `biblatex` is loaded, we declare a bibliography heading that uses `\pptBanner` instead of `\section*`. The `crumbs` package redefines `\section` in a way that does not understand its starred form, so `biblatex`'s default heading (which issues `\section*{\refname}`) would otherwise render as a bogus numbered section. This also makes the references slide visually consistent with other `\pptBanner` headings:

```
71 \AtBeginDocument{%
```

```
72 \ifdefined\printbibliography%
```

```

73     \defbibheading{bibliography}{\refname}{\pptBanner{#1}}%
74     \fi%
75 }

```

`\pptChapter` Then, we define the `\pptChapter` and `\pptChapterLabel` commands:

```

76 \newcommand\pptChapterLabel{Chapter}
77 \newcommand\pptChapter[2] [] {%
78   \crumbection{#1}{#2}%
79   \begin{pptMiddle}%
80     {\large\ttfamily \pptChapterLabel{} \#\the\value{section}:}%
81     \newline%
82     \rotatebox{1}{\pptBanner[blue]{\LARGE#2}}%
83   \end{pptMiddle}%
84 }

```

`\pptSection` Then, we define the `\pptSection` command:

```

85 \newcommand\pptSection[2] [] {%
86   \subcrumbection{#1}{#2}%
87   \pptHeader{#2}%
88 }

```

`\pptHeader` Then, we define the `\pptHeader` command:

```

89 \newcommand\pptHeader[1] {%
90   \vspace*{-0.75in}\hspace*{-0.5in}%
91   \rotatebox{1}{\pptBanner[orange]{\Large #1}}\vspace*{0.15in}%
92   \par%
93 }

```

`\pptTitle` Then, we define the `\pptTitle` command, which prints the `title` and `subtitle` on the first page. The first argument is the title, the second is the subtitle. If the second argument is empty, it won't be printed:

```

94 \makeatletter
95 \newcommand\pptTitle[2] {%
96   {\pptBanner{\Huge #1}}%
97   \def\ppt@param{#2}%
98   \ifx\ppt@param\empty\else%
99     {\pptBanner{\large #2}}%
100   \fi%
101 }
102 \makeatother

```

`\pptToc` Then, we define the `\pptToc` command. With [clicks](#), the table is handed to `\print`, which defers it until `\flush`. Without it, the table is typeset once the list of chapters is collected:

```

103 \newcommand*\pptTOC{}
104 \makeatletter
105 \newcommand\pptToc[1] [] {%
106   \renewcommand*\pptTOC{}%
107   \def\ppt@toc{%
108     \def\ppt@param{#1}%
109     \ifx\ppt@param\empty\else%
110       \pptBanner{#1}%
111     \fi%

```

```

112 \tikz{%
113 \node (z) {%
114 \begin{varwidth}{\textwidth}%
115 \ifx\pptTOC\empty\else%
116 \begin{itemize}[label={}]
117 \pptTOC
118 \end{itemize}
119 \fi%
120 \end{varwidth}%
121 };%
122 \path [draw=blue,line width=4pt] (z.north west) -- (z.south west);%
123 }%
124 }%
125 \ifdefined\print\print{\ppt@toc}\fi%
126 \begin{@empty}%
127 \renewcommand\crumb[2]{%
128 \ifx\pptTOC\empty\else\ifdefined\click\click[0]\fi\fi
129 \gappto\pptTOC{\item ##2}
130 }
131 \crumbs
132 \end{@empty}%
133 \ifdefined\print\else\ppt@toc\par\fi%
134 \ifdefined\flush\flush[1]\fi%
135 }
136 \makeatother

```

`\pptQuote` Then, we define the `\pptQuote` command, with one optional and three mandatory arguments:

```

137 \makeatletter
138 \newcommand\pptQuote[4][]{%
139 \begin{tabularx}{\columnwidth}{c>{\raggedright\arraybackslash}X}%
140 \raisebox{\dimexpr-\height+\ht\strutbox}{%
141 \parbox{0.25\textwidth+2pt}{%
142 \raggedright%
143 \pptPic{0.25}{#2}%
144 \def\ppt@person{#1}%
145 \ifx\ppt@person\empty\else%
146 \par%
147 \small%
148 \ppt@person%
149 \fi%
150 }%
151 }%
152 &%
153 \enquote{#3{}}}%
154 \def\ppt@param{#4}%
155 \ifx\ppt@param\empty\else%
156 \par\vspace*{1em}%
157 \small%
158 \textendash{} \ppt@param%
159 \par%
160 \fi%
161 \\%
162 \end{tabularx}%

```

```

163 }
164 \makeatother

```

\pptPic Then, we define the \pptPic command:

```

165 \newcommand\pptPic[2]{%
166   {%
167     \setlength{\fboxsep}{0pt}%
168     \setlength{\fboxrule}{1pt}%
169     \fcolorbox{gray}{white}{%
170       \includegraphics[width=#1\columnwidth]{#2}%
171     }%
172   }%
173 }

```

\pptPin Then, we define the \pptPin command:

```

174 \newcommand\pptPin[2][right]{%
175   \begin{tikzpicture}[remember picture,overlay]%
176     \ifthenelse{\equal{#1}{left}}{%
177       {%
178         \node[anchor=north east] at
179           ($(\current page text area.north west) - (24pt, 18pt)$) {%
180           \begin{minipage}{0.25\textwidth}%
181             \raggedleft #2%
182           \end{minipage}%
183         };%
184       }%
185     }{%
186       \node[anchor=north east] at (\current page text area.north east) {%
187       \begin{minipage}{0.25\textwidth}%
188         \raggedright #2%
189       \end{minipage}%
190     };%
191   }%
192   \end{tikzpicture}%
193 }

```

\pptThought Then, we define the \pptThought command:

```

194 \newcommand\pptThought[1]{%
195   \begin{pptMiddle}%
196     \tikz{%
197       \node [inner sep=18pt] (z) {%
198         \begin{varwidth}{0.8\textwidth}%
199           \raggedright\Large #1%
200         \end{varwidth}%
201       };
202       \path [draw=green,line width=8pt] (z.north west) -- (z.south west);
203     }%
204   \end{pptMiddle}%
205 }

```

\pptPunch Then, we define the \pptPunch command, which places its block on the grid that only the template sets up, so it refuses to work without one:

```

206 \makeatletter
207 \newcommand\pptPunch[1]{%

```

```

208 \ifdefined\ppt@template%
209   \begin{textblock}[16][0.5,0.5](8,8)\leavevmode\color{.}%
210     \centering #1%
211   \end{textblock}%
212 \else%
213   \PackageError{ppt-slides}{The \string\pptPunch\space command
214     requires the 'template' package option}{}%
215   \fi%
216 }
217 \makeatother

```

`\pptSnippet` Then, we define the `\pptSnippet` command:

```

218 \newcommand\pptSnippet[2][\small]{%
219   \begin{samepage}%
220     #1\verbatiminput{#2}%
221   \end{samepage}%
222 }

```

`\pptQR` Then, we define the `\pptQR` command:

```

223 \newcommand\pptQR[2][2in]{%
224   \tikz{%
225     \node[draw=white]{%
226       \href{#2}{%
227         \qr{height=#1}{#2}%
228       }%
229     }%
230   }%
231 }

```

`\pptPinQR` Then, we define the `\pptPinQR` command:

```

232 \newcommand\pptPinQR[2][right]{%
233   \pptPin[#1]{%
234     \begin{minipage}{1.4in}%
235       \raggedleft%
236       \pptQR[1in]{#2}%
237       \\\[3pt]
238       \ttfamily\fontsize{8pt}{10pt}\selectfont%
239       \seqsplit{#2} \href{#2}{\${\rightarrow}}%
240     \\\
241     \end{minipage}%
242   }%
243 }

```

`pptWideOne` (*env.*) Finally, we define the `pptWide` and `pptWideOne` environments:

```

pptWide (env.) 244 \newenvironment{pptWideOne}
245   {\begin{adjustwidth}{-2in}{-1in}}
246   {\end{adjustwidth}}
247 \newenvironment{pptWide}[1]
248   {\begin{pptWideOne}\begin{multicols}{#1}}
249   {\end{multicols}\end{pptWideOne}}

250 \endinput

```

Change History

0.0.1	specified.	9
General: First draft.	<code>\pptThought</code> : The <code>\pptThought</code>	
0.1.4	command aligns text to the left.	10
General: The <code>nominutes</code> package	0.3.1	
option added.	General: The <code>calc</code> package added.	7
0.1.5	0.4.0	
General: The default directory	<code>\pptChapter</code> : The	
fixed.	<code>\pptChapterLabel</code> command	
0.2.0	introduced.	8
General: We migrated to the	0.5.0	
<code>ltxdoc</code> format and <code>.dtx</code> file.	General: The <code>nocrumbs</code> package	
0.2.1	option was introduced.	5
<code>pptMiddle</code> : The <code>\pptMiddle</code>	0.7.0	
command turned into	<code>\pptPunch</code> : The <code>\pptPunch</code>	
<code>pptMiddle</code> environment.	command places content right	
0.3.0	in the middle of the slide, both	
<code>\pptQuote</code> : The <code>\pptQuote</code>	vertically and horizontally.	10
command has got an optional	0.8.0	
parameter, where the name of	General: A <code>biblatex</code> -compatible	
the person on the photo can be	bibliography heading defined.	7

Index

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

Symbols	F	95, 103, 105, 138, 165, 174, 194, 207, 218, 223, 232
\# 80	\fboxrule 168	
A	\fboxsep 67, 167	
\arraybackslash ... 139	\fcolorbox 169	\newenvironment 62, 244, 247
\AtBeginDocument ... 71	\fill 63, 64	\newline 81
B	\flush 134	\nocrums 5
\begin 79, 114, 116, 126, 139, 175, 180, 187, 195, 198, 209, 219, 234, 245, 248	\fontsize 238	\node 113, 178, 186, 197, 225
C	G	\nominutes 5
\centering 210	\gappto 129	
\click 128	H	
\color 68, 209	\height 140	\PackageError 213
\colorbox 68	\href 226, 239	\par 66, 69, 92, 133, 146, 156, 159
\columnwidth .. 139, 170	\hspace 90	\parbox 141
\crumb 127	\ht 140	\PassOptionsToPackage 26, 53
\crumblection 78	\Huge 96	\path 122, 202
\crumbs 131	I	\pgfkeys 3
D	\ifdefined 27, 33, 36, 72, 125, 128, 133, 134, 208	\ppt@directory .. 9, 19, 20, 21, 28, 38, 39
\def 34, 97, 107, 108, 144, 154	\IfFileExists 38	\ppt@nocrums 12
\defbibheading 73	\ifthenelse 176	\ppt@nominutes 11
\dimexpr 140	\ifx 98, 109, 115, 128, 145, 155	\ppt@param 97, 98, 108, 109, 154, 155, 158
\directory 5	\includegraphics .. 170	\ppt@person 144, 145, 148
E	\input 28, 39, 40	\ppt@scheme 8, 33, 34, 36, 38, 39, 40
\empty 98, 109, 115, 128, 145, 155	\item 129	\ppt@template 6, 27, 28, 33, 208
\end 83, 118, 120, 132, 162, 182, 189, 192, 200, 204, 211, 221, 241, 246, 249	L	\ppt@toc .. 107, 125, 133
\endinput 250	\LARGE 82	\pptBanner . 3, 65, 73, 82, 91, 96, 99, 110
\enquote 153	\Large 91, 199	\pptChapter 1, 76
environments:	\large 68, 80, 99	\pptChapterLabel 2, 76, 80
pptMiddle 3, 62	\leavevmode 209	\pptHeader 3, 87, 89
pptWide 4, 244	M	\pptLeft 2
pptWideOne ... 4, 244	\makeatletter 2, 18, 27, 33, 36, 94, 104, 137, 206	pptMiddle (env.) .. 3, 62
\equal 176	\makeatother ... 16, 24, 32, 35, 44, 102, 136, 164, 217	\pptPic 2, 143, 165
\ExplSyntaxOff 25	\message 30, 42	\pptPin 4, 174, 233
\ExplSyntaxOn 17	N	\pptPinQR 4, 232
	\newcommand 65, 76, 77, 85, 89,	\pptPunch 206
		\pptQR 4, 223, 236
		\pptQuote 3, 137
		\pptRight 2
		\pptSection 2, 85

