

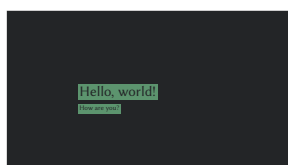
ppt-slides: L^AT_EX Package for Slide Decks à la Power Point*

Yegor Bugayenko
yegor256@gmail.com

2023-11-08, 0.3.0

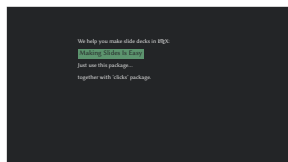
1 Introduction

This package helps you render slide decks in 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 to use this package together with [clicks](#):



```
1 \documentclass{article}
2 \usepackage[template,scheme=dark]{ppt-slides}
3 \usepackage[static]{clicks}
4 \begin{document}
5 We help you make slide decks in \LaTeX:
6 \click{\pptBanner{Making Slides Is Easy}}
7 \click{Just use this package...}
8 \plush{together with 'clicks' package.}
9 They will look similar to what PowerPoint can
10 make, but with the precision of \LaTeX. We
11 recommend using us together with ‘clicks.’
12 \end{document}
```

2 Layout

`\pptToc` First, split your story 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 will render the table of contents in an interactive “clickable” format. Thanks to the use of **crumbs**, there will be a navigation at the top left corner.

\pptSection Then, put sections inside 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 used together will render nice two-level navigation menu at the top left corner.

\pptLeft You can place the title of the presentation and your name at the bottom of each slide,
 \pptRight 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}

```

If you don’t use the \pptLeft or \pptRight commands, nothing will be printed at the bottom.

3 Commands

\pptPic You can add an image to the slide (the first argument is the width of the image in relation to \textwidth, while the second one is the path of it):



```

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}

```

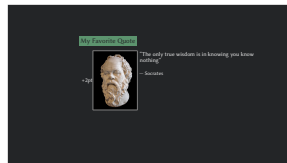
`\pptHeader` You can add a piece of code to the slide (we recommend using [ffcode](#)):



```
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}
```

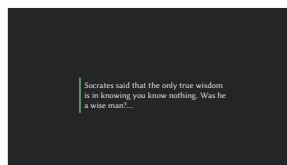
Don't use `\plick` or `\plush`, they won't work with code snippets. The `\pptHeader` command prints a header similar to what the `\pptSection` command prints, but doesn't start a new section.

`\pptQuote` You can put a quote next to the image, by using the `\pptQuote` command:
`\pptBanner`



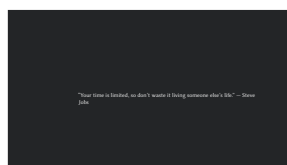
```
1 \documentclass{article}
2 \usepackage[template,scheme=dark]{ppt-slides}
3 \begin{document}
4 \pptBanner{My Favorite Quote}
5 \pptQuote{socrates.jpg}{The only true wisdom
6 is in knowing you know nothing.}{Socrates}
7 \end{document}
```

`\pptThought` For an interesting thought in the middle of the slide, use `\pptThought`:



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

`pptMiddle` You can put your content vertically in the middle of the page, using the `pptMiddle` environment:



```
1 \documentclass{article}
2 \usepackage[template,scheme=dark]{ppt-slides}
3 \begin{document}
4 \begin{pptMiddle}
5 ‘‘Your time is limited, so don't waste it
6 living someone else's life.’’ --- Steve Jobs
7 \end{pptMiddle}
8 \end{document}
```

`\pptQR` Sometimes it's convenient to show a QR code to your audience instead of a URL, since it's easier to use — they can scan it:

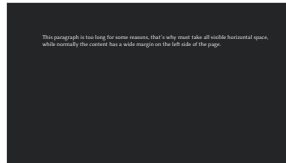


```

1 \documentclass{article}
2 \usepackage[template,scheme=dark]{ppt-slides}
3 \begin{document}
4 Check my blog post by this link: \\
5 \pptQR{https://www.yegor256.com}
6 \end{document}

```

pptWide
pptWideOne



```

1 \documentclass{article}
2 \usepackage[template,scheme=dark]{ppt-slides}
3 \begin{document}
4 \begin{pptWideOne}
5 This paragraph is too long for some reasons,
6 that's why must take all visible horizontal
7 space, while normally the content has a wide
8 margin on the left side of the page.
9 \end{pptWideOne}
10 \end{document}

```

You can also use `\begin{pptWide}{X}`, where `X` is the number of columns to render.

4 Package Options

template You start with a template for your slide deck:

```

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

```

There is only one template, which is used by default: `9x6`. If you don't specify the name, it will be used. If you don't use the `template` option at all, the default `article` will be rendered, which is not what you want.

scheme You can choose a color scheme for your slides, using the `scheme` option of the package:

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

There are a few out-of-the-box schemes available: `light`, `dark`, `light-mono`, and `dark-mono`. You can design your own, using `ppt-light.tex` file as an example:

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

directory You can change the location of the templates and schemes by means of the `directory` package option:

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

nominutes By default, if you render your slide deck in non-static mode (the `static` option for the “clicks” package), there will be minutes tracking in the right top corner of each slide. You can turn this off by using the `nominutes` option of the package:

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

5 Implementation

First, we parse the package options:

```
1 \RequirePackage{pgfplots}
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  nomminutes/.estore in = \ppt@nomminutes,
12  directory
13 }
14 \ProcessPgfPackageOptions{/ppt-slides}
15 \makeatother
```

Then, we include the template:

```
16 \makeatletter\ifdefined\ppt@template
17   \input{\ppt@directory ppt-templates/ppt-\ppt@template.tex}
18 \else
19   \message{^^Jppt: No template is loaded, because
20     the 'template' option is not specified}
21 \fi\makeatother
```

Then, we include the scheme:

```
22 \makeatletter\ifdefined\ppt@scheme
23   \RequirePackage{xcolor}
24   \input{\ppt@directory ppt-schemes/ppt-\ppt@scheme.tex}
25 \else
26   \message{^^Jppt: No color scheme is loaded, because
27     the 'scheme' option is not specified}
28 \fi\makeatother
```

Then, we include a few packages:

```
29 \RequirePackage{ifthen} % for if/then/else in LaTeX
30 \RequirePackage{href-ul} % for nice hyperlinks
31 \RequirePackage{pagecolor} % to change page color
32 \RequirePackage{varwidth} % for pptWide
33 \RequirePackage{qrcode} % to print QR codes
34 \RequirePackage{tikz} % for drawings in TOC
35 \usetikzlibrary{calc} % for positioning of pinned boxes
36 \RequirePackage{tikzpagenodes} % for pptPin
37 \RequirePackage[inline]{enumitem} % for lists in crumbs
38 \RequirePackage{crumbs} % for crumbs
```

`\pptMiddle` Then, we define the `\pptMiddle` environment:

```
39 \newenvironment{pptMiddle}
40   {\vspace*{\fill}}
41   {\vspace*{\fill}}
```

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

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

```

43 \par%
44 {\setlength{\fboxsep}{6pt}%
45 \colorbox{#1}{\color{\thepagecolor}\large #2}}%
46 \par%
47 }

```

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

```

48 \newcommand\pptChapter[2] [] {%
49 \crumbection{#1}{#2}%
50 \begin{pptMiddle}%
51 {\large\ttfamily Chapter \#\the\value{section}:}%
52 \newline%
53 \rotatebox{1}{\pptBanner[blue]{\LARGE#2}}%
54 \end{pptMiddle}%
55 }

```

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

```

56 \newcommand\pptSection[2] [] {%
57 \subcrumbection{#1}{#2}%
58 \pptHeader{#2}%
59 }

```

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

```

60 \newcommand\pptHeader[1] {%
61 \vspace*{-0.75in}\hspace*{-0.5in}%
62 \rotatebox{1}{\pptBanner[orange]{\Large #1}}\vspace*{0.15in}%
63 \par%
64 }

```

`\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:

```

65 \newcommand\pptTitle[2] {%
66 {\pptBanner{\Huge #1}}%
67 \def\param{#2}%
68 \ifx\param\empty\else%
69 {\pptBanner{\large #2}}%
70 \fi%
71 }

```

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

```

72 \newcommand*\pptTOC{}
73 \newcommand\pptToc[1] [] {%
74 \renewcommand*\pptTOC{}%
75 \print{%
76 \def\param{#1}%
77 \ifx\param\empty\else%
78 \pptBanner{#1}%
79 \fi%
80 \tikz{%
81 \node (z) {%
82 \begin{varwidth}{\textwidth}%
83 \ifx\pptTOC\empty\else%

```

```

84         \begin{itemize}[label={}]
85         \pptTOC
86         \end{itemize}
87     \fi%
88 \end{varwidth}%
89 };%
90 \path [draw=blue,line width=4pt] (z.north west) -- (z.south west);%
91 }%
92 }%
93 \begin{@empty}%
94     \renewcommand\crumb[2]{%
95         \ifx\pptTOC\empty\else\ifdefined\click\click[0]\fi\fi
96         \gappto\pptTOC{\item ##2}
97     }
98     \crumbs
99 \end{@empty}%
100 \ifdefined\flush\flush[1]\fi%
101 }

```

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

```

102 \RequirePackage{tabularx}
103 \newcommand\pptQuote[4][{}]{%
104     \begin{tabularx}{\columnwidth}{c>\raggedright\arraybackslashX}%
105         \raisebox{\dimexpr-\height+\ht\strutbox}{
106             \parbox{0.25\textwidth+2pt}{%
107                 \raggedright%
108                 \pptPic{0.25}{#2}%
109                 \def\person{#1}%
110                 \ifx\person\empty\else%
111                     \par%
112                     \small%
113                     \person%
114                 \fi%
115             }%
116         }%
117         &%
118         ‘‘#3’’%
119         \def\param{#4}%
120         \ifx\param\empty\else%
121             \newline\newline%
122             \small%
123             --- \param%
124         \fi%
125         \\%
126     \end{tabularx}%
127 }

```

\pptPic Then, we define the \pptPic command:

```

128 \newcommand\pptPic[2]{%
129     {%
130         \setlength{\fboxsep}{0pt}%
131         \setlength{\fboxrule}{1pt}%
132         \fcolorbox{gray}{white}{%

```

```

133     \includegraphics[width=#1\columnwidth]{#2}%
134 }%
135 }%
136 }

```

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

```

137 \newcommand\pptPin[2][right]{%
138   \begin{tikzpicture}[remember picture,overlay]%
139     \ifthenelse{\equal{#1}{left}}{%
140       {%
141         \node[anchor=north east] at
142           ($(current page text area.north west) - (24pt, 18pt)$) {%
143           \begin{minipage}{0.25\textwidth}%
144             \raggedleft #2%
145           \end{minipage}%
146         };%
147       }%
148     {%
149       \node[anchor=north east] at (current page text area.north east) {%
150       \begin{minipage}{0.25\textwidth}%
151         \raggedright #2%
152       \end{minipage}%
153     };%
154   }%
155 \end{tikzpicture}%
156 }

```

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

```

157 \newcommand\pptThought[1]{%
158   \begin{pptMiddle}%
159     \tikz{%
160       \node [inner sep=18pt] (z) {%
161         \begin{varwidth}{0.8\textwidth}%
162           \raggedright\Large #1%
163         \end{varwidth}%
164       };
165       \path [draw=green,line width=8pt] (z.north west) -- (z.south west);
166     }%
167   \end{pptMiddle}%
168 }

```

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

```

169 \newcommand\pptSnippet[2][\small]{%
170   \begin{samepage}%
171     #1\verbatiminput{#2}%
172   \end{samepage}%
173 }

```

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

```

174 \newcommand\pptQR[2][2in]{%
175   \tikz{%
176     \node[draw=white]{%
177       \href{#2}{%
178         \qrcode[height=#1]{#2}%

```



```

179     }%
180   }%
181 }%
182 }

\pptPinQR Then, we define the \pptPinQR command:
183 \RequirePackage{seqsplit}
184 \newcommand\pptPinQR[2][right]{%
185   \pptPin[#1]{%
186     \begin{minipage}{1.4in}%
187       \raggedleft%
188       \pptQR[1in]{#2}%
189       \[3pt]
190       \ttfamily\fontsize{8pt}{10pt}\selectfont%
191       \seqsplit{#2} \href{#2}{$\rightarrow$}%
192     }%
193   }%
194 }%
195 }

```

pptWide Finally, we define the `pptWide` and `pptWideOne` environments:

```

196 \newenvironment{pptWideOne}
197 {\begin{adjustwidth}{-2in}{-1in}}
198 {\end{adjustwidth}}
199 \newenvironment{pptWide}[1]
200 {\begin{pptWideOne}\begin{multicols}{#1}}
201 {\end{multicols}\end{pptWideOne}}

```

Change History

0.0.1	General: First draft. 5	0.2.1	\pptMiddle: The \pptMiddle command turned into pptMiddle environment. 5
0.1.4	General: The <code>nominutes</code> package option added. 5	0.3.0	\pptQuote: The \pptQuote command has got an optional parameter, where the name of the person on the photo can be specified. 7
0.1.5	General: The default directory fixed. . . 5		
0.2.0	General: We migrated to the <code>ltxdoc</code> format and <code>.dtx</code> file. 5		\pptThought: The \pptThought command aligns text to the left. . . 8

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	A
\# 51	\arraybackslash . . . 104

B		\input 17, 24	\pptThought 157
\begin	50, 82, 84, 93, 104, 138, 143, 150, 158, 161, 170, 186, 197, 200	\item 96	\pptTitle 65
C		L	
\click	95	\LARGE 53	\pptToc 72, 74, 83, 85, 95, 96
\color	45	\Large 62, 162	\pptToc 72
\colorbox	45	\large 45, 51, 69	\pptWide 196
\columnwidth	104, 133	\print 75	
\crumb	94	\ProcessPgfpPackageOptions 14	
\crumbection	49	M	
\crumbs	98	\makeatletter .. 2, 16, 22	Q
D		\makeatother ... 15, 21, 28	
\def	67, 76, 109, 119	\message 19, 26	\qrcode 178
\dimexpr	105	N	
E		\newcommand 42, 48, 56, 60, 65, 72, 73, 103, 128, 137, 157, 169, 174, 184	R
\empty	68, 77, 83, 95, 110, 120	\newenvironment 39, 196, 199	
\end	54, 86, 88, 99, 126, 145, 152, 155, 163, 167, 172, 193, 198, 201	\newline 52, 121	\raggedleft 144, 187
\equal	139	\node . 81, 141, 149, 160, 176	\raggedright 104, 107, 151, 162
F		\raisebox 105	
\fboxrule	131	\renewcommand 74, 94	
\fboxsep	44, 130	\RequirePackage 1, 23, 29, 30, 31, 32, 33, 34, 36, 37, 38, 102, 183	
\fcolorbox	132	\rightarrow 191	
\fill	40, 41	\rotatebox 53, 62	
\flush	100	P	
\fontsize	190	\par 43, 46, 63, 111	S
G		\param 67, 68, 76, 77, 119, 120, 123	
\gappto	96	\parbox 106	\selectfont 190
H		\path 90, 165	\seqsplit 191
\height	105	\person 109, 110, 113	\setlength .. 44, 130, 131
\href	177, 191	\pgfkeys 3	\small 112, 122, 169
\hspace	61	\ppt@directory . 9, 17, 24	\strutbox 105
\ht	105	\ppt@nominutes 11	\subcrumbection 57
\Huge	66	\ppt@scheme 8, 22, 24	T
I		\ppt@template .. 6, 16, 17	
\ifdefined	16, 22, 95, 100	\pptBanner 42, 53, 62, 66, 69, 78	\textwidth 82, 106, 143, 150, 161
\ifthenelse	139	\pptChapter 48	\the 51
\ifx	68, 77, 83, 95, 110, 120	\pptHeader 58, 60	\thepagecolor 45
\includegraphics	133	\pptMiddle 39	\tikz 80, 159, 175
		\pptPic 108, 128	\ttfamily 51, 190
		\pptPin 137, 185	U
		\pptPinQR 183	
		\pptQR 174, 188	\usetikzlibrary 35
		\pptQuote 102	
		\pptSection 56	V
		\pptSnippet 169	
			\value 51
			\verbatiminput 171
			\vspace 40, 41, 61, 62