

Slides and Course Notes*

Michael Kohlhasse
FAU Erlangen-Nürnberg
<http://kwarc.info/kohlhasse>

December 5, 2018

Abstract

We present a document class from which we can generate both course slides and course notes in a transparent way.

Contents

1	Introduction	2
2	The User Interface	2
2.1	Package Options	2
2.2	Notes and Slides	2
2.3	Header and Footer Lines	4
2.4	Colors and Highlighting	4
2.5	Front Matter, Titles, etc	4
2.6	Miscellaneous	4
3	Limitations	4
4	The Implementation	4
4.1	Class and Package Options	4
4.2	Notes and Slides	6
4.3	Header and Footer Lines	8
4.4	Colors and Highlighting	10
4.5	Sectioning	11
4.6	Miscellaneous	12

*Version ? (last revised ?)

1 Introduction

The `mikoslides` document class is derived from `beamer.cls` [Tana], it adds a “notes version” for course notes derived from the `omdoc` class [Kohlhase:smomdl] that is more suited to printing than the one supplied by `beamer.cls`.

2 The User Interface

The `mikoslides` class takes the notion of a slide frame from Till Tantau’s excellent `beamer` class and adapts its notion of frames for use in the \TeX and OMDoc. To support semantic course notes, it extends the notion of mixing frames and explanatory text, but rather than treating the frames as images (or integrating their contents into the flowing text), the `mikoslides` package displays the slides as such in the course notes to give students a visual anchor into the slide presentation in the course (and to distinguish the different writing styles in slides and course notes).

In practice we want to generate two documents from the same source: the slides for presentation in the lecture and the course notes as a narrative document for home study. To achieve this, the `mikoslides` class has two modes: *slides mode* and *notes mode* which are determined by the package option.

2.1 Package Options

The `mikoslides` class takes a variety of class options:¹

<code>slides</code>	• The options <code>slidesnd</code> and <code>notesnotes</code> switch between slides mode and notes mode (see Section 2.2).
<code>a</code>	
<code>sectocframes</code>	• If the option <code>sectocframes</code> is given, then special frames with section table of contents are produced headers ²
<code>showmeta</code>	• <code>showmeta</code> . If this is set, then the metadata keys are shown (see [Koh16] for details and customization options).
<code>frameimages</code>	• If the option <code>frameimages</code> is set, then slide mode also shows the <code>\frameimage-</code> generated frames.
<code>topsect</code>	• <code>topsect=<sect></code> can be used to specify the top-level sectioning level; the default for <code><setc></code> is <code>section</code> .

2.2 Notes and Slides

<code>frame</code>	Slides are represented with the <code>frame</code> just like in the <code>beamer</code> class, see [Tanb] for details. The <code>mikoslides</code> class adds the <code>note</code> environment for encapsulating the
<code>note</code>	

¹EDNOTE: leaving out noproblems for the moment until we decide what to do with it.

²EDNOTE: document the functionality

course note fragments.¹

 Note that it is essential to start and end the `notes` environment at the start of the line – in particular, there may not be leading blanks – else \LaTeX becomes confused and throws error messages that are difficult to decipher.

```
\begin{note}
  We start this course with ...
\end{note}

\begin{frame}
  \frametitle{The first slide}
  ...
\end{frame}
\begin{note}
  ... and more explanatory text
\end{note}

\begin{frame}
  \frametitle{The second slide}
  ...
\end{frame}
...
```

Example 1: A typical Course Notes File

By interleaving the `frame` and `note` environments, we can build course notes as shown in Figure 1.

Sometimes, we want to integrate slides as images after all – e.g. because we already have a PowerPoint presentation, to which we want to add \LaTeX notes. In this case we can use `\frameimage[<opt>]{<path>}`, where *<opt>* are the options of `\includegraphics` from the `graphicx` package [CR99] and *<path>* is the file path (extension can be left off like in `\includegraphics`).

If we want to transclude the contents of a file as a note, we can use the `\ninputref` macro. `\ninputref{foo}` is equivalent to


```
\begin{note}
\inputref{foo}
\end{note}
```

There are some environments that tend to occur at the top-level of `note` environments. We make convenience versions of these: e.g. the `nomtext` environment is just an `omtext` inside a `note` environment (but looks nicer in the source, since it avoids one level of source indenting). Similarly, we have the `nomgroup` environment.

¹MK: it would be very nice, if we did not need this environment, and this should be possible in principle, but not without intensive \LaTeX trickery. Hints to the author are welcome.

2.3 Header and Footer Lines

2.4 Colors and Highlighting

`\textwarning` The `\textwarning` macro generates a warning sign: 

2.5 Front Matter, Titles, etc

2.6 Miscellaneous

3 Limitations

In this section we document known limitations. If you want to help alleviate them, please feel free to contact the package author. Some of them are currently discussed in the `sTeX`GitHub repository [sTeX].

1. when option `book` which uses `\pagestyle{headings}` is given and semantic macros are given in the `omgroup` titles, then they sometimes are not defined by the time the heading is formatted. Need to look into how the headings are made. This is a problem of the underlying `omdoc` package.

4 The Implementation

4.1 Class and Package Options

We define some Package Options and switches for the `mikoslides` class and activate them by passing them on to `beamer.cls` and `omdoc.cls` and the `mikoslides` package. We pass the `nontheorem` option to the `statements` package when we are not in notes mode, since the `beamer` package has its own (overlay-aware) theorem environments.

```
1 <*cls>
2 \RequirePackage{kvoptions}
3 \SetupKeyvalOptions{family=mks@cls,prefix=mks@cls@}
4 \DeclareBoolOption{notes}
5 \DeclareComplementaryOption{slides}{notes}
6 \DeclareDefaultOption{\PassOptionsToClass{\CurrentOption}{omdoc}}
7 \PassOptionsToClass{\CurrentOption}{beamer}
8 \PassOptionsToPackage{\CurrentOption}{mikoslides}}
9 \ProcessKeyvalOptions{mks@cls}
10 </cls>
```

now we do the same for the `mikoslides` package.

```
11 <*package>
12 \RequirePackage{kvoptions}
13 \SetupKeyvalOptions{family=mks@sty,prefix=mks@sty@}
14 \DeclareStringOption[section]{topsect}
15 \DeclareBoolOption{mh}
16 \AddToKeyvalOption*{mh}{
```

```

17 \PassOptionsToPackage{mh}{stex}
18 \PassOptionsToPackage{mh}{smglom}
19 \PassOptionsToPackage{mh}{tikzinput}}
20 \newif\ifnotes\notesttrue
21 \DeclareBoolOption{notes}
22 \AddToKeyvalOption*{notes}{\notesttrue\PassOptionsToPackage{notes}{statements}}
23 \DeclareComplementaryOption{slides}{notes}
24 \AddToKeyvalOption*{slides}{\notestfalse\PassOptionsToPackage{nontheorem}{statements}}
25 \DeclareBoolOption{sectocframes}
26 \AddToKeyvalOption*{sectocframes}{\PassOptionsToPackage{msection}{statements}}
27 \DeclareBoolOption{frameimages}
28 \DeclareBoolOption{nopproblems}
29 \DeclareDefaultOption{\PassOptionsToPackage{\CurrentOption}{stex}
30 \PassOptionsToPackage{\CurrentOption}{smglom}
31 \PassOptionsToPackage{\CurrentOption}{tikzinput}}
32 \ProcessKeyvalOptions{mks@sty}
33 \end{package}

```

Depending on the options, we either load the `article`-based `omdoc` or the `beamer` class (and set some counters).

```

34 \begin{document}
35 \ifmks@cls@notes
36 \LoadClass{omdoc}
37 \else
38 \LoadClass[10pt,nontheorems]{beamer}
39 \newcounter{Item}
40 \newcounter{paragraph}
41 \newcounter{subparagraph}
42 \newcounter{Hfootnote}
43 \fi

```

now it only remains to load the `mikoslides` package that does all the rest.

```

44 \RequirePackage{mikoslides}
45 \end{document}

```

In `notes` mode, we also have to make the `beamer`-specific things available to `article` via the `beamerarticle` package. We use options to avoid loading theorem-like environments, since we want to use our own from the `STEX` packages. The first batch of packages we want are loaded on `mikoslides.sty`. These are the general ones, we will load the `STEX`-specific ones after we have done some work (e.g. defined the counters `m*`). Only the `stex-logo` package is already needed now for the default theme.

```

46 \begin{package}
47 \ifmks@sty@notes
48 \RequirePackage{a4wide}
49 \RequirePackage{marginnote}
50 \RequirePackage{xcolor}
51 \RequirePackage{mdframed}
52 \RequirePackage[noxcolor,noamsthm]{beamerarticle}
53 \fi

```

```

54 \ifmks@sty@mh\RequirePackage{mikosides-mh}\fi
55 \RequirePackage{etoolbox}
56 \RequirePackage{amssymb}
57 \RequirePackage{amsmath}
58 \RequirePackage{comment}
59 \RequirePackage{textcomp}
60 \RequirePackage{url}
61 \RequirePackage{graphicx}
62 \RequirePackage{stex-logo}
63 \RequirePackage{pgf}
64 \ifmks@sty@notes
65 \RequirePackage[bookmarks,bookmarksopen,bookmarksnumbered,breaklinks,
66 linkcolor=black,citecolor=black,urlcolor=cyan,filecolor=cyan,colorlinks]{hyperref}
67 \fi

```

finally, we require the `metakeys` package from `STEX`, so that we can use the `\addmetakey` mechanism.

```

68 \RequirePackage{metakeys}

```

4.2 Notes and Slides

For the lecture notes cases, we also provide the `\usetheme` macro that would otherwise from the the `beamer` class. While the latter loads `beamertheme<theme>.sty`, the notes version loads `beamernotestheme<theme>.sty`.³

```

69 \ifmks@sty@notes
70 \renewcommand\usetheme[2] [] {\usepackage[#1]{beamernotestheme#2}}
71 \fi

```

We define the sizes of slides in the notes. Somehow, we cannot get by with the same here.

```

72 \newcounter{slide}
73 \newlength{\slidewidth}\setlength{\slidewidth}{12.8cm}
74 \newlength{\slideheight}\setlength{\slideheight}{9cm}

```

note The `note` environment is used to leave out text in the `slides` mode. It does not have a counterpart in OMDoc. So for course notes, we define the `note` environment to be a no-operation otherwise we declare the `note` environment as a comment via the `comment` package.

```

75 \ifmks@sty@notes%
76 \renewenvironment{note}{\ignorespaces}{}%
77 \else%
78 \excludecomment{note}%
79 \fi%

```

`\ninputref`

```

80 \newcommand\ninputref[2] [] {\ifmks@sty@notes\ninputref[#1]{#2}\fi}

```

³EDNOTE: MK: This is not ideal, but I am not sure that I want to be able to provide the full theme functionality there.

We first set up the slide boxes in `article` mode. We set up sizes and provide a box register for the frames and a counter for the slides.

```
81 \ifmks@sty@notes
82 \newlength{\slideframewidth}
83 \setlength{\slideframewidth}{1.5pt}
```

`frame` We first define the keys.

```
84 \addmetakey{frame}{label}
85 \addmetakey[yes]{frame}{allowframebreaks}
86 \addmetakey{frame}{allowdisplaybreaks}
87 \addmetakey[yes]{frame}{fragile}
88 \addmetakey[yes]{frame}{shrink}
89 \addmetakey[yes]{frame}{squeeze}
90 \addmetakey[yes]{frame}{t}
```

We define the environment, read them, and construct the slide number and label.

```
91 \renewenvironment{frame}[1][{}]{%
92 \metasetkeys{frame}{#1}%
93 \stepcounter{slide}%
94 \def\@currentlabel{\theslide}%
95 \ifx\frame@label\@empty%
96 \else%
97 \label{\frame@label}%
98 \fi%
```

We redefine the `itemize` environment so that it looks more like the one in `beamer`.

```
99 \def\itemize@level{outer}%
100 \def\itemize@outer{outer}%
101 \def\itemize@inner{inner}%
102 \renewcommand\newpage{}%
103 \renewcommand\metakeys@show@keys[2]{\marginnote{\scriptsize ##2}}%
104 \renewenvironment{itemize}{%
105 \ifx\itemize@level\itemize@outer%
106 \def\itemize@label{$\rhd$}%
107 \fi%
108 \ifx\itemize@level\itemize@inner%
109 \def\itemize@label{$\scriptstyle\rhd$}%
110 \fi%
111 \begin{list}%
112 {\itemize@label}%
113 {\setlength{\labelsep}{.3em}%
114 \setlength{\labelwidth}{.5em}%
115 \setlength{\leftmargin}{1.5em}%
116 }%
117 \edef\itemize@level{\itemize@inner}%
118 }{%
119 \end{list}%
120 }%
```

We create the box with the `mdframed` environment from the `equinymous` package.

```

121     \begin{mdframed}[linewidth=\slideframewidth,skipabove=1ex,skipbelow=1ex,userdefinedwidth=\s
122   }{%
123     \medskip\miko@slidelabel\end{mdframed}%
124   }%

```

Now, we need to redefine the frametitle (we are still in course notes mode).

```

\frametitle
125   \renewcommand{\frametitle}[1]{\Large\bf\sf\color{blue}{#1}}\medskip}%
126 \fi %ifnotes

```

EdN:4

```

\frameimage We have to make sure that the width is overwritten, for that we check the
\Gin@ewidth macro from the graphicx package4
127 \newrobustcmd\frameimage[2] []{%
128   \stepcounter{slide}%
129   \ifmks@sty@frameimages%
130     \def\Gin@ewidth{}\setkeys{Gin}{#1}%
131     \ifmks@sty@notes\else\vfill\fi%
132     \ifx\Gin@ewidth\@empty%
133       \mycgraphics[width=\slidewidth,#1]{#2}\else\mycgraphics[#1]{#2}%
134     \fi%
135     \par\strut\hfill{\footnotesize Slide \arabic{slide}}}%
136     \ifmks@sty@notes\else\vfill\fi%
137   \fi%
138 }% ifframeimages

```

EdN:5

```

\pause 5
139 \ifmks@sty@notes\newcommand\pause{}\fi

nomtext
140 \ifmks@sty@notes\newenvironment{nomtext}[1] [] {\begin{omtext}[#1]}\end{omtext}}%
141 \else\excludecomment{nomtext}\fi%

nomgroup
142 \ifmks@sty@notes\newenvironment{nomgroup}[2] [] {\begin{omgroup}[#1]{#2}}\end{omgroup}}%
143 \else\excludecomment{nomgroup}\fi%

```

4.3 Header and Footer Lines

Now, we set up the infrastructure for the footer line of the slides, we use boxes for the logos, so that they are only loaded once, that considerably speeds up processing.

```

\setslidelogo The default logo is the logo of Jacobs University. Customization can be done by
\setslidelogo{\logo name}.
144 \newlength{\slidelogoheight}

```

⁴EdNOTE: MK@DG; we need to do that in the LaTeXML binding as well!

⁵EdNOTE: MK: fake it in notes mode for now


```

145 \ifmks@sty@notes%
146 \setlength{\slidelogoheight}{.4cm}%
147 \else%
148 \setlength{\slidelogoheight}{1cm}%
149 \fi%
150 \newsavebox{\slidelogo}%
151 \sbox{\slidelogo}{\sTeX}%
152 \newrobustcmd{\setslidelogo}[1]{%
153 \sbox{\slidelogo}{\includegraphics[height=\slidelogoheight]{#1}}%
154 }%

\setsource \source stores the writer's name. By default it is Michael Kohlhase since he is
the main user and designer of this package. \setsource{<name>} can change the
writer's name.

155 \def\source{Michael Kohlhase}% customize locally
156 \newrobustcmd{\setsource}[1]{\def\source{#1}}%

\setlicensing Now, we set up the copyright and licensing. By default we use the Creative
Commons Attribution-ShareAlike license to strengthen the public domain. If
package hyperref is loaded, then we can attach a hyperlink to the license logo.
\setlicensing[<url>]{<logo name>} is used for customization, where <url> is op-
tional.

157 \def\copyrightnotice{\footnotesize\copyright:\hspace{.3ex}{\source}}%
158 \newsavebox{\cclogo}%
159 \sbox{\cclogo}{\includegraphics[height=\slidelogoheight]{cc_somerights}}%
160 \newif\ifcchref\cchreffalse%
161 \AtBeginDocument{%
162 \ifpackageloaded{hyperref}{\cchreftrue}{\cchreffalse}
163 }%
164 \def\licensing{%
165 \ifcchref%
166 \href{http://creativecommons.org/licenses/by-sa/2.5/}{\usebox{\cclogo}}%
167 \else%
168 {\usebox{\cclogo}}%
169 \fi%
170 }%
171 \newrobustcmd{\setlicensing}[2][ ]{%
172 \def\@url{#1}%
173 \sbox{\cclogo}{\includegraphics[height=\slidelogoheight]{#2}}%
174 \ifx\@url\@empty%
175 \def\licensing{\usebox{\cclogo}}%
176 \else%
177 \def\licensing{%
178 \ifcchref%
179 \href{#1}{\usebox{\cclogo}}%
180 \else%
181 {\usebox{\cclogo}}%
182 \fi%
183 }%

```

```

184 \fi%
185 }%

```

EdN:6

`\slidelabel` Now, we set up the slide label for the `article` mode.⁶

```

186 \newrobustcmd\miko@slidelabel{%
187 \vbox to \slidelogoheight{%
188 \vss\hbox to \slidewidth%
189 {\licensing\hfill\copyrightnotice\hfill\arabic{slide}\hfill\usebox{\slidelogo}}%
190 }%
191 }%

```

4.4 Colors and Highlighting

We first specify sans serif fonts as the default.

```

192 \sffamily

```

Now, we set up an infrastructure for highlighting phrases in slides. Note that we use content-oriented macros for highlighting rather than directly using color markup. The first thing to do is to adapt the green so that it is dark enough for most beamers

```

193 \AtBeginDocument{%
194 \definecolor{green}{rgb}{0,.5,0}%
195 \definecolor{purple}{cmyk}{.3,1,0,.17}%
196 }%

```

We customize the `\defemph`, `\notemph`, and `\stDMemph` macros with colors for the use in the `statements` package. Furthermore we customize the `\@@lec` macro for the appearance of line end comments in `\lec`.

```

197 % \def\STpresent#1{\textcolor{blue}{#1}}
198 \def\defemph#1{\textcolor{magenta}{#1}}
199 \def\notemph#1{\textcolor{magenta}{#1}}
200 \def\stDMemph#1{\textcolor{blue}{#1}}
201 \def\@@lec#1(\textcolor{green}{#1})

```

I like to use the dangerous bend symbol for warnings, so we provide it here.

`\textwarning` as the macro can be used quite often we put it into a box register, so that it is only loaded once.

```

202 \pgfdeclareimage[width=.8em]{miko@small@dbend}{dangerous-bend}
203 \def\smalltextwarning{%
204 \pgfuseimage{miko@small@dbend}%
205 \xspace%
206 }%
207 \pgfdeclareimage[width=1.2em]{miko@dbend}{dangerous-bend}
208 \newrobustcmd\textwarning{%
209 \raisebox{-.05cm}{\pgfuseimage{miko@dbend}}%
210 \xspace%
211 }%

```

⁶EdNOTE: see that we can use the themes for the slides some day. This is all fake.

```

212 \pgfdeclareimage[width=2.5em]{miko@big@dbend}{dangerous-bend}%
213 \newrobustcmd\bigtextwarning{%
214   \raisebox{- .05cm}{\pgfuseimage{miko@big@dbend}}}%
215   \xspace%
216 }%

217 \newrobustcmd\putgraphicsat[3]{%
218   \begin{picture}(0,0)\put(#1){\includegraphics[#2]{#3}}\end{picture}%
219 }%
220 \newrobustcmd\putat[2]{%
221   \begin{picture}(0,0)\put(#1){#2}\end{picture}%
222 }%

```

4.5 Sectioning

If the `sectocframes` option is set, then we make section frames. We first define a set of counters⁷

```

223 \ifmks@sty@sectocframes%
224 \ifdefstring\mks@sty@topsect{part}{%
225   \newcounter{mpart}\newcounter{mchapter}\newcounter{msection}[mchapter]}
226 {\ifdefstring\mks@sty@topsect{chapter}{%
227   \newcounter{mchapter}\newcounter{msection}[mchapter]}
228   {\newcounter{msection}}}%
229 \newcounter{msubsection}[msection]%
230 \newcounter{msubsubsection}[msubsection]%
231 \newcounter{msubsubsubsection}[msubsubsection]%
232 \fi% ifsectocframes

```

Now that we have defined the counters, we can load the \TeX -specific packages (in particular `statements` that needs these counters).

```

233 \RequirePackage{stex}
234 \RequirePackage{smglom}
235 \RequirePackage{tikzinput}

```

`\section@level` Finally, we set the `\section@level` macro that governs sectioning.

```

236 \section@level=2
237 \ifdefstring{\mks@sty@topsect}{part}{\section@level=0}{%
238 \ifdefstring{\mks@sty@topsect}{chapter}{\section@level=1}{%

```

Now \TeX is loaded, we redefine the `omgroup` environment to produce section toc frames (if the option `sectocframes` is specified.)⁸

```

239 \ifmks@sty@notes\else% only in slides
240 \def\part@prefix{\ifdefstring\mks@sty@topsect{part}{\arabic{mchapter}.}{}}
241 \renewenvironment{omgroup}[2][{}]{%

```

⁷EDNOTE: I forget: why not use the counters from `beamer/article`? –i document this.

⁸EDNOTE: MK: we should probably just redefine `omgroup@num` and `omgroup@nonum`, since they do the actual work so that we can add the `sectocframes` behavior here without having to copy the internals. Then there is less material that can get out of sync. Additionally, we should have a hook in the original code of those so that we can increment the slides counter in notes node (to keep slides in sync)

```

242 \metasetkeys{omgroup}{#1}\sref@target%
243 \advance\section@level by 1%
244 \ifmks@sty@sectocframes%
245 \stepcounter{slide}
246 \begin{frame}[noframenumbering]%
247 \vfill\Large\centering%
248 \red{%
249 \ifcase\section@level\or
250 \stepcounter{mpart}
251 \def\@@label{Part \Roman{mpart}}
252 \def\currentsectionlevel{part}
253 \or%
254 \stepcounter{mchapter}
255 \def\@@label{Chapter \arabic{mchapter}}
256 \def\currentsectionlevel{chapter}
257 \or
258 \stepcounter{msection}
259 \def\@@label{\part@prefix\arabic{msection}}
260 \def\currentsectionlevel{section}
261 \or
262 \stepcounter{msubsection}
263 \def\@@label{\part@prefix\arabic{msection}.\arabic{msubsection}}
264 \def\currentsectionlevel{subsection}
265 \or
266 \stepcounter{msubsubsection}
267 \def\@@label{\part@prefix\arabic{msection}.\arabic{msubsection}.\arabic{msubsubsection}}
268 \def\currentsectionlevel{subsubsection}
269 \or
270 \stepcounter{msubsubsubsection}
271 \def\@@label{\part@prefix\arabic{msection}.\arabic{msubsection}.\arabic{msubsubsubsection}.\ar
272 \def\currentsectionlevel{subsubsubsection}
273 \fi% end ifcase
274 \@@label\sref@label@id\@@label
275 \quad #2%
276 }%
277 \vfill%
278 \end{frame}%
279 \fi %ifmks@sty@sectocframes
280 }
281 {\advance\section@level by -1}%
282 \fi% ifmks@sty@notes

```

4.6 Miscellaneous

We set up a beamer template for theorems like ams style, but without a block environment.

```

283 \def\inserttheorembodyfont{\normalfont}
284 \def\beamertemplate{theorem begin}{miko}
285 {\inserttheoremheadfont\inserttheoremname\inserttheoremnumber

```

```

286 \ifx\inserttheoremaddition\@empty\else\ (\inserttheoremaddition)\fi%
287 \inserttheorempunctuation\inserttheorembodyfont\xspace}
288 \defbeamertemplate{theorem end}{miko}{%
    and we set it as the default one.
289 \setbeamertemplate{theorems}[miko]
    The following fixes an error I do not understand, this has something to do with
    beamer compatibility, which has similar definitions but only up to 1.
290 \expandafter\def\csname Parent2\endcsname{}
    We need to disregard the columns macros introduced by the beamer class in the
    notes.
291 \ifmks@sty@notes%
292 \renewenvironment{columns}[1][]{%
293 \par\noindent%
294 \begin{minipage}%
295 \slidewidth\centering\leavevmode%
296 }{%
297 \end{minipage}\par\noindent%
298 }%
299 \newsavebox\columnbox%
300 \renewenvironment<>{column}[2][]{%
301 \begin{lrbox}{\columnbox}\begin{minipage}{#2}%
302 }{%
303 \end{minipage}\end{lrbox}\usebox\columnbox%
304 }%
305 \fi% ifnotes
306 \ifmks@sty@nopproblems%
307 \newenvironment{problems}{}{}%
308 \else%
309 \excludecomment{problems}%
310 \fi%
311 </package>

```

Change History

v0.1		the <code>frame</code> environment in notes	
General: Initial Version	1	mode	1
v0.2		numbered sectocframes	1
General: course notes back on		this is almost done	1
seminar	1		
v0.3		v1.0	
General: changing to Jacobs logo	1	General: adding <code>\frameimage</code>	1
v0.4		v1.1	
General: moving line-end-comment		General: moving MathHub support	
to <code>omdoc.dtx</code>	1	out to separate package	1
re-basing the whole thing on		reinterpreting <code>omgroup</code>	1
beamer	1	Removing the old title macros	
v0.5		(use the regular ones instead)	1
General: eliminating			
mytwocolumns, this is better		v1.2	
done by <code>beamer.cls</code>	1	General: changed to keyval	
v0.9		class/package options, allowed	
General: basic options handling for		arbitrary classes	1

References

- [CR99] David Carlisle and Sebastian Rathz. *The graphicx package*. Part of the T_EX distribution. The Comprehensive T_EX Archive Network. 1999. URL: <https://www.tug.org/texlive/devsrc/Master/texmf-dist/doc/latex/graphics/graphicx.pdf>.
- [Koh16] Michael Kohlhase. *metakeys.sty: A generic framework for extensible Metadata in L^AT_EX*. Tech. rep. Comprehensive T_EX Archive Network (CTAN), 2016. URL: <http://mirror.ctan.org/macros/latex/contrib/stex/sty/metakeys/metakeys.pdf>.
- [sTeX] KWARC/sTeX. URL: <https://github.com/KWARC/sTeX> (visited on 05/15/2015).
- [Tana] Till Tantau. *beamer – A L^AT_EX class for producing presentations and slides*. URL: <http://ctan.org/pkg/beamer> (visited on 01/07/2014).
- [Tanb] Till Tantau. *User Guide to the Beamer Class*. URL: <http://ctan.org/macros/latex/contrib/beamer/doc/beameruserguide.pdf>.