Slides and Course Notes*

Michael Kohlhase
FAU Erlangen-Nürnberg
http://kwarc.info/kohlhase

June 17, 2021

Abstract

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

Contents

---

*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 STEXand 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

EdN:1

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

slides
notes
- The options `slides` and `notes` switch between slides mode and notes mode (see Section 2.2).

sectocframes
- If the option `sectocframes` is given, then for the `omgroups`, special frames with the `omgroup` title (and number) are generated.

showmeta
- `showmeta`. If this is set, then the metadata keys are shown (see [Koh20b] for details and customization options).

frameimages
fiboxed
- If the option `frameimages` is set, then slide mode also shows the `\frameimage`-generated frames. If also the `fiboxed` option is given, the slides are surrounded by a box.

topsect
- `topsect=⟨sect⟩` can be used to specify the top-level sectioning level; the default for ⟨sect⟩ is `section`.

mh
- The `mh` option turns on MathHub support; see [Koh20a].

2.2 Notes and Slides

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

---

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

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.

```
\ifnotes\maketitle\else
\frame[noframenumbering]\maketitle\fi

\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.

\ifnotes   Note the use of the `\ifnotes` conditional, which allows different treatment between `notes` and `slides` mode – manually setting `\notestrue` or `\notesfalse` is strongly discouraged however.

⚠: We need to give the title frame the `noframenumbering` option so that the frame numbering is kept in sync between the slides and the course notes.

⚠: The `beamer` class recommends not to use the `allowframebreaks` option on frames (even though it is very convenient). This holds even more in the `mikoslides` case: At least in conjunction with `\newpage`, frame numbering behaves funnily (we have tried to fix this, but who knows).

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 sTeXnotes. In
\frameimage   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`). We have added the `label`

---

[1]MK: 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.

key that allows to give a frame label that can be referenced like a regular `beamer` frame.[2]

\inputref*

    If we want to transclude a the contents of a file as a note, we can use a new variant `\inputref*` of the `\inputref` macro from [KGA20]: `\inputref*{foo}` is equivalent to `\begin{note}\inputref{foo}\end{note}`.

nomtext

    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`, `ndefinition`, `nexample`, `nsproof`, and `nassertion` environments.

nomgroup
ndefinition
nexample
nsproof
nassertion

2.3   Header and Footer Lines of the Slides

\setslidelogo

The default logo provided by the `mikoslides` package is the SₜEX logo it can be customized using `\setslidelogo{⟨logo name⟩}`.

    The default footer line of the `mikoslides` package mentions copyright and licensing. In the `beamer` class, `\source` stores the author's name as the copyright holder . By default it is *Michael Kohlhase* in the `mikoslides` package

\setsource

since he is the main user and designer of this package. `\setsource{⟨name⟩}` can change the writer's name. For licensing, we use the Creative Commons Attribuition-ShareAlike license by default to strengthen the public domain. If package `hyperref` is loaded, then we can attach a hyperlink to the license logo.

\setlicensing

`\setlicensing[⟨url⟩]{⟨logo name⟩}` is used for customization, where ⟨url⟩ is optional.

2.4   Colors and Highlighting

\textwarning

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

2.5   Front Matter, Titles, etc.

2.6   Excursion

In course notes, we sometimes want to point to an "excursion" – material that is either presupposed or tangential to the course at the moment – e.g. in an appendix. The typical setup is the following:

```
\excursion{founif}{../ex/founif}{We will cover first-order unification in}
...
\begin{appendix}\printexcursions\end{appendix}
```

\excursion
\activateexcursion

    The `\excursion{⟨ref⟩}{⟨path⟩}{⟨text⟩}` is syntactic sugar for

```
\begin{nomtext}[title=Excursion]
```

---

[2]EDNOTE: MK: the hyperref link does not seem to work yet. I wonder why but do not have the time to fix it.

```
    \activateexcursion{founif}{../ex/founif}
    We will cover first-order unification in \sref{founif}.
  \end{nomtext}
```

\activateexcursion   where \activateexcursion{⟨*path*⟩} augments the \printexcursions macro
\printexcursions   by a call \inputref{⟨*path*⟩}. In this way, the3 \printexcursions macro (usually
in the appendix) will collect up all excursions that are specified in the main text.

Sometimes, we want to reference – in an excursion – part of another. We can
\excursionref   use \excursionref{⟨*label*⟩} for that.

Finally, we usually want to put the excursions into an omgroup environ-
ment and add an introduction, therefore we provide the a variant of the
\excursiongroup   \printexcursions macro: \excursiongroup[id=⟨*id*⟩,intro=⟨*path*⟩] is equiva-
lent to

```
\begin{omgroup}[id=<id>]{Excursions}
  \inputref{<path>}
  \printexcursions
\end{omgroup}
```

2.7   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 sTEXGitHub 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 acti-
vate 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{modules}
3 \RequirePackage{kvoptions}
4 \RequirePackage{etoolbox}
```

```
 5 \SetupKeyvalOptions{family=mks@cls,prefix=mks@cls@}
 6 \DeclareStringOption[article]{class}
 7 \AddToKeyvalOption*{class}{\PassOptionsToClass{class=\mks@cls@class}{omdoc}
 8   \ifdefstring{\mks@cls@class}{book}{\PassOptionsToPackage{defaulttopsect=part}{mikoslides}}{}
 9   \ifdefstring{\mks@cls@class}{report}{\PassOptionsToPackage{defaulttopsect=part}{mikoslides}}}
10 \DeclareBoolOption{notes}
11 \DeclareComplementaryOption{slides}{notes}
12 \DeclareDefaultOption{%
13   \PassOptionsToClass{\CurrentOption}{omdoc}
14   \PassOptionsToClass{\CurrentOption}{beamer}
15   \PassOptionsToPackage{\CurrentOption}{mikoslides}}
16 \ProcessKeyvalOptions{mks@cls}
17 ⟨/cls⟩
```

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

```
18 ⟨∗package⟩
19 \RequirePackage{stex-base}
20 \RequirePackage{kvoptions}
21 \SetupKeyvalOptions{family=mks@sty,prefix=mks@sty@}
22 \DeclareStringOption{topsect}
23 \DeclareStringOption{defaulttopsect}
24 \DeclareBoolOption{mh}
25 \AddToKeyvalOption*{mh}{
26   \PassOptionsToPackage{mh}{stex}
27   \PassOptionsToPackage{mh}{smglom}
28   \PassOptionsToPackage{mh}{tikzinput}}
29 \newif\ifnotes\notestrue
30 \DeclareBoolOption{notes}
31 \AddToKeyvalOption*{notes}{\notestrue\PassOptionsToPackage{notes}{statements}}
32 \DeclareComplementaryOption{slides}{notes}
33 \AddToKeyvalOption*{slides}{\notesfalse\PassOptionsToPackage{nontheorem}{statements}}
34 \DeclareBoolOption{sectocframes}
35 \DeclareBoolOption{frameimages}
36 \DeclareBoolOption{fiboxed}
37 \DeclareBoolOption{noproblems}
38 \DeclareDefaultOption{%
39   \PassOptionsToPackage{\CurrentOption}{stex}
40   \PassOptionsToPackage{\CurrentOption}{smglom}
41   \PassOptionsToPackage{\CurrentOption}{tikzinput}}
42 \ProcessKeyvalOptions{mks@sty}
```

we give ourselves a macro `\@@topsect` that needs only be evaluated once, so that the `\ifdefstring` conditionals work below.

```
43 \ifx\mks@sty@topsect\@empty\edef\@@topsect{\mks@sty@defaulttopsect}
44 \else\edef\@@topsect{\mks@sty@topsect}\fi
45 ⟨/package⟩
```

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

```
46 ⟨∗cls⟩
47 \ifmks@cls@notes
```

```
48    \LoadClass{omdoc}
49 \else
50    \LoadClass[10pt,notheorems,xcolor={dvipsnames,svgnames}]{beamer}
51    \newcounter{Item}
52    \newcounter{paragraph}
53    \newcounter{subparagraph}
54    \newcounter{Hfootnote}
55 \fi
```

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

```
56 \RequirePackage{mikoslides}
57 ⟨/cls⟩
```

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.

```
58 ⟨*package⟩
59 \ifmks@sty@notes
60 \RequirePackage{a4wide}
61 \RequirePackage{marginnote}
62 \RequirePackage[dvipsnames,svgnames]{xcolor}
63 \if@latexml\else\RequirePackage{mdframed}\fi
64 \RequirePackage[noxcolor,noamsthm]{beamerarticle}
65 \fi
66 \RequirePackage{etoolbox}
67 \RequirePackage{amssymb}
68 \RequirePackage{amsmath}
69 \RequirePackage{comment}
70 \RequirePackage{textcomp}
71 \RequirePackage{url}
72 \RequirePackage{graphicx}
73 \RequirePackage{stex-logo}
74 \RequirePackage{pgf}
75 \ifmks@sty@notes
76 \RequirePackage[bookmarks,bookmarksopen,bookmarksnumbered,breaklinks,hidelinks]{hyperref}
77 \fi
```

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

```
78 \RequirePackage{metakeys}
```

4.2   Notes and Slides

For the lecture notes cases, we also provide the `\usetheme` macro that would otherwise come from the the `beamer` class. While the latter loads `beamertheme`⟨*theme*⟩.sty,

the notes version loads `beamernotestheme⟨theme⟩.sty`.[3]

```
79 \ifmks@sty@notes
80 \renewcommand\usetheme[2][]{\usepackage[#1]{beamernotestheme#2}}
81 \fi
```

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

```
82 \newcounter{slide}
83 \newlength{\slidewidth}\setlength{\slidewidth}{13.5cm}
84 \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.

```
85 \ifmks@sty@notes%
86   \renewenvironment{note}{\ignorespaces}{}%
87 \else%
88   \excludecomment{note}%
89 \fi%
```

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.

```
90 \ifmks@sty@notes
91   \newlength{\slideframewidth}
92   \setlength{\slideframewidth}{1.5pt}
```

frame   We first define the keys.

```
93   \addmetakey{frame}{label}
94   \addmetakey[yes]{frame}{allowframebreaks}
95   \addmetakey{frame}{allowdisplaybreaks}
96   \addmetakey[yes]{frame}{fragile}
97   \addmetakey[yes]{frame}{shrink}
98   \addmetakey[yes]{frame}{squeeze}
99   \addmetakey[yes]{frame}{t}
```

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

```
100   \renewenvironment{frame}[1][]{%
101     \metasetkeys{frame}{#1}%
102     \sffamily%
103     \stepcounter{slide}%
104     \def\@currentlabel{\theslide}%
105     \ifx\frame@label\@empty%
106     \else\ifreinput\else%
107       \label{\frame@label}%
108     \fi\fi%
```

---

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

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

```
109      \def\itemize@level{outer}%
110      \def\itemize@outer{outer}%
111      \def\itemize@inner{inner}%
112      \renewcommand\newpage{\addtocounter{framenumber}{1}}%
113      \renewcommand\metakeys@show@keys[2]{\marginnote{{\scriptsize ##2}}}%
114      \renewenvironment{itemize}{%
115        \ifx\itemize@level\itemize@outer%
116          \def\itemize@label{$\rhd$}%
117        \fi%
118        \ifx\itemize@level\itemize@inner%
119          \def\itemize@label{$\scriptstyle\rhd$}%
120        \fi%
121        \begin{list}%
122        {\itemize@label}%
123        {\setlength{\labelsep}{.3em}%
124         \setlength{\labelwidth}{.5em}%
125         \setlength{\leftmargin}{1.5em}%
126        }%
127        \edef\itemize@level{\itemize@inner}%
128      }{%
129        \end{list}%
130      }%
```

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

```
131      \if@latexml\else\begin{mdframed}[linewidth=\slideframewidth,skipabove=1ex,skipbelow=1ex,use
132    }{%
133      \medskip\miko@slidelabel\if@latexml\else\end{mdframed}\fi%
134    }%
```

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

`\frametitle`

```
135      \renewcommand{\frametitle}[1]{{\Large\bf\sf\color{blue}{#1}}\medskip}%
136 \fi %ifmks@sty@notes
```

`\frameimage` We have to make sure that the width is overwritten, for that we check the
`\Gin@ewidth` macro from the `graphicx` package. We also add the `label` key.

```
137 \define@key{Gin}{label}{\def\@currentlabel{\arabic{slide}}\label{#1}}
138 \newrobustcmd\frameimage[2][]{%
139   \stepcounter{slide}%
140   \ifmks@sty@frameimages%
141     \def\Gin@ewidth{}\setkeys{Gin}{#1}%
142     \ifmks@sty@notes\else\vfill\fi%
143     \begin{center}
144       \ifmks@sty@fiboxed%
145         \fbox{\ifx\Gin@ewidth\@empty\includegraphics[width=\slidewidth,#1]{#2}\else\mygraphics[
146       \else
147         \ifx\Gin@ewidth\@empty\includegraphics[width=\slidewidth,#1]{#2}\else\mygraphics[#1]{#2
148       \fi% ifmks@sty@fiboxed
```

9

```
149    \end{center}
150    \par\strut\hfill{\footnotesize Slide \arabic{slide}}%
151    \ifmks@sty@notes\else\vfill\fi%
152  \fi} % ifmks@sty@frameimages
```

EdN:4   \pause [4]

```
153 \ifmks@sty@notes\newcommand\pause{}\fi
```

nomtext

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

nomgroup

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

ndefinition

```
158 \ifmks@sty@notes\newenvironment{ndefinition}[2][]{\begin{definition}[#1]{#2}}{\end{definition}}
159 \else\excludecomment{ndefinition}\fi%
```

nassertion

```
160 \ifmks@sty@notes\newenvironment{nassertion}[2][]{\begin{assertion}[#1]{#2}}{\end{assertion}}%
161 \else\excludecomment{nassertion}\fi%
```

nsproof

```
162 \ifmks@sty@notes\newenvironment{nsproof}[2][]{\begin{sproof}[#1]{#2}}{\end{sproof}}%
163 \else\excludecomment{nsproof}\fi%
```

nexample

```
164 \ifmks@sty@notes\newenvironment{nexample}[2][]{\begin{example}[#1]{#2}}{\end{example}}%
165 \else\excludecomment{nexample}\fi%
```

\inputref@*skip   We customize the hooks for in \inputref.

```
166 \def\inputref@preskip{\smallskip}
167 \def\inputref@postskip{\medskip}
```

\inputref*

```
168 \let\orig@inputref\inputref
169 \def\inputref{\@ifstar\ninputref\orig@inputref}
170 \newcommand\ninputref[2][]{\ifmks@sty@notes\orig@inputref[#1]{#2}\fi}
```

---
[4]EdNote: MK: fake it in notes mode for now

10

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 sTeX logo. Customization can be done by `\setslidelogo{⟨logo name⟩}`.

```
171 \newlength{\slidelogoheight}
172 \ifmks@sty@notes%
173   \setlength{\slidelogoheight}{.4cm}%
174 \else%
175   \setlength{\slidelogoheight}{1cm}%
176 \fi%
177 \newsavebox{\slidelogo}%
178 \sbox{\slidelogo}{\sTeX}%
179 \newrobustcmd{\setslidelogo}[1]{%
180   \sbox{\slidelogo}{\includegraphics[height=\slidelogoheight]{#1}}%
181 }%
```

\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.

```
182 \def\source{Michael Kohlhase}% customize locally
183 \newrobustcmd{\setsource}[1]{\def\source{#1}}%
```

\setlicensing Now, we set up the copyright and licensing. By default we use the Creative Commons Attribuition-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 optional.

```
184 \def\copyrightnotice{\footnotesize\copyright:\hspace{.3ex}{\source}}%
185 \newsavebox{\cclogo}%
186 \sbox{\cclogo}{\includegraphics[height=\slidelogoheight]{cc_somerights}}%
187 \newif\ifcchref\cchreffalse%
188 \AtBeginDocument{%
189   \@ifpackageloaded{hyperref}{\cchreftrue}{\cchreffalse}
190 }%
191 \def\licensing{%
192   \ifcchref%
193     \href{http://creativecommons.org/licenses/by-sa/2.5/}{\usebox{\cclogo}}%
194   \else%
195     {\usebox{\cclogo}}%
196   \fi%
197 }%
198 \newrobustcmd{\setlicensing}[2][]{%
199   \def\@url{#1}%
200   \sbox{\cclogo}{\includegraphics[height=\slidelogoheight]{#2}}%
201   \ifx\@url\@empty%
```

```
202     \def\licensing{{\usebox{\cclogo}}}%
203   \else%
204     \def\licensing{%
205       \ifcchref%
206       \href{#1}{\usebox{\cclogo}}%
207       \else%
208       {\usebox{\cclogo}}%
209       \fi%
210     }%
211   \fi%
212 }%
```

     \slidelabel   Now, we set up the slide label for the `article` mode.[5]

```
213 \newrobustcmd\miko@slidelabel{%
214   \vbox to \slidelogoheight{%
215     \vss\hbox to \slidewidth%
216     {\licensing\hfill\copyrightnotice\hfill\arabic{slide}\hfill\usebox{\slidelogo}}%
217   }%
218 }%
```

4.4   Colors and Highlighting

We first specify sans serif fonts as the default.

```
219 \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 to is to adapt the green so that it is dark enough for most beamers

```
220 \AtBeginDocument{%
221 \definecolor{green}{rgb}{0,.5,0}%
222 \definecolor{purple}{cmyk}{.3,1,0,.17}%
223 }%
```

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.

```
224 % \def\STpresent#1{\textcolor{blue}{#1}}
225 \def\defemph#1{{\textcolor{magenta}{#1}}}
226 \def\termemph#1{{\textcolor{cyan}{#1}}}
227 \def\notemph#1{{\textcolor{magenta}{#1}}}
228 \def\stDMemph#1{{\textcolor{blue}{#1}}}
229 \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.

---

[5]EDNOTE: see that we can use the themes for the slides some day. This is all fake.

```
230 \pgfdeclareimage[width=.8em]{miko@small@dbend}{dangerous-bend}
231 \def\smalltextwarning{%
232   \pgfuseimage{miko@small@dbend}%
233   \xspace%
234 }%
235 \pgfdeclareimage[width=1.2em]{miko@dbend}{dangerous-bend}
236 \newrobustcmd\textwarning{%
237   \raisebox{-.05cm}{\pgfuseimage{miko@dbend}}%
238   \xspace%
239 }%
240 \pgfdeclareimage[width=2.5em]{miko@big@dbend}{dangerous-bend}%
241 \newrobustcmd\bigtextwarning{%
242   \raisebox{-.05cm}{\pgfuseimage{miko@big@dbend}}%
243   \xspace%
244 }%

245 \newrobustcmd\putgraphicsat[3]{%
246   \begin{picture}(0,0)\put(#1){\includegraphics[#2]{#3}}\end{picture}%
247 }%
248 \newrobustcmd\putat[2]{%
249   \begin{picture}(0,0)\put(#1){#2}\end{picture}%
250 }%
```

4.5 Sectioning

If the `sectocframes` option is set, then we make section frames. We first define counters for `part` and `chapter`, which `beamer.cls` does not have and we make the `section` counter which it does dependent on `chapter`.

```
251 \ifmks@sty@sectocframes%
252 \ifdefstring\@@topsect{part}{%
253   \newcounter{chapter}\counterwithin*{section}{chapter}}
254 {\ifdefstring\@@topsect{chapter}{\newcounter{chapter}\counterwithin*{section}{chapter}}{}}
255 \fi% ifsectocframes
```

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

```
256 \RequirePackage{stex}
257 \RequirePackage{smglom}
258 \RequirePackage{tikzinput}
259 \ifmks@sty@mh\RequirePackage{mikoslides-mh}\fi
```

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

```
260 \section@level=2
261 \ifdefstring{\@@topsect}{part}
262 {\section@level=0%
263   \def\thesection{\arabic{chapter}.\arabic{section}}%
264   \def\part@prefix{\arabic{chapter}.}}{}
265 \ifdefstring{\@@topsect}{chapter}
266 {\section@level=1%
267   \def\thesection{\arabic{chapter}.\arabic{section}}%
```

13

```
268    \def\part@prefix{\arabic{chapter}.}}{}
269 \ifmks@sty@notes\else% only in slides
270 \renewenvironment{omgroup}[2][]{%
271    \metasetkeys{omgroup}{#1}\sref@target%
272    \advance\section@level by 1%
273    \advance\omgroup@level by 1%
274    \ifmks@sty@sectocframes%
275    \stepcounter{slide}
276    \begin{frame}[noframenumbering]%
277    \vfill\Large\centering%
278    \red{%
279      \ifcase\section@level\or
280      \stepcounter{part}
281      \def\@@label{\omdoc@part@kw~\Roman{part}}
282      \def\currentsectionlevel{\omdoc@part@kw}
283      \or%
284      \stepcounter{chapter}
285      \def\@@label{\omdoc@chapter@kw~\arabic{chapter}}
286      \def\currentsectionlevel{\omdoc@chapter@kw}
287      \or
288      \stepcounter{section}
289      \def\@@label{\part@prefix\arabic{section}}
290      \def\currentsectionlevel{\omdoc@section@kw}
291      \or
292      \stepcounter{subsection}
293      \def\@@label{\part@prefix\arabic{section}.\arabic{subsection}}
294      \def\currentsectionlevel{\omdoc@subsection@kw}
295      \or
296      \stepcounter{subsubsection}
297      \def\@@label{\part@prefix\arabic{section}.\arabic{subsection}.\arabic{subsubsection}}
298      \def\currentsectionlevel{\omdoc@subsubsection@kw}
299      \or
300      \stepcounter{mparagraph}
301      \def\@@label{\part@prefix\arabic{section}.\arabic{msubsection}.\arabic{subsubsection}.\arab
302      \def\currentsectionlevel{\omdoc@paragraph@kw}
303      \fi% end ifcase
304      \@@label\sref@label@id\@@label
305      \quad #2%
306    }%
307    \vfill%
308    \end{frame}%
309    \fi %ifmks@sty@sectocframes
310 }
311 {\advance\section@level by -1}%
312 \fi% ifmks@sty@notes
```

4.6 Excursions

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

```
313 \def\inserttheorembodyfont{\normalfont}
314 \ifmks@sty@notes\else% only in slides
315 \defbeamertemplate{theorem begin}{miko}
316 {\inserttheoremheadfont\inserttheoremname\inserttheoremnumber
317   \ifx\inserttheoremaddition\@empty\else\ (\inserttheoremaddition)\fi%
318   \inserttheorempunctuation\inserttheorembodyfont\xspace}
319 \defbeamertemplate{theorem end}{miko}{}
```

and we set it as the default one.

```
320 \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.

```
321 \expandafter\def\csname Parent2\endcsname{}
322 \fi% ifmks@sty@notes
323 \ifmks@sty@notes%
324   \renewenvironment{columns}[1][]{%
325     \par\noindent%
326     \begin{minipage}%
327     \slidewidth\centering\leavevmode%
328   }{%
329     \end{minipage}\par\noindent%
330   }%
331   \newsavebox\columnbox%
332   \renewenvironment<>{column}[2][]{%
333     \begin{lrbox}{\columnbox}\begin{minipage}{#2}%
334   }{%
335     \end{minipage}\end{lrbox}\usebox\columnbox%
336   }%
337 \fi% ifmks@sty@notes

338 \ifmks@sty@noproblems%
339   \newenvironment{problems}{}{}%
340 \else%
341   \excludecomment{problems}%
342 \fi%
```

`\excursion*` The excursion macros are very simple, we define a new internal macro in `\excursionref` and use it in `\excursion`, which is just an `\inputref` that checks if the new macro is defined before formatting the file in the argument.

```
343 \gdef\printexcursions{}
344 \newcommand\activateexcursion[2][]{\gappto\printexcursions{\inputref[#1]{#2}}}
345 \newcommand\excursionref[2]{% label, text
346 \ifnotes\begin{omtext}[title=Excursion]#2 \sref[fallback=the appendix]{#1}.\end{omtext}\fi}
347 \newcommand\excursion[4][]{% opt, label, path, text
348   \ifnotes\activateexcursion[#1]{#3}\excursionref{#2}{#4}\fi}
```

`\excursiongroup`

```
349 \srefaddidkey{excursiongroup}%
350 \addmetakey{excursiongroup}{intro}%
351 \newcommand\excursiongroup[1][]{%
352   \metasetkeys{excursiongroup}{#1}%
353   \ifdefempty\printexcursions{}% only if there are excursions
354   {\begin{omgroup}[#1]{Excursions}%
355       \ifdefempty\excursiongroup@intro{}{\inputref{\excursiongroup@intro}}%
356       \printexcursions%
357     \end{omgroup}}}
358 ⟨/package⟩
```

4.7 Miscellaneous

Change History

References

[CR99]   David Carlisle and Sebastian Rathz. *The* graphicxl *package*. Part of the TEX distribution. The Comprehensive TEX Archive Network. 1999. URL: https://www.tug.org/texlive/devsrc/Master/texmf-dist/doc/latex/graphics/graphicx.pdf.

[KGA20]   Michael Kohlhase, Deyan Ginev, and Rares Ambrus. *modules.sty: Semantic Macros and Module Scoping in sTeX*. Tech. rep. 2020. URL: https://github.com/sLaTeX/sTeX/raw/master/sty/modules/modules.pdf.

[Koh20a]   Michael Kohlhase. *MathHub Support for sTEX*. Tech. rep. 2020. URL: https://github.com/sLaTeX/sTeX/raw/master/sty/mathhub/mathhub.pdf.

[Koh20b]   Michael Kohlhase. *metakeys.sty: A generic framework for extensible Metadata in LATEX*. Tech. rep. 2020. URL: https://github.com/sLaTeX/sTeX/raw/master/sty/metakeys/metakeys.pdf.

[sTeX]   *sTeX: A semantic Extension of TeX/LaTeX*. URL: https://github.com/sLaTeX/sTeX (visited on 05/11/2020).

[Tana]   Till Tantau. *beamer – A LaTeX 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.