

Slides and Course Notes for Jacobs University*

Michael Kohlhasse
Jacobs University, Bremen
<http://kwarc.info/kohlhasse>

July 22, 2015

Abstract

We present a document class from which we can generate both course slides and course notes in a transparent way. Furthermore, we present a set of \LaTeX bindings for these, so that we can also generate OMDoc-based course materials, e.g. for inclusion in the ACTIVEMATH system.

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	3
2.4	Colors and Highlighting	3
2.5	Front Matter, Titles, etc	3
2.6	Miscellaneous	3
2.7	Support for MathHub	3
3	Limitations	4
4	The Implementation	5
4.1	Initialization and Class Options	5
4.2	Notes and Slides	7
4.3	Header and Footer Lines	10
4.4	Colors and Highlighting	11
4.5	Front Matter, Titles, etc	12
4.6	Sectioning	14
4.7	Miscellaneous	15
4.8	Support for MathHub	17
4.9	Finale	17

*Version ? (last revised ?)

1 Introduction

This Document class is derived from `beamer.cls` [`beamerclass:on`], specializes it with Jacobs stuff and adds a notes version 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 `STEX`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

- `slides` • The options `slidesnd` `notesnotes` switch between slides mode and notes mode (see Section 2.2).
- `a` • If the option `sectocframes` is given, then special frames with section table of contents are produced headers²
- `sectocframes` • `showmeta`. If this is set, then the metadata keys are shown (see [`Kohlhase:metakeys:ctan`] for details and customization options).
- `showmeta` • If the option `frameimages` is set, then slide mode also shows the `\frameimage`-generated frames.
- `frameimages`

2.2 Notes and Slides

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

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

²EDNOTE: document the functionality

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.

```

\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 `STEXnotes`. In this case we can use `\frameimage[⟨opt⟩]{⟨path⟩}`, where `⟨opt⟩` are the options of `\includegraphics` from the `graphicx` package [CarRah:tpp99] and `⟨path⟩` is the file path (extension can be left off like in `\includegraphics`).

`\frameimage`

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

2.7 Support for MathHub

Much of the `STEX` content is hosted on MathHub (<http://MathHub.info>), a portal and archive for flexiformal mathematics. MathHub offers GIT repositories (public and private escrow) for mathematical documentation projects, online and offline authoring and document development infrastructure, and a rich, interactive reading interface. The `modules` package supports repository-sensitive operations on MathHub.

Note that **MathHub** has two-level repository names of the form $\langle group \rangle / \langle repo \rangle$, where $\langle group \rangle$ is a **MathHub**-unique repository group and $\langle repo \rangle$ a repository name that is $\langle group \rangle$ -unique. The file and directory structure of a repository is arbitrary – except that it starts with the directory **source** because they are Math Archives in the sense of [HorIacJuc:cscpnrr11]. But this structure can be hidden from the \TeX author with **MathHub**-enabled versions of the **modules** macros.

\mhframeimage The **\mhframeimage** macro is a variant of **\frameimage** with repository support. Instead of writing

```
\defpath{MathHub}{/user/foo/lmh/MathHub}
\frameimage{\MathHub{fooMH/bar/source/baz/foobar}}
```

we can simply write (assuming that **\MathHub** is defined as above)

```
\mhframeimage[fooMH/bar]{baz/foobar}
```

Note that the **\mhframeimage** form is more semantic, which allows more advanced document management features in **MathHub**.

If **baz/foobar** is the “current module”, i.e. if we are on the **MathHub** path **...MathHub/fooMH/bar...**, then stating the repository in the first optional argument is redundant, so we can just use

```
\mhframeimage{baz/foobar}
```

Of course, neither \LaTeX nor \LaTeXML know about the repositories when they are called from a file system, so we can use the **\mhcurrentrepos** macro from the **modules** package to tell them. But this is only needed to initialize the infrastructure in the driver file. In particular, we do not need to set it in each module, since the **\importmhmodule** macro sets the current repository automatically.

Caveat if you want to use the **MathHub** support macros (let’s call them **mh**-variants), then every time a module is imported or a document fragment is included from another repos, the **mh**-variant **\importmhmodule** must be used, so that the “current repository” is set accordingly. To be exact, we only need to use **mh**-variants, if the imported module or included document fragment use **mh**-variants.

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 \TeX TRAC [sTeX:online].

1. the class should be divided into concerns. [sTeX:online], issue 1684
2. when option **book** or **report** is given together with **sectocframes** chapter-level omgroups generate a spurious slide with a bare heading. This has something to do with the fact that beamer does not support **\chapter**

4 The Implementation

The `mikoslides` package generates two files: the \LaTeX package (all the code between `\package` and `\endpackage`) and the \LaTeX XML bindings (between `\ltxml` and `\endltxml`). We keep the corresponding code fragments together, since the documentation applies to both of them and to prevent them from getting out of sync.

4.1 Initialization and Class Options

For the \LaTeX XML bindings, we make sure the right perl packages are loaded.

```
1 \ltxml
2 # -*- CPERL -*-
3 package LaTeXXML::Package::Pool;
4 use strict;
5 use LaTeXXML::Package;
6 DeclareOption('showmeta', sub {PassOptions('metakeys','sty',ToString(Digest(T_CS('\CurrentOption'))));});
7 DeclareOption('defindex', sub {PassOptions('statements','sty',ToString(Digest(T_CS('\CurrentOption'))));});
8 DeclareOption('notes', '');
9 DeclareOption('slides', '');
10 DeclareOption('nopproblems', '');
11 DeclareOption('sectocframes', '');
12 DeclareOption('frameimages', '');
13 DeclareOption('report', sub {PassOptions('omdoc','cls',ToString(Digest(T_CS('\CurrentOption'))));});
14 DeclareOption('book', sub {PassOptions('omdoc','cls',ToString(Digest(T_CS('\CurrentOption'))));});
15 DeclareOption(undef, sub {PassOptions('omdoc','cls',ToString(Digest(T_CS('\CurrentOption'))));});
16 ProcessOptions();
17 \ltxml
```

For \LaTeX we define some Package Options and switches for the `mikoslides` class and activate them by passing them on to `beamer.cls` the appropriate packages.

```
18 \cls
19 \DeclareOption{showmeta}{\PassOptionsToPackage{\CurrentOption}{metakeys}}
20 \DeclareOption{defindex}{\PassOptionsToPackage{\CurrentOption}{statements}}
21 \newif\ifnotes\notesfalse
22 \newif\ifsectocframes\sectocframesfalse
23 \newif\ifframeimages\frameimagesfalse
24 \newif\ifproblems\problemstrue
25 \DeclareOption{notes}{\notesttrue}
26 \DeclareOption{slides}{\notesfalse}
27 \DeclareOption{nopproblems}{\problemfalse}
28 \DeclareOption{sectocframes}{\sectocframetrue}
29 \DeclareOption{frameimages}{\frameimagestrue}
```

the next two define the `frontmatter` environment so that the later `\renewcommand` does not lead to trouble.

```
30 \newif\if@part\@partfalse
31 \DeclareOption{report}{\@parttrue\PassOptionsToClass{\CurrentOption}{omdoc}}
```

```

32 \DeclareOption{book}{\@parttrue\PassOptionsToClass{\CurrentOption}{omdoc}}
33 \DeclareOption*{\PassOptionsToClass{\CurrentOption}{omdoc}}
34 \PassOptionsToClass{\CurrentOption}{beamer}}
35 \ProcessOptions
36 </cls>
37 <*ltxml>
38 RawTeX('newif\ifnotes\notesfalse');
39 RawTeX('newif\ifproblems\problemsfalse');
40 </ltxml>

```

Depending on the options, we either load the `article`-based `omdoc` or the `beamer` class. In the first case, 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 \LaTeX packages.

```

41 <*cls>
42 \ifnotes
43 \LoadClass{omdoc}
44 \RequirePackage{a4wide}
45 \RequirePackage{marginnote}
46 \RequirePackage{mdframed}
47 \RequirePackage[notheorems,noamsthm,noxcolor]{beamerarticle}
48 \else
49 \LoadClass[notheorems,noamsthm,10pt]{beamer}
50 \newcounter{Item}
51 \newcounter{paragraph}
52 \newcounter{subparagraph}
53 \newcounter{Hfootnote}
54 \usetheme{Jacobs}
55 \fi
56 </cls>
57 <*ltxml>
58 LoadClass('omdoc');
59 RequirePackage('tikzinput');
60 DefConstructor('usetheme{'',''');
61 </ltxml>

```

EdN:3

now, we load the remaining packages for both versions. ³

```

62 <*cls>
63 \RequirePackage{stex}
64 \RequirePackage{tikzinput}
65 \RequirePackage{latexml}
66 \RequirePackage{amssymb}
67 \usepgflibrary{shapes}
68 \usetikzlibrary{arrows}
69 \usetikzlibrary{positioning}
70 \usetikzlibrary{tikzmark}%experimental/beta but very useful
71 \usetikzlibrary{fit}

```

³EDNOTE: MK: eventually (when tikz support is fully realized in \LaTeX ML) get rid of the standalone package

```

72 \RequirePackage{url}
73 \RequirePackage{amsmath}
74 \RequirePackage{comment}
75 \RequirePackage{standalone}
76 \RequirePackage{textcomp}
77 \</cls>
78 \<*txml>
79 \RequirePackage('stex');
80 \RequirePackage('latexml');
81 \RequirePackage('amssymb');
82 \RequirePackage('graphicx');
83 \RequirePackage('tikz');
84 \RequirePackage('url');
85 \RequirePackage('amsmath');
86 \</txml>

```

4.2 Notes and Slides

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

```

87 \<cls>
88 \newcounter{slide}
89 \newlength{\slidewidth}\setlength{\slidewidth}{12.8cm}
90 \newlength{\slideheight}\setlength{\slideheight}{9cm}
91 \</cls>
92 \<*txml>
93 \DefRegister('\slidewidth'      => Dimension('13.6cm'));
94 \DefRegister('\slideheight'    => Dimension('9cm'));
95 \</txml>

```

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.

```

96 \<cls>
97 \ifnotes%
98   \renewenvironment{note}{%
99     \ignorespaces%
100   }{}%
101 \else%
102   \excludecomment{note}%
103 \fi%
104 \</cls>
105 \<*txml>
106 \DefEnvironment('{note}', '#body');
107 \</txml>

```

We start by giving the L^AT_EX_ML binding for the `frame` environment from the `beamer` class. 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.

```

108 <*cls>
109 \ifnotes
110   \newlength{\slideframewidth}
111   \setlength{\slideframewidth}{1.5pt}

```

frame We first define the keys.

```

112   \addmetakey{frame}{label}
113   \addmetakey[yes]{frame}{allowframebreaks}
114   \addmetakey{frame}{allowdisplaybreaks}
115   \addmetakey[yes]{frame}{fragile}
116   \addmetakey[yes]{frame}{shrink}
117   \addmetakey[yes]{frame}{squeeze}

```

We redefine the `itemize` environment so that it looks more like the one in `beamer` with `Jacobs` theme. We create the box with the `mdframed` environment from the `equinymous` package. Then we define the environment, read them, and construct the slide number and label.

```

118   \renewenvironment{frame}[1][]{%
119     \metasetkeys{frame}{#1}%
120     \stepcounter{slide}%
121     \def\@currentlabel{\theslide}%
122     \ifx\frame@label\@empty%
123       \else%
124         \label{\frame@label}%
125       \fi%

```

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

```

126   \def\itemize@level{outer}%
127   \def\itemize@outer{outer}%
128   \def\itemize@inner{inner}%
129   \renewcommand\newpage{}%
130   \renewcommand\metakeys@showkeys[2]{\marginnote{{\scriptsize ##2}}}%
131   \renewenvironment{itemize}{%
132     \ifx\itemize@level\itemize@outer%
133       \def\itemize@label{${\rhd}$}%
134     \fi%
135     \ifx\itemize@level\itemize@inner%
136       \def\itemize@label{${\scriptstyle\rhd}$}%
137     \fi%
138     \begin{list}%
139     {\itemize@label}%
140     {\setlength{\labelsep}{.3em}%
141      \setlength{\labelwidth}{.5em}%
142      \setlength{\leftmargin}{1.5em}%
143     }%
144     \edef\itemize@level{\itemize@inner}%
145   }{%
146     \end{list}%

```



```

147     }%
    We create the box with the mdframed environment from the equinymous package.
148     \begin{mdframed}[linewidth=\slideframewidth,skipabove=1ex,skipbelow=1ex,userdefinedwidth=\s
149     ]{
150     \medskip\miko@slidelabel\end{mdframed}%
151     }%
152 \</cls>
153 \<*txml>
154 DefEnvironment('{frame}[]',
155     "<omdoc:omgroup layout='slide'>"
156     .   "#body\n"
157     . "</omdoc:omgroup>\n\n",
158     afterDigestBegin=>sub {
159     $_[1]->setProperty(theory=>LookupValue('current_module')); });
160 \</txml>#\$

```

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

```

\frametitle
161 \<*cls>
162 \renewcommand{\frametitle}[1]{\Large\bf\sf\color{blue}{#1}}\medskip}%
163 \fi %ifnotes
164 \</cls>
165 \<*txml>
166 DefConstructor('\frametitle{',
167     "\n<omdoc:metadata><dc:title>#1</dc:title></omdoc:metadata>");
168 \</txml>

```

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
169 \<*cls>
170 \newrobustcmd\frameimage[2][{}]{
171     \stepcounter{slide}%
172     \ifframeimages%
173     \def\Gin@ewidth{}\setkeys{Gin}{#1}%
174     \ifnotes%
175     \else%
176     \vfill%
177     \fi%
178     \ifx\Gin@ewidth\@empty%
179     \mycgraphics[width=\slidewidth,#1]{#2}\else\mycgraphics[#1]{#2}%
180     \fi%
181     \par\strut\hfill{\footnotesize Slide \arabic{slide}}}%
182     \ifnotes%
183     \else%
184     \vfill%
185     \fi%

```

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

```

186 \fi%
187 }% iframeimages
188 \end{class}
189 \end{document}
190 DefMacro('frameimage[]{}', '\@frameimage{\includegraphics[#1,width=\slidewidth]{#2}}');
191 DefConstructor('\@frameimage{}', "<omdoc:omgroup layout='slide'>#1</omdoc:omgroup>\n");
192 \end{document}

```

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>}`.

```

193 \begin{class}
194 \newlength{\slidelogoheight}
195 \ifnotes%
196 \setlength{\slidelogoheight}{.4cm}%
197 \else%
198 \setlength{\slidelogoheight}{1cm}%
199 \fi%
200 \newsavebox{\slidelogo}%
201 \sbox{\slidelogo}{\includegraphics[height=\slidelogoheight]{jacobs-logo}}%
202 \newrobustcmd{\setslidelogo}[1]{%
203 \sbox{\slidelogo}{\includegraphics[height=\slidelogoheight]{#1}}%
204 }%

```

`\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.

```

205 \def\source{Michael Kohlhase}% customize locally
206 \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 optional.

```

207 \def\copyrightnotice{\footnotesize\copyright:\hspace{.3ex}{\source}}%
208 \newsavebox{\cclogo}%
209 \sbox{\cclogo}{\includegraphics[height=\slidelogoheight]{cc_somerights}}%
210 \newif\ifcchref\cchreffalse%
211 \AtBeginDocument{%
212 \ifpackageloaded{hyperref}{\cchreftrue}{\cchreffalse}%
213 }%
214 \def\licensing{%
215 \ifcchref%

```

```

216 \href{http://creativecommons.org/licenses/by-sa/2.5/}{\usebox{\cclogo}}%
217 \else%
218 {\usebox{\cclogo}}%
219 \fi%
220 }%
221 \newrobustcmd{\setlicensing}[2][]{%
222 \def\@url{#1}%
223 \sbox{\cclogo}{\includegraphics[height=\slidelogoheight]{#2}}%
224 \ifx\@url\@empty%
225 \def\licensing{\usebox{\cclogo}}%
226 \else%
227 \def\licensing{%
228 \ifcchref%
229 \href{#1}{\usebox{\cclogo}}%
230 \else%
231 {\usebox{\cclogo}}%
232 \fi%
233 }%
234 \fi%
235 }%

```

EdN:5

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

```

236 \newrobustcmd{\miko@slidelabel}{%
237 \vbox to \slidelogoheight{%
238 \vss\hbox to \slidewidth{%
239 {\licensing\hfill\copyrightnotice\hfill\arabic{slide}\hfill\usebox{\slidelogo}}%
240 }%
241 }%

```

4.4 Colors and Highlighting

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

```

242 \AtBeginDocument{%
243 \definecolor{green}{rgb}{0,.5,0}%
244 \definecolor{purple}{cmyk}{.3,1,0,.17}%
245 }%

```

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`.

```

246 % \def\STpresent#1{\textcolor{blue}{#1}}
247 \def\defemph#1{{\textcolor{magenta}{#1}}}
248 \def\notemph#1{{\textcolor{magenta}{#1}}}
249 \def\stDMemph#1{{\textcolor{blue}{#1}}}

```

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

```

250 \def\@lec#1{(\textcolor{green}{#1})}
251 \end{class}
252 \end{document}
253 \defmacro{'}{\textcolor{magenta}{#1}};
254 \defmacro{'}{\textcolor{magenta}{#1}};
255 \end{document}

```

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.

```

256 \end{class}
257 \pgfdeclareimage[width=.9em]{miko@small@dbend}{dangerous-bend}
258 \def\smalltextwarning{%
259   \pgfuseimage{miko@small@dbend}%
260   \xspace%
261 }%
262 \pgfdeclareimage[width=1.5em]{miko@dbend}{dangerous-bend}
263 \newrobustcmd\textwarning{%
264   \raisebox{-.05cm}{\pgfuseimage{miko@dbend}}%
265   \xspace%
266 }%
267 \pgfdeclareimage[width=2.5em]{miko@big@dbend}{dangerous-bend}%
268 \newrobustcmd\bigtextwarning{%
269   \raisebox{-.05cm}{\pgfuseimage{miko@big@dbend}}%
270   \xspace%
271 }%
272 \end{class}
273 \end{document}
274 \defmacro{'}{\textwarning', '@textwarning\xspace'};
275 \defmacro{'}{\textwarning', ""};
276 \end{document}

```

4.5 Front Matter, Titles, etc

We need to redefine the frontmatter macros inherited from the `beamer` class for LaTeXML, since there they take an optional argument.

```

277 \end{document}
278 \defmacro{'}{\title{'}', '@add@frontmatter{ltx:title}{#1}};
279 \defmacro{'}{\date{'}', '@add@frontmatter{ltx:date}{role=creation}{#1}};
280 \defmacro{'}{\author{'}', sub { andSplit(T_CS('@author'),$_[1]); });#$
281 \end{document}

```

Now, we specialize the slide environment that we have implemented above or inherited from `seminar.cls` for some abbreviations, e.g. separator slides and title slides.

```

282 \end{class}
283 \ifnotes%
284   \newrobustcmd\titleframe{\maketitle}%
285 \else%

```

```

286 \newrobustcmd\titleframe{%
287   \begin{frame}%
288   \titlepage%
289   \end{frame}%
290 }%
291 \fi%
292 \newenvironment{titleframewith}{%
293   \begin{frame}%
294   \titlepage%
295 }{%
296   \end{frame}%
297 }%
298 \newenvironment{tttitle}{%
299   \begin{center}%
300   \LARGE%
301   \begin{tabular}{|c|}%
302   \hline%
303   }{%
304     \\ \hline%
305   \end{tabular}%
306   \end{center}%
307   \vspace{1ex minus 1ex}%
308 }%
309 \newenvironment{tttitlejoint}[1]{%
310   \newbox\boxwith%
311   \setbox\boxwith\hbox{%
312     \begin{tabular}{c}{\em joint work with}\#1\end{tabular}%
313   }%
314   \begin{center}%
315   \LARGE%
316   \begin{tabular}{c}%
317   \color{red}%
318   }{%
319     \\ \box\boxwith%
320   \end{tabular}%
321   \end{center}%
322   \vspace{1ex minus 1ex}%
323 }%
324 \</cls>
325 \<*\txml>
326 DefConstructor('\titleframe',"<omdoc:ignore>titleframe elided here</omdoc:ignore>");
327 DefEnvironment('{titleframewith}',
328   "<omdoc:ignore>begin elided titleframe</omdoc:ignore>"
329   . "#body"
330   . "<omdoc:ignore>end elided titleframe</omdoc:ignore>");
331 DefEnvironment('{titleslide}', "");
332 DefEnvironment('{titleslide}', "<omdoc:omgroup>#body</omdoc:omgroup>");
333 DefEnvironment('{tttitle}', "\n<dc:title>#body</dc:title>");
334 \</txml>

```

```

335 %      Must be first command on slide to make positioning work.
336 <*cls>
337 \newrobustcmd\putgraphicsat[3]{%
338   \begin{picture}(0,0)\put(#1){\includegraphics[#2]{#3}}\end{picture}%
339 }%
340 \newrobustcmd\putat[2]{%
341   \begin{picture}(0,0)\put(#1){#2}\end{picture}%
342 }%
343 </cls>

```

4.6 Sectioning

If the `sectocframes` option is set, then we make section frames.

```

344 <*cls>
345 \ifsectocframes%
346   \if@part%
347     \newcounter{mpart}%
348     \newcounter{mchapter}%
349     \newcounter{msection}[mchapter]%
350   \else%
351     \newcounter{msection}%
352   \fi%
353   \newcounter{msubsection}[msection]%
354   \newcounter{msubsubsection}[msubsection]%
355   \newcounter{msubsubsubsection}[msubsubsection]%
356   \ifnotes%
357   \else% only in slides
358     \renewcommand\at@begin@omgroup[3][ ]{%
359       \begin{frame}%
360       \vfill\Large\centering%
361       \red{%
362         \ifcase\section@level\or%
363           \stepcounter{mpart}Part \Roman{mpart}\or%
364           \stepcounter{mchapter}Chapter \arabic{mchapter}\or
365           \stepcounter{msection}\if@part\arabic{mchapter}.\fi\arabic{msection}\or
366           \stepcounter{msubsection}\if@part\arabic{mchapter}.\fi\arabic{msection}.\arabic{msubsec
367           \stepcounter{msubsubsection}\if@part\arabic{mchapter}.\fi\arabic{msection}.\arabic{msub
368           \stepcounter{msubsubsubsection}\if@part\arabic{mchapter}.\fi\arabic{msection}.\arabic{m
369           \quad #3%
370         }%
371       \vfill%
372       \end{frame}%
373     }%
374   \fi% ifnotes
375 \fi% ifsectocframes
376 </cls>

```

4.7 Miscellaneous

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.

```

377 <*cls>
378 \expandafter\def\csname Parent2\endcsname{}
379 % \begin{macrocode}
380 %
381 % We need to disregard the columns macros introduced by the |beamer| class
382 % \begin{macrocode}
383 \ifnotes%
384 \renewenvironment{columns}{%
385 \par\noindent%
386 \begin{minipage}%
387 \slidewidth\centering\leavevmode%
388 }{%
389 \end{minipage}\par\noindent%
390 }%
391 \newsavebox\columnbox%
392 \renewenvironment{column}[1]{%
393 \begin{lrbox}{\columnbox}\begin{minipage}{#1}%
394 }{%
395 \end{minipage}\end{lrbox}\usebox\columnbox%
396 }%
397 \fi%
398 </cls>
399 <*txml>
400 DefEnvironment('{columns}','"#body");
401 DefEnvironment('{column}{}','"#body");

```

We also need to deal with overlay specifications introduced by the `beamer` class.⁶

```

7
402 DefConstructor('\uncover','#1');
403 #Define a Beamer Overlay Parameter type
404 DefParameterType('BeamerOverlay', sub {
405 my ($gullet) = @_;
406 my $tok = $gullet->readXToken;
407 if (ref $tok && ToString($tok) eq '<') {
408 $gullet->readUntil(T_OTHER('>'));
409 } else {
410 $gullet->unread($tok) if ref $tok;
411 undef; },
412 reversion=> sub {
413 (T_OTHER('<'), $_[0]->revert, T_OTHER('>'));
414 });
415

```

⁶EDNOTE: this is just to keep latexml quiet, no real functionality here.

⁷EDNOTE: Deyan: We reuse the CMP itemizations defined in the `omdoc.cls` latexml binding, adjusting the parameters to be overlay-sensitive

```

416 #Take the "from" field of the overlay range
417 sub overlayFrom {
418   return "" unless defined $_[0];
419   my $overlay=ToString($_[0]); $overlay =~ /\d+/; $1;}
420
421 #Reuse the CMP itemizations, only adjust the \item constructors.
422 DefMacro('\beamer@group@item[] OptionalBeamerOverlay IfBeginFollows', sub {
423   my($gullet,$tag,$overlay,$needwrapper)=@_;
424   $overlay=$overlay||T_OTHER("");
425   ( T_CS('\group@item@maybe@unwrap'),
426     ($needwrapper ? (Invocation(T_CS('\beamer@group@item@wrap'),$tag,$overlay)->unlist) : ()))
427 DefConstructor('\beamer@group@item@wrap {} OptionalBeamerOverlay',
428   "<omdoc:omtext ?#2(overlay='&overlayFrom(#2)')()>"
429   . "?#1(<dc:title>#1</dc:title>())"
430   . "<omdoc:CMP>",
431   beforeDigest=>sub {
432     Let('\group@item@maybe@unwrap','\group@item@unwrap');
433     $_[0]->bgroup;
434   return; },
435   properties=>sub{ RefStepItemCounter(); });
436 #DefConstructor('\beamer@itemize@item[] OptionalBeamerOverlay',
437 #   "<omdoc:li ?#2(overlay='&overlayFrom(#2)')() >"
438 #   . "?#1(<dc:title>#1</dc:title>())",
439 #   properties=>sub{ RefStepItemCounter(); });
440 DefConstructor('\beamer@enumerate@item[] OptionalBeamerOverlay',
441   "<omdoc:li ?#2(overlay='&overlayFrom(#2)')() >"
442   . "?#1(<dc:title>#1</dc:title>())",
443   properties=>sub{ RefStepItemCounter(); });
444 DefConstructor('\beamer@description@item[] OptionalBeamerOverlay',
445   "<omdoc:di ?#2(overlay='&overlayFrom(#2)')() >"
446   . "?#1(<omdoc:dt>#1</omdoc:dt>())<omdoc:dd>", # trust di and dt to autoclose
447   properties=>sub{ RefStepItemCounter(); });
448 </ltxml>#&

```

Now, some things that are imported from the `pgf` and `beamer` packages:

```

449 <*ltxml>
450 DefMacro('\putgraphicsat{}{}{}','\mygraphics[#2]{#3}');
451 DefMacro('\putat{}{}','\#2');
452 </ltxml>
453 <*cls>
454 \ifproblems%
455   \newenvironment{problems}{}{}%
456 \else%
457   \excludecomment{problems}%
458 \fi%
459 </cls>
460 <*ltxml>
461 DefEnvironment('{problems}','\#body');
462 </ltxml>

```


4.8 Support for MathHub

`\mhframeimage` Use the current value of `\mh@currentrepos` or the value of the `mhrepos` key if it is given in `\frameimage`.

```
463 <cls>\addmetakey{Gin}{mhrepos}
464 <ltxml>DefKeyVal('Gin','mhrepos','Semiverbatim');
465 <ltxml>RawTeX('
466 <*ltxml | cls>
467 \newcommand\mhframeimage[2][]{%
468   \metasetkeys{Gin}{#1}%
469   \edef\mh@crepos{\mh@currentrepos}%
470   \ifx\Gin@mhrepos\empty%
471     \frameimage[#1]{\MathHub{\mh@currentrepos/source/#2}}%
472   \else%
473     \frameimage[#1]{\MathHub{\Gin@mhrepos/source/#2}}%
474   \fi%
475 }%
476 </ltxml | cls>
477 <ltxml>');
```

4.9 Finale

Finally, we set the slide body font to the sans serif, and we terminate the \LaTeX ML bindings file with a success mark for perl.

```
478 <cls>\ifnotes\else\sffamily
479 <ltxml>1;
```