

Slides and Course Notes for Jacobs University*

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

January 7, 2014

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	9
4.4	Colors and Highlighting	9
4.5	Front Matter, Titles, etc	10
4.6	Sectioning	11
4.7	Miscellaneous	12
4.8	Support for MathHub	13
4.9	Finale	14

*Version ? (last revised ?)

1 Introduction

This Document class is derived from `beamer.cls` [Tana], 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 \LaTeX 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

- The options `slides` and `notes` switch between slides mode and notes mode (see Section 2.2).
- If the option `sectocframes` is given, then special frames with section table of contents are produced headers²
- `showmeta`. If this is set, then the metadata keys are shown (see [Koh13] for details and customization options).

2.2 Notes and Slides

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 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.

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

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

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[$\langle opt \rangle$]{ $\langle path \rangle$ }`, where $\langle opt \rangle$ are the options of `\includegraphics` from the `graphicx` package [CR99] and $\langle path \rangle$ is the file path (extension can be left off like in `\includegraphics`).

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 \LaTeX 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 [Hor+11]. But this structure can be hidden from the `\TeXauthor` 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 `STEXTRAC` [sTeX].

1. the class should be divided into concerns. [sTeX], 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 # -*- PERL -*-
3 package LaTeXXML::Package::Pool;
4 use strict;
5 use LaTeXXML::Package;
6 \endltxml
```

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.

```
7 \cls
8 \DeclareOption{showmeta}{\PassOptionsToPackage{\CurrentOption}{metakeys}}
9 \newif\ifnotes\notesfalse
10 \newif\ifsectocframes\sectocframesfalse
11 \newif\ifproblems\problemstrue
12 \DeclareOption{notes}{\notestrue}
13 \DeclareOption{slides}{\notesfalse}
14 \DeclareOption{nopproblems}{\problemsfalse}
15 \DeclareOption{sectocframes}{\sectocframestrue}
```

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

```
16 \newif\if@part\@partfalse
17 \DeclareOption{report}{\@parttrue\PassOptionsToClass{\CurrentOption}{omdoc}}
18 \DeclareOption{book}{\@parttrue\PassOptionsToClass{\CurrentOption}{omdoc}}
19 \DeclareOption*{\PassOptionsToClass{\CurrentOption}{omdoc}
20               \PassOptionsToClass{\CurrentOption}{beamer}}
21 \ProcessOptions
22 \cls
23 \ltxml
24 \RawTeX{'\newif\ifnotes\notesfalse'};
25 \RawTeX{'\newif\ifproblems\problemsfalse'};
26 \endltxml
```

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.

```

27 <*cls>
28 \ifnotes
29 \LoadClass{omdoc}
30 \RequirePackage{a4wide}
31 \RequirePackage{marginnote}
32 \RequirePackage{mdframed}
33 \RequirePackage[notheorems,noamsthm,noxcolor]{beamerarticle}
34 \else
35 \if@part% report or book class
36 \renewenvironment{frontmatter}{}{}
37 \fi
38 \LoadClass[notheorems,noamsthm,10pt]{beamer}
39 \newcounter{Item}
40 \newcounter{paragraph}
41 \newcounter{subparagraph}
42 \newcounter{Hfootnote}
43 \usetheme{Jacobs}
44 \fi
45 </cls>
46 <*ltxml>
47 LoadClass('omdoc');
48 RequirePackage('tikzinput');
49 DefConstructor('\usetheme{','}');
50 </ltxml>

```

EdN:3

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

```

51 <*cls>
52 \RequirePackage{tikzinput}
53 \RequirePackage{stex}
54 \RequirePackage{latexml}
55 \RequirePackage{amssymb}
56 \RequirePackage{tikz}
57 \usepgflibrary{shapes}
58 \usetikzlibrary{arrows}
59 \usetikzlibrary{positioning}
60 \usetikzlibrary{tikzmark}%experimental/beta but very useful
61 \usetikzlibrary{fit}
62 \RequirePackage{url}
63 \RequirePackage{amsmath}
64 \RequirePackage{comment}
65 \RequirePackage{standalone}
66 \RequirePackage{textcomp}
67 </cls>
68 <*ltxml>
69 RequirePackage('stex');

```

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

```

70 \RequirePackage('latexml');
71 \RequirePackage('amssymb');
72 \RequirePackage('graphicx');
73 \RequirePackage('tikz');
74 \RequirePackage('url');
75 \RequirePackage('amsmath');
76 \ltxml

```

4.2 Notes and Slides

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

```

77 \*cls
78 \newcounter{slide}
79 \newlength{\slidewidth}\setlength{\slidewidth}{12.5cm}
80 \newlength{\slideheight}\setlength{\slideheight}{9cm}
81 \*cls
82 \ltxml
83 \DefRegister('slidewidth'      => Dimension('13.5cm'));
84 \DefRegister('slideheight'    => Dimension('9cm'));
85 \ltxml

```

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.

```

86 \*cls
87 \ifnotes\renewenvironment{note}{\ignorespaces}{\else\excludecomment{note}\fi
88 \*cls
89 \ltxml
90 \DefEnvironment('{note}','body');
91 \ltxml

```

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.

```

92 \*cls
93 \ifnotes
94 \newlength{\slideframewidth}\setlength{\slideframewidth}{1.5pt}

```

frame We first define the keys.

```

95 \addmetakey{frame}{label}
96 \addmetakey[yes]{frame}{allowframebreaks}
97 \addmetakey{frame}{allowdisplaybreaks}
98 \addmetakey[yes]{frame}{fragile}
99 \addmetakey[yes]{frame}{shrink}
100 \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.

```

101 \renewenvironment{frame}[1][]{%
102 {\metasetkeys{frame}{#1}%
103 \stepcounter{slide}\def\@currentlabel{\theslide}%
104 \ifx\frame@label\@empty\else\label{\frame@label}\fi
105 \def\itemize@level{outer}%
106 \def\itemize@outer{outer}%
107 \def\itemize@inner{inner}%
108 \renewcommand\newpage{}%
109 \renewcommand\metakeys@show@keys[2]{\marginnote{{\scriptsize ##2}}}%
110 \renewenvironment{itemize}%
111 {\ifx\itemize@level\itemize@outer\def\itemize@label{\$ \rhd$}\fi%
112 \ifx\itemize@level\itemize@inner\def\itemize@label{\$ \scriptstyle \rhd$}\fi%
113 \begin{list}%
114   {\itemize@label}%
115   {\setlength{\labelsep}{.3em}\setlength{\labelwidth}{.5em}\setlength{\leftmargin}{1.5em}}%
116   \edef\itemize@level{\itemize@inner}}%
117 {\end{list}}

```

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

```

118 \begin{mdframed}[linewidth=\slideframewidth,skipabove=1ex,skipbelow=1ex,
119 userdefinedwidth=\slidewidth,align=center]\sf}
120 {\medskip\miko@slidelabel\end{mdframed}}
121 \end{cls}
122 \end{*txml}
123 DefEnvironment('frame') ,
124   "<omdoc:omgroup layout='slide'>"
125   .   "#body\n"
126   . "</omdoc:omgroup>\n\n",
127   afterDigestBegin=>sub {
128     $_[1]->setProperty(theory=>LookupValue('current_module')); };
129 \end{*txml}#

```

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

```

\frametitle
130 \end{cls}
131 \renewcommand{\frametitle}[1]{\Large\bf\sffcolor{blue}{#1}\medskip}
132 \fi
133 \end{cls}
134 \end{*txml}
135 DefConstructor('\frametitle') ,
136   "\n<omdoc:metadata><dc:title>#1</dc:title></omdoc:metadata>";
137 \end{*txml}

```


EdN:4

```
\frameimage We have to make sure that the width is overwritten4
138 <*cls>
139 \newcommand\frameimage[2] [] {\def\Gin@ewidth{}\setkeys{Gin}{#1}%
140 \ifx\Gin@ewidth\empty\mycgraphics[#1]{#2}\else\mycgraphics[width=\slidewidth,#1]{#2}\fi}
141 </cls>
142 <*txml>
143 DefMacro(' \frameimage[] {}', '\@frameimage{\includegraphics[#1,width=\slidewidth]{#2}});
144 DefConstructor(' \@frameimage{}', "<omdoc:omgroup layout='slide'>#1</omdoc:omgroup>\n");
145 </txml>
```

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.

```
146 <*cls>
147 \newlength{\slidelogoheight}
148 \ifnotes\setlength{\slidelogoheight}{.4cm}\else\setlength{\slidelogoheight}{1cm}\fi
149 \newsavebox{\slidelogo}\sbox{\slidelogo}{\includegraphics[height=\slidelogoheight]{jacobs-logo}}
```

Now, we set up the copyright and licensing, the copyright remains with the author, but we use the Creative Commons Attribution-ShareAlike license to strengthen den public domain. Here the problem is that we want a hyperref on the CC logo, if hyperref is loaded, and otherwise not. As hyperref is always loaded, we have to find out at the beginning of the document whether it is, set up a switch, and later in the footer line decide what to do.

```
150 \def\source{Michael Kohlase}% customize locally
151 \def\copyrightnotice{\footnotesize\copyright:\hspace{.3ex}{\source}}
152 \newsavebox{\cclogo}\sbox{\cclogo}{\includegraphics[height=\slidelogoheight]{cc_somerights}}
153 \newif\ifcchref\cchreffalse
154 \AtBeginDocument{\ifpackageloaded{hyperref}{\cchreftrue}{\cchreffalse}}
155 \def\licensing{\ifcchref\href{http://creativecommons.org/licenses/by-sa/2.5/}{\usebox{\cclogo}}
```

EdN:5

Now, we set up the slide label for the article mode⁵

```
\slidelabel
156 \newcommand\miko@slidelabel%
157 {\vbox to \slidelogoheight{\vss\hbox to \slidewidth%
158 {\licensing\hfill\copyrightnotice\hfill\arabic{slide}\hfill\usebox{\slidelogo}}}}
```

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

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

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

```
159 \AtBeginDocument{\definecolor{green}{rgb}{0,.5,0}\definecolor{purple}{cmyk}{.3,1,0,.17}}
```

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`.

```
160 % \def\STpresent#1{\textcolor{blue}{#1}}
161 \def\defemph#1{\textcolor{magenta}{#1}}
162 \def\notemph#1{\textcolor{magenta}{#1}}
163 \def\stDMemph#1{\textcolor{blue}{#1}}
164 \def\@@lec#1{\textcolor{green}{#1}}
165 \<cls>
166 \<ltxml>
167 #DefMacro('defemph','\textcolor{magenta}{#1}');
168 #DefMacro('notemph','\textcolor{magenta}{#1}');
169 \</ltxml>
```

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.

```
170 \<cls>
171 \pgfdeclareimage[width=.9em]{miko@small@dbend}{dangerous-bend}
172 \def\smalltextwarning{\pgfuseimage{miko@small@dbend}\xspace}
173 \pgfdeclareimage[width=1.5em]{miko@dbend}{dangerous-bend}
174 \def\textwarning{\raisebox{-.05cm}{\pgfuseimage{miko@dbend}}\xspace}
175 \pgfdeclareimage[width=2.5em]{miko@big@dbend}{dangerous-bend}
176 \def\bigtextwarning{\raisebox{-.05cm}{\pgfuseimage{miko@big@dbend}}\xspace}
177 \</cls>
178 \<ltxml>
179 DefMacro('textwarning','\textwarning\xspace');
180 DefConstructor('\textwarning',"");
181 \</ltxml>
```

4.5 Front Matter, Titles, etc

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

```
182 \<ltxml>
183 DefMacro('title[]{}', '\@add@frontmatter{ltx:title}{#1}');
184 DefMacro('date[]{}', '\@add@frontmatter{ltx:date}[role=creation]{#1}');
185 DefMacro('author[]{}', sub { andSplit(T_CS('\@author'),$_[1]); };\#\$
186 \</ltxml>
```

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.

```
186 \<cls>
187 \ifnotes\newcommand\titleframe{\maketitle}\else
188 \newcommand\titleframe{\begin{frame}\titlepage\end{frame}}\fi
189 \newenvironment{titleframewith}{\begin{frame}\titlepage}{\end{frame}}
```

```

190 \newenvironment{tttitle}{\begin{center}\LARGE\begin{tabular}{|c|}\hline}%
191 {\hline\end{tabular}\end{center}\vspace{1ex minus 1ex}}
192 \newenvironment{tttitlejoint}[1]%
193 {\newbox\boxwith\setbox\boxwith\hbox{\begin{tabular}{c}{\em joint work with}\#1\end{tabular}}}%
194 \begin{center}\LARGE\begin{tabular}{c}\color{red}}}%
195 {\box\boxwith\end{tabular}\end{center}%
196 \vspace{1ex minus 1ex}}
197 \end{cls}
198 \end{*txml}
199 DefConstructor('titleframe',"<omdoc:ignore>titleframe elided here</omdoc:ignore>");
200 DefEnvironment('titleframewith',
201               "<omdoc:ignore>begin elided titleframe</omdoc:ignore>"
202               . "#body"
203               . "<omdoc:ignore>end elided titleframe</omdoc:ignore>");
204 DefEnvironment('titleslide','');
205 DefEnvironment('titleslide',"<omdoc:omgroup>#body</omdoc:omgroup>");
206 DefEnvironment('tttitle',"<dc:title>#body</dc:title>");
207 \end{*txml}

208 %      Must be first command on slide to make positioning work.
209 \end{*cls}
210 \newcommand\putgraphicsat[3]{%
211   \begin{picture}(0,0)\put(#1){\includegraphics[#2]{#3}}\end{picture}}
212 \newcommand\putat[2]{\begin{picture}(0,0)\put(#1){#2}\end{picture}}
213 \end{cls}

```

4.6 Sectioning

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

```

214 \end{*cls}
215 \ifsectocframes
216 \if@part\newcounter{mpart}
217 \newcounter{mchapter}
218 \newcounter{msection}[mchapter]
219 \else
220 \newcounter{msection}
221 \fi
222 \newcounter{msubsection}[msection]
223 \newcounter{msubsubsection}[msubsection]
224 \newcounter{msubsubsubsection}[msubsubsection]
225 \ifnotes\else% only in slides
226 \renewcommand\at@begin@omgroup[3][\begin{frame}%
227 \vfill\LARGE\centering
228 \red{\ifcase\section@level\or
229 \stepcounter{mpart}Part \Roman{mpart}\or%
230 \stepcounter{mchapter}Chapter \arabic{mchapter}\or
231 \stepcounter{msection}\if@part\arabic{mchapter}.\fi\arabic{msection}\or
232 \stepcounter{msubsection}\if@part\arabic{mchapter}.\fi\arabic{msection}.\arabic{msubsection}\or
233 \stepcounter{msubsubsection}\if@part\arabic{mchapter}.\fi\arabic{msection}.\arabic{msubsection}
234 \stepcounter{msubsubsubsection}\if@part\arabic{mchapter}.\fi\arabic{msection}.\arabic{msubsecti

```

```

235 \quad #3}\vfill
236 \end{frame}}
237 \fi% ifnotes
238 \fi% ifsectocframes
239 \</cls>

```

4.7 Miscellaneous

We need to disregard the columns macros introduced by the `beamer` class

```

240 \<*<cls>
241 \ifnotes
242 \renewenvironment{columns}%
243 {\par\noindent\begin{minipage}\slidewidth\centering\leavevmode}%
244 {\end{minipage}\par\noindent}
245 \newsavebox\columnbox
246 \renewenvironment{column}[1]%
247 {\begin{lrbox}{\columnbox}\begin{minipage}{#1}}%
248 {\end{minipage}\end{lrbox}\usebox\columnbox}
249 \fi
250 \</cls>
251 \<*<ltxml>
252 DefEnvironment('{columns}','#body');
253 DefEnvironment('{column}{}','#body');

```

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

```

254 DefConstructor('\uncover','#1');
255 #Define a Beamer Overlay Parameter type
256 DefParameterType('BeamerOverlay', sub {
257   my ($gullet) = @_;
258   my $tok = $gullet->readXToken;
259   if (ref $tok && ToString($tok) eq '<') {
260     $gullet->readUntil(T_OTHER('>'));
261   } else {
262     $gullet->unread($tok) if ref $tok;
263     undef; }},
264   reversion=> sub {
265     (T_OTHER('<'), $_[0]->revert, T_OTHER('>'));
266   });
267
268 #Take the "from" field of the overlay range
269 sub overlayFrom {
270   return "" unless defined $_[0];
271   my $overlay=ToString($_[0]); $overlay =~ /\^(\\d+)/; $1;}
272

```

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

```

273 #Reuse the CMP itemizations, only adjust the \item constructors.
274 DefMacro('\beamer@group@item[] OptionalBeamerOverlay IfBeginFollows', sub {
275   my($gullet,$tag,$overlay,$needwrapper)=@_;
276   $overlay=$overlay||T_OTHER("");
277   ( T_CS('\group@item@maybe@unwrap'),
278     ($needwrapper ? (Invocation(T_CS('\beamer@group@item@wrap'),$tag,$overlay)->unlist) : ()) )
279 DefConstructor('\beamer@group@item@wrap {} OptionalBeamerOverlay',
280   "<omdoc:omtext ?#2(overlay='&overlayFrom(#2)')()>"
281   . "?#1(<dc:title>#1</dc:title>())"
282   . "<omdoc:CMP>",
283   beforeDigest=>sub {
284     Let('\group@item@maybe@unwrap','\group@item@unwrap');
285     $_[0]->bgroup;
286     return; },
287   properties=>sub{ RefStepItemCounter(); });
288 #DefConstructor('\beamer@itemize@item[] OptionalBeamerOverlay',
289 #   "<omdoc:li ?#2(overlay='&overlayFrom(#2)')() >"
290 #   . "?#1(<dc:title>#1</dc:title>())",
291 #   properties=>sub{ RefStepItemCounter(); });
292 DefConstructor('\beamer@enumerate@item[] OptionalBeamerOverlay',
293   "<omdoc:li ?#2(overlay='&overlayFrom(#2)')() >"
294   . "?#1(<dc:title>#1</dc:title>())",
295   properties=>sub{ RefStepItemCounter(); });
296 DefConstructor('\beamer@description@item[] OptionalBeamerOverlay',
297   "<omdoc:di ?#2(overlay='&overlayFrom(#2)')() >"
298   . "?#1(<omdoc:dt>#1</omdoc:dt>())<omdoc:dd>", # trust di and dt to autoclose
299   properties=>sub{ RefStepItemCounter(); });
300 </ltxml>#&

```

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

```

301 <*ltxml>
302 DefMacro('\putgraphicsat{}{}{}','\mygraphics[#2]{#3}');
303 DefMacro('\putat{}{}','\#2');
304 </ltxml>
305 <*cls>
306 \ifproblems\newenvironment{problems}{}{}\else\excludecomment{problems}\fi
307 </cls>
308 <*ltxml>
309 DefEnvironment('{problems}','\#body');
310 </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`.

```

311 <ltxml>RawTeX('
312 <*ltxml | cls>
313 \addmetakey{Gin}{mhrepos}
314 \newcommand\mhframeimage[2][\metasetkeys{Gin}{#1}%

```

```

315 \edef\mh@@repos{\mh@currentrepos}%
316 \ifx\Gin@mhrepos@empty\frameimage[#1]{\MathHub{\mh@currentrepos/source/#2}}%
317 \else\frameimage[#1]{\MathHub{\Gin@mhrepos/source/#2}}\fi}
318 \</ltxml | cls>
319 \<ltxml>' );

```

4.9 Finale

Finally, we set the slide body font to the sans serif, and we terminate the L^AT_EXML bindings file with a success mark for perl.

```

320 \<cls>\ifnotes\else\sans\fi
321 \<ltxml>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>.
- [Hor+11] Fulya Horozal et al. “Combining Source, Content, Presentation, Narration, and Relational Representation”. In: *Intelligent Computer Mathematics*. Ed. by James Davenport et al. LNAI 6824. Springer Verlag, 2011, pp. 212–227. ISBN: 978-3-642-22672-4. URL: http://kwarc.info/frabe/Research/HIJKR_dimensions_11.pdf.
- [Koh13] Michael Kohlhase. *metakeys.sty: A generic framework for extensible Metadata in L^AT_EX*. Self-documenting L^AT_EX package. Comprehensive T_EX Archive Network (CTAN), 2013. URL: <http://www.ctan.org/tex-archive/macros/latex/contrib/stex/metakeys/metakeys.pdf>.
- [sTeX] *Semantic Markup for L^AT_EX*. Project Homepage. URL: <http://trac.kwarc.info/sTeX/> (visited on 02/22/2011).
- [Tana] Till Tantau. *beamer – A L^AT_EX class for producing presentations and slides*. URL: <http://www.ctan.org/pkg/beamer> (visited on 01/07/2014).
- [Tanb] Till Tantau. *User Guide to the Beamer Class*. URL: <http://www.ctan.org/tex-archive/macros/latex/contrib/beamer/doc/beameruserguide.pdf>.