

hwexam.sty/cls: An Infrastructure for formatting Assignments and Exams*

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

November 5, 2015

Abstract

The **hwexam** package and class allows individual course assignment sheets and compound assignment documents using problem files marked up with the **problem** package.

Contents

1	Introduction	2
2	The User Interface	2
2.1	Package and Class Options	2
2.2	Assignments	2
2.3	Typesetting Exams	2
2.4	Including Assignments	3
3	Limitations	4
4	Implementation: The hwexam Class	5
4.1	Class Options	5
5	Implementation: The hwexam Package	6
5.1	Package Options	6
5.2	Assignments	7
5.3	Including Assignments	10
5.4	Typesetting Exams	11
5.5	Leftovers	13

*Version v1.1 (last revised 2015/11/04)

1 Introduction

The `hwexam` package and class supplies an infrastructure that allows to format nice-looking assignment sheets by simply including problems from problem files marked up with the `problem` package [Koh15c]. It is designed to be compatible with `problems.sty`, and inherits some of the functionality.

2 The User Interface

2.1 Package and Class Options

The `hwexam` package and class take the options `solutions`, `notes`, `hints`, `pts`, `min`, and `boxed` that are just passed on to the `problems` package (cf. its documentation for a description of the intended behavior).

`showmeta` If the `showmeta` option is set, then the metadata keys are shown (see [Koh15a] for details and customization options).

The `hwexam` class additionally accepts the options `report`, `book`, `chapter`, `part`, and `showignores`, of the `omdoc` package [Koh15b] on which it is based and passes them on to that. For the `extrefs` option see [Koh15d].

2.2 Assignments

`assignment` This package supplies the `assignment` environment that groups problems into assignment sheets. It takes an optional `KeyVal` argument with the keys `number` (for the assignment number; if none is given, 1 is assumed as the default or — in multi-assignment documents — the ordinal of the `assignment` environment), `title` (for the assignment title; this is referenced in the title of the assignment sheet), `type` (for the assignment type; e.g. “quiz”, or “homework”), `given` (for the date the assignment was given), and `due` (for the date the assignment is due).

2.3 Typesetting Exams

`multiple` Furthermore, the `hwexam` package takes the option `multiple` that allows to combine multiple assignment sheets into a compound document (the assignment sheets are treated as section, there is a table of contents, etc.).

`test` Finally, there is the option `test` that modifies the behavior to facilitate formatting tests. Only in `test` mode, the macros `\testspace`, `\testnewpage`, and `\testemptypage` have an effect: they generate space for the students to solve the given problems. Thus they can be left in the \LaTeX source.

`\testspace` `\testspace` takes an argument that expands to a dimension, and leaves vertical space accordingly. `\testnewpage` makes a new page in `test` mode, and `\testemptypage` generates an empty page with the cautionary message that this page was intentionally left empty.

`testheading` Finally, the `\testheading` takes an optional keyword argument where the keys `duration` specifies a string that specifies the duration of the test, `min` specifies the equivalent in number of minutes, and `reqpts` the points that are required for a

perfect grade.

```

\title{320101 General Computer Science (Fall 2010)}
\begin{testheading}[duration=one hour,min=60,reqpts=27]
  Good luck to all students!
\end{testheading}

```

formats to

Name:

Matriculation Number:

320101 General Computer Science (Fall 2010)

November 5, 2015

You have one hour(sharp) for the test;
 Write the solutions to the sheet.
 The estimated time for solving this exam is 58 minutes, leaving you 2 minutes for
 revising your exam.
 You can reach 30 points if you solve all problems. You will only need 27 points
 for a perfect score, i.e. 3 points are bonus points.

*Different problems test different skills and knowledge, so
 do not get stuck on one problem.*

	To be used for grading, do not write here								
prob.	1.1	2.1	2.2	2.3	3.1	3.2	3.3	Sum	grade
total	4	4	6	6	4	4	2	30	
reached									

good luck

Example 1: A generated test heading.

2.4 Including Assignments

\includeassignment

number
title
type
given
due

The `\includeassignment` macro can be used to include an assignment from another file. It takes an optional `KeyVal` argument and a second argument which is a path to the file containing the problem (the macro assumes that there is only one `assignment` environment in the included file). The keys `number`, `title`, `type`, `given`, and `due` are just as for the `assignment` environment and (if given) overwrite the ones specified in the `assignment` environment in the included file.

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. none reported yet.

4 Implementation: The hwexam Class

The functionality is spread over the `hwexam` class and package. The class provides the `document` environment and pre-loads some convenience packages, whereas the package provides the concrete functionality.

`hwexam.dtx` generates four files: `hwexam.cls` (all the code between `<*cls>` and `</cls>`), `hwexam.sty` (between `<*package>` and `</package>`) and their L^AT_EXML bindings (between `<*ltxml.cls>` and `</ltxml.cls>` and `<*ltxml.sty>` and `</ltxml.sty>` respectively). We keep the corresponding code fragments together, since the documentation applies to both of them and to prevent them from getting out of sync.

first the general setup for L^AT_EXML(for the class and package)

```
1 <ltxml.cls | ltxml.sty>
2 # -*- CPERL -*-
3 package LaTeXML::Package::Pool;
4 use strict;
5 use LaTeXML::Package;
6 use LaTeXML::Util::Pathname;
7 use Cwd qw(cwd abs_path);
8 </ltxml.cls | ltxml.sty>
```

4.1 Class Options

To initialize the `hwexam` class, we declare and process the necessary options by passing them to the respective packages and classes they come from.

```
9 <*cls>
10 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{hwexam}}
11 \PassOptionsToClass{\CurrentOption}{omdoc}
12 \ProcessOptions
13 </cls>
14 <*ltxml.cls>
15 DeclareOption(undef,sub {PassOptions('hwexam','sty',ToString(Digest(T_CS('\CurrentOption'))));});
16 DeclareOption(undef,sub {PassOptions('omdoc','cls',ToString(Digest(T_CS('\CurrentOption'))));});
17 ProcessOptions();
18 </ltxml.cls>
```

We load `omdoc.cls`, and the desired packages. For the L^AT_EXML bindings, we make sure the right packages are loaded.

```
19 <*cls>
20 \LoadClass{omdoc}
21 \RequirePackage{stex}
22 \RequirePackage{hwexam}
23 \RequirePackage{tikzinput}
24 \RequirePackage{graphicx}
25 \RequirePackage{a4wide}
26 \RequirePackage{amssymb}
27 \RequirePackage{amstext}
28 \RequirePackage{amsmath}
29 </cls>
```

```

30 <*ltxml.cls>
31 LoadClass('omdoc');
32 RequirePackage('stex');
33 RequirePackage('hwexam');
34 RequirePackage('tikzinput', options => ['image']);
35 RequirePackage('graphicx');
36 RequirePackage('amssymb');
37 RequirePackage('amstext');
38 RequirePackage('amsmath');
39 </ltxml.cls>

```

EdN:1 Finally, we register another keyword for the document environment¹

```

40 <*cls>
41 \newcommand\assig@default@type{\hwexam@assignment@kw}
42 \addmetakey[\assig@default@type]{document}{hwexamtype}
43 </cls>

```

5 Implementation: The hwexam Package

5.1 Package Options

The first step is to declare (a few) package options that handle whether certain information is printed or not. Some come with their own conditionals that are set by the options, the rest is just passed on to the `problems` package.

```

44 <*package>
45 \newif\iftest\testfalse
46 \DeclareOption{test}{\testtrue}
47 \newif\ifmultiple\multiplefalse
48 \DeclareOption{multiple}{\multipletrue}
49 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{problem}}
50 \ProcessOptions
51 </package>

```

Then we make sure that the necessary packages are loaded (in the right versions).

```

52 <*package>
53 \RequirePackage{keyval}[1997/11/10]
54 \RequirePackage{problem}
55 </package>

```

Here comes the equivalent header information for L^AT_EXML, we also initialize the package inclusions. Since L^AT_EXML does not handle options yet, we have nothing to do.

```

56 <*ltxml>
57 DeclareOption('test', '');
58 DeclareOption('multiple', '');
59 DeclareOption(undef, sub {PassOptions('problem','sty',ToString(Digest(T_CS('\CurrentOption'))))}
60 ProcessOptions();

```

¹EdNOTE: MK: this still needs to be internationalized.

```

61 RequirePackage('problem');
62 </ltxml>

    Then we register the namespace of the requirements ontology
63 <*ltxml>
64 RegisterNamespace('assig'=>"http://omdoc.org/ontology/assignments#");
65 RegisterDocumentNamespace('assig'=>"http://omdoc.org/ontology/assignments#");
66 </ltxml>

```

\hwexam@*@kw For multilinguality, we define internal macros for keywords that can be specialized in *.ldf files.

```

67 <*package>
68 \AfterBabelLanguage{ngerman}{\input{hwexam-ngerman.ldf}}
69 \newcommand\hwexam@assignment@kw{Assignment}
70 \newcommand\hwexam@given@kw{Given}
71 \newcommand\hwexam@due@kw{Due}
72 </package>

```

5.2 Assignments

Then we set up a counter for problems and make the problem counter inherited from `problem.sty` depend on it. Furthermore, we specialize the `\prob@label` macro to take the assignment counter into account.

```

73 <*package>
74 \newcounter{assignment}
75 \numberproblemsin{assignment}
76 \renewcommand\prob@label[1]{\arabic{assignment}.#1}

```

We will prepare the keyval support for the `assignment` environment.

```

77 \srefaddidkey{assig}
78 \addmetakey{assig}{number}
79 \addmetakey*{assig}{title}
80 \addmetakey{assig}{type}
81 \addmetakey{assig}{given}
82 \addmetakey{assig}{due}
83 \addmetakey[false]{assig}{loadmodules}[true]

```

The next three macros are intermediate functions that handle the case gracefully, where the respective token registers are undefined.

The `\given@due` macro prints information about the given and due status of the assignment. Its arguments specify the brackets.

```

84 \newcommand\given@due[2]{%
85 \ifx \inclassig@given\@empty
86 \ifx \assig@given\@empty
87 \ifx \inclassig@due\@empty
88 \ifx \assig@due\@empty% all empty do nothing
89 \else #1%
90 \fi
91 \else #1%

```

```

92 \fi
93 \else #1%
94 \fi
95 \else #1%
96 \fi
97 \ifx\inclassig@given\@empty
98 \ifx\assig@given\@empty% do nothing
99 \else \hwexam@given@kw\xspace \assig@given%
100 \fi
101 \else \hwexam@given@kw\xspace \inclassig@given%
102 \fi
103 \ifx \inclassig@due\@empty
104 \ifx \assig@due\@empty% do nothing
105 \else
106 \ifx \inclassig@given\@empty
107 \ifx \assig@given\@empty% do nothing
108 \else ,~%
109 \fi
110 \else ,~%
111 \fi
112 \fi
113 \else
114 \ifx \inclassig@given\@empty
115 \ifx \assig@given\@empty% do nothing
116 \else ,~%
117 \fi
118 \else ,~%
119 \fi
120 \fi
121 \ifx \inclassig@due\@empty
122 \ifx \assig@due\@empty% do nothing
123 \else \hwexam@due@kw\xspace \assig@due%
124 \fi
125 \else \hwexam@due@kw\xspace \inclassig@due%
126 \fi
127 \ifx \inclassig@given\@empty
128 \ifx \assig@given\@empty
129 \ifx \inclassig@due\@empty
130 \ifx \assig@due\@empty% all empty do nothing
131 \else #2%
132 \fi
133 \else #2%
134 \fi
135 \else #2%
136 \fi
137 \else #2%
138 \fi
139 }

```

`\assignment@title` This macro prints the title of an assignment, the local title is overwritten, if there

is one from the `\includeassignment`. `\assignment@title` takes three arguments the first is the fallback when no title is given at all, the second and third go around the title, if one is given.

```
140 \newcommand\assignment@title[3]
141 {\ifx\inclassig@title\empty% if there is no outside title
142 \ifx\assig@title\empty{#1}\else{#2\assig@title{#3}}\fi
143 \else{#2}\inclassig@title{#3}\fi}% else show the outside title
```

`\assignment@number` Like `\assignment@title` only for the number, and no around part.

```
144 \newcommand\assignment@number%
145 {\ifx\inclassig@number\empty% if there is no outside number
146 \ifx\assig@number\empty\else\assig@number\fi
147 \else\inclassig@number\fi}% else show the outside number
```

With them, we can define the central `assignment` environment. This has two forms (separated by `\ifmultiple`) in one we make a title block for an assignment sheet, and in the other we make a section heading and add it to the table of contents. We first define an assignment counter

`assignment` For the `assignment` environment we delegate the work to the `@assignment` environment that depends on whether `multiple` option is given.

```
148 \newenvironment{assignment}[1] [] {\metasetkeys{assig}{#1}\sref@target%
149 \edef\@@num{\assignment@number}%
150 \ifx\@@num\empty\stepcounter{assignment}\else\setcounter{assignment}{\@@num}\fi%
151 \setcounter{problem}{0}%
152 \def\current@section@level{\document@hwexamtype}%
153 \sref@label{id{\document@hwexamtype \thesection}%
154 \begin{@assignment}}
155 {\end{@assignment}}}
```

In the multi-assignment case we just use the `omdoc` environment for suitable sectioning.

```
156 \ifmultiple
157 \newenvironment{@assignment}%
158 {\ifx\assig@loadmodules\@true
159 \begin{omgroup}[loadmodules]{\protect\document@hwexamtype~\arabic{assignment}%
160 \assignment@title{\;({})\;}\given@due{}}
161 \else
162 \begin{omgroup}{\protect\document@hwexamtype~\arabic{assignment}%
163 \assignment@title{\;({})\;}\given@due{}}
164 \fi%
165 {\protect\document@hwexamtype~\arabic{assignment}%
166 \assignment@title{\;({})\;}\given@due{}}}
167 {\end{omgroup}}}
```

for the single-page case we make a title block from the same components.

```
168 \else
169 \newenvironment{@assignment}
170 {\begin{center}\bf
```

```

171 \Large\@title\strut\
172 \document@hwexamtype~\arabic{assignment}\assignment@title{\;}{:\;}{\;\}%
173 \large\given@due{--\;}{\;\;--}
174 \end{center}}
175 {}
176 \fi% multiple
177 \</package>

178 \<!--*xml-->
179 DefEnvironment('{assignment} OptionalKeyVals:assig',
180 " <omdoc:omgroup ?&GetKeyVal(#1,'id')(xml:id='&GetKeyVal(#1,'id')')() "
181 . " assig:dummy='for the namespace'"
182 . " <omdoc:metadata>"
183 . " <dc:title>"
184 . " Assignment ?&GetKeyVal(#1,'num')(&GetKeyVal(#1,'num').)()"
185 . " ?&GetKeyVal(#1,'title')(&GetKeyVal(#1,'title')))"
186 . " </dc:title>"
187 . " ?&GetKeyVal(#1,'given')(<omdoc:meta property='assig:given'>&GetKeyVal(#1,'given')</omdoc:meta>)"
188 . " ?&GetKeyVal(#1,'due')(<omdoc:meta property='assig:due'>&GetKeyVal(#1,'due')</omdoc:meta>)"
189 . " ?&GetKeyVal(#1,'pts')(<omdoc:meta property='assig:pts'>&GetKeyVal(#1,'pts')</omdoc:meta>)"
190 . " </omdoc:metadata>"
191 . " #body"
192 . " </omdoc:omgroup>\n"#,
193 # afterDigest=> sub {
194 # my ($stomach, $kv) = @_;
195 # my $kvi = LookupValue('inclassig');
196 # my @keys = qw(id num title pts given due);
197 # my @vals = $kvi && map($kvi->getValue($_), @keys);
198 # foreach my $i(0..$#vals) {
199 # $kv->setValue($keys[$i],$vals[$i]) if $vals[$i];
200 # }
201 );#$
202 \</xml-->

```

5.3 Including Assignments

`\in*assignment` This macro is essentially a glorified `\include` statement, it just sets some internal macros first that overwrite the local points. Importantly, it resets the `inclassig` keys after the input.

```

203 \<!--*package-->
204 \addmetakey{inclassig}{number}
205 \addmetakey*{inclassig}{title}
206 \addmetakey{inclassig}{type}
207 \addmetakey{inclassig}{given}
208 \addmetakey{inclassig}{due}
209 \addmetakey{inclassig}{mhrepos}
210 \clear@inclassig@keys%initially
211 \newcommand\includeassignment[2][\metasetkeys{inclassig}{#1}%
212 \include{#2}\clear@inclassig@keys}
213 \newcommand\inputassignment[2][\metasetkeys{inclassig}{#1}%

```

```

214 \input{#2}\clear@inclassig@keys}
215 \end{package}
216 \end{*txml}
217 DefMacro('includeassignment [] {}', sub {
218   my ($stomach, $arg1, $arg2) = @_;
219   AssignValue('inclassig', $arg1) if $arg1;
220   (Invocation(T_CS('input'), $arg2)->unlist);
221 });
222 DefMacro('inputassignment [] {}', 'includeassignment[#1]{#2}');
223 \end{*txml}

```

5.4 Typesetting Exams

\quizheading

```

224 \begin{package}
225 \addmetakey{quizheading}{tas}
226 \newcommand\quizheading[1]{\def\@tas{#1}%
227 \large\noindent NAME: \hspace{8cm} MAILBOX:\[2ex]%
228 \ifx\@tas\empty\else%
229 \noindent TA: \@for\@I:=\@tas\do{\Large$\Box$}\@I\hspace*{1em}}\[2ex]\fi}

```

\testheading

```

230 \addmetakey{testheading}{min}
231 \addmetakey{testheading}{duration}
232 \addmetakey{testheading}{reqpts}
233 \newenvironment{testheading}[1][\metasetkeys{testheading}{#1}
234 {\noindent\large{Name: \hfill Matriculation Number:\hspace*{2cm}\strut}\[1ex]
235 \begin{center}\Large\textbf{\@title}\[1ex]\large\@date\[3ex]\end{center}
236 {\textbf{You have
237 \ifx\test@heading@duration\empty\testheading@min minutes\else\testheading@duration\fi
238 (sharp) for the test}};\[1ex] Write the solutions to the sheet.}\par\noindent
239
240 \newcount\check@time\check@time=\testheading@min
241 \advance\check@time by -\theassignment@totalmin
242 The estimated time for solving this exam is {\theassignment@totalmin} minutes,
243 leaving you {\the\check@time} minutes for revising your exam.
244
245 \newcount\bonus@pts\bonus@pts=\theassignment@totalpts
246 \advance\bonus@pts by -\testheading@reqpts
247 You can reach {\theassignment@totalpts} points if you solve all problems. You will only need
248 {\testheading@reqpts} points for a perfect score, i.e.\ {\the\bonus@pts} points are
249 bonus points. \vfill
250 \begin{center}
251   {\Large\em
252 % You have ample time, so take it slow and avoid rushing to mistakes!\[2ex]
253   Different problems test different skills and knowledge, so do not get stuck on
254   one problem.}\vfill\par\correction@table \[3ex]
255 \end{center}}
256 {\newpage}

```

```

257 </package>
258 <*!xml>
259 DefEnvironment('{testheading}OptionalKeyVals:omdoc','');
260 </!xml>

\testspace
261 <*package>
262 \newcommand\testspace[1]{\iftest\vspace*{#1}\fi}
263 </package>
264 <*!xml>
265 DefConstructor('\testspace{}','');
266 </!xml>

\testnewpage
267 <*package>
268 \newcommand\testnewpage{\iftest\newpage\fi}
269 </package>
270 <*!xml>
271 DefConstructor('\testnewpage','');
272 </!xml>

\testemptypage
273 <*package>
274 \newcommand\testemptypage[1][\iftest\begin{center}This page was intentionally left
275     blank for extra space\end{center}\vfill\eject\else\fi}
276 </package>
277 <*!xml>
278 DefConstructor('\testemptypage','');
279 </!xml>

\@problem This macro acts on a problem's record in the *.aux file. Here we redefine it to
generate the correction table.
280 <*package>
281 \renewcommand\@problem[3]{\stepcounter{assignment@probs}
282 \def\@@pts{#2}\ifx\@@pts\@empty\else\addtocounter{assignment@totalpts}{#2}\fi
283 \def\@@min{#3}\ifx\@@min\@empty\else\addtocounter{assignment@totalmin}{#3}\fi
284 \xdef\correction@probs{\correction@probs & #1}%
285 \xdef\correction@pts{\correction@pts & #2}
286 \xdef\correction@reached{\correction@reached &}}
287 </package>

\correction@table This macro generates the correction table
288 <*package>
289 \newcounter{assignment@probs}
290 \newcounter{assignment@totalpts}
291 \newcounter{assignment@totalmin}
292 \newcommand\correction@probs{prob.}%
293 \newcommand\correction@pts{total}%
294 \newcommand\correction@reached{reached}%

```

```

295 \stepcounter{assignment@probs}
296 \newcommand\correction@table{\begin{tabular}{|l|*{\theassignment@probs}{c|}|l|}\hline%
297 &\multicolumn{\theassignment@probs}{c|}|%|
298 {\footnotesize To be used for grading, do not write here} &\\\hline
299 \correction@probs & Sum & grade\\\hline
300 \correction@pts & \theassignment@totalpts & \\\hline
301 \correction@reached & & \[.7cm]\hline
302 \end{tabular}}
303 \end{package}

```

5.5 Leftovers

at some point, we may want to reactivate the logos font, then we use

```

here we define the logos that characterize the assignment
\font\bierfont=../assignments/bierglas
\font\denkerfont=../assignments/denker
\font\uhrfont=../assignments/uhr
\font\warnschildfont=../assignments/achtung

\newcommand\bierglas{{\bierfont\char65}}
\newcommand\denker{{\denkerfont\char65}}
\newcommand\uhr{{\uhrfont\char65}}
\newcommand\warnschild{{\warnschildfont\char 65}}
\newcommand\hardA{\warnschild}
\newcommand\longA{\uhr}
\newcommand\thinkA{\denker}
\newcommand\discussA{\bierglas}

```

Finally, we need to terminate the file with a success mark for perl.

```

304 \ltxml>1;

```

Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used.

L^AT_EX^{ML}, 5, 6

References

- [Koh15a] Michael Kohlhase. *metakeys.sty: A generic framework for extensible Metadata in L^AT_EX*. Tech. rep. Comprehensive T_EX Archive Network (CTAN), 2015. URL: <http://www.ctan.org/tex-archive/macros/latex/contrib/stex/metakeys/metakeys.pdf>.
- [Koh15b] Michael Kohlhase. *omdoc.sty/cls: Semantic Markup for Open Mathematical Documents in L^AT_EX*. Tech. rep. Comprehensive T_EX Archive Network (CTAN), 2015. URL: <http://www.ctan.org/tex-archive/macros/latex/contrib/stex/omdoc/omdoc.pdf>.
- [Koh15c] Michael Kohlhase. *problem.sty: An Infrastructure for formatting Problems*. Tech. rep. Comprehensive T_EX Archive Network (CTAN), 2015. URL: <http://www.ctan.org/tex-archive/macros/latex/contrib/stex/problem/problem.pdf>.
- [Koh15d] Michael Kohlhase. *sref.sty: Semantic Crossreferencing in L^AT_EX*. Tech. rep. Comprehensive T_EX Archive Network (CTAN), 2015. URL: <http://www.ctan.org/tex-archive/macros/latex/contrib/stex/sref/sref.pdf>.
- [sTeX] KWARC/sTeX. URL: <https://svn.kwarc.info/repos/stex> (visited on 05/15/2015).