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

Michael Kohlhase
Jacobs University, Bremen
`http://kwarc.info/kohlhase`

November 7, 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
`number` 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` `title` (for the assignment title; this is referenced in the title of the assignment
`type` sheet), `type` (for the assignment type; e.g. "quiz", or "homework"), `given` (for
`given` the date the assignment was given), and `due` (for the date the assignment is due).
`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 ver-
`\testnewpage` tical space accordingly. `\testnewpage` makes a new page in `test` mode, and
`\testemptypage` `\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` `duration` specifies a string that specifies the duration of the test, `min` specifies the
`min` equivalent in number of minutes, and `reqpts` the points that are required for a
`reqpts`

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 7, 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`

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.

`number`
`title`
`type`
`given`
`due`

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 LaTeXML bindings (between ⟨*ltxml.cls⟩ and ⟨/ltxml.cls⟩ and ⟨*ltxml.sty⟩ and ⟨/ltxml.sty⟩ respetively). 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 LaTeXML(for the class and package)

```
⟨*ltxml.cls | ltxml.sty⟩
# -*- CPERL -*-
package LaTeXML::Package::Pool;
use strict;
use LaTeXML::Package;
use LaTeXML::Util::Pathname;
use Cwd qw(cwd abs_path);
⟨/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.

```
⟨*cls⟩
\DeclareOption*{\PassOptionsToPackage{\CurrentOption}{hwexam}
                              \PassOptionsToClass{\CurrentOption}{omdoc}}
\ProcessOptions
⟨/cls⟩
⟨*ltxml.cls⟩
DeclareOption(undef,sub {PassOptions('hwexam','sty',ToString(Digest(T_CS('\CurrentOption')))); }
DeclareOption(undef,sub {PassOptions('omdoc','cls',ToString(Digest(T_CS('\CurrentOption')))); }
ProcessOptions();
⟨/ltxml.cls⟩
```

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

```
⟨*cls⟩
\LoadClass{omdoc}
\RequirePackage{stex}
\RequirePackage{hwexam}
\RequirePackage{tikzinput}
\RequirePackage{graphicx}
\RequirePackage{a4wide}
\RequirePackage{amssymb}
\RequirePackage{amstext}
\RequirePackage{amsmath}
⟨/cls⟩
```

```
⟨*ltxml.cls⟩
LoadClass('omdoc');
RequirePackage('stex');
RequirePackage('hwexam');
RequirePackage('tikzinput', options => ['image']);
RequirePackage('graphicx');
RequirePackage('amssymb');
RequirePackage('amstext');
RequirePackage('amsmath');
⟨/ltxml.cls⟩
```

EdN:1 Finally, we register another keyword for the `document` environment[1]

```
⟨*cls⟩
\newcommand\assig@default@type{\hwexam@assignment@kw}
\addmetakey[\assig@default@type]{document}{hwexamtype}
⟨/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.

```
⟨*package⟩
\newif\iftest\testfalse
\DeclareOption{test}{\testtrue}
\newif\ifmultiple\multiplefalse
\DeclareOption{multiple}{\multipletrue}
\DeclareOption*{\PassOptionsToPackage{\CurrentOption}{problem}}
\ProcessOptions
⟨/package⟩
```

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

```
⟨*package⟩
\RequirePackage{keyval}[1997/11/10]
\RequirePackage{problem}
⟨/package⟩
```

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

```
⟨*ltxml⟩
DeclareOption('test', '');
DeclareOption('multiple', '');
DeclareOption(undef, sub {PassOptions('problem','sty',ToString(Digest(T_CS('\CurrentOption'))))
ProcessOptions();
```

[1] EdNote: MK: this still needs to be internationalized.

```
RequirePackage('problem');
⟨/ltxml⟩
```

Then we register the namespace of the requirements ontology

```
⟨*ltxml⟩
RegisterNamespace('assig'=>"http://omdoc.org/ontology/assignments#");
RegisterDocumentNamespace('assig'=>"http://omdoc.org/ontology/assignments#");
⟨/ltxml⟩
```

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

```
⟨*package⟩
\AfterBabelLanguage{ngerman}{\input{hwexam-ngerman.ldf}}
\newcommand\hwexam@assignment@kw{Assignment}
\newcommand\hwexam@given@kw{Given}
\newcommand\hwexam@due@kw{Due}
⟨/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.

```
⟨*package⟩
\newcounter{assignment}
\numberproblemsin{assignment}
\renewcommand\prob@label[1]{\arabic{assignment}.#1}
```

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

```
\srefaddidkey{assig}
\addmetakey{assig}{number}
\addmetakey*{assig}{title}
\addmetakey{assig}{type}
\addmetakey{assig}{given}
\addmetakey{assig}{due}
\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.

```
\newcommand\given@due[2]{%
\ifx \inclassig@given\@empty
\ifx \assig@given\@empty
\ifx \inclassig@due\@empty
\ifx \assig@due\@empty% all empty do nothing
\else #1%
\fi
\else #1%
```

```
\fi
\else #1%
\fi
\else #1%
\fi
\ifx\inclassig@given\@empty
\ifx\assig@given\@empty% do nothing
\else \hwexam@given@kw\xspace \assig@given%
\fi
\else \hwexam@given@kw\xspace \inclassig@given%
\fi
\ifx \inclassig@due\@empty
\ifx \assig@due\@empty% do nothing
\else
\ifx \inclassig@given\@empty
\ifx \assig@given\@empty% do nothing
\else ,~%
\fi
\else ,~%
\fi
\fi
\else
\ifx \inclassig@given\@empty
\ifx \assig@given\@empty% do nothing
\else ,~%
\fi
\else ,~%
\fi
\fi
\ifx \inclassig@due\@empty
\ifx \assig@due\@empty% do nothing
\else \hwexam@due@kw\xspace \assig@due%
\fi
\else \hwexam@due@kw\xspace \inclassig@due%
\fi
\ifx \inclassig@given\@empty
\ifx \assig@given\@empty
\ifx \inclassig@due\@empty
\ifx \assig@due\@empty% all empty do nothing
\else #2%
\fi
\else #2%
\fi
\else #2%
\fi
\else #2%
\fi
}
```

`\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.

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

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

```
\newcommand\assignment@number%
{\ifx\inclassig@number\@empty% if there is no outside number
\ifx\assig@number\@empty\else\assig@number\fi
\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.

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

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

```
\ifmultiple
\newenvironment{@assignment}%
{\ifx\assig@loadmodules\@true
\begin{omgroup}[loadmodules]{\protect\document@hwexamtype~\arabic{assignment}%
\assignment@title{}{\;(}{)\;}\given@due{}{}}
\else
\begin{omgroup}{\protect\document@hwexamtype~\arabic{assignment}%
\assignment@title{}{\;(}{)\;}\given@due{}{}}
\fi%
{\protect\document@hwexamtype~\arabic{assignment}%
\assignment@title{}{\;(}{)\;}\given@due{}{}}}
{\end{omgroup}}
```

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

```
\else
\newenvironment{@assignment}
{\begin{center}\bf
```

```
\Large\@title\strut\\
\document@hwexamtype~\arabic{assignment}\assignment@title{\;}{:\;}{\\}%
\large\given@due{--\;}{\;--}
\end{center}}
{}
\fi% multiple
```
⟨/package⟩

```
⟨*ltxml⟩
DefEnvironment('{assignment} OptionalKeyVals:assig',
  "<omdoc:omgroup ?&GetKeyVal(#1,'id')(xml:id='&GetKeyVal(#1,'id')')() "
  .  "assig:dummy='for the namespace'>"
  .  "<omdoc:metadata>"
  .    "<dc:title>"
  .       "Assignment ?&GetKeyVal(#1,'num')(&GetKeyVal(#1,'num').)()"
  .       "?&GetKeyVal(#1,'title')((&GetKeyVal(#1,'title')))"
  .    "</dc:title>"
  .    "?&GetKeyVal(#1,'given')(<omdoc:meta property='assig:given'>&GetKeyVal(#1,'given')</omdo
  .    "?&GetKeyVal(#1,'due')(<omdoc:meta property='assig:due'>&GetKeyVal(#1,'due')</omdoc:meta
  .    "?&GetKeyVal(#1,'pts')(<omdoc:meta property='assig:pts'>&GetKeyVal(#1,'pts')</omdoc:meta
  .  "</omdoc:metadata>"
  .  "#body"
  ."</omdoc:omgroup>\n"#,
#  afterDigest=> sub {
#    my ($stomach, $kv) = @_;
#    my $kvi = LookupValue('inclassig');
#    my @keys = qw(id num title pts given due);
#    my @vals = $kvi && map($kvi->getValue($_), @keys);
#    foreach my $i(0..$#vals) {
#       $kv->setValue($keys[$i],$vals[$i]) if $vals[$i];
#     }}
);#$
```
⟨/ltxml⟩

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.

```
⟨*package⟩
\addmetakey{inclassig}{number}
\addmetakey*{inclassig}{title}
\addmetakey{inclassig}{type}
\addmetakey{inclassig}{given}
\addmetakey{inclassig}{due}
\addmetakey{inclassig}{mhrepos}
\clear@inclassig@keys%initially
\newcommand\includeassignment[2][]{\metasetkeys{inclassig}{#1}%
\include{#2}\clear@inclassig@keys}
\newcommand\inputassignment[2][]{\metasetkeys{inclassig}{#1}%
```

```
\input{#2}\clear@inclassig@keys}
⟨/package⟩
⟨*ltxml⟩
DefMacro('\includeassignment [] {}', sub {
  my ($stomach, $arg1, $arg2) = @_;
  AssignValue('inclassig',$arg1) if $arg1;
  (Invocation(T_CS('\input'),$arg2)->unlist);
});
DefMacro('\inputassignment [] {}','\includeassignment[#1]{#2}');
⟨/ltxml⟩
```

5.4 Typesetting Exams

\quizheading

```
⟨*package⟩
\addmetakey{quizheading}{tas}
\newcommand\quizheading[1]{\def\@tas{#1}%
\large\noindent NAME: \hspace{8cm}  MAILBOX:\\[2ex]%
\ifx\@tas\@empty\else%
\noindent TA: \@for\@I:=\@tas\do{{\Large$\Box$}\@I\hspace*{1em}}\\[2ex]\fi}
```

\testheading

```
\addmetakey{testheading}{min}
\addmetakey{testheading}{duration}
\addmetakey{testheading}{reqpts}
\newenvironment{testheading}[1][]{\metasetkeys{testheading}{#1}
{\noindent\large{}Name: \hfill Matriculation Number:\hspace*{2cm}\strut\\[1ex]
\begin{center}\Large\textbf{\@title}\\[1ex]\large\@date\\[3ex]\end{center}
{\textbf{You have
\ifx\test@heading@duration\@empty\testheading@min minutes\else\testheading@duration\fi
(sharp) for the test}};\\ Write the solutions to the sheet.}\par\noindent

\newcount\check@time\check@time=\testheading@min
\advance\check@time by -\theassignment@totalmin
The estimated time for solving this exam is {\theassignment@totalmin} minutes,
leaving you {\the\check@time} minutes for revising your exam.

\newcount\bonus@pts\bonus@pts=\theassignment@totalpts
\advance\bonus@pts by -\testheading@reqpts
You can reach {\theassignment@totalpts} points if you solve all problems. You will only need
{\testheading@reqpts} points for a perfect score, i.e.\ {\the\bonus@pts} points are
bonus points. \vfill
\begin{center}
  {\Large\em
%  You have ample time, so take it slow and avoid rushing to mistakes!\\[2ex]
  Different problems test different skills and knowledge, so do not get stuck on
  one problem.}\vfill\par\correction@table \\[3ex]
\end{center}}
{\newpage}
```

```
⟨/package⟩
⟨*ltxml⟩
DefEnvironment('{testheading}OptionalKeyVals:omdoc','');
⟨/ltxml⟩
```

\testspace

```
⟨*package⟩
\newcommand\testspace[1]{\iftest\vspace*{#1}\fi}
⟨/package⟩
⟨*ltxml⟩
DefConstructor('\testspace{}','');
⟨/ltxml⟩
```

\testnewpage

```
⟨*package⟩
\newcommand\testnewpage{\iftest\newpage\fi}
⟨/package⟩
⟨*ltxml⟩
DefConstructor('\testnewpage','');
⟨/ltxml⟩
```

\testemptypage

```
⟨*package⟩
\newcommand\testemptypage[1][]{\iftest\begin{center}This page was intentionally left
    blank for extra space\end{center}\vfill\eject\else\fi}
⟨/package⟩
⟨*ltxml⟩
DefConstructor('\testemptypage','');
⟨/ltxml⟩
```

\@problem This macro acts on a problem's record in the `*.aux` file. Here we redefine it to generate the correction table.

```
⟨*package⟩
\renewcommand\@problem[3]{\stepcounter{assignment@probs}
\def\@@pts{#2}\ifx\@@pts\@empty\else\addtocounter{assignment@totalpts}{#2}\fi
\def\@@min{#3}\ifx\@@min\@empty\else\addtocounter{assignment@totalmin}{#3}\fi
\xdef\correction@probs{\correction@probs & #1}%
\xdef\correction@pts{\correction@pts & #2}
\xdef\correction@reached{\correction@reached &}}
⟨/package⟩
```

\correction@table This macro generates the correction table

```
⟨*package⟩
\newcounter{assignment@probs}
\newcounter{assignment@totalpts}
\newcounter{assignment@totalmin}
\newcommand\correction@probs{prob.}%
\newcommand\correction@pts{total}%
\newcommand\correction@reached{reached}%
```

```
\stepcounter{assignment@probs}
\newcommand\correction@table{\begin{tabular}{|l|*{\theassignment@probs}{c|}|l|}\hline%
&\multicolumn{\theassignment@probs}{c||}%|
{\footnotesize To be used for grading, do not write here} &\\\hline
\correction@probs & Sum & grade\\\hline
\correction@pts &\theassignment@totalpts & \\\hline
\correction@reached & & \\[.7cm]\hline
\end{tabular}}
⟨/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.

```
⟨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.

LaTeXML, 5, 6

References

[Koh15a] Michael Kohlhase. *metakeys.sty: A generic framework for extensible Metadata in LaTeX*. Tech. rep. Comprehensive TeX 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 LaTeX*. Tech. rep. Comprehensive TeX 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 TeX 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 LaTeX*. Tech. rep. Comprehensive TeX 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).