

The `thorshammer` Package

D. P. Story*
Email: `dpstory@acrotex.net`

processed June 26, 2021

Contents

1	Introduction	2
2	Preliminaries	3
2.1	Options for this package	3
2.2	Required packages	4
3	Setting the initial view	4
4	Declaring instructor and class information	5
5	Running headers	5
6	Declaring a cover page	6
7	Basic methods	7
7.1	Configuring the basic methods experience	7
7.2	Post creation document assembly	8
8	Form field commands	9
8.1	Controls above the <code>quiz</code> environment	9
8.2	Commands that usually follow the <code>quiz</code>	13
8.3	Controls inside the <code>quiz</code> environment	15
9	Field level JavaScript for form field commands	16

*The author acknowledges Thorsten G. (a.k.a., Thor) who proposed this workflow and who contributed many ideas; Jürgen G. (a.k.a., Loki) also contributed many good ideas, enthusiasm, questions, bug detecting, and motivation. High regards and respect to both. D. P. Story (a.k.a., Odon).

10 Modifications and redefinitions of AeB	21
10.1 Quiz components modified	21
10.2 Modify margin points markup	26
10.3 Modifications to the summary table	26
10.4 Boom! Thor’s thunders: “Thor needs solutions!”	27
10.5 Modifications to the <code>web</code> package	28
11 The <code>useclass</code> option and above	28
11.1 Declaring class members	29
11.2 Some process controls	30
11.3 Working with multiple quizzes in one source	30
11.4 Building quizzes with <code>makeClassFiles</code> & <code>\sadQuizzes</code>	32
12 Batch support files	38
12.1 The container file for Thor’s way	38
12.2 The batch termination file	40
13 Configuration files	41
13.1 Class configuration	41
13.2 Load the configuration file	42
14 Index	43
15 Change History	48
1 <code>*package</code>	

1 Introduction

Thorsten G. has asked me to assist him in creating a quiz system, based on AcroTEX, to be delivered to his classes. His workflow for this assessment system is a follows:¹

1. The `quiz` environment is used to pose the questions, which consist of MC, numerical fill-in the blank, and extended response questions.

Though the `quiz` environment is used, the student does not get his/her score reported back immediately upon finishing the quiz.

2. When the student finishes the exam, taken in AR, he/she presses the End Quiz control and saves the document as `\jobname-ID.pdf`, where ID is a student identification. I am informed the ID is the student name.

3. At some point, the instructor’s script moves the document to the instructors folder.

¹As my occasional friend Jürgen says, this workflow is a real *hammer*, so I titled this package ‘Thors(ten) hammer’, or simply `thorshammer`.

- The instructor opens the PDF and finishes marking the extended response questions and assigns a grade.

This package supports the Thorsten's workflow by providing the necessary form elements and JavaScript to carry out his(her) plan. What happens to the quiz after that, I do not know.

2 Preliminaries

```
2 \RequirePackage{xkeyval}
3 \edef\th@quoteCat{\the\catcode'\"}
4 \catcode'\ "=12\relax
```

2.1 Options for this package

- | | |
|-------------------|---|
| nocfg | If this option is taken, <code>thorshammer.cfg</code> is not input. |
| | <pre>5 \DeclareOptionX{nocfg}{\let\th@loadCFG\dl@NO} 6 \let\th@loadCFG\dl@YES</pre> |
| testmode | If this option is taken, quizzes can be used in the normal way. |
| | <pre>7 \newif\ifthtestmode\thtestmodefalse 8 \DeclareOptionX{testmode}{\thtestmodetrue} 9 \DeclareOptionX{!testmode}{\thtestmodefalse}</pre> |
| ordinary | This is an experimental option to see if we can produce a regular quiz with selected features of a <code>thorshammer</code> workflow. |
| | <pre>10 \newif\ifthordinary \thordinaryfalse 11 \DeclareOptionX{ordinary}{\thtestmodetrue\thordinarytrue}</pre> |
| useclass | Use this option to bring in addition code to declare each member of the class, to automatically build a quiz for each class member, to distribute these quizzes to a designated folder of the instructor, and to distribute the quizzes the respective class folder. |
| | <pre>12 \newif\ifbasicmethods\basicmethodstrue 13 \newif\ifuseclassOpt\useclassOptfalse 14 \def\bUseClass{false} 15 \DeclareOptionX{useclass}{\useclassOpttrue} 16 \def\bUseClass{true}\basicmethodsfalse 17 }</pre> |
| usebatch | This option should be used with the <code>useclass</code> option. When this option is taken, the no freeze quiz button is created. The batch sequence <code>Thor's way</code> does that, and the presence of the freeze button, the instructor may press it without thinking. We don't want that to happen. |
| | <pre>18 \newif\ifth@allowfreeze \th@allowfreezetrue 19 \DeclareOptionX{usebatch}{\th@allowfreezefalse} 20 \ExecuteOptionsX{useclass}</pre> |
| batchdistr | This option declares the instructor's intention of using Acrobat to apply security using <code>'Thor distributes.sequ'</code> or <code>'Thor protects and distributes.sequ'</code> to apply security and to distribute the files to the students's folders. This option simply expands |

`\distrToStudentsOff`, and redefines the two commands so the author can't use them. This option has no effect on writing quizzes to the instructor's folder.

```
21 \DeclareOptionX{batchdistr}{\ExecuteOptionsX{usebatch}}%
22 \AtEndOfPackage{\distrToStudentsOff
23 \let\distrToStudentsOff\relax\let\distrToStudentsOff\relax}}
```

Process the options

```
24 \ProcessOptionsX
25 \edef\thOrdQz{\ifthordinary true\else false\fi}
```

2.2 Required packages

```
26 \RequirePackage{insdljs}[2021/06/19]
```

We use the `usealtadobe` option of `insdljs`, but not directly. If `\inputAltAdbFncs` is `\relax` than the functions have not already been input above `thorshammer`.

```
27 \ifx\usedAdbFncs\dl@NO
28 \def\inputAltAdbFncs{\InputIfFileExists{altadbfncs.def}}%
29 {\PackageInfo{insdljs}{Inputting code for usealtadobe option}}%
30 {\PackageWarning{insdljs}{Cannot find altadbfncs.def.\MessageBreak
31 Reinstall or refresh your file name database.}}}%
32 \let\usedAdbFncs\dl@YES
33 \else
34 \let\inputAltAdbFncs\relax
35 \fi
36 \inputAltAdbFncs
37 \RequirePackage{exerquiz}[2021/05/29]
38 \RequirePackage{eq-save}[2021/04/27]
39 \let\execjs\dl@YES
40 \@ifundefined{CommentStream}{\newwrite\CommentStream}{}
41 \def\csarg#1#2{\expandafter#1\csname#2\endcsname}
42 \providecommand{\eqSP}{\string\040}
43 \def\thPageOne{\setcounter{page}{1}}
```

3 Setting the initial view

We require the document to be opened on the first page, but the initial magnification is under the control of the document author.

```
\setInitMag{fitpage|actualsize|fitwidth|fitheight|fitvisible|inheritzoom}
```

This command determines the initial magnification. There are a choice of six values for the argument; the default is `fitpage`

```
44 \def\setInitMag#1{\setkeys{thim}{mag=#1}}
45 \define@choicekey+{thim}{mag}[\val\nr]%
46 {fitpage,actualsize,fitwidth,fitheight,%
47 fitvisible,inheritzoom}[fitpage]%
48 {\edef\th@initmag{\@nameuse{dl@val}}}%
49 {\PackageWarning{thorshammer}{%
50 Bad choice for initial magnification,\MessageBreak
51 permissible values are fitpage, actualsize,\MessageBreak
```

```

52   fitwidth, fitheight, fitvisible, and\MessageBreak
53   inheritzoom. Try again}}
54 \def\th@initmag{\dl@fitpage}

```

The `\addToDocOpen` is a command from `insdljs`. We turn off calculations as the student does not need to see the calculation icon each time s/he enters a response. When the instructor presses the Mark It button, calculations are turned on again. In the second line below, we set the initial view to page 1 and the magnification set by the `\setInitMag` command above.

```

55 \addToDocOpen{JS{%
56   var stmot=app.setTimeout("this.calculate=false;",100);}}
57 \addToDocOpen{\GoToD[\Page{1}]\th@initmag}

```

4 Declaring instructor and class information

`\instrPath*`{*path*} The path to the instructor's folder. It is assumed that this *path* is an absolute path. If the star option is taken, then the path is relative to the current folder. The `\instrPathIsCHTTP` declaration is available if the path to the instructor is a WebDAV address. This info is passed on the a JavaScript method.

```

58 \def\instrPathIsCHTTP{\def\thInstrFS{CHTTP}}
59 \let\thInstrFS@empty
60 \newcommand\instrPath{\ifstar
61   {\gdef\InstrPathFull{false}\instrPath@i}
62   {\gdef\InstrPathFull{true}\instrPath@i}}
63 \def\instrPath@i#1{\gdef\InstrPath{"#1"}}
64 \def\InstrPathFull{true}
65 %\let\InstrPath@empty
66 \def\InstrPath{this.path.replace(reRmFn,"")}

```

`\classPath*`{*path*} The path to the class folder. It is assumed that this *path* is an absolute path. If the star option is taken, then the path is relative to the current folder. The `\instrPathIsCHTTP` declaration is available if the path to the class folders is a WebDAV address. This info is passed on the a JavaScript method.

```

67 \def\classPathIsCHTTP{\def\thClassFS{CHTTP}}
68 \let\thClassFS@empty
69 \newcommand\classPath{\ifstar
70   {\gdef\ClassPathFull{false}\classPath@i}
71   {\gdef\ClassPathFull{true}\classPath@i}}
72 \def\classPath@i#1{\gdef\ClassPath{"#1"}}
73 \def\ClassPath{this.path.replace(reRmFn,"")}
74 \def\ClassPathFull{true}

```

5 Running headers

The scheme used here assumes no other \LaTeX package has been used to take over the running headers (and footers). If that is the case, use the values of the `\thQzHeaderL` commands below to design your own $\langle text \rangle$. This is inserted into the left running

header for the quiz pages.

```
75 \def\thQzHeaderL#1{\def\th@QzHeaderLQ{\makebox[0pt][l]{#1}}}  
76 \def\th@QzHeaderL{\th@QzHeaderLQ}  
77 \thQzHeaderL{Thor's Class}  
78 \def\th@QzHeaderLS{\th@HeaderOffset\th@QzHeaderLQ}  
\thQzHeaderCQ{text} This is inserted into the center running header for the quiz pages.  
79 \def\thQzHeaderCQ#1{\def\th@QzHeaderCQ{\makebox[0pt][c]{#1}}}  
80 \thQzHeaderCQ{Quiz \thQuizName}  
81 \def\th@QzHeaderC{\th@QzHeaderCQ}
```

```
\thQzHeaderCS{text} This is inserted into the center running header for the solution pages.  
82 \def\thQzHeaderCS#1{\def\th@QzHeaderCS{\makebox[0pt][c]{#1}}}  
83 \thQzHeaderCS{Solutions: \thQuizName}
```

```
\thQzHeaderR{text} This is inserted into the right running header for the solution pages.  
84 \def\thQzHeaderR#1{\def\th@QzHeaderR{\makebox[0pt][r]{#1}}}  
85 \thQzHeaderR{\thepage}
```

We apply the running headings, depending on whether web loaded by testing for the `webheadings` page style.

```
86 \ifundefined{ps@webheadings}{%  
87 \def\th@setHeaders{%  
88 \renewcommand{\@oddhead}{\th@QzHeaderL\hfil\th@QzHeaderC\hfil  
89 \th@QzHeaderR}\renewcommand{\@evenhead}{\@oddhead}}%  
90 }{%  
91 \def\th@setHeaders{%  
92 \lheader{\th@QzHeaderL}%  
93 \cheader{\th@QzHeaderC}%  
94 \rheader{\th@QzHeaderR}}%  
95 }
```

Change the header for the solution section

```
96 \def\eq@normallheader{%  
97 \ifundefined{ps@webheadings}{%  
98 \def\th@QzHeaderL{\th@QzHeaderLS}%  
99 \def\th@QzHeaderC{\th@QzHeaderCS}%  
100 }{%  
101 \lheader{\th@QzHeaderLS}%  
102 \cheader{\th@QzHeaderCS}%  
103 }  
104 }
```

```
\rhPgNumsOnly Originally, this package only exhibited page numbers in the running header, expanding \rhPgNumsOnly in the preamble restores that original experience.
```

```
105 \def\rhPgNumsOnly{\thQzHeaderL}\thQzHeaderCQ}\thQzHeaderCS}  
106 \AtBeginDocument{\th@setHeaders}
```

6 Declaring a cover page

`\DeclareCoverPage{pgNum}` A cover page, if declared, is appended to the beginning of the quiz. The

page specified by $\langle pgNum \rangle$ is the cover page. The cover page is a single page and must occur prior to any quiz. Valid in the *preamble only*.

```

107 \newif\ifthCoverPage \thCoverPagefalse
108 \newcommand{\DeclareCoverPage}[1]{\thCoverPagetrue
109 \def\thIsCP{true}\def\thCvrPg{#1}}
110 \def\thIsCP{false}\def\thCvrPg{0}
111 \@onlypreamble\DeclareCoverPage

```

7 Basic methods

Thor is tormenting me with the basic methods option. The basic methods option is no options other than perhaps `nocfgs`. As a consequence, the student names are not pre-filled into the name fields. When multiple quizzes are produced, they are named differently, `\jobname-1.pdf`, `\jobname-2.pdf`, and so on. A lot of work has gone in to the basic methods so it works link the non-basic methods (option `useclass` or higher). The commands `\instrPath` and `\classPath` are supported; to use the `\classMember` command, use `useclass` or higher. In addition to `\instrPath` and `\classPath`, we define special basic method commands, as describe in the next section.

7.1 Configuring the basic methods experience

`\useNameToCustomize` (Basic methods) `\useNameToCustomize` can be used to modify the file name of the quiz to include student name; the default is to use the original quiz file name. This command is implemented through the Freeze Quiz button. This command has no effect in the non-basic setting.

```

112 \def\useNameToCustomize{\def\thUseNameToCustomize{true}}
113 \def\thUseNameToCustomize{false}

```

`\enumQuizzes` $\langle num \rangle$ (Basic methods) This file specifies that the quizzes should be replicated $\langle num \rangle$ times and named `\jobname-1`, `\jobname-2`, ..., `\jobname- $\langle num \rangle$` . The default is not to enumerate.

```

114 \def\enumQuizzes#1{\def\bUseClass{true}\basicmethodsfalse
115 \ClassEntriestrue\def\ClassPathFull{true}\def\InstrPathFull{true}}%
116 \def\ClassPath{this.path.replace(reRmFn,"")}%
117 \def\InstrPath{this.path.replace(reRmFn,"")}%
118 \bgroup\@tempcnta#1\relax
119 \@whilenum\@tempcnta>\z@\do{\classMember{}-{}-{}%
120 \advance\@tempcnta\m@ne}\egroup
121 \def\thEnumQuizzes{#1}\def\bEnumQuizzes{true}}
122 \def\thEnumQuizzes{0}\def\bEnumQuizzes{false}

```

`\distrQuizzes` $\{\langle folder_1 \rangle\}\{\langle folder_2 \rangle\}\dots\{\langle folder_n \rangle\}$ (Basic methods) If `\distrQuizzes` is used, `\enumQuizzes` command is ignored. The quizzes are enumerated, as described above, but the number of quizzes created is the number of folders declared. The script `\sadQuizzes` also distributes the individual quizzes to the appropriate folder, on the path determined by the `\classPath` command.

```

123 \newcommand{\distrQuizzes}{%
124   \ifuseclassOpt
125     \def\th@next{\PackageWarning{thorshammer}
126       {Use have specified the useclass option or higher\MessageBreak
127         yet you employ \string\distrQuizzes, these are\MessageBreak
128         incompatible. Assuming the specified package option}}%
129   \else
130     \let\th@next\th@distrQuizzes
131   \fi\th@next
132 }
133 \def\th@distrQuizzes{\def\bUseClass{true}\basicmethodsfalse
134   \ClassEntriestrue\bgroup\@makeother\_ \th@distrQuizzes@i}
135 \def\rmSTAR#1*\@nil{\def\@folder{#1}}
136 \def\tstForSTAR#1{\tstForSTAR@i#1*\@nil}
137 \def\tstForSTAR#1*#2*\@nil{\def\@rgi{#1}\ifx\@rgi\@empty
138   \def\ISSTAR*{\rmSTAR#2\@nil}\else\let\ISSTAR\@empty\fi}%
139 \def\th@distrQuizzes@i#1{\@tempcnta\z@
140   \@tfor\@folder:=#1\do{\advance\@tempcnta\@ne
141     Determine if \@folder begin with *, remove it and return the path as \@folder
142     \expandafter\tstForSTAR\@folder**\@nil
143     \edef\x{\noexpand\classMember{}{}\ISSTAR{\@folder}}\x
144   }\xdef\enumQuizzes{the\@tempcnta}%
145 }
146 \def\bDistrQuizzes{false}

```

Just auto-save the document - not recommended The `\executeSave()` command previously figured in importantly, as this package developed, use of `\executeSave()` cannot be recommended. This command was implemented early in the development process

```

147 \@ifundefined{executeSave}
148   {\def\executeSave(){%
149     console.println("automatically saving this file...");^^J%
150     var retn=aebTrustedFunctions(this,aebDocSaveAs,%
151       {cPath:this.path,bCopy:false})}}{}

```

`docassembly` The `docassembly` environment was created early in development and was meant to be used with `\executSave()`. The environment definition was updated to be equivalent to the `makeClassFiles` environment. An environment by the same name, and the same functionality, is defined in `aeb_pro`.

```

152 \@ifundefined{docassembly}
153   {\newenvironment{docassembly}{%
154     \execJS[\mkClFlsSpcls]{docassembly}}{\endexecJS}}
155   {\renewenvironment{docassembly}{%
156     \execJS[\mkClFlsSpcls]{docassembly}}{\endexecJS}}

```

7.2 Post creation document assembly

`\rasSolns` The `\sadQuizzes` command is placed within the `docassembly` or `makeClassFiles`

environment, `\sadQuizzes` in the body of the environment.

```
\begin{docassembly}
\sadQuizzes
\end{docassembly}
\begin{document}
```

Originally, we defined a command `\rasSolns`, this command has been `\let` to `\sadQuizzes`, which now performs its duties.

8 Form field commands

We define two types of controls: (1) those placed outside the quiz; (2) those placed within the quiz.

8.1 Controls above the quiz environment

Commands that occur above the quiz environment

The student needs to sign in with his/her first and last name. The commands `\FirstName` and `\LastName` are defined for that purpose.

```
\FirstName
\LastName 157 \ifbasicmethods\let\th@namePresets\@empty\else
```

If the option `useclass`, or higher, is taken, we make these fields read only and the JavaScript code of `\sadQuizzes` will fill name field in for the student.

```
158 \def\th@namePresets{\Ff\FfReadOnly\BC{}}\fi
159 \newcommand\FirstName[3][\th@bMrkQz\textField[%
160 \presets{\th@namePresets}#1]{Name.first}{#2}{#3}}
161 \newcommand\LastName[3][\textField[%
162 \presets{\th@namePresets}#1]{Name.last}{#2}{#3}}
```

The `\sadQuizzes` command uses the name fields to identify on which page a quiz begins. This worries me a little if a document designer places more than one name field for a quiz. We attempt to make the first use of the name field per quiz. We define `\th@bMrkQz`. These fields are place exactly once for each quiz and is attached to the name fields.

```
163 \def\th@bMrkQz{\@ifundefined{bMrkQz\currQuiz}
164 {\rlap{\textField[\Ff\FfReadOnly\BC{}\BG{}}]{bMrkQz}{0pt}{0pt}}%
165 \@namedef{bMrkQz\currQuiz}{}}{}}
```

`\FullName` [*options*] {*wd*} {*ht*} The first and last name fields are required; however, when `useclass` or higher is used, they are automatically filled in. The `\FullName` field uses the calculate event to extract the first and last names and displays them together in one field. The format for the this name field can be changed through

```
\thfullnameFmt the declaration \thfullnameFmt.
166 \def\th@fullnamePresets{\Ff\FfReadOnly\BG{}}\BC{}}
167 \def\thfullnameFmt#1{\def\th@fullnameFmt##1##2{#1}}
168 \thfullnameFmt{#1+" "+#2}
169 \newcommand{\FullName}[3][\textField[%
```

```

170 \presets{\th@fullnamePresets}#1\AAcalculate{%
171   var fName=this.getField("Name.first").value;\r
172   var lName=this.getField("Name.last").value;\r
173   event.value=\th@fullnameFmt{fName}{lName};}{FullName}{#2}{#3}}

```

`\pwdInstrFld`[(*options*)]{(*pwd*)}{(*wd*)}{(*ht*)} In this workflow, when the instructor opens a quiz file, he/she enters a password. On success, the non-extended response questions of the student's quiz is marked, and various hidden form elements are made visible.

`\pwdInstrFldTU`[(*pdfstr*)] can be redefined to provide a tool tip for this field.

```

174 \def\rwdInstrFldTU#1{\def\rwdInstrFld@TU{#1}}
175 \pwdInstrFldTU{Enter password to mark this quiz}
    The definition of \pwdInstrFld
176 \newcommand{\pwdInstrFld}[4][\% opts, pwd, wd, ht
177   \@ifundefined{\currQuiz-nqs}{\def\nqs{0}}
178   {\edef\nqs{\@nameuse{\currQuiz-nqs}}}%
179   \textField[\cmd{\bParams{\currQuiz}{\nqs}{"#2"}\eParams}
180   \Ff\FfPassword\AAkeystroke{\pwdKeyJS}
181   \protect\AA\protect\Ff\TU{\pwdInstrFld@TU}#1%
182 ]{pwtxt}{#3}{#4}}

```

`\markQz`[(*options*)]{(*wd*)}{(*ht*)} Loki suggested another idea to have the password field hidden until the instructor opens the file. Well if you are doing that, why have a password field? Instead, a push button is provided.

`\markQzFldCA`[(*jsstr*)] The caption for this button

```

183 \def\rmarkQzFldCA#1{\def\rmarkQzFld@CA{#1}}
184 \markQzFldCA{Mark It}

```

`\markQzFldTU`[(*jsstr*)] The tool tip for this button

```

185 \def\rmarkQzFldTU#1{\def\rmarkQzFld@TU{#1}}
186 \markQzFldTU{Press to mark this quiz}

```

Important! The code for `\markQz`. **Important!** For obvious reasons, we don't want the push button to be seen by the students. As a result, it is initially hidden. The key to having the push button visible when the instructor is the private JavaScript variable `_thorshammer` (this can be changed). The following code is placed in the `config.js`

```
var _thorshammer=true;
```

Acrobat reads this file only once when it's opened. When the instructor opens the student's quiz PDF in Acrobat, some underlying JavaScript code tests whether the `_thorshammer` variable is defined and is `true`. If these conditions are met, JavaScript makes the `\markQuizFld` and `\freezeQz` fields visible.

```

187 \newcommand{\markQz}[3][\%
188   \@ifundefined{\currQuiz-nqs}{\def\nqs{0}}%
189   {\edef\nqs{\@nameuse{\currQuiz-nqs}}}%

```

This text field is placed underneath the push button. It is the one that causes the `\markQz` field to be visible.

```

190 \makebox[0pt][l]{\textField[\BC]{\BG}{\H{S}\AAformat{%
191   var f=this.getField("MarkIt");\r
192   var g=this.getField("freezeQz");\r
193   if(typeof _thorshammer!="undefined" && _thorshammer){\r\t
194     if(f!=null)f.display=display.visible;\r\t
195 } else{\r\t
196   if(f!=null)f.display=display.hidden;\r\t
197   if(g!=null)g.display=display.hidden;\r
198 }]}{hideTxtFldMI}{Opt}{Opt}}%

```

The push button seen by the instructor to mark the quiz.

```

199 \pushButton[\cmd{\bParams{\currQuiz}{\nQs}\eParams}\F\FHIDDEN
200   \AAMouseup{\commonPassKey}\CA{\markQzFld@CA}
201   \TU{\markQzFld@TU}\protect\AA\protect\F#1%
202 ]{MarkIt}{#2}{#3}

```

`\freezeQuiz` [*options*] [*wd*] [*ht*] The `\freezeQuiz` makes all form fields readonly. After the instructor finishes marking the quiz, he/she presses the freeze quiz button before he moves it to the student's folder for review. This is done so the student cannot modify the quiz in any case and beg for more points. (They never beg for fewer points). The freeze quiz button makes itself hidden as well.

`\freezeQuizFldTU` [*pdfstr*] can be redefined to provide a tool tip for this field.

```

203 \def\freezeQuizFldTU#1{\def\freezeQuizFld@TU{#1}}
204 \freezeQuizFldTU{Make all fields readonly, cannot be undone}

```

`\freezeQuizFldCA` [*pdfstr*] can be redefined to provide a button caption for this field.

```

205 \def\freezeQuizFldCA#1{\def\freezeQuizFld@CA{#1}}
206 \freezeQuizFldCA{Freeze Quiz}

```

The definition of `\freezeQuiz`. If the `usebatch` option is taken, we do not create the push button.

```

207 \newcommand\freezeQuiz[3][\pushButton[\cmd{\let\%defjsLB}
208   \CA{\freezeQuizFld@CA}\F\FHIDDEN
209   \TU{\freezeQuizFld@TU}\AAMouseup{freezeQuizMU()}
210   \protect\AA\protect\F
211   #1]{freezeQz}{#11bp}}

```

`\instrSave` [*options*] [*wd*] [*ht*] A companion macro to `\freezeQuiz`. This macro is substituted for `\freezeQuiz` when the `usebatch` option is taken. The `\instrSave` and `\freezeQuiz` should not appear in the same document; we give them the same field name so the JavaScript treats them the same, in terms of making them hidden and visible. `\ifth@allowfreeze`

`\instrSaveFldTU` [*pdfstr*] can be redefined to provide a tool tip for this field.

```

212 \def\instrSaveFldTU#1{\def\instrSaveFld@TU{#1}}
213 \instrSaveFldTU{Save and close this file to the current folder}

```

`\instrSaveFldCA` [*pdfstr*] can be redefined to provide a button caption for this field.

```

214 \def\instrSaveFldCA#1{\def\instrSaveFld@CA{#1}}
215 \instrSaveFldCA{Save \string& Close}

```

The definition of `\instrSave`. If the `usebatch` option is taken, we do not create the push button.

```

216 \newcommand\instrSave[3][\pushButton[%
217 \CA{\instrSaveFld@CA}\F\FHidden
218 \TU{\instrSaveFld@TU}\AAmouseup{%
219   var f=this.getField("studentenGrade");\r
220   var str=""+f.value;\r
221   str=str.replace(/\string\s/g,"");\r
222   if (str=="")\r\t
223     app.alert("You did not award the student a final mark."
224     +"\n\nAward the mark and then save.");\r
225   else {\r\t
226     aebTrustedFunctions(this,aebSaveAs);\r\t
227     this.closeDoc(true);\r
228   }}\protect\AA\protect\F
229 #1]{freezeQz}{11bp}}

```

`\freezeOrSave[(options)]{(wd)}{(ht)}` is the recommended way of inserting `\freezeQuiz` or `\instrSave`. If `usebatch` is taken, `freezeOrSave` expands to `\instrSave`; otherwise it expands to `\freezeQuiz`.

```

230 \AtEndOfPackage{\ifth@allowfreeze\let\freezeOrSave\freezeQuiz
231 \else\let\freezeOrSave\instrSave\fi}

```

`\studentReport[(options)]{(wd)}{(ht)}` In Thor's way of things, a summary report is placed at the top of the document. This readonly field shows the number of points awarded and the total points.

```

232 \newcommand\studentReport[3][\%
233 \textField[\BC{\BG{\Q1}\F\FHidden\Ff\FfReadOnly\protect\Ff#1%
234 ]{studentenReport}{#2}{#3}}

```

`\studentGrade[(options)]{(wd)}{(ht)}` Again, in Thor's way of things, a text field is available to assign grade. This field is initially hidden, but becomes visible when instructor signs in.

```

235 \newcommand\studentGrade[3][\textField[\F\FHidden\protect\F
236 \BC{red}\BG{\Q1}\textSize{12}\textColor{blue}
237 \AAkeystroke{event.change=event.change.toUpperCase()}#1%
238 ]{studentenGrade}{#2}{#3}}

```

`\thQHFirstName[(options)]{(wd)}{(ht)}` The first name of the student

```

239 \def\thQHFirstName#1{\def\th@QHFirstName{\textbf{#1}\space}}
240 \thQHFirstName{First name:}

```

`\thQHLastName[(options)]{(wd)}{(ht)}` The last name of the student

```

241 \def\thQHLastName#1{\def\th@QHLastName{\textbf{#1}\space}}
242 \thQHLastName{Last name:}

```

`\thQHPoints[(options)]{(wd)}{(ht)}` Number of points, displayed in the form '10 / 20'.

```

243 \def\thQHPoints#1{\def\th@QHPoints{\textbf{#1}\space}}
244 \thQHPoints{Points:}

```

`\thQHGrade[(options)]{(wd)}{(ht)}` Some grade mark (A, B, C, etc., or 1, 2, 3, etc.)

```
245 \def\thQHGrade#1{\def\th@QHGrade{\textbf{#1}\space}}
```

```
246 \thQHGrade{Grade:}
```

`\thQuizHeader*` The above commands typically appear above the quiz and placed in some beautiful way. We bundle these commands into a single one, according to my own happiness, but you may seek happiness some other way by redefining `\thQuizHeaderLayout`. (The command `\thQuizHeader` just picks up on the `*`-option, then expands `\thQuizHeaderLayout`.) The command is placed beneath the `\DeclareQuiz` command and above the `quiz` environment. The command automatically emits a `\newpage`, unless the `*`-option is taken.

Preferred Placement	Alternate Placement
<code>\DeclareQuiz{<i>(qz-name)</i>}</code>	<code>\begin{document}</code>
<code>...</code>	<code>\DeclareQuiz{<i>(qz-name)</i>}</code>
<code>\begin{document}</code>	<code>\thQuizHeader</code>
<code>...</code>	<code>...</code>
<code>\thQuizHeader</code>	<code><quiz-begins></code>
<code>...</code>	
<code><quiz-begins></code>	

```
247 \newcommand{\thQuizHeader}{\let\Hy@EveryPageAnchor\relax
```

```
248 \ifstar{\thPageOne\thQuizHeaderLayout}
```

```
249 {\newpage\thPageOne\thQuizHeaderLayout}%
```

```
250 }
```

`\thQuizHeaderLayout` The body of this command contains the arraignment of the above defined commands. It is this command that may be redefined.

```
251 \newcommand\thQuizHeaderLayout{\noindent
252 \th@QHFirstName\FirstName{1.5in}{13bp}\vcgBdry[3pt]
253 \th@QHLastName\LastName{1.5in}{13bp}\vcgBdry[6pt]
254 \begin{minipage}[t]{1.2in}\kern0pt
255 \makebox[Opt][r]{\raggedleft\markQz}{11bp}%
256 \hspace{\marginparsep}}%
257 \th@QHPoints\studentReport{\widthof{000/000}}{11bp}\vcgBdry[6pt]
258 \makebox[Opt][r]{\raggedleft\freezeOrSave}{11bp}%
259 \hspace{\marginparsep}}%
260 \th@QHGrade\studentGrade{14bp}{14bp}\vcgBdry[6pt]
261 \end{minipage}\hfill
262 \begin{minipage}[t]{\linewidth-1em-1.2in}\kern0pt
263 \begin{sumryTblAux}{\currQuiz}
264 \displaySumryTbl[ntables=1,showmarkup]{\currQuiz}
265 \end{sumryTblAux}
266 \end{minipage}}
```

This assumes the English language and the `usesumrytbls` option of `exerquiz`.

8.2 Commands that usually follow the quiz

`\completeMsgFldv{(pdfstr)}` is the message that is displayed in this multi-line text field.

`\completeMsgFld`{*options*}{*wd*}{*ht*} When the student completes the quiz, a hidden text field appears and reminds the student to save the document.

```
267 \def\completeMsgFldV#1{\def\completeMsgFld@V{#1}}
268 \completeMsgFldV{Congratulations, you have completed the quiz,
269 before doing anything else, you need to save this document.}
```

The definition of `\completeMsgFld`

```
270 \newcommand{\completeMsgFld}[3][\]{\textfield[\F\FHidden\F\FMultiline
271 \F\FFReadOnly\V{\completeMsgFld@V}
272 \DV{\completeMsgFld@V}]{postQzMsg}{#2}{#3}}
```

`\ShrtPtsFld`{*options*}{*quiz-name*} This is `\PointsField`, but with special format script.

`\ShrtPtsFldFmt`{*js-str*} is the formatting string used; *js-str* should incorporate the event property `event.value` in its definition. See default definition below.

```
273 \def\ShrtPtsFldFmt{\bgroup\obeyspaces\ShrtPtsFldFmt@i}
274 \def\ShrtPtsFldFmt@i#1{\egroup\flJSstr[noquotes]{\ShrtPtsFld@Fmt}{#1}}
275 \ShrtPtsFldFmt{"Short Pts: "+event.value}
```

The definition of `\ShrtPtsFld`

```
276 \newcommand{\ShrtPtsFld}[2][\]{%
277 \PointsField[\AAformat{if(event.value!="")}
278 event.value=\ShrtPtsFld@Fmt}\F\FHidden\protect\AA
279 \protect\F#1}{#2}}
```

`\LngPtsFld`{*options*}{*quiz-name*} This field will hold the total points for the essay or extended response questions. It is modeled after `\PointsField`, having the same defaults and width and height.

`\LngPtsFldFmt`{*js-str*} is the formatting string used; *js-str* should incorporate the event property `event.value` in its definition. See default definition below.

```
280 \def\LngPtsFldFmt{\bgroup\obeyspaces\LngPtsFldFmt@i}
281 \def\LngPtsFldFmt@i#1{\egroup\flJSstr[noquotes]{\LngPtsFld@Fmt}{#1}}
282 \LngPtsFldFmt{"Long Pts: "+event.value}
```

The definition of `\LngPtsFld`

```
283 \newcommand{\LngPtsFld}[2][\]{%
284 \textfield[\presets{\PointsFieldDefaults}\F\FHidden
285 \AAformat{if(event.value!="")} event.value=\LngPtsFld@Fmt}
286 \AAcalculate{var f=this.getField("essayMrkUp");\r
287 if(f!=null)EFSimple_Calculate("SUM",%
288 new Array("essayMrkUp.\currQuiz"));}
289 ]{EssayField.\currQuiz}{\PtFW}{\DefaultHeightOfWidget}}
```

`\TotalsFld`{*options*}{*quiz-name*} This is modeled after `\PointsField`, but with special format script.

`\TotalsFldFmt`{*js-str*} is the formatting string used; *js-str* should incorporate the event property `event.value` in its definition. See default definition below.

```
290 \def\TotalsFldFmt{\bgroup\obeyspaces\TotalsFldFmt@i}
291 \def\TotalsFldFmt@i#1{\egroup\flJSstr[noquotes]{\TotalsFld@Fmt}{#1}}
```

```

292 \TotalsFldFmt{"Total: "+event.value+"\space\eqOutOf\space"%
293 +NPointTotal}
    The definition of \TotalsFld
294 \newcommand{\TotalsFld}[2] [] {%
295   \textField[\presets{\PointsFieldDefaults}\F\FHidden
296   \AAformat{try{event.value=(\TotalsFld@Fmt)}catch(e){}}
297   \AAcalculate{EFSimple_Calculate("SUM",%
298 new Array("PointsField.\currQuiz","EssayField.\currQuiz"));}\r
299   var\eqSP f=this.getField("studentenReport");\r
300   f.value=(1*event.value)+"\eqSP/\eqSP"+\theeqpointvalue;
301   }]{TotalsField.\currQuiz}{\PtFW}{\DefaultHeightOfWidget}}

```

`\thQuizTrailer` The above commands can be arranged in some way following the quiz; one such arrangement is found in the command `\thQuizTrailer`.

```

302 \newcommand{\thQuizTrailer}{\raisebox{\baselineskip-\fboxsep}%
303   {\makebox[Opt][l]{\parbox[t]{3in}{\kernOpt
304   \completeMsgFld{3in}{3\baselineskip}}}}%
305   \makebox[Opt][l]{\hspace{3in}\quad
306   \ifthtestmode\CorrButton{\currQuiz}\else
307   \stuSaveBtn{}{11bp}\fi}\parbox[t]{3in}
308   {\ShrtPtsFld{\currQuiz}\vcgBdry[6pt]
309   \LngPtsFld{\currQuiz}\vcgBdry[6pt]
310   \TotalsFld{\currQuiz}}}

```

8.3 Controls inside the quiz environment

`\essayQ{nPts}` When we have an essay type question we need to mark it prior to the `\item`. In order to test whether the instructor has put in more credit than specified by the `\PTs` command, we need to pass the number of points for this question.

`\essayQFldTU{pdfstr}` is the tool tip for this field.

```

311 \def\essayQFldTU#1{\def\essayQFld@TU{#1}}
312 \essayQFldTU{Assign points to extended responses}

```

`\EsW` We fix the width `\EsW` and the height `\EsH` of `\essayQ` using commands, these `\EsH` can be redefined.

```

313 \def\Esw{33bp}\def\EsH{14bp}

```

Now for the definition of `\essayQ`

```

314 \def\essayQ#1{\let\qMark@HookSave\qMark@Hook
315   \def\qMark@Hook{\makebox[Opt][r]{\smash
316   {\raisebox{-7bp+\fboxsep}{\stepcounter{questionno}\textField[%
317   \cmd{\bParams{#1}\eParams}\F\FHidden\Q{1}
318   \AAkeystroke{\essayQKey}
319 %   \AAonfocus{var essayPtsAssigned=(1*event.value);}
320   \AAformat{if(event.value!="") event.value=event.value
321   +((event.value==1)?" \eqptLabel":" \eqptsLabel")}}
322   \TU{\essayQFld@TU}
323   ]{essayMrkUp.\currQuiz.\thequestionno}{\EsW}{\EsH}%
324   \addtocounter{questionno}{-1}}}\global

```

```
325 \let\qMark@Hook\qMark@HookSave}}
```

The way you pose an essay question is as follows:

```
\essayQ{5}
\item\PTs{5} The question ...\[3pt]
\RespBoxEssay{4in}{4\baselineskip}
```

`\essayitem{<num>}` We simplify this workflow a little, define `\essayitem`:

```
326 \def\essayitem#1{\essayQ{#1}\item\PTs{#1}}
```

Thus, we can now type,

```
\essayitem{5} The question ...\[3pt]
\RespBoxEssay{4in}{4\baselineskip}
```

9 Field level JavaScript for form field commands

`\pwdInstrFld` JS Keystroke action for `\pwdInstrFld`. This script uses three parameters passed to it through `\pwdInstrFld`: `@p(1)` is the quiz name (`\currQuiz`); `@p(2)` is the number of questions; and `@p(3)` is the password.

```
327 \begin{defineJS}[\makeesc\@]{\pwdKeyJS}
328 if (event.willCommit) {
329   if (event.value==@p(3)) {
330     @commonPassKey
331   }
332 }
333 \end{defineJS}
334 \begin{defineJS}[\makeesc\@makecmt\%]{\commonPassKey}
```

Added code from `\qz@IDTtxtField` to avoid the dreaded ‘q1 is undefined’ JavaScript error message. This happens when the Mark It control and the Begin Quiz controls are on different pages. When Mark It is pressed, ‘q1’ has not been defined yet, not until the next page.

```
335 if(typeof aQuizzesInDoc=="undefined")
336   var aQuizzesInDoc=new Array();
337 if (aQuizzesInDoc.indexOf("@oField"))
338   aQuizzesInDoc.push("@oField");
339 if (typeof @oField=="undefined")
340   var @oField=new Object;
341 restoreQuizData();
342 this.calculate=true;
343 @ifthtestmode@else%
344 var f=this.getField("postQzMsg");
345 if (f!=null) f.display=display.hidden;@fi
346 var f=this.getField("pbStuSvCl");
347 if (f!=null) f.display=display.hidden;
348 var f=this.getField("ScoreField.@p(1)");
349 if (f!=null) f.display=display.visible;
350 var f=this.getField("PointsField.@p(1)");
```



```

351 if (f!=null) f.display=display.visible;
352 var f=this.getField("EssayField.@p(1)");
353 if (f!=null) f.display=display.visible;
354 var f=this.getField("TotalsField.@p(1)");
355 if (f!=null) f.display=display.visible;
356 var f=this.getField("essayMrkUp");
357 if (f!=null) f.display=display.visible;
358 correctQuiz("@p(1)",@p(2));
359 var f=this.getField("qzreset");
360 if (f!=null) f.display=display.visible;
361 var f=this.getField("freezeQz");
362 if (f!=null) f.display=display.visible;
363 var f=this.getField("studentenReport");
364 if (f!=null) f.display=display.visible;
365 var f=this.getField("studentenGrade");
366 if (f!=null) f.display=display.visible;
367 if (typeof correctSumryTbl == "function")
368   correctSumryTbl("@p(1)",@p(2));
369 \end{defineJS}

```

`\essayQKey` Keystroke JS action for `\essayQ`. The `@p(1)` parameter is the weight of this essay question, it is passed to this script by `\essayQ`.

`\NoNumEnteredMsg{jsstr}` When you enter a non-number, and an alert box pops up with this as its message.

```

370 \def\NoNumEnteredMsg#1{\flJSStr*[noquotes]{\cNoNumEnteredMsg}{#1}}
371 \NoNumEnteredMsg{"You did not enter a number, %
372 enter a nonnegative number only"}

```

`\TooMuchCreditMsg{jsstr}` When you assign too much credit for the problem, an alert box appears containing this message.

```

373 \def\TooMuchCreditMsg#1{\flJSStr*[noquotes]{\cTooMuchCredit}{#1}}
374 \TooMuchCreditMsg{"You've assigned too much credit for this %
375 problem, assigning the maximum instead"}

```

Now the definition of `\essayQKey`

```

376 \begin{defineJS}[\makeesc\@makecmt\%]{\essayQKey}
377 if (event.willCommit) {
378   var qpts=(1*event.value);
379   if (isNaN(qpts)) {
380     app.alert(@cNoNumEnteredMsg);
381     event.rc=false;
382   } else if (qpts<0) {
383     event.value=-1*event.value;
384     qpts=1*event.value;
385   }
386   if (event.rc) {
387     if (qpts > @p(1) ) {
388       app.alert(@cTooMuchCredit);
389       qpts=@p(1);
390     }

```

```

391 // update ProbDist array
392 ProbDist[@thequestionno]=qpts;
393 // see if table is present
394 if (typeof correctSumryTbl == "function") {
395     f=this.getField("%
396 @dlcombine(@currQuiz)(SanityCheckPts).@thequestionno");
397     var thesePts= qpts + (( qpts == 1 )?%
398 " @eqptLabel":" @eqptsLabel");
399     f.value=thesePts;
400     // add color
401     var cb=this.getField("%
402 @dlcombine(@currQuiz)(SanityCheck).@thequestionno");
403     if (qpts==@p(1)) cb.strokeColor=@rghtColorJS;
404     else if (qpts>0) cb.strokeColor=@partialColorJS;
405     else cb.strokeColor=@wrngColorJS;
406 }
407 event.value=qpts;
408 }
409 }
410 \end{defineJS}

```

`\instrAutoSaveOn` When the instructor presses the freeze quiz control, there is an option to automatically save the document or not. `\instrAutoSaveOn` saves the document; `\instrAutoSaveOff` however, if `\instrAutoSaveOff` is expanded in the preamble, no automatic save is performed. The default is `\instrAutoSaveOn`.

```

411 \def\instrAutoSaveOn{\def\instrAutoSave{true}}
412 \def\instrAutoSaveOff{\def\instrAutoSave{false}}
413 \instrAutoSaveOn

```

`\instrAutoCloseOn` When the instructor presses the freeze quiz control, there is an option to silently close the document or not. `\instrAutoCloseOn` closes the document; however, if `\instrAutoCloseOff` is expanded in the preamble, no automatic closing occurs. The default is `\instrAutoCloseOn`.

```

414 \def\instrAutoCloseOn{\def\instrAutoClose{true}}
415 \instrAutoCloseOn
416 \def\instrAutoCloseOff{\def\instrAutoClose{false}}

```

`freezeQuizMU()` The mouse up JavaScript for `freezeQuiz()`. It makes all form fields *in the entire document* readonly. Use *only* after all markups are finished and document is ready to be moved into the student's folder.

```

417 \def\MarkWarningMsg#1{\d1JSStr*[noquotes]{\MarkWarning@Msg}{#1}}
418 \MarkWarningMsg{"You did not award the student a final mark.\
419 \n\nAward the mark and then save."}

```

`\flattenOn` The `\flattenOn` turns on flattening, while `\flattenOff` turns flattening off. The reason you would turn flattening off is to use Thor's way for basic methods and for the `useclass` option. The default is `\flattenOff` for basic methods and `\flattenOn` for `usebatch`. Applies only when the Freeze Quiz button is present.

```

420 \def\flattenOn{\def\bFlattenState{false}}

```

```

421 \def\flattenOff{\def\bFlattenState{true}}
422 \ifbasicmethods\flattenOff\else\flattenOn\fi
    The definition of freezeMU().
423 \begin{insDLJS}{jsforthor}{thorshammer: Freeze/Save Doc}
424 var sndSaveWarning=\SecondSave@Msg;
425 var isthereCvrPg=\thIsCP;
426 var cvrPgNum="\thCvrPg";
427 function freezeQuizMU() {
428 var f, fname;
429 var bOK=true;
430 var f=this.getField("studentenGrade");
431 var str="+f.value;
432 str=str.replace(/s/g,"");
433 if (str=="") {
434   app.alert(\MarkWarning@Msg);
435   bOK=false;
436 }
    Determine if there are solution pages, and if so, re-insert them.
437 var SolnSet=this.info.SolnSet;
438 if (bOK&&SolnSet!=""){
439 var SolnPath=this.info.SolnPath;
440 // var SolnSet=this.info.SolnSet;
441 var qzbasename=this.info.qzBaseName;
442   aebTrustedFunctions(this,aebInsertPages,{
443     nPage: (this.numPages-1),
444     cPath: SolnPath+"/"+qzbasename+"-"+SolnSet+".pdf"
445   })
446 };
    If \thUseNameToCustomize is true, we use the current file name; otherwise we use
    the original file name (\jobname)
447 if(\instrAutoSave&&bOK) {
448 //   var cSave="\jobname";
449   var docFN=this.documentFileName;
450   docFN=docFN.substring(0,docFN.length-4);
451   var cSave=(\thUseNameToCustomize)?"\jobname":docFN;
    If \thUseNameToCustomize is true, we append student and "-g" to signal that
    this file has been graded.
452   if(\thUseNameToCustomize) {
453     var f=this.getField("Name.first");
454     if(f!=null)cSave+=("-"+f.value+"_");
455     f=this.getField("Name.last");
456     if(f!=null)cSave+=(f.value);
457     cSave+=("-g");
458   }
459   var oRetn=aebTrustedFunctions(this,aebBrowseForDoc,{bSave:true,%
460 cFilenameInit: cSave });
461   bOK=(typeof oRetn=="object");

```

```
462  if(bOK) {
```

If the file name and path are chosen, we make all files readonly

```
463    for (var i=0; i<this.numFields; i++) {
464      fname=this.getNthFieldName(i);
465      f=this.getField(fname);
466      f.readonly=true;
467    }
```

After making all fields readonly, we hide the freeze quiz button itself.

```
468    var f=this.getField("MarkIt");
469    if (f!=null)f.display=display.hidden;
470    f=this.getField("freezeQz");
471    if (f!=null)f.display=display.hidden;
```

(2019/06/30) Jürgen suggested to flatten the document to add more security.

```
472    if(typeof _flattenThisDoc=="undefined")this.flattenPages();
```

Now we are ready to save the file

```
473    oRecordOfQuizData=undefined;
```

If instructor uses Thor's way, we don't want to reattach the solution page as it has already been reattached in this workflow.

```
474    this.info.SolnSet="";
475    var retn=aebTrustedFunctions(this,aebDocSaveAs,%
476 {cPath:oRetn.cPath,cFS:oRetn.cFS});
477 }
```

```
478 }
```

```
479 if(!instrAutoClose&& bOK) this.closeDoc(true);
```

```
480 }
```

```
481 \end{insDLJS}
```

```
482 \begin{defineJS}[\makeesc\@]{\freezeQuizMU}
```

```
483 var f, fname;
```

```
484 var bOK=true;
```

```
485 if(@instrAutoSave) {
```

```
486   var cSave="@jobname";
```

```
487   var f=this.getField("Name.first");
```

```
488   if(f!=null)cSave+=("-"+f.value+"_");
```

```
489   f=this.getField("Name.last");
```

```
490   if(f!=null)cSave+=(f.value);
```

```
491   var oRetn=aebTrustedFunctions(this,aebBrowseForDoc,{bSave:true,@%
```

```
492 cFilenameInit: cSave });
```

```
493   bOK=(typeof oRetn=="object");
```

```
494   if(bOK) {
```

If the file name and path are chosen, we make all files readonly

```
495     for (var i=0; i<this.numFields; i++) {
496       fname=this.getNthFieldName(i);
497       f=this.getField(fname);
498       f.readonly=true;
499     }
```

After making all fields readonly, we hide the freeze quiz button itself.

```
500     var f=this.getField("MarkIt");
501     if (f!=null)f.display=display.hidden;
502     f=this.getField("freezeQz");
503     if (f!=null)f.display=display.hidden;
```

(2019/06/30) Jürgen suggested to flatten the document to add more security.

```
504     this.flattenPages();
```

Now we are ready to save the file

```
505     var retn=aebTrustedFunctions(this,aebDocSaveAs,@%
506 {cFS:oRetn.cFS,cPath: oRetn.cPath });
507 }
508 }
509 if(@instrAutoClose&&b0K) this.closeDoc(true);
510 \end{defineJS}
```

10 Modifications and redefinitions of AeB

We modify various commands of exerquiz to conform the goals of the mighty Thor.

10.1 Quiz components modified

We begin by modifying the `\RespBoxEssay` action.

```
511 \def\@@RespBoxEssayActions{%
512   \AA{\if\eqQuizType\isQZ
513     \AAKeystroke{%
514       if(event.willCommit){\jsR\jsT
515         RecordPointValue(\eqPTs,\thequestionno);\jsR\jsT
516         RecordProblemType("\eqQT",\thequestionno);\jsR
```

The next three lines are inserted. After user has left the text field, we determine if he/she did anything. If `event.value`, stripped of all white space, is empty, nothing was done and we mark the response as `undefined`; otherwise we mark it as `"<essay>"`. The fact that the `Responses` array is nonempty for this question will cause a check mark to appear in the summary table.

```
517     var stripResp=stripWhiteSpace(event.value);\jsR\jsT
518     if(stripResp=="")Responses[\thequestionno]=undefined;\jsR\jsT
519     else Responses[\thequestionno]="<essay>";\jsR\jsT
520     if ( typeof fieldPopTbl == "function" ) fieldPopTbl("\currQuiz");
521   }\jsR
522   if (!isQuizInitialized("\curr@quiz")) {\jsR\jsT
523     \eqObjAlert\space eqAppAlert(
524       InitMsg("\bqlabelISO"),3);\jsR\jsT
525     event.rc = false;\jsR
526   }%
527 }%
528 \fi
529 }
```

530 }

We redefine `\@initQuiz` from `exerquiz` to first test whether name fields have been entered.

```
531 \def\InitQzMsg#1{\flJSStr*[noquotes]{\InitQzMsg@Msg}{#1}}
532 \InitQzMsg{"You cannot begin the quiz before entering
533 your first and last names in the fields provided.\n\n
534 Enter the name as you are known in the class; otherwise,
535 you will receive no credit for your work."}
536 \def\IfbQzChkSnippet{%
537 this.calculate=false;\jsR
538 if(\thOrdQz) bOk=true\jsR
539 else {\jsR\jsT
540 var f=this.getField("Name.first");\jsR\jsT
541 var str1=stripWhiteSpace(f.value);\jsR\jsT
542 var f=this.getField("Name.last");\jsR\jsT
543 var str2=stripWhiteSpace(f.value);\jsR\jsT
544 bOk=(str1!="&&str2!=");\jsR
545 }
546 if(bOk)}
547 \expandafter\def\expandafter\@initQuiz\expandafter
548 {\expandafter\IfbQzChkSnippet\expandafter{\@initQuiz}
549 else app.alert({cMsg:\InitQzMsg@Msg,cTitle:\ThorsAlert@Title});
550 }
```

`\postSubmitQuiz` Modify `\postSubmitQuiz`. When the End Quiz control is pressed, we make visible the post quiz message, placed in the document by the `\completeMsgFld` command.

```
551 \toks@=\expandafter{\postSubmitQuiz\t\t
552 oRecordOfQuizData["ProbDist.\oField"]=ProbDist;\r\t\t
553 oRecordOfQuizData["RightWrong.\oField"]=RightWrong;\r\t\t
554 \ifhtestmode\else
555 var f=this.getField("postQzMsg");\r\t\tfi
556 if (f!=null) f.display=display.visible;\r\t\t
557 var f=this.getField("pbStuSvCl");\r\t\t
558 if (\stuAutoSave&&f!=null)f.display=display.visible;} %\r\t\t
559 \edef\postSubmitQuiz{\the\toks@}
```

The action for the End Quiz button, we modify it to give the student a chance to reconsider his decision to end the quiz.

`\EndQzWarningMsg{jsstr}` The message that appears on the alert box asking to student to verify ending the quiz.

```
560 \def\EndQzWarningMsg#1{\flJSStr*[noquotes]{\EndQzWarning@Msg}{#1}}
561 \EndQzWarningMsg{"When you end the quiz, you cannot change
562 any of your answers without starting the quiz over from the
563 beginning.\n\n Press \"Yes\" to end the quiz."}
564 \def\ThorsAlertTitle#1{\flJSStr*[noquotes]{\ThorsAlert@Title}{#1}}
565 \ThorsAlertTitle{"Thor's Hammer"}
```

`\eq@EndQzBtnScriptThor` The modified script for the end of the `dps0624` quiz button. We rework the script of `\eq@EndQuizButtonActions`, taken from `exerquiz`.

```

566 \begin{defineJS}[\makeesc\*\makecmt\%]{\eq@EndQzBtnScriptThor}
567 if (!isQuizInitialized("*currQuiz"))
568   eqAppAlert(InitMsg("*bqlabelISO"),3);
569 else {
570   var retn=app.alert({cMsg: *EndQzWarning@Msg,%
571 cTitle: *ThorsAlert@Title, nIcon: 2, nType: 2});
572   if (retn==4) {
573     if (*minQuizResp(*thequestionno)&&_ModalNotOn){
574       *currQuiz.PtValues=(new %
575 Array(*pointValuesArray));
576       ProbType=[*ptypeArray];
577 *if@inclkey
578       *currQuiz.CorrAns=(new %
579 Array(*corrAnsArray));
580 *fi%
581       DisplayQuizResults("*currQuiz",*theeqpointvalue,%
582 *thequestionno);
583       var h=this.getField("ScoreData.*currQuiz");
584       h.value=Score+";"+NQuestions+";"%
585 +ptScore+";"+NPointTotal;
586 %       *eq@submitURL
587       *postSubmitQuiz
588       resetQuiz("*currQuiz");
589     }
590   }
591 }
592 \end{defineJS}

```

Now, we redefine \eq@@EndQuizButtonActions of exerquiz.

```

593 \def\eq@@EndQuizButtonActions{\A{\JS{\eq@EndQzBtnScriptThor}}} % dps0624
594 \let\eq@@EndQuizButtonActionsThorSave\eq@@EndQuizButtonActions % dps0624

```

\useEndQuizThor Define \useEndQuizThor to restore the End Quiz control to the action defined in this package. (Other packages may removed this End Quiz action.)

```

595 \def\useEndQuizThor{\let\eq@@EndQuizButtonActions
596 \eq@@EndQuizButtonActionsThorSave}

```

Add a SaveAs menu item to end of the quiz

\stuAutoSaveOn When expanded in the preamble, a save button will appear (\stuSaveBtn) when the End Quiz control is pressed. A dialog appears to save the file, the student can choose the file location and the file name at that time. When \stuAutoSaveOff is in effect, the save button does not appear, and the student must press the save button the on Adobe Reader toolbar. The default is \stuAutoSaveOn.

```

597 \let\stuASOn\ef@YES
598 \def\stuAutoSaveOn{\let\stuASOn\ef@YES
599 \def\stuAutoSaveScript{\t app.execMenuItem("SaveAs");\r}%
600 \def\stuAutoSave{true}}
601 \def\stuAutoSaveOff{\let\stuASOn\ef@NO
602 \let\stuAutoSaveScript@empty
603 \def\stuAutoSave{false}}

```

```

604 \stuAutoSaveOn
\stuAutoCloseOn This command is obeyed only if \stuAutoSaveOn is in effect. After the student
presses the save button (\stuSaveBtn), the document is closed after the student
save the document. Note that if the student cancels saving the document and
if the document still needs saving, the document is not closed. The default is
\stuAutoCloseOn.
605 \def\stuAutoCloseOn{\def\stuAutoCloseScript{\t
606 if(!this.dirty)this.closeDoc(true);\r}%
607 \def\stuAutoClose{true}}
608 \stuAutoCloseOn
609 \def\stuAutoCloseOff{\let\stuAutoCloseScript\@empty
610 \def\stuAutoClose{false}}
\stuSaveBtnCA{jsstr} The caption for this button
611 \def\stuSaveBtnCA#1{\def\stuSaveBtn@CA{#1}}
612 \stuSaveBtnCA{Save}
\stuSaveBtnTU{jsstr} The tool tip for this button
613 \def\stuSaveBtnTU#1{\def\stuSaveBtn@TU{#1}}
614 \stuSaveBtnTU{Press to save and close the document}
\stuSaveBtn[options]{wd}{ht} This button is initially hidden and becomes visible within
the student presses the End Quiz control; provided \stuAutoSaveOn is in effect.
The button saves and optionally closes the document.
\autoSaveStuJS is a revised version of the JavaScript action for \stuSaveBtn. If the JavaScript
method aebTrustedFunctions is undefined, we use the old code; otherwise, we
use the new code.
\SecondSaveMsg{msg} is an alert dialog message stating the the document was not saved. This
declaration mush occur on the preamble of in the CFG file, or is has no effect.
615 \def\SecondSaveMsg#1{\d1JSstr*[noquotes]{\SecondSave@Msg}{#1}}
616 \SecondSaveMsg{"Alert! This document has not been saved, do not
617 exit before saving!"}
When aebTrustedFunctions is defined for AR, we offer two methods for the
student to save the document: (1) \useStuSaveAsDialogOff (the default) is the
most seamless method, no save-as dialog is offered, the student asked to confirm
the save; (2) \useStuSaveAsDialogOn offers the save-as dialog (but streamlined).
These two commands must appear on the preamble or CFG file, or they have no
effect.
618 \newif\ifUseStuSaveAsDialog\UseStuSaveAsDialogfalse
619 \def\useStuSaveAsDialogOn{\UseStuSaveAsDialogtrue}
620 \def\useStuSaveAsDialogOff{\UseStuSaveAsDialogfalse}
621 \begin{defineJS}[\makeesc*\makecmt\%]{\autoSaveStuJS}
622 var bOK=true;
623 global.bOkClose=true;
624 var _path=this.path;
625 var pos=_path.lastIndexOf("/");
626 var currentFolder=_path.substring(0,pos+1);

```



```

627 var docFN=this.documentFileName;
628 docFN=docFN.substring(0,docFN.length-4);
629 var cSave>(*thUseNameToCustomize)?"*jobname":docFN;
630 currentFolder=currentFolder+cSave+".pdf";
631 if (typeof aebTrustedFunctions=="undefined")
632     app.execMenuItem("SaveAs");
633 else {

```

In this controlled environment of taking PDF quizzes at an institution, the AeB special function `aebTrustedFunctions` is defined, along with supporting functions.

```

634 *ifUseStuSaveAsDialog%
635     var oRetn=aebTrustedFunctions(this,aebBrowseForDoc,%
636 {bSave:true,cFilenameInit: cSave });
637     bOK=(typeof oRetn=="object");
638 *fi%

```

If the user dismisses the browse-for-doc dialog, the return value (`oRetn`) is undefined; in this case, we do not save or close the document. The user must initiate the action again.

```

639     aebDocSaveAs.msg="";
640     aebDocSaveAs.action=%
641 'global.bOkClose=false;app.alert(""+sndSaveWarning+')';
642     if (bOK) var retn=aebTrustedFunctions(this,aebDocSaveAs,%
643 {cPath:*ifUseStuSaveAsDialog oRetn.cPath*else currentFolder*fi });
644     else app.alert(sndSaveWarning);
645 }
646 if(*stuAutoClose&&bOk&&global.bOkClose&&!this.dirty)
647     delete global.bOkClose;
648     this.closeDoc(true);
649 \end{defineJS}

```

We finally reach the definition of `\stuSaveBtn`

```

650 \newcommand\stuSaveBtn[3] []{\pushButton[\F\FHidden
651 \CA{\stuSaveBtn@CA}\TU{\stuSaveBtn@TU}

```

Here, we leverage the new `\cmd` command to test if auto save is on, if not we gobble the `\AAmouseup` action.

```

652 \cmd{\ifx\stuASOn\ef@NO\let\@eqAAmouseup\@gobble\fi}
653 \AAmouseup{if(\stuAutoSave){\r
654     \autoSaveStuJS
655 %     \stuAutoSaveScript\stuAutoCloseScript
656 }}\protect\AA\protect\F#1
657 ]{\pbStuSvCl}{#2}{#3}}

```

`\DeclareQuiz{<gz-name>}` We modify the `\DeclareQuiz` command of `exerquiz`, by appending some code that defines `\eq@prior@endQuiz` to write the number of questions and the number of points to the AUX file. This command *must appear* at the top of the source file, just after `\begin{document}` or in the *preamble*, it defines `\currQuiz` for the rest of the document, and write to AUX file. When using multi-quizzes in one source file, this package redefines the `\currQuiz`; for example, if originally, you

Required in preamble or at top of file

declared `\DeclareQuiz{Quiz1}`, the first rendition uses the quiz name `Quiz1a`, the second `Quiz1b`, and so on. (Limit of 26 renditions). To preserve the original quiz name, we define `\thQuizName`, this command expands to `\Quiz1` within all renditions; consequently, can be used in the running head to consistently display the quiz name.

```

658 \let\DeclareQuizSAVE\DeclareQuiz
659 \def\DeclareQuiz#1{\def\thQuizName{#1}\th@DeclareQuiz{#1}}
660 \def\th@DeclareQuiz#1{\DeclareQuizSAVE{#1}%
661   \expandafter\gdef\expandafter
662   \eq@prior@endQuiz\expandafter{\eq@prior@endQuiz\wrtQzInfo}}

```

`\thQzName{friendly-qz-name}` This is the friendly (human readable) quiz name, suitable for use in the running header and elsewhere.

```

663 \def\thQzName#1{\def\thqzname{#1}}
664 \thQzName{\thQuizName}

665 \def\wrtQzInfo{\eq@IWAuxOut{\string
666   \csarg\string\gdef{\currQuiz-nQs}{\thequestionno}^^J\string
667   \csarg\string\gdef{\currQuiz-nPts}{\theqpointvalue}}}

```

`\eqQuizPointsMsg` We modify `\eqQuizPointsMsg`, its default definition is the string

```
"\eqptScore\space"+ptScore+" \eqOutOf\space"+nPointTotal
```

but for Thor's way, we simplify to `ptScore`.

```
668 \renewcommand\eqQuizPointsMsg{ptScore}
```

10.2 Modify margin points markup

Make the markup boxes in the margins larger.

```

669 \renewcommand{\aeb@creditmarkup}{\bgroup
670   \edef\markupWidth{\EsW}\edef\markupHeight{\EsH}%
671   \textField[\Ff\FfReadOnly\BC{}]\F\FHidden
672   \textColor{\pcMarkupColor}\textSize{\markupTextSize}\autoCenter{y}%
673   \DV{0 \eqptsLabel}\V{0 \eqptsLabel}}%
674   {qMark.\currQuiz.\thequestionno.\arabic{qMarkCnt}}%
675   {\markupWidth}{\markupHeight}\egroup}

```

10.3 Modifications to the summary table

We modify the summary table to make the markup points larger and left align the second column. Thor may come down on me with his mighty hammer, but I'll take the chance.

```

676 \def\eq@begintab{% second column left aligned
677   \begin{tabular}[t]{llc}\sumryTbl1Q&\sumryTbl1R&\sumryTbl1P\\\sthline
678   {\Large\strut}}%
679 \let\st@scndclmnSAVE\st@scndclmn

```

Offset the check boxes by 2bp to better align with the heading

```
680 \def\st@scndclmn{\kern2bp\st@scndclmnSAVE}
```

Increase the width of the markup boxes (from 12bp to 20bp, and change to a fixed text size

```
681 \def\stmarkupWidth{20bp} % normally 12bp
682 \def\stmarkupHeight{9bp} % unchanged
683 \def\stmarkupTextSize{8} % normally Opt
```

Offset the check boxes by 2bp to better align with the heading

```
684 \def\stmarkupbox{\mbox} % normally {\makebox[Opt][l]}
685 %\def\sumrytblLinkHook#1{\the\value{page}}
686 \def\st@thrdclmn#1{\setLink[\linktxtcolor{black}}
687 \A{\JS{this.pageNum=(this.pageNum+#1-1)}}{\sumrytblLinkHook{#1}}
```

10.4 Boom! Thor’s thunders: “Thor needs solutions!”

The package was complete, then it wasn’t. `\RespBoxEssay` never supported solutions, so that needed to be fixed, now requiring `exerquiz` dated 2019/08/13 or later.

The first issue addressed here is the labeling of the solutions to the quiz. We try a simple enumeration of the solutions. For that, the `\fancyQuizHeaders` is used from `exerquiz`.

```
688 \fancyQuizHeaders
689 \setsolnspace{}
690 \let\FncyHdrsFmtNoTitleQuiz\@empty
```

The question numbers protrude into the left margin, to disguise this, we shift the running header over a little

```
691 \setlength{\eflength}{\widthof{\textbf{00.}\space}}
692 \edef\th@leftShiftHdr{\the\eflength}
693 \def\th@HeaderOffset{\hskip-\th@leftShiftHdr\relax}
694 \def\doNotShirtSonsHdrs{\let\th@HeaderOffset\relax}
```

`\thQzSolnMrkr` `\thQzSolnMrkr` is a small text field that is inserted under the section title. This is used to identify on what page the solutions begin. Later used by `\sadQuizzes`.

```
695 \def\thQzSolnMrkr{\textField[BC]{\thsolns4.\currQuiz}{1bp}{1bp}}
```

The quiz numbers will go in the left margin, so we’ll shift the section title over a little to disguise this.

```
696 \def\quizSolnsHeadnToc{\section*
697   {\makebox[Opt][l]{\th@HeaderOffset
698     \thQzSolnMrkr\sqlsectitle}}%
699   \addcontentsline{toc}{section}{%
700     \@ifundefined{web@latextoc}{}%
701     \ifx\web@latextoc\eq@YES\else
702     \protect\numberline{}\fi}\sqlsectitle}}
703 \renewcommand\eq@sqlsectitle{Solutions to the Quiz}
```

`\myFQHfmt` describes the numbering scheme for the solutions.

```
704 \newcommand\myFQHfmt{%
705   \string\bfseries\string\color{\fncyQHdrsColor}%
706   \ifx\aeBTitleQuiz\@empty
```

```

707     \ifnum\@eqquestiondepth>0\relax
708         \FncyHdrsFmtNoTitleQuiz\fi\else
709         \aebTitleQuiz\protect\
710         \ifnum\@eqquestiondepth=0\else\\\relax
711         \FncyHdrsFmtQuestion\fi
712 \fi %\space
713 \ifcase\@eqquestiondepth
714     \ifx\aebTitleQuiz\@empty\FncyHdrsFmtNoTitleQuiz\fi
715     \or
716     \string\llap{\arabic{eqquestionnoi}.\space}%
717     \or
718     \string\llap{\arabic{eqquestionnoi}.\space}%
719     (\alph{eqquestionnoi})\space
720     \or
721     \string\llap{\arabic{eqquestionnoi}.\space}%
722     (\alph{eqquestionnoi})%
723     (\roman{eqquestionnoiii})\space
724 \fi
725 }
726 \dclrFncyQzHdrsFmt{\myFQHfmt}
    No return symbol or link to the question.
727 \let\ReturnTo\@gobbletwo

```

10.5 Modifications to the **web** package

The TEX template file (`tex-template.tex`, generated by `thmclass.ps1`) specifies the `web` package and uses many command particular to that package. Here, we create a special command to input customization commands that are specified in the `web.cfg` file. Place `\inputWebCfg` in the preamble to input the `web.cfg`; any `\ExecuteOptions` commands are ignored. Customization commands are placed between the two marks `\bWebCustomize` and `\eWebCustomize`.

```

728 \let\bWebCustomize\endinput
729 \let\eWebCustomize\relax
730 \providecommand{\inputWebCfg}{%
731     \let\bWebCustomize\relax
732     \let\eWebCustomize\endinput
733     \let\ExecuteOptions@SAVE\ExecuteOptions
734     \let\ExecuteOptions\@gobble
735     \makeatletter
736     \InputIfFileExists{web.cfg}{-}{-}\makeatother
737     \let\ExecuteOptions\ExecuteOptions@SAVE
738     \let\bWebCustomize\endinput
739     \let\eWebCustomize\relax
740 }

```

11 The useclass option and above

These options (useclass, usebatch, and batchdistr) are designed for the mass production of the quizzes, one for each student in the class. The quiz is built and saved (for each student), saved to the instructor's designated folder, as declared by `\instrPath`, and to the student's personal folder as declared within the `\classMember` entry and `\classEntries` array.

11.1 Declaring class members

In conjunction with `\instrPath` and `\classPath`, use `\classMember` to declare the identity of each member of the class.

`\classMember*{⟨first-name⟩{⟨last-name⟩}*{⟨folder|path⟩}` Enter the first name, last name, and folder name of each student in the class. When the star form is used, `⟨first-name⟩` and `⟨last-name⟩` are first passed through `\pdfstringdef`. If the second star-open is specified between the second and third arguments, the third argument should be the absolute path to the (exceptional) student.

There are several ways of producing characters in the Latin-1 character set:

- `\u` • unicode method: `\classMember{J\u00FCrgen}{Loki}{B}`
- `\classMember*` • using `\pdfstringdef`, in this case use the star version of `\classMember`
 `\classMember*{J"u}rgen}{Loki}{B}`
- `\oct` • octal method: `\classMember{J\oct374rgen}{Loki}{B}` or
 `\classMember{J\string\374rgen}{Loki}{B}`

```
741 \def\classEntriesDef{["", "", "", false]}
742 \newif\ifClassEntries\ClassEntriesfalse
743 \let\classEntries@gobble
744 \def\classMember{\ClassEntrytrue\@ifstar
745   {\let\th@star\ef@YES\classMember@i}
746   {\let\th@star\ef@NO\classMember@i}}
747 \newcommand\classMember@i[2]{%
748   \@ifstar{\let\th@exstar\ef@YES\classMember@ii{#1}{#2}}
749   {\let\th@exstar\ef@NO\classMember@ii{#1}{#2}}}
750 \newcommand\classMember@ii[3]{%
751   \ifx\th@exstar\ef@YES\def\AbsPth{true}\else
752     \def\AbsPth{false}\fi
753   \ifx\th@star\ef@NO
754     \ifx\th@exstar\ef@YES
755       \g@addto@macro\classEntries{["#1", "#2", "#3", true]}\else
756       \g@addto@macro\classEntries{["#1", "#2", "#3", false]}\fi
757   \else
758     \g@addto@macro\classEntries{,["}%
759     \pdfstringdef\x{#1}\expandafter
760     \g@addto@macro\expandafter\classEntries\expandafter{x}%
761     \g@addto@macro\classEntries{" ,"}%
```

```

762 \pdfstringdef\x{#2}\expandafter
763 \g@addto@macro\expandafter\classEntries\expandafter{x}%
764 \g@addto@macro\classEntries{","}%
765 \pdfstringdef\x{#3}\expandafter
766 \g@addto@macro\expandafter\classEntries\expandafter{x}%
767 \ifx\th@exstar\ef@YES
768 \g@addto@macro\classEntries{",true"]\else
769 \g@addto@macro\classEntries{",false"]\fi
770 \fi}

```

11.2 Some process controls

During document development, you don't want to copy the files each time you build and review the document. Set `\autoCopyOff` during quiz development, and declare `\autoCopyOn`. The default is `\autoCopyOn`.

```

771 \def\autoCopyOn{\def\autoCopy{true}}
772 \def\autoCopyOff{\def\autoCopy{false}}
773 \autoCopyOn

```

`\cFS{empty|CHTTP}` (This command is obsolete, and should be removed. Its functionality is accessed through the optional arguments of `\instrPath` and `\classPath`.) The `Doc.saveAs()` method has a `cFS` key for determining the file system, the value of the key is either `empty` or the string `CHTTP`. We offer this option. This key is recognized for the `\classPath`, we assume the `\instrPath` is in his local file system; however, it is easy to incorporate the `cFS` key here as well.

```

774 \newcommand{\cFS}[1]{\def\@rgi{#1}\ifx\@rgi\@empty
775 \let\cFSth\@empty\else\def\cFSth{CHTTP}\fi}
776 \let\cFSth\@empty

```

`\distrToStudentsOff` Allow the instructor to turn off the distribution of the quizzes to the students' folders using `\distrToStudentsOff`, the default is `\distrToStudentsOn`.

```

777 \def\distrToStudentsOn{\def\distrToStudents{true}}\distrToStudentsOn
778 \def\distrToStudentsOff{\def\distrToStudents{false}}

```

`\distrToInstrOff` Allow the instructor to turn off the distribution of the quizzes to himself by using `\distrToInstrOff`, the default is `\distrToInstrOn`.

```

779 \def\distrToInstrOn{\def\distrToInstr{true}}\distrToInstrOn
780 \def\distrToInstrOff{\def\distrToInstr{false}}

```

(2019/08/26) Combined `\sadMultQuizzes` with `\sadQuizzes`

11.3 Working with multiple quizzes in one source

This section we develop some ideas of creating a single source file with multiple quizzes, these quizzes should be roughly equivalent. One such approach is to have a single quiz, and randomly permute the questions as well as randomly permute any MC or MS choice fields.

`\declareQuizBody{<name>}` This macro defines a verbatim environment with `<name>`. Such an environment cuts and saves its contents under the name of `<name>.cut`. It typically is designed to enclose the ‘body of a quiz,’ the ‘quiz body’ can then be input later using `\InputQuizBody`, define below. Associated with the declaration is a `\QzVer` version number, `\QzVer`, which may be used in the titles, refer to `thexrt.tex` in the `examples/misc` folder.

```

781 \def\th@QzVer{0}
782 \def\QzVer{1}
783 \newcommand{\declareQuizBody}[1]{%
784   \bgroup\@tempcnta\th@QzVer\relax
785   \advance\@tempcnta@ne
786   \edef\th@qbCnt{\the\@tempcnta}%
787   \csarg\xdef{#1-QzVer}{\th@qbCnt}\egroup
788   \csarg\def{#1}{\immediate\openout\CommentStream #1.cut
789     \let\verbatim@out\CommentStream
790     \immediate\write\verbatim@out{\string
791       \def\string\QzVer{\@nameuse{#1-QzVer}}}%
792     \verbatimwrite}%
793   \csarg\def{end#1}{\endverbatimwrite
794     \immediate\closeout\CommentStream}}

```

`\InputQuizBody{<name>}` We input a ‘quiz body’ that has been earlier CUT and saved under the name of `<name>.cut`.

```

795 \newcounter{th@qzCnt}
796 \def\theth@qzCnt{\alph{th@qzCnt}}
797 \let\qzLtr\theth@qzCnt % dps5-29
798 \newcommand{\InputQuizBody}[1]{\newpage %\thPageOne
799   \@ifundefined{thisQuizOrig}{\edef\thisQuizOrig{\thisQuiz}
800     \let\Hy@EveryPageAnchor\relax}{\stepcounter{th@qzCnt}}%
801   \edef\x{\thisQuizOrig\theth@qzCnt}\expandafter
802     \th@DeclareQuiz\expandafter{x}%
803   \renewcommand\sqlsecrunhead{}%
804   \InputIfFileExists{#1.cut}{-}{-}

```

Before we include quiz solutions, we close the `\quiz@solns` stream.

```

805 \immediate\closeout\quiz@solns %\th@QzHeaderLS
806 \let\eq@normallheader\relax
807 \newpage
808 \@ifundefined{ps@webheadings}{-%
809   \def\th@QzHeaderL{\th@QzHeaderLS}%
810   \def\th@QzHeaderC{\th@QzHeaderCS}%
811 }-%
812 \lheader{\th@QzHeaderLS}%
813 \cheader{\th@QzHeaderCS}%
814 }
815 \includequizzesolutions*\relax
816 \global\therearequizzesolutionsfalse
817 \renewcommand\sqlsecrunhead{\eq@sqlsecrunhead}%
818 \eq@noformstrue

```

Putting `\eq@noformtrue` assures us that the solution file will not be input a `\end{document}`. Next, we open a new quiz solution file stream so the another rendition to write solutions to a fresh file.

```

819 \immediate\openout \quiz@solns \jobname.qsl
820 \ifthordinary\else
821   \@ifundefined{ps@webheadings}{%
822     \def\th@QzHeaderL{\th@QzHeaderLQ}%
823     \def\th@QzHeaderC{\th@QzHeaderCQ}%
824   }{%
825     \lheader{\th@QzHeaderLQ}%
826     \cheader{\th@QzHeaderCQ}%
827   }%
828 \fi
829 }
```

11.4 Building quizzes with `makeClassFiles` & `\sadQuizzes`

Central to this whole process is building customized quizzes. This done by the `\sadQuizzes` expanded within the `makeClassFiles` environment.

`makeClassFiles` is an `execJS` environment, the base name has been preset to be `mcftThor`. The contents of this environment is `\sadQuizzes`. Beginning the 2019/07/15 of `insdljs`, `execJS` has an option argument that is used to pass a command to the `.djs` file.

`\oct` We use this to make special definitions to support `\oct` and `\u`.

```

\u
830 \def\mkClFlsSpcls{\let\oct\eqbs\let\u\relax}
831 \newenvironment{makeClassFiles}{%
832 \execJS[\mkClFlsSpcls]{mcftThor}}{\endexecJS}
```

`\sadQuizzes` (save and distribute quizzes) is a script for the `makeClassFiles` environment.

```

\begin{makeClassFiles}
\sadQuizzes
\end{makeClassFiles}
\begin{document}
...
```

The above is placed just above `\begin{document}`. The script populates, for each entry in the `\classEntries` array, the `Name.first` and `Name.last` fields with the student's first and last name. It then saves a copy of the document to the instructor's folder under the name `\jobname-⟨first-name⟩-⟨last-name⟩`. It does the same thing for the student's private folder. Finally, it clears the `Name` fields, and saves itself to the source folder. The script may be redefined in the preamble of the document using the `defineJS*` environment. The script also deals with solution and cover pages.

```

833 \def\setClassArray{\ifClassEntries
834   \classEntries\else\classEntriesDef\fi}
835 \def\setArrayLength{\ifbasicmethods0\else lst.length\fi}
836 \def\setfilesuffix{\ifuseclassOpt"-"+fN+"_" +1N\else
837   \ifbasicmethods""\else"-"+(i+1)\fi\fi+".pdf"}
```


Begin \sadQuizzes here.

```
838 \begin{defineJS}[\dfnJSCR{^^J}\let\u\relax\makeesc\@]{\sadQuizzes}
```

If \autoCopyOff, then this script does nothing

```
839 if(@bFlattenState)
```

```
840   this.addScript({
```

```
841     cName: "thorshammer: Do not flatten",
```

```
842     cScript:"var _flattenThisDoc=false;"
```

```
843   });
```

```
844 if (@autoCopy) {
```

JavaScript variables common to building quizzes

```
845   var bUseClass=@bUseClass;
```

```
846   var reRmFn=new RegExp(this.documentFileName,"i");
```

```
847   var instrPath=@InstrPath;
```

```
848   var cLast=instrPath[instrPath.length-1];
```

```
849   if (cLast=="/")
```

```
850     instrPath=instrPath.substring(0,instrPath.length-1);
```

```
851   var classPath=@ClassPath;
```

```
852   cLast=classPath[classPath.length-1];
```

```
853   if (cLast=="/")
```

```
854     classPath=classPath.substring(0,classPath.length-1);
```

```
855   var thInstrFS="@thInstrFS";
```

```
856   var thClassFS="@thClassFS";
```

```
857   var isthereCvrPg=@thIsCP;
```

```
858   var cvrPgNum="@thCvrPg";
```

```
859   console.println("autocopy "+((@autoCopy)?"on":"off"));
```

```
860   var retN;
```

```
861   var solnSuffix="";
```

```
862   var oSolnSuffix=new Object;
```

```
863   var parentoDoc=this;
```

```
864   var workingFolder=this.path;
```

```
865   var pos=workingFolder.lastIndexOf("/");
```

```
866   workingFolder=workingFolder.substring(0,pos+1);
```

```
867   var _workingFolder="";
```

```
868   // console.println("working folder: " + workingFolder);
```

```
869   var willSaveScript='isAQuizUnfinishedAtSave();\r'
```

```
870     +'if (oRecordOfQuizData !=undefined) collectQuizData();';
```

```
871   var oHSD=this.getField("holdScoreData");
```

```
872   var Rect=oHSD.rect;
```

```
873   this.removeField("holdScoreData");
```

```
874   var hsdFmt='if(typeof oRecordOfQuizData=="undefined")\r\t\
```

```
875     oRecordOfQuizData=new Object;';
```

```
876   var restQD='@restoreQD';
```

Cover page or pages. Determine if there are cover pages, if yes, extract it and save it to the instructor's folder. Cover pages are declared in the preamble with the command \DeclareCoverPage, the argument of which is either a single number (usually 0) of a range of zero-based page numbers.

```
877   if(isthereCvrPg) {// ----- extract cover pages -----
```

```

878     var aCvrPgRng=cvrPgNum.split("-");
879     if (aCvrPgRng[0]=="") {
880         console.println("Start Range not specified, using 0 instead");
881         var bPg=0;
882     } else var bPg=aCvrPgRng[0];
883     var ePg=(aCvrPgRng.length>1)?aCvrPgRng[1]:aCvrPgRng[0];
884     var oDoc=aebTrustedFunctions(this,aebExtractPages,
885         {nStart: bPg, nEnd: ePg});
886     // save cover page(s) to the instructor folder
887     _workingFolder=(@InstrPathFull)?":workingFolder;
888     aebDocSaveAs.msg="Cannot access the local folder "
889         + _workingFolder+instrPath+"/@jobname-cvrpg.pdf";
890     var retn=aebTrustedFunctions(oDoc,aebDocSaveAs,
891         {cFS:thInstrFS,cPath: _workingFolder+instrPath+"/@jobname-"
892         +"cvrpg.pdf",bCopy:false});
893     oDoc.dirty=false;
894     oDoc.closeDoc(true);
895     // delete cover page before continuing, will reinsert it later
896     this.deletePages({nStart: bPg, nEnd: ePg});
897     this.dirty=false;
898 }
899 // ----- end cover page code -----
900 var nQz=aQuizzesInDoc.length; // number of quizzes this doc

```

Solution pages. Before getting into generating the customized quizzes for the class, we must first see if there are any solutions in this document, if so, we separate the solution page(e) from the quiz.

```

901 var bOkBasicSolns=false;
902 var f=this.getField("thsolns4");
903 if (f != null ) {
904     console.println("There are solutions");
905     bOkBasicSolns=true;
906
907     Save the solution suffixes for later use
908     var g=f.toArray();
909     for (var i=0; i<g.length; i++) {
910         var solnSuffix=g[i].name; // eg solns4.q1a
911         var pos=solnSuffix.indexOf(".");
912         var qzName=solnSuffix.substring(pos+1); // eg q1a
913         solnSuffix=solnSuffix.replace(/\./g,"-"); // eg solns4-q1a
914         // oSolnSuffix records whether a quiz has solution pages
915         oSolnSuffix[qzName]=solnSuffix;
916     }

```

Extract the solution page save it, close it, delete the page from master document

```

915 // ----- extraction-----
916 // console.println("aQuizzesInDoc: " + aQuizzesInDoc.toSource());
917 // console.println("nQz="+nQz);
918 for (var i=0; i< nQz; i++) {
919     var f=this.getField("thsolns4");
920     var g=f.toArray();

```

```

921     var qzName=aQuizzesInDoc[i];
922     var bOk2Extract=(typeof oSolnSuffix[qzName]!="undefined");
923     if (bOk2Extract) {
924 // console.println(g[0].name + ", begins on page "+g[0].page);
925         var bPg=g[0].page;
926 // console.println("bPg= " + bPg);
927 //         var f=this.getField("Name.first."+i);
928         var f=this.getField("bMrkQz."+i);
929         var ePg=(f==null)?(this.numPages-1):(f.page-1);
930 // console.println("ePg= " + ePg);
931         var solnSuffix=g[0].name; // solns4.<qzName>
932         solnSuffix=solnSuffix.replace(/\.\/g,"-"); // solns4-<qzName>
933         var oDoc=aebTrustedFunctions(this,aebExtractPages,
934             {nStart: bPg, nEnd: ePg});
935         // save
936         _workingFolder=(@InstrPathFull)?":workingFolder;
937         aebDocSaveAs.msg="Cannot access the local folder "
938             + _workingFolder+instrPath+"/@jobname-"+solnSuffix+".pdf";
939         var retn=aebTrustedFunctions(oDoc,aebDocSaveAs,
940             {cFS:thInstrFS,cPath: _workingFolder+instrPath+"/@jobname-"
941                 +solnSuffix+".pdf",bCopy:false});
942         // close
943         oDoc.dirty=false;
944         oDoc.closeDoc(true);
945         // delete solution page before continuing
946         this.deletePages({
947             nStart: bPg,
948             nEnd: ePg});
949         this.dirty=false;
950     }
951 }
952 }
953 // ----- begin creating custom quizzes for class members -----
954 var cnt=0; // determines which quiz to generate
955 var lst=new Array(@setClassArray);
956 var l=(bUseClass)?@setArrayLength:1; // dps
957 for (var i=0; i < l; i++) {
958     var qzName=aQuizzesInDoc[cnt];
959 // console.println("Working on " + qzName);
960     var fN=lst[i][0];
961     var lN=lst[i][1];
962     var folder=lst[i][2];
963     var isAbsPth=lst[i][3]; // dps
964     if (folder!="")folder+="/";
965     // pre-populate with the student's name
966     this.getField("Name.first").value=fN;
967     this.getField("Name.last").value=lN;
968     // extract quiz
969 //     var f=this.getField("Name.first."+cnt);
970     var f=this.getField("bMrkQz."+cnt);

```

```

971     var bPg=f.page;
972 //     var f=this.getField("Name.first."+cnt+1));
973     var f=this.getField("bMrkQz."+cnt+1));
974     var ePg=(f==null)?(this.numPages-1):(f.page-1);

```

Extraction: We extract a quiz from the master document. Extracting causes some problems: the restore quiz data is lost, the WillSave event is lost, and the holdScoreData field is lost. We try to overcome this problems.

```

975     // ----- extraction of quiz pages -----
976     if(bUseClass)var oDoc=aebTrustedFunctions(this,aebExtractPages,
977         {nStart: bPg, nEnd: ePg});
978     else oDoc=this; // dps

```

Restore the WillSave event.

```

979     // extracting preserves doc JS but not doc actions
980     oDoc.setAction({cTrigger: "WillSave", cScript: willSaveScript});

```

We have had problems with oRecordOfQuizData is undefined, to (finally) overcome this, we place some code at the document level.

```

981     oDoc.addScript({
982         cName: "oRecordOfQuizData Obj Declaration", cScript: hsdFmt});

```

The holdScoreData is on the first page of the document, this first page may not survive on the extracted pages, so we place this text field and the top of each of the first pages of the extracted quizzes.

```

983     var oDocHSD=oDoc.addField({
984         cName: "holdScoreData",
985         cFieldType: "text",
986         nPageNum: 0,
987         oCoords: Rect
988     });

```

Create a page open action to execute the restore quiz data

```

989     oDoc.setPageAction({
990         nPage: 0,
991         cTrigger: "Open",
992         cScript: restQD
993     });

```

We save custom info to each of the files being saved: StudentPath has the path to the students' folders and cFS is the file system. These two are used by 'protect and distribute'.

```

994     oDoc.info.qzBaseName="@jobname";
995     if(isAbsPth) // dps
996         oDoc.info.StudentPath=folder;
997     else
998         oDoc.info.StudentPath=classPath+"/"+folder;
999     var bOkSolns=(typeof oSolnSuffix[qzName]!="undefined");
1000     if(bOkSolns) oDoc.info.SolnSet=oSolnSuffix[qzName];
1001     oDoc.info.SolnPath=_workingFolder+instrPath+"/";
1002     if(isthereCvrPg) oDoc.info.CvrPg="cvrpg";
1003     oDoc.info.cFS=thInstrFS;

```

Insert cover page, if any

```
1004 // Insert cover page, if any.
1005 if (isthereCvrPg) {
1006     if (typeof bCVMsg == "undefined") {
1007         console.println("Inserting cover page from "
1008             + _workingFolder+instrPath+"/@jobname-cvrpg.pdf");
1009         var bCVMsg=true;
1010     }
1011     if (cnt==0) var _cvrPath=_workingFolder+instrPath;
1012     aebTrustedFunctions(oDoc,aebInsertPages,
1013         {nPage: -1,
1014         cPath: _cvrPath+"/@jobname-cvrpg.pdf"});
1015 }
1016 var filesuffix=@setfilesuffix;
1017 // Now save this as a copy
```

Instructor's folder: Save a copy to the instructor's folder

```
1018 if(bUseClass) {
1019     _workingFolder=(@InstrPathFull)?":workingFolder;
1020     aebDocSaveAs.msg="Cannot access the local folder "
1021         + _workingFolder+instrPath+"/@jobname-"+fN+"_"+lN+".pdf";
1022     if(@distrToInstr) retn=aebTrustedFunctions(oDoc,aebDocSaveAs,
1023         {cFS:thInstrFS,
1024         cPath: _workingFolder+instrPath+"/@jobname"+filesuffix,
1025         bCopy:true});
```

Student's folder: Save a copy to the student's folder

```
1026     var _workingFolderC=(@ClassPathFull)?":workingFolder;
1027     if(isAbsPth)
1028         var cPath=folder+"@jobname"+filesuffix;
1029     else
1030         var cPath=_workingFolderC+classPath+"/"+folder
1031             +"@jobname"+filesuffix
1032     aebDocSaveAs.msg="Cannot access the path "+ cPath;
1033     if(@distrToStudents) retn=aebTrustedFunctions(oDoc,aebDocSaveAs,
1034         {cFS:thClassFS,
1035         cPath: cPath,
1036         bCopy:true});
1037     oDoc.dirty=false;
1038     oDoc.closeDoc(true);
1039 }
1040 cnt=++cnt % nQz
1041 }
1042 }
1043 this.resetForm(["Name"]);
1044 console.println("automatically saving this file...");
```

Remove the following line

```
// this.oRecordOfQuizData=undefined;
```

Save the source document: Finally, save any changes that have occurred source document. Before saving, we make some adjustments in the case this is the basic method case.

```

1045 var toSa=app.setTimeout("aebTrustedFunctions(this,aebSaveAs);\
1046 app.clearTimeout(toSa);",50);
1047 \end{defineJS}
1048 \let\sadMultQuizzes\sadQuizzes
1049 \let\rasSolns\sadQuizzes

1050 </package>
1051 <*container>

```

12 Batch support files

These files are designed for a workflow wherein the `usebatch` option is taken. This option, though largely symbolic, declares the instructor is going to use a batch sequence to process the students' quizzes, after they have taken the quiz and returned to the instructor. A custom batch file *Thor's way* was written for this purpose.

Thor's way. This batch sequence is run after the instructor has has a final look at each quiz (additional markup needed in the case of essay questions and assigning a final mark. the batch sequence performs a number of actions, for each quiz file selected, it acquires minimal quiz data (first name, last name, number of points awarded, total points, and final mark.

12.1 The container file for Thor's way

This file should be brought into Acrobat, its fields filled, and the button pushed. It sets global JavaScript variables `global.qzName` and `global.gradedPath`.

```

1052 \documentclass{article}
1053 \usepackage[designi]{web}
1054 \usepackage{eforms}[2020/12/14]
1055 \hypersetup{pdfpagemode=UseAttachments}
1056 %% \previewOn\pmpvOn
1057 \parindent0pt \parskip6pt
1058 \begin{defineJS}{\pbContainer}
1059 var f=this.getField("qzName");
1060 if(typeof global.RcrdData=="undefined")
1061   global.RcrdData=1;

```

If the 'Record class data' box is checked we create a new attachment to receive the data; otherwise, no new attachment file is created.

```

1062 if(global.RcrdData) {
1063   global.qzName=f.value;
1064   if (global.qzName=="") {
1065     f.value="qzData";
1066     global.qzName="qzData.txt";

```

```

1067 } else global.qzName=f.value+".txt";
1068 var d=this.dataObjects;
1069 if (d!=null) {
1070     for(var i=0; i< d.length; i++)
1071         this.removeDataObject(d[i].name);
1072 }
1073 this.createDataObject({
1074     cName: global.qzName,
1075     cValue: "First\\tSecond\\tPoints\\tTotal\\tGrade"
1076 });
1077 }
1078 var _path=this.path;
1079 var pos=_path.lastIndexOf("/");
1080 global.containerPath=_path.substring(0,pos+1);
1081 var f=this.getField("gradedPath");
1082 var v=f.value;
1083 var pos=v.indexOf(":");
1084 if(pos!=-1||v[0]=="/") global.gradedPath=v;
1085 else global.gradedPath=global.containerPath+v;
1086 aebTrustedFunctions(this,aebSaveAs);
1087 this.closeDoc(true);
1088 \end{defineJS}

```

(2019/11/21) Put JS for Clear Btn in defineJS environment

```

1089 \begin{defineJS}{\clrContainer}
1090 this.resetForm(["qzName","gradedPath"]);
1091 try{

```

(2019/11/21) Fixed a bug with exception thrown

```

1092 if (typeof global.qzName!="undefined")
1093     delete global.qzName; global.qzName="";
1094 if (typeof global.gradedPath!="undefined")
1095     delete global.gradedPath; global.gradedPath="";
1096 if (typeof global.appndSolns!="undefined")
1097     delete global.appndSolns; global.appndSolns=true;
1098 if (typeof global.RcrdDat!="undefined")
1099     delete global.RcrdDat; global.RcrdDat=true;
1100 } catch(e){}
1101 \end{defineJS}
1102 \begin{document}
1103 This file contains the quiz data as an attachment. Before you
1104 start the batch action \textsf{Thor's way}, build and
1105 \emph{place this file in the class folder of the instructor}.
1106 \begin{center}
1107 \begin{tabular}{rl}
1108 \pushButton[\TU{Fill in the two fields then push this button
1109 before starting the batch sequence}]{CA{Push}}\AAmouseup{\pbContainer}
1110 ]{\pbContainer}{}{13bp}&%
1111 \parbox[c]{1.5in}{\textField[\TU{Enter base name of the file that
1112 stores quiz results}]{qzName}{1.5in}{13bp}\vcgBdry[3bp]
1113 \textField[\TU{The path to the folder that will hold the graded

```

```

1114 quizzes, it may be a relative or an absolute path}
1115 ]{gradedPath}{1.5in}{13bp}}\cgBdry[\columnsep]\makebox[0pt][l]
1116 {\pushButton[\CA{Clear}
1117 \Amouseup{\clrContainer}]{clear}{}{13bp}}\[\[12pt]
1118 \checkBox[\V{Yes}\DV{Yes}\Amouseup{%
1119     global.appendSolns=(event.target.isBoxChecked());
1120 }]{AppdSolns}{11bp}{11bp}{Yes}&%
1121 \rlap{Append solutions, if they exist}\[\[6pt]
1122 \checkBox[\V{Yes}\DV{Yes}\Amouseup{%
1123     global.RcrdData=(event.target.isBoxChecked());
1124 }]{RecordData}{11bp}{11bp}{Yes}&%
1125 \rlap{Record class data}
1126 \end{tabular}
1127 \end{center}
1128 Fill in the base name of the file in the text field above. After
1129 you push the button, the file is saved, then start
1130 \textsf{Thor's way} action. After the batch sequence finishes,
1131 this file is opened again. Open the attachments panel and save
1132 the attached file. The file just saved is a tab delimited text
1133 file that can be opened in Microsoft Excel.
1134 \end{document}
1135 </container>
1136 <*bterminate>

```

12.2 The batch termination file

When running Thor's way, it is important that `terminate-batch.pdf` is the last file processed by the batch. The batch test each file initially, if the current file being process, the batch exits 'gracefully', otherwise, the batch processes the current file. The role `terminate-batch.pdf` plays it that it is detected by the batch, it performs no actions.

```

1137 \documentclass{article}
1138 \usepackage[designi]{web}
1139 \parindent0pt\parskip6pt
1140 \begin{document}
1141 \null\vfil
1142 \begin{center}
1143 \fbox{\begin{minipage}{.67\linewidth}
1144 This file is the last to be processed by \textsf{Thor's way},
1145 the batch action identifies it and gracefully terminates.
1146 \end{minipage}}
1147 \end{center}
1148 \vfil
1149 \end{document}
1150 </bterminate>
1151 <*package>

```


13 Configuration files

13.1 Class configuration

It was suggested by Loki that we should be able to have a configuration file for the class. In the case an instructor has several classes, this is a great convenience.

`\InputClassData{base-name}` Input the class file with name `<base-name>.cfg`. The class configuration file (`<base-name>.cfg`) consists of a series of declarations of the form,

```
\instrPath{<path>}
\classPath{<path>}
\classMember*{<first-name>}{<last-name>}{<folder>}
...
```

where the `*` is optionally included.

```
1152 \def\InputClassData#1{\def\InputClassData{#1}\mkClassData
1153 \InputIfFileExists{#1.cfg}
1154   {\PackageInfo{thorhammer}{Inputting class file #1.cfg}}
1155   {\PackageWarning{thorhammer}
1156     {Cannot find the class file #1.cfg}}}
```

`\InputFormattedClass[<cmd>]{base-name}` We define a more general input method. The format for the class configuration file is as follows:

```
\instrPath{<path>}
\classPath{<path>}
\bClassData
<entry1>
<entry2>
...
```

Prior to `\bClassData`, the entries are passed through, so you can, for example, set the `\instrPath` and `\classPath` here. After `\bClassData` comes a series of lines, one each student. Each line (`<entry>`) should be marked up so it can be parsed and information extracted; at the minimum, each line should contain `<first-name>`, `<last-name>`, and `<folder>`, with possibly more. The default for `<cmd>` is `\classMember`. For each line following `\bClassData`, `<cmd>` is placed in front of each entry like so, `<cmd><entry>`.

The first optional argument is the command (`<cmd>`) that parses each line. The end purpose of the command is to construct a data structure,

```
\classMember*{<first-name>}{<last-name>}{<folder>}
```

A simple example is the following (filename is `myclassroll.cfg`):

```
\instrPath{myClass}
\classPath{/z/Users/thor/myClass}
\bClassData
{Peter}{Pan}{A}
*{J\"{u}rgen}{Loki}{B}
{Thors}{Hammer}{C}
```

We can then input this CFG file with `\InputFormattedClass{myclassroll}`. For each line after `\bClassData`, the default (`\cmd`) is placed in front; for example, the first two lines of class data become `\classMember{Peter}{Pan}{A}` and `\classMember*{J}{u}{rgen}{Loki}{B}`, and so on.

```

1157 \newcommand\InputFormattedClass[2][\classMember]{\ClassEntriesttrue
1158 \begingroup
1159 \mkClFlsSpcls
1160 \endlinechar=-1
1161 \let\procThisLine\relax
1162 \let\bClassData\relax
1163 \let\re@dOK\d1@YES
1164 \newread\fmtclass
1165 \immediate\openin\fmtclass=#2.cfg
1166 \loop
1167 \read\fmtclass to \classmember
1168 \ifeof\fmtclass\let\re@dOK\d1@NO
1169 \else
1170 \expandafter
1171 \ifx\classmember\endinput\let\re@dOK\d1@NO
1172 \else
1173 \ifx\classmember\@empty %\let\procThisLine\relax
1174 \else
1175 \expandafter\procThisLine\classmember
1176 \expandafter\ifx\classmember\bClassData

```

After we process the current line, if this line is `\bClassData`, we switch over to #1 as the definition of `\procThisLine`.

```

1177 \def\procThisLine{#1}\fi
1178 \fi
1179 \fi
1180 \fi
1181 \ifx\re@dOK\d1@YES\repeat
1182 \immediate\closein\fmtclass
1183 \endgroup
1184 }

```

13.2 Load the configuration file

```

1185 \def\th@InputCFG{\InputIfFileExists{thorshammer.cfg}
1186 {\PackageInfo{thorshammer}
1187 {Inputting the configuration file}}
1188 {\PackageInfo{thorshammer}
1189 {No configuration file found}}}
1190 \ifx\th@loadCFG\d1@YES\expandafter\th@InputCFG\fi

1191 \catcode'\=\th@dquoteCat
1192 </package>

```

14 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.

Symbols	
<code>\%</code>	207, 334, 376, 566, 621
<code>\@RespBoxEssayActions</code>	511
<code>\@eqAAmouseup</code>	652
<code>\@eqquestiondepth</code>	707, 710, 713
<code>\@evenhead</code>	89
<code>\@folder</code>	135, 140–142
<code>\@initQuiz</code>	547, 548
<code>\@makeoother</code>	134
<code>\@oddhead</code>	88, 89
<code>\@onlypreamble</code>	111
<code>\@rgi</code>	137, 774
<code>_</code>	134
<code>_</code>	418, 709, 874, 1045
A	
<code>\A</code>	593, 687
<code>\AA</code>	181, 201, 210, 228, 278, 512, 656
<code>\AAcalculate</code>	170, 286, 297
<code>\AAformat</code>	190, 277, 285, 296, 320
<code>\AAkeystroke</code>	513
<code>\AAkeystroke</code>	180, 237, 318
<code>\AAmouseup</code>	200, 209, 218, 653, 1109, 1117, 1118, 1122
<code>\AAonfocus</code>	319
<code>\AbsPth</code>	751, 752
<code>\addcontentsline</code>	699
<code>\addtocounter</code>	324
<code>\addToDocOpen</code>	55, 57
<code>\aeb@creditmarkup</code>	669
<code>\aebTitleQuiz</code>	706, 709, 714
<code>\AtBeginDocument</code>	106
<code>\AtEndOfPackage</code>	22, 230
<code>\autoCenter</code>	672
<code>\autoCopy</code>	771, 772
<code>\autoCopyOff</code>	30, 772
<code>\autoCopyOn</code>	30, 771, 773
<code>\autoSaveStuJS</code>	24, 621, 654
B	
<code>\baselineskip</code>	302, 304
<code>\basicmethodsfalse</code>	16, 114, 133
<code>\basicmethodstrue</code>	12
<code>batchdistr (option)</code>	3
<code>\BC</code>	158, 164, 166, 190, 233, 236, 671, 695
<code>\bClassData</code>	1162, 1176
<code>\bDistrQuizzes</code>	144, 146
<code>\bEnumQuizzes</code>	121, 122
<code>\bFlattenState</code>	420, 421
<code>\BG</code>	164, 166, 190, 233, 236
<code>\bParams</code>	179, 199, 317
<code>\bqlabelISO</code>	524
<code>\bUseClass</code>	14, 16, 114, 133
<code>\bWebCustomize</code>	728, 731, 738
C	
<code>\CA</code>	200, 208, 217, 651, 1109, 1116
<code>\cFS</code>	30, 774
<code>\cFSth</code>	775, 776
<code>\cgBdry</code>	1115
<code>\cheader</code>	93, 102, 813, 826
<code>\checkbox</code>	1118, 1122
<code>\classEntries</code>	743, 755, 756, 758, 760, 761, 763, 764, 766, 768, 769, 834
<code>\classEntriesDef</code>	741, 834
<code>\ClassEntriesfalse</code>	742
<code>\ClassEntriestrue</code>	115, 134, 744, 1157
<code>\classMember</code>	29, 119, 142, 744, 1157
<code>\classmember</code>	1167, 1171, 1173, 1175, 1176
<code>\classMember*</code>	29
<code>\classMember@i</code>	745–747
<code>\classMember@ii</code>	748–750
<code>\ClassPath</code>	72, 73, 116
<code>\classPath</code>	5, 69
<code>\classPath@i</code>	70–72
<code>\ClassPathFull</code>	70, 71, 74, 115
<code>\classPathIsCHTTP</code>	5, 67
<code>\clrContainer</code>	1089, 1117
<code>\cmd</code>	179, 199, 207, 317, 652
<code>\cNoNumEnteredMsg</code>	370
<code>\color</code>	705
<code>\columnsep</code>	1115
<code>\CommentStream</code>	40, 788, 789, 794
<code>\commonPassKey</code>	200, 334
<code>\completeMsgFld</code>	13, 270, 304
<code>\completeMsgFld@V</code>	267, 271, 272
<code>\completeMsgFldV</code>	13, 267, 268
<code>\CorrButton</code>	306
<code>\csarg</code>	41, 666, 667, 787, 788, 793

<code>\cTooMuchCredit</code>	373	<code>\eParams</code>	179, 199, 317
<code>\curr@quiz</code>	522	<code>\eq@@EndQuizButtonActions</code>	593–595
<code>\currQuiz</code>	163, 165, 177–	<code>\eq@@EndQuizButtonActionsThorSave</code>	594, 596
179, 188, 189, 199, 263, 264, 288, 289, 298,		<code>\eq@begintab</code>	676
301, 306, 308–310, 323, 520, 666, 667, 674, 695		<code>\eq@EndQzBtnScriptThor</code>	22, 566, 593
D			
<code>\dclrFncyQzHdrsFmt</code>	726	<code>\eq@IWAuxOut</code>	665
<code>\DeclareCoverPage</code>	6, 108, 111	<code>\eq@noformstrue</code>	818
<code>\DeclareOptionX</code>	5, 8, 9, 11, 15, 19, 21	<code>\eq@normallheader</code>	96, 806
<code>\DeclareQuiz</code>	25, 658, 659	<code>\eq@prior@endQuiz</code>	662
<code>\declareQuizBody</code>	30, 783	<code>\eq@ssqlsecrunhead</code>	817
<code>\DeclareQuizSAVE</code>	658, 660	<code>\eq@ssqlsectitle</code>	703
<code>\DefaultHeightOfWidget</code>	289, 301	<code>\eq@YES</code>	701
<code>\define@choicetype</code>	45	<code>\eqbs</code>	830
<code>\defjsLB</code>	207	<code>\eqObjAlert</code>	523
<code>\dfnJSCR</code>	838	<code>\eqOutOf</code>	292
<code>\displaySumryTbl</code>	264	<code>\eqptLabel</code>	321
<code>\distrQuizzes</code>	7, 123, 127	<code>\eqPTs</code>	515
<code>\distrToInstr</code>	779, 780	<code>\eqptsLabel</code>	321, 673
<code>\distrToInstrOff</code>	30, 780	<code>\eqQT</code>	516
<code>\distrToInstrOn</code>	779	<code>\eqQuizPointsMsg</code>	26, 668
<code>\distrToStudents</code>	777, 778	<code>\eqQuizType</code>	512
<code>\distrToStudentsOff</code>	22, 23, 30, 778	<code>\eqSP</code>	42, 299, 300
<code>\distrToStudentsOn</code>	30, 777	<code>\EsH</code>	15, 313, 323, 670
<code>\dl@fitpage</code>	54	<code>\essayitem</code>	16, 326
<code>\dl@NO</code>	5, 27, 1168, 1171	<code>\essayQ</code>	15, 314, 326
<code>\dl@YES</code>	6, 32, 39, 1163, 1181, 1190	<code>\essayQFldTU</code>	311, 322
<code>\dlJSStr</code>	417, 615	<code>\essayQFldTU</code>	15, 311, 312
<code>docassembly (environment)</code>	8	<code>\essayQKey</code>	17, 318, 376
<code>\documentclass</code>	1052, 1137	<code>\EsW</code>	15, 313, 323, 670
<code>\doNotShirtSonsHdrs</code>	694	<code>\eWebCustomize</code>	729, 732, 739
<code>\DV</code>	272, 673, 1118, 1122	<code>\execJS</code>	154, 156, 832
E			
<code>\ef@NO</code>	601, 652, 746, 749, 753	<code>\execjs</code>	39
<code>\ef@YES</code>	597, 598, 745, 748, 751, 754, 767	<code>\ExecuteOptions</code>	733, 734, 737
<code>\eflength</code>	691, 692	<code>\ExecuteOptions@SAVE</code>	733, 737
<code>\egroup</code>	120, 144, 274, 281, 291, 675, 787	<code>\ExecuteOptionsX</code>	20, 21
<code>\emph</code>	1105	<code>\executeSave</code>	148
<code>\endexecJS</code>	154, 156, 832	F	
<code>\endinput</code>	728, 732, 738, 1171	<code>\F</code>	199, 201, 208, 210, 217, 228, 233,
<code>\endlinechar</code>	1160	235, 270, 278, 279, 284, 295, 317, 650, 656, 671	
<code>\EndQzWarning@Msg</code>	560	<code>\fancyQuizHeaders</code>	688
<code>\EndQzWarningMsg</code>	22, 560, 561	<code>\fbox</code>	1143
<code>\endverbatimwrite</code>	793	<code>\fboxsep</code>	302, 316
<code>\enumQuizzes</code>	7, 114, 143	<code>\Ff</code>	158, 164, 166, 180, 181, 233, 270, 271, 671
environments:		<code>\FfMultiline</code>	270
<code>docassembly</code>	8	<code>\FfPassword</code>	180
<code>makeClassFiles</code>	32	<code>\FfReadOnly</code>	158, 164, 166, 233, 271, 671
		<code>\FHidden</code>	199, 208,
		217, 233, 235, 270, 278, 284, 295, 317, 650, 671	
		<code>\FirstName</code>	9, 159, 252

<code>\flattenOff</code>	18, 421, 422	<code>\instrAutoCloseOn</code>	18, 414, 415
<code>\flattenOn</code>	18, 420, 422	<code>\instrAutoSave</code>	411, 412, 447
<code>\flJSStr</code>	274, 281, 291, 370, 373, 531, 560, 564	<code>\instrAutoSaveOff</code>	18, 412
<code>\fmtclass</code>	1164, 1165, 1167, 1168, 1182	<code>\instrAutoSaveOn</code>	18, 411, 413
<code>\FncyHdrsFmtNoTitleQuiz</code>	690, 708, 714	<code>\InstrPath</code>	63, 65, 66, 117
<code>\FncyHdrsFmtQuestion</code>	711	<code>\instrPath</code>	5, 60
<code>\fncyQHdrsColor</code>	705	<code>\instrPath@i</code>	61–63
<code>\freezeOrSave</code>	12, 230, 231, 258	<code>\InstrPathFull</code>	61, 62, 64, 115
<code>\freezeQuiz</code>	11, 207, 230	<code>\instrPathIsCHTTP</code>	5, 58
<code>\freezeQuizFld@CA</code>	205, 208	<code>\instrSave</code>	11, 216, 231
<code>\freezeQuizFld@TU</code>	203, 209	<code>\instrSaveFld@CA</code>	214, 217
<code>\freezeQuizFld@CA</code>	11, 205, 206	<code>\instrSaveFld@TU</code>	212, 218
<code>\freezeQuizFld@TU</code>	11, 203, 204	<code>\instrSaveFldCA</code>	11, 214, 215
<code>\freezeQuizMU</code>	482	<code>\instrSaveFldTU</code>	11, 212, 213
<code>freezeQuizMU()</code> (JS function)	18	<code>\isQZ</code>	512
<code>\FullName</code>	9, 169	<code>\ISSTAR</code>	138, 142
		<code>\item</code>	326
G			
<code>\g@addto@macro</code>		J	
. 755, 756, 758, 760, 761, 763, 764, 766, 768, 769		<code>\JS</code>	55, 593, 687
<code>\GoToD</code>	57	JS functions:	
		<code>freezeQuizMU()</code>	18
H			
<code>\H</code>	190	K	
<code>\Hy@EveryPageAnchor</code>	247, 800	<code>\kern</code>	254, 262, 303, 680
<code>\hypersetup</code>	1055	L	
I			
<code>\ifbasicmethods</code>	12, 157, 422, 835, 837	<code>\Large</code>	678
<code>\IfbQzChkSnippet</code>	536, 548	<code>\LastName</code>	9, 161, 253
<code>\ifClassEntries</code>	742, 833	<code>\lheader</code>	92, 101, 812, 825
<code>\ifeof</code>	1168	<code>\linktxtcolor</code>	686
<code>\ifth@allowfreeze</code>	18, 230	<code>\llap</code>	716, 718, 721
<code>\ifthCoverPage</code>	107	<code>\LngPtsFld</code>	14, 283, 309
<code>\ifthordinary</code>	10, 25, 820	<code>\LngPtsFld@Fmt</code>	281, 285
<code>\ifthtestmode</code>	7, 306, 554	<code>\LngPtsFldFmt</code>	14, 280, 282
<code>\ifuseclassOpt</code>	13, 124, 836	<code>\LngPtsFldFmt@i</code>	280, 281
<code>\ifUseStuSaveAsDialog</code>	618	<code>\loop</code>	1166
<code>\includequizsolutions</code>	815	M	
<code>\InitQzMsg</code>	531, 532	<code>\m@ne</code>	120
<code>\InitQzMsg@Msg</code>	531, 549	<code>\makeatletter</code>	735
<code>\inputAltAdbFnCS</code>	28, 34, 36	<code>\makeatother</code>	736
<code>\InputClassData</code>	1152	<code>makeClassFiles</code> (environment)	32
<code>\InputClassData</code>	41, 1152	<code>\makecmt</code>	334, 376, 566, 621
<code>\InputFormattedClass</code>	41, 1157	<code>\makeesc</code>	327, 334, 376, 482, 566, 621, 838
<code>\InputIfFileExists</code>	28, 736, 804, 1153, 1185	<code>\marginparsep</code>	256, 259
<code>\InputQuizBody</code>	31, 798	<code>\markQz</code>	10, 187, 255
<code>\inputWebCfg</code>	28, 730	<code>\markQzFld@CA</code>	183, 200
<code>\instrAutoClose</code>	414, 416, 479	<code>\markQzFld@TU</code>	185, 201
<code>\instrAutoCloseOff</code>	18, 416	<code>\markQzFldCA</code>	10, 183, 184

<code>\markQzFldTU</code>	10, 185, 186	<code>\providecommand</code>	42, 730
<code>\markupHeight</code>	670, 675	<code>\PtFW</code>	289, 301
<code>\markupTextSize</code>	672	<code>\PTs</code>	326
<code>\markupWidth</code>	670, 675	<code>\pushButton</code>	199, 207, 216, 650, 1108, 1116
<code>\MarkWarning@Msg</code>	417, 434	<code>\pwdInstrFld</code>	10, 16, 176
<code>\MarkWarningMsg</code>	417, 418	<code>\pwdInstrFld@TU</code>	174, 181
<code>\mkClFlsSpcls</code>	154, 156, 830, 832, 1152, 1159	<code>\pwdInstrFldTU</code>	10, 174, 175
<code>\myFQHFmt</code>	27, 704, 726	<code>\pwdKeyJS</code>	180, 327

N

<code>\n</code>	533, 563
<code>\newwrite</code>	40
<code>nocfg</code> (option)	3
<code>\NoNumEnteredMsg</code>	17, 370, 371
<code>\nQs</code>	177–179, 188, 189, 199
<code>\null</code>	1141
<code>\numberline</code>	702

O

<code>\obeyspaces</code>	273, 280, 290
<code>\oct</code>	29, 32, 830
<code>\oField</code>	552, 553
<code>\openin</code>	1165
<code>\openout</code>	788, 819
options:	
<code>batchdistr</code>	3
<code>nocfg</code>	3
<code>ordinary</code>	3
<code>testmode</code>	3
<code>usebatch</code>	3
<code>useclass</code>	3
ordinary (option)	3

P

<code>\PackageInfo</code>	29, 1154, 1186, 1188
<code>\PackageWarning</code>	30, 49, 125, 1155
<code>\Page</code>	57
<code>\parindent</code>	1057, 1139
<code>\parskip</code>	1057, 1139
<code>\pbContainer</code>	1058, 1109
<code>\pcMarkupColor</code>	672
<code>\pdfstringdef</code>	759, 762, 765
<code>\pmpvOn</code>	1056
<code>\PointsField</code>	277
<code>\PointsFieldDefaults</code>	284, 295
<code>\postSubmitQuiz</code>	22, 551, 559
<code>\presets</code>	160, 162, 170, 284, 295
<code>\previewOn</code>	1056
<code>\ProcessOptionsX</code>	24
<code>\procThisLine</code>	1161, 1173, 1175, 1177

Q

<code>\Q</code>	236, 317
<code>\qMark@Hook</code>	314, 315, 325
<code>\qMark@HookSave</code>	314, 325
<code>\quiz@solns</code>	805, 819
<code>\quizSolnsHeadnToc</code>	696
<code>\qzLtr</code>	797
<code>\QzVer</code>	30, 782, 791

R

<code>\r</code>	171, 172, 191–197, 219–222, 224–227, 286, 298, 299, 552, 553, 555–558, 599, 606, 653, 869, 874
<code>\raggedleft</code>	255, 258
<code>\raisebox</code>	302, 316
<code>\rasSolns</code>	8, 1049
<code>\re@dOK</code>	1163, 1168, 1171, 1181
<code>\read</code>	1167
<code>\repeat</code>	1181
<code>\RequirePackage</code>	2, 26, 37, 38
<code>\ReturnTo</code>	727
<code>\rheader</code>	94
<code>\rhPgNumsOnly</code>	6, 105
<code>\rlap</code>	164, 1121, 1125
<code>\rmSTAR</code>	135, 138
<code>\roman</code>	723

S

<code>\s</code>	221, 432
<code>\sadMultQuizzes</code>	1048
<code>\sadQuizzes</code>	32, 838, 1048, 1049
<code>\SecondSave@Msg</code>	424, 615
<code>\SecondSaveMsg</code>	24, 615, 616
<code>\section</code>	696
<code>\setArrayLength</code>	835
<code>\setClassArray</code>	833
<code>\setcounter</code>	43
<code>\setfilesuffix</code>	836
<code>\setInitMag</code>	4, 44
<code>\setLink</code>	686
<code>\setsolnspace</code>	689
<code>\ShrtPtsFld</code>	14, 276, 308

<code>\ShrtPtsFld@Fmt</code>	274, 278	<code>\th@bMrkQz</code>	159, 163
<code>\ShrtPtsFldFmt</code>	14, 273, 275	<code>\th@DeclareQuiz</code>	659, 660, 802
<code>\ShrtPtsFldFmt@i</code>	273, 274	<code>\th@distrQuizzes</code>	130, 133
<code>\smash</code>	315	<code>\th@distrQuizzes@i</code>	134, 139
<code>\sqlsecurunhead</code>	803, 817	<code>\th@dquoteCat</code>	3, 1191
<code>\sqlsectitle</code>	698, 702	<code>\th@exstar</code>	748, 749, 751, 754, 767
<code>\st@scndclmn</code>	679, 680	<code>\th@fullnameFmt</code>	167, 173
<code>\st@scndclmnSAVE</code>	679, 680	<code>\th@fullnamePresets</code>	166, 170
<code>\st@thrdclmn</code>	686	<code>\th@HeaderOffset</code>	78, 693, 694, 697
<code>\stepcounter</code>	316, 800	<code>\th@initmag</code>	48, 54, 57
<code>\sthline</code>	677	<code>\th@InputCFG</code>	1185, 1190
<code>\stmarkupbox</code>	684	<code>\th@leftShiftHdr</code>	692, 693
<code>\stmarkupHeight</code>	682	<code>\th@loadCFG</code>	5, 6, 1190
<code>\stmarkupTextSize</code>	683	<code>\th@namePresets</code>	157, 158, 160, 162
<code>\stmarkupWidth</code>	681	<code>\th@next</code>	125, 130, 131
<code>\strut</code>	678	<code>\th@qbCnt</code>	786, 787
<code>\stuASOn</code>	597, 598, 601, 652	<code>\th@QHFIRSTName</code>	239, 252
<code>\stuAutoClose</code>	607, 610	<code>\th@QHGrade</code>	245, 260
<code>\stuAutoCloseOff</code>	609	<code>\th@QHLastName</code>	241, 253
<code>\stuAutoCloseOn</code>	23, 605, 608	<code>\th@QHPoints</code>	243, 257
<code>\stuAutoCloseScript</code>	605, 609, 655	<code>\th@QzHeaderC</code>	81, 88, 93, 99, 810, 823
<code>\stuAutoSave</code>	558, 600, 603, 653	<code>\th@QzHeaderCQ</code>	79, 81, 823, 826
<code>\stuAutoSaveOff</code>	23, 601	<code>\th@QzHeaderCS</code>	82, 99, 102, 810, 813
<code>\stuAutoSaveOn</code>	23, 598, 604	<code>\th@QzHeaderL</code>	76, 88, 92, 98, 809, 822
<code>\stuAutoSaveScript</code>	599, 602, 655	<code>\th@QzHeaderLQ</code>	75, 76, 78, 822, 825
<code>\studentGrade</code>	12, 235, 260	<code>\th@QzHeaderLS</code>	78, 98, 101, 805, 809, 812
<code>\studentReport</code>	12, 232, 257	<code>\th@QzVer</code>	781, 784
<code>\stuSaveBtn</code>	24, 307, 650	<code>\th@setHeaders</code>	87, 91, 106
<code>\stuSaveBtn@CA</code>	611, 651	<code>\th@star</code>	745, 746, 753
<code>\stuSaveBtn@TU</code>	613, 651	<code>\thClassFS</code>	67, 68
<code>\stuSaveBtnCA</code>	24, 611, 612	<code>\thCoverPagefalse</code>	107
<code>\stuSaveBtnTU</code>	24, 613, 614	<code>\thCoverPagetrue</code>	108
<code>\sumrytblLinkHook</code>	685, 687	<code>\thCvrPg</code>	109, 110, 426
<code>\sumryTblP</code>	677	<code>\theeqpointvalue</code>	300, 667
<code>\sumryTblQ</code>	677	<code>\thEnumQuizzes</code>	121, 122
<code>\sumryTblR</code>	677	<code>\thepage</code>	85
		<code>\thequestionno</code>	323, 515, 516, 518, 519, 666, 674
		<code>\therearequizzesolutionsfalse</code>	816
		<code>\theth@QzCnt</code>	796, 797, 801
<code>\t</code>	193–196,	<code>\thfullnameFmt</code>	9, 167, 168
	222, 225, 226, 551–553, 555–558, 599, 605, 874	<code>\thInstrFS</code>	58, 59
<code>\t@hQzHeaderR</code>	84, 89, 94	<code>\thIsCP</code>	109, 110, 425
<code>testmode (option)</code>	3	<code>\ththisQuiz</code>	799
<code>\textbf</code>	239, 241, 243, 245, 691	<code>\ththisQuizOrig</code>	799, 801
<code>\textColor</code>	236, 672	<code>\thordinaryfalse</code>	10
<code>\textField</code>	159, 161, 164, 169, 179, 190,	<code>\thordinarytrue</code>	11
	233, 235, 270, 284, 295, 316, 671, 695, 1111, 1113	<code>\thOrdQz</code>	25, 538
<code>\textsf</code>	1104, 1130, 1144	<code>\ThorsAlert@Title</code>	549, 564
<code>\textSize</code>	236, 672	<code>\ThorsAlertTitle</code>	564, 565
<code>\th@allowfreezefalse</code>	19	<code>\thPageOne</code>	43, 248, 249, 798
<code>\th@allowfreezettrue</code>	18		

T

<code>\thQHFirstName</code>	12, 239, 240
<code>\thQHGrade</code>	12, 245, 246
<code>\thQHLastName</code>	12, 241, 242
<code>\thQHPoints</code>	12, 243, 244
<code>\thQuizHeader</code>	13, 247
<code>\thQuizHeaderLayout</code>	13, 248, 249, 251
<code>\thQuizName</code>	25, 80, 83, 659, 664
<code>\thQuizTrailer</code>	15, 302
<code>\thQzHeaderCQ</code>	6, 79, 80, 105
<code>\thQzHeaderCS</code>	6, 82, 83, 105
<code>\thQzHeaderL</code>	5, 75, 77, 105
<code>\thQzHeaderR</code>	6, 84, 85
<code>\thQzName</code>	26, 663, 664
<code>\thqzname</code>	663
<code>\thQzSolnMrkr</code>	27, 695, 698
<code>\thtestmodfalse</code>	7, 9
<code>\thtestmodetrue</code>	8, 11
<code>\thUseNameToCustomize</code>	112, 113, 451, 452
<code>\TooMuchCreditMsg</code>	17, 373, 374
<code>\TotalsFld</code>	14, 294, 310
<code>\TotalsFld@Fmt</code>	291, 296
<code>\TotalsFldFmt</code>	14, 290, 292
<code>\TotalsFldFmt@i</code>	290, 291
<code>\tstForSTAR</code>	136, 137, 141
<code>\tstForSTAR@i</code>	136
<code>\TU</code>	181, 201, 209, 218, 322, 651, 1108, 1111, 1113

U

<code>\u</code>	29, 32, 830, 838
-----------------	------------------

<code>usebatch</code> (option)	3
<code>useclass</code> (option)	3
<code>\useclassOptfalse</code>	13
<code>\useclassOpttrue</code>	15
<code>\usedAdbFuncs</code>	27, 32
<code>\useEndQuizThor</code>	23, 595
<code>\useNameToCustomize</code>	7, 112
<code>\usepackage</code>	1053, 1054, 1138
<code>\UseStuSaveAsDialogfalse</code>	618, 620
<code>\useStuSaveAsDialogOff</code>	24, 620
<code>\useStuSaveAsDialogOn</code>	24, 619
<code>\UseStuSaveAsDialogtrue</code>	619

V

<code>\V</code>	271, 673, 1118, 1122
<code>\vcgBdry</code>	252, 253, 257, 260, 308, 309, 1112
<code>\verbatim@out</code>	789, 790
<code>\verbatimwrite</code>	792
<code>\vfil</code>	1141, 1148

W

<code>\web@latextoc</code>	701
<code>\widthof</code>	257, 691
<code>\write</code>	790
<code>\wrtQzInfo</code>	662, 665

X

<code>\x</code>	142, 759, 760, 762, 763, 765, 766, 801, 802
-----------------	---

15 Change History

v1.1 (2019/06/30)	quiz	23
General: Added flattening		20, 21
v1.1.1 (2019/06/30)	v1.1.5 (2019/07/06)	
General: Corrected a bug in <code>\essayQKey</code> that	General: If document is dirty, do not save	23
caused a miscalculation.		17
v1.1.2 (2019/07/02)	v1.1.6 (2019/07/06)	
General: Added try/catch in <code>\AAformat</code> to avoid	General: Added end quiz warning	22
exceptions thrown in the case of distiller;	Added save as requiring acrobat	20, 21
added pdf space in the <code>\TotalsFld JS</code> to	Check on name fields	21
avoid unexpected wraps in the case of		
dvips/distiller workflow.	v1.1.8 (2019/07/08)	
v1.1.3 (2019/07/03)	General: make studentenReport initially hidden	12
General: Make <code>\LngPtsFld</code> a calculation field		14
Remove lines not needed since <code>\LngPtsFld</code>	v1.1.9 (2019/07/15)	
became a calculation field	General: Added optional argument	
v1.1.4 (2019/07/04)	<code>\mkClFlsSpcls</code>	32
General: Add a SaveAs menu item to end of the	v1.2 (2019/07/16)	
	General: Added <code>\freezeOrSave</code>	12
	Added batch support files	38
	v1.2.1 (2019/07/19)	
	General: Inserted test for <code>oRecordOfQuizData</code>	

before saving, this avoids the message that appears in the console that 'f is null'	20	v1.4.3 (2019/08/22)	General: let \Hy@EveryPageAnchor to \relax . . .	13
v1.3 (2019/07/20)		v1.4.5 (2019/08/25)	General: Added command to remove shift for solns headers	27
General: Added multiple quizzes in one source	30	v1.4.6 (2019/08/26)	General: Combined \sadMultQuizzes with \sadQuizzes	30
v1.3.1 (2019/07/20)			Renamed \sadMultQuizzes to \sadQuizzes, removed old \sadQuizzes.	32
General: container: Push now closes the container	38	v1.4.7 (2019/08/27)	General: Added format code from \qz@IDTxtField	16
v1.3.5 (2019/08/06)			Added marker field attached to name field	9
General: Use \usedAdbFuncs to detect usealtadobe option	4		Added version support for quiz bodies	31
v1.3.6 (2019/08/07)		v1.4.8 (2019/09/05)	General: Added \distrQuizzes	7
General: freezeQuizMU: Cannot freeze unless grade given and added suffix "-g" to document name on saving.	7		Added \enumQuizzes	7
v1.3.7 (2019/08/11)			Added \useNameToCustomize	7
General: fixed typo in loading cfg	42		Append solution pages if there is one	19
v1.4 (2019/08/11)		v1.5.1 (2019/10/23)	General: Allow web.cfg to be imported in the preamble.	28
General: Begin major change to this package, leaving v1.3.8 as our best working version prior to this update.	3		v1.5.10 (2021/05/31)	
Changes to \InputBodyQuiz to support solution sets	31		General: Added \qzLtr the public version of \theth@qzCnt	31
v1.4.1 (2019/08/16)			v1.5.11 (2021/06/24)	
General: Added cFS to aebDocSaveAs	21		General: Define \useEndQuizThor	23
Implement the concept of a cover page.	6		Require insdljs dated 2021/06/19 or later, which itself requires acrotex-js.	4
v1.4.10 (2019/09/11)			v1.5.2 (2019/11/21)	
General: Fixed a bug in calculation when there are no essay questions	14		General: Fixed a bug with exception thrown	39
v1.4.12 (2019/09/11)			Put JS for Clear Btn in defineJS	39
General: Correction a problem with \MarkWarningMsg in the dvips/distiller workflow	18		v1.5.4 (2019/11/30)	
v1.4.13 (2019/09/12)			General: Reworked \distrQuizzes to account for second star option of \classMember	7
General: Added \flattenOn and \flattenOff	18		v1.5.5 (2019/12/08)	
v1.4.14 (2019/09/13)			General: Use aebTrustedFunctions if present	24
General: Inserted \mkClFlsSpcls as optional argument of docassembly	8		v1.5.7 (2020/01/13)	
v1.4.17 (2019/10/03)			General: Added delete global.bOkClose	25
General: Defined _workingFolderC to avoid redefinition of _workingFolder	37		v1.5.8 (2020/01/21)	
Rewrote \LngPtsFldFmt to produce pdf spaces	14		General: Published password of "acrotex" for the two security related action sequences	3
Rewrote \ShrtPtsFldFmt to produce pdf spaces	14		v1.5.9 (2020/05/29)	
Rewrote \TotalsFldFmt to produce pdf spaces	14		General: Defined public \qzLtr version of \theth@qzCnt	31
Saved path to cover page	36			
v1.4.2 (2019/08/22)				
General: Insert cover page in \sadQuizzes	36			