ximera — Simultaneously write print and online interactive materials.*

Jim Fowler   Jeramiah Hocutt   Oscar Levin   Jason Nowell
Hans Parshall   Bart Snapp

Released 2018/10/28

Abstract

"Ximera begins where TEX ends." The ximera class aids in the creation of handouts, worksheets, exercises, and sections of textbooks to be used either individually or "glued" together via a xourse file. All ximera documents can be deployed in an online interactive form via `xake` See: Ximera Project and the source code on GitHub.

1 Introduction

2 ximera.cls

2.1 Options for the class

We start by listing the options for the ximera document class. Note, since the xourse class is based on the ximera class, all listed options are available there too.

```
1 ⟨∗classXimera⟩
```

handout The default behavior of the class is to display **all** content. This means that if any questions are asked, all answers are shown. Moreover, some content will only have a meaningful presentation when displayed online. When compiled without any options, this content will be shown too. This option will supress such content and generate a reasonable printiable "handout."

```
2 \newif\ifhandout
3 \handoutfalse
4 \DeclareOption{handout}{\handouttrue}
```

noauthor By default, authors are listed at the bottom of the first page of a document. This option will supress the listing of the authors.

```
5 \newif\ifnoauthor
6 \noauthorfalse
7 \DeclareOption{noauthor}{\noauthortrue}
```

nooutcomes By default, learning outcomes are listed at the bottom of the first page of a document. This option will supress the listing of the learning outcomes.

```
8 \newif\ifnooutcomes
9 \nooutcomesfalse
10 \DeclareOption{nooutcomes}{\nooutcomestrue}
```

instructornotes This option will turn on (and off) notes written for the instructor.

```
11 \newif\ifinstructornotes
12 \instructornotesfalse
13 \DeclareOption{instructornotes}{\instructornotestrue}
```

---

*This file describes version v1.0, last revised 2018/10/28.

noinstructornotes This option will turn off (and on) notes written for the instructor.

```
14 \DeclareOption{noinstructornotes}{\instructornotestrue}
```

hints When the `handout` options is used, hints are not shown. This option will make hints visible in handout mode.

```
15 \newif\ifhints
16 \hintsfalse
17 \DeclareOption{hints}{\hintstrue}
```

newpage This option will start each problem-like environment (`exercise`, `question`, `problem`, and `exploration`) start on a new page.

```
18 \newif\ifnewpage
19 \newpagefalse
20 \DeclareOption{newpage}{\newpagetrue}
```

numbers This option will number the titles of the activity. By default the activities are unnumbered.

```
21 \newif\ifnumbers
22 \numbersfalse
23 \DeclareOption{numbers}{\numberstrue}
```

wordchoicegiven This option will replace the choices shown by `wordChoice` with the correct choice. No indication of the `wordChoice` environment will be shown.

```
24 \newif\ifwordchoicegiven
25 \wordchoicegivenfalse
26 \DeclareOption{wordchoicegiven}{\wordchoicegiventrue}
27 \newif\iffirstinlinechoice% Support for other wordchoice command contents.
28 \firstinlinechoicetrue
29
30 \newif\ifxake
31 \xakefalse
32 \DeclareOption{xake}{\xaketrue}
33
34 \newif\iftikzexport
35 \tikzexportfalse
36 \DeclareOption{tikzexport}{%
37   \tikzexporttrue%
38   \handoutfalse%
39   \numbersfalse%
40   \newpagefalse%
41   \hintsfalse%
42   \nooutcomesfalse%
43 }
44
45 \DeclareOption*{%
46   \PassOptionsToClass{\CurrentOption}{article}%
47 }
48 \ProcessOptions\relax
49 \LoadClass{article}
50
51 \ifdefined\HCode
52   \xaketrue%
53   \tikzexporttrue%
54   \handoutfalse%
55   \numbersfalse%
56   \newpagefalse%
57   \hintsfalse%
58   \nooutcomesfalse%
59 \fi
60
61 ⟨/classXimera⟩
62 ⟨∗classXimera⟩
```

2.2 Loading packages

Since we want `\cancel` to work, we load it here to avoid polluting the `.jax` output.

```
63 \RequirePackage[makeroom]{cancel}
```

Quite a few packages are required by the document class. This is a list of required packages. As packages are added to this list, we should include a comment as to where they are being utilized. This will help keep this list from being redundant and/or outdated.

```
64 \RequirePackage[inline]{enumitem}
65 \RequirePackage[pagestyles]{titlesec}
66 \RequirePackage{titletoc}
67 \RequirePackage{titling}
68 \RequirePackage{url}
69 \RequirePackage[table]{xcolor}
70 \RequirePackage{tikz}
71 \RequirePackage{pgfplots}
72 \usepgfplotslibrary{groupplots}
73 \usetikzlibrary{calc}
74 \RequirePackage{fancyvrb}
```

Load `forloop` for the problem environment dynamic naming and building.

```
75 \RequirePackage{forloop}
```

Now we load even more packages.

```
76 \RequirePackage{environ}% Included to allow saving of environment contents. This does *not* 
77 \RequirePackage{amssymb}% Included to have access to math typeset.
78 \RequirePackage{amsmath}% Included to have access to math typeset.
79 \RequirePackage{amsthm}%  Included to have access to math typeset.
80 \RequirePackage{xifthen}% http://ctan.org/pkg/xifthen
81 \RequirePackage{multido}% http://ctan.org/pkg/multido
82 \RequirePackage{listings} %% is this required???
83
84 \RequirePackage{xkeyval}
85
86 \RequirePackage{comment}
87 ⟨/classXimera⟩
```

Various packages must be loaded early to avoid polluting the `.jax` file.

```
88 ⟨*classXimera⟩
89 \RequirePackage{gettitlestring}
90 \RequirePackage{nameref}
91 \RequirePackage{epstopdf}
92 ⟨/classXimera⟩
```

2.3 Page setup

We want non-indented spaced-out paragraphs.

```
93 ⟨*classXimera⟩
94 \setlength{\parindent}{0pt}
95 \setlength{\parskip}{5pt}
96 ⟨/classXimera⟩
```

To avoid weird margins in 2-sided mode, change the margins.

```
97 ⟨*classXimera⟩
98 \oddsidemargin 62pt
99 \evensidemargin 62pt
100 \textwidth 345pt
101 \headheight 14pt
102 ⟨/classXimera⟩
```

On the HTML side, there is more complicated page setup to perform.

```
103 ⟨*cfgXimera⟩
104 \Preamble{xhtml}
105
106 % We don't want to translate font suggestions with ugly wrappers like
```

```
107 % <span class="cmti-10"> for italic text
108 \NoFonts
109
110 % Don't output xml version tag
111 \Configure{VERSION}{}
112
113 % Output HTML5 doctype instead of the default for HTML4
114 \Configure{DOCTYPE}{\HCode{<!doctype html>\Hnewline}}
115
116 % Custom page opening
117 \Configure{HTML}{\HCode{<html lang="en">\Hnewline}}{\HCode{\Hnewline</html>}}
118
119 % Reset <head>, aka delete all default boilerplate; alternatively set up new content
120 \Configure{@HEAD}{\HCode{<meta name="generator" content="TeX4ht (http://www.cse.ohio-state.e
121 \Configure{@HEAD}{\HCode{<meta name="ximera" content="version 0.0.1" />\Hnewline}}
122 \Configure{@HEAD}{\HCode{<link href="https://ximera.osu.edu/public/stylesheets/standalone.css
123 \Configure{@HEAD}{\HCode{<script type="text/javascript" async src="https://ximera.osu.edu/pub
124 ⟨/cfgXimera⟩
```

Disable certain ligatures in HTML.

```
125 ⟨∗htXimera⟩
126 \usepackage{microtype}
127 \DisableLigatures[f]{encoding=*}
128 ⟨/htXimera⟩
```

I am not sure what this does.

```
129 ⟨∗htXimera⟩
130 \NewEnviron{html}{\HCode{\BODY}}
131 ⟨/htXimera⟩
```

2.4 Structure

2.4.1 Macros

Makes everymath display style even when inline, could be optional.

```
132 ⟨∗classXimera⟩
133 \everymath{\displaystyle}
134 ⟨/classXimera⟩
```

Ok not everything, we also need to configure "display style" limits.

```
135 ⟨∗classXimera⟩
136 \let\prelim\lim
137 \renewcommand{\lim}{\displaystyle\prelim}
138 ⟨/classXimera⟩
```

2.4.2 Theorem and theorem-like environments

On the web, a theorem is emitted as a special `<div>`.

```
139 ⟨∗htXimera⟩
140 \newcommand{\ConfigureTheoremEnv}[1]{%
141 \renewenvironment{#1}[1][]{\refstepcounter{problem}%
142 \ifthenelse{\equal{##1}{}}{}{%
143   \HCode{<span class="theorem-like-title">}##1\HCode{</span>}%
144 }}{}
145 \ConfigureEnv{#1}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div class="
146 }
147 ⟨/htXimera⟩
148 ⟨classXimera⟩\theoremstyle{definition} % No italic (because this makes also text in TikZ itali
```

The key is to make sure that the theorem environments are defined in a corresponding
fashion on the web and on paper.

theorem (*env.*)   Theorem

```
149 ⟨classXimera⟩        \newtheorem{theorem}{Theorem}
150 ⟨htXimera⟩          \ConfigureTheoremEnv{theorem}
```

algorithm (*env.*)	Algorithm	
	151 ⟨classXimera⟩	\newtheorem{algorithm}{Algorithm}
	152 ⟨htXimera⟩	\ConfigureTheoremEnv{algorithm}
axiom (*env.*)	Axiom	
	153 ⟨classXimera⟩	\newtheorem{axiom}{Axiom}
	154 ⟨htXimera⟩	\ConfigureTheoremEnv{axiom}
claim (*env.*)	Claim	
	155 ⟨classXimera⟩	\newtheorem{claim}{Claim}
	156 ⟨htXimera⟩	\ConfigureTheoremEnv{claim}
conclusion (*env.*)	Conclusion	
	157 ⟨classXimera⟩	\newtheorem{conclusion}{Conclusion}
	158 ⟨htXimera⟩	\ConfigureTheoremEnv{conclusion}
condition (*env.*)	Condition	
	159 ⟨classXimera⟩	\newtheorem{condition}{Condition}
	160 ⟨htXimera⟩	\ConfigureTheoremEnv{condition}
conjecture (*env.*)	Conjecture	
	161 ⟨classXimera⟩	\newtheorem{conjecture}{Conjecture}
	162 ⟨htXimera⟩	\ConfigureTheoremEnv{conjecture}
corollary (*env.*)	Corollary	
	163 ⟨classXimera⟩	\newtheorem{corollary}{Corollary}
	164 ⟨htXimera⟩	\ConfigureTheoremEnv{corollary}
criterion (*env.*)	Criterion	
	165 ⟨classXimera⟩	\newtheorem{criterion}{Criterion}
	166 ⟨htXimera⟩	\ConfigureTheoremEnv{criterion}
definition (*env.*)	Definition	
	167 ⟨classXimera⟩	\newtheorem{definition}{Definition}
	168 ⟨htXimera⟩	\ConfigureTheoremEnv{definition}
example (*env.*)	Example	
	169 ⟨classXimera⟩	\newtheorem{example}{Example}
	170 ⟨htXimera⟩	\ConfigureTheoremEnv{example}
explanation (*env.*)	Explanation	
	171 ⟨classXimera⟩	\newtheorem*{explanation}{Explanation}
	172 ⟨htXimera⟩	\ConfigureTheoremEnv{explanation}
fact (*env.*)	Fact	
	173 ⟨classXimera⟩	\newtheorem{fact}{Fact}
	174 ⟨htXimera⟩	\ConfigureTheoremEnv{fact}
lemma (*env.*)	Lemma	
	175 ⟨classXimera⟩	\newtheorem{lemma}{Lemma}
	176 ⟨htXimera⟩	\ConfigureTheoremEnv{lemma}
formula (*env.*)	Formula	
	177 ⟨classXimera⟩	\newtheorem{formula}{Formula}
	178 ⟨htXimera⟩	\ConfigureTheoremEnv{formula}
idea (*env.*)	Idea	
	179 ⟨classXimera⟩	\newtheorem{idea}{Idea}
	180 ⟨htXimera⟩	\ConfigureTheoremEnv{idea}
notation (*env.*)	Notation	
	181 ⟨classXimera⟩	\newtheorem{notation}{Notation}
	182 ⟨htXimera⟩	\ConfigureTheoremEnv{notation}
model (*env.*)	Model	
	183 ⟨classXimera⟩	\newtheorem{model}{Model}
	184 ⟨htXimera⟩	\ConfigureTheoremEnv{model}
observation (*env.*)	Observation	
	185 ⟨classXimera⟩	\newtheorem{observation}{Observation}
	186 ⟨htXimera⟩	\ConfigureTheoremEnv{observation}

proposition (*env.*)	Proposition	
	187 ⟨classXimera⟩	`\newtheorem{proposition}{Proposition}`
	188 ⟨htXimera⟩	`\ConfigureTheoremEnv{proposition}`
paradox (*env.*)	Paradox	
	189 ⟨classXimera⟩	`\newtheorem{paradox}{Paradox}`
	190 ⟨htXimera⟩	`\ConfigureTheoremEnv{paradox}`
procedure (*env.*)	Procedure	
	191 ⟨classXimera⟩	`\newtheorem{procedure}{Procedure}`
	192 ⟨htXimera⟩	`\ConfigureTheoremEnv{procedure}`
remark (*env.*)	Remark	
	193 ⟨classXimera⟩	`\newtheorem{remark}{Remark}`
	194 ⟨htXimera⟩	`\ConfigureTheoremEnv{remark}`
summary (*env.*)	Summary	
	195 ⟨classXimera⟩	`\newtheorem{summary}{Summary}`
	196 ⟨htXimera⟩	`\ConfigureTheoremEnv{summary}`
template (*env.*)	Template	
	197 ⟨classXimera⟩	`\newtheorem{template}{Template}`
	198 ⟨htXimera⟩	`\ConfigureTheoremEnv{template}`
warning (*env.*)	Warning	
	199 ⟨classXimera⟩	`\newtheorem{warning}{Warning}`
	200 ⟨htXimera⟩	`\ConfigureTheoremEnv{warning}`

2.4.3 Enumerate fixes

Make enumerate use a letter

```
201 ⟨*classXimera⟩
202 \renewcommand{\theenumi}{\textup{(\alph{enumi})}}
203 \renewcommand{\labelenumi}{\theenumi}
204 \renewcommand{\theenumii}{\textup{(\roman{enumii})}}
205 \renewcommand{\labelenumii}{\theenumii}
206 ⟨/classXimera⟩
```

2.4.4 Proofs

proof (*env.*) A mathematical proof environment.

```
207 ⟨*classXimera⟩
208 \renewcommand{\qedsymbol}{$\blacksquare$}
209 \renewenvironment{proof}[1][\proofname]
210   {\begin{trivlist}\item[\hskip \labelsep \itshape \bfseries #1{}\hspace{2ex}]]}
211 {\qed\end{trivlist}}
212 ⟨/classXimera⟩
```

2.4.5 Problem environments

These are problem environment decorations (these should be user invoked, not default). The decoration for these environments were inspired by http://tex.stackexchange.com/questions/11098/nice-formatting-for-theorems

```
213 ⟨*classXimera⟩
```

latexProblemContent Added for those that want to use UF problems without using the problem filter code. This command is renewed into something meaningful in the 'ProblemSelector.sty'.

```
214 \providecommand{\latexProblemContent}[1]{#1}
215 % Iterate count for problem counts.
216 \Make@Counter{Iteration@probCnt}
```

```
217 \newcommand{\hang}{% top theorem decoration
218   \begingroup%
219   \setlength{\unitlength}{.005\linewidth}% \linewidth/200
220     \begin{picture}(0,0)(1.5,0)%
221       \linethickness{1pt} \color{black!50}%
222       \put(-3,2){\line(1,0){206}}% Top line
223       \multido{\iA=2+-1,\iB=50+-10}{5}{% Top hangs
224         \color{black!\iB}%
225         \put(-3,\iA){\line(0,-1){1}}% Top left hang
226         %\put(203,\iA){\line(0,-1){1}}% Top right hang
227       }%
228     \end{picture}%
229   \endgroup%
230 }%
231 \newcommand{\hung}{% bottom theorem decoration
232   \nobreak
233   \begingroup%
234     \setlength{\unitlength}{.005\linewidth}% \linewidth/200
235     \begin{picture}(0,0)(1.5,0)%
236       \linethickness{1pt} \color{black!50}%
237       \put(60,0){\line(1,0){143}}% Bottom line
238       \multido{\iA=0+1,\iB=50+-10}{5}{% Bottom hangs
239         \color{black!\iB}%
240         %\put(-3,\iA){\line(0,1){1}}% Bottom left hang
241         \put(203,\iA){\line(0,1){1}}% Bottom right hang
242         \put(\iB,0){\line(60,0){10}}% Left fade out
243       }%
244     \end{picture}%
245   \endgroup%
246 }%
```

Configure environment configuration commands

The command `\problemNumber` contains all the format code to determine the number (and the format of the number) for any of the problem environments.

```
247 \MakeCounter{problem}
248 \newcommand{\problemNumber}{
249 % First we determine if we have a counter for this question depth level.
250 \ifcsname c@depth\Roman{problem@Depth}Count\endcsname% Check to see if counter exists
251 %If so, do nothing.
252 \else
253 %If not, create it.
254 \expandafter\newcounter{depth\Roman{problem@Depth}Count}
255 \expandafter\setcounter{depth\Roman{problem@Depth}Count}{0}
256 \fi
257
258 \expandafter\stepcounter{depth\Roman{problem@Depth}Count}
259 \arabic{depthICount}% The first problem depth, what use to be |\theproblem|.
260
261 \forloop{Iteration@probCnt}{2}{\arabic{Iteration@probCnt} < \numexpr \value{problem@Depth} +
262     .\expandafter\arabic{depth\Roman{Iteration@probCnt}Count}% Get the problem number of the
263 }
264 %\@ifpackageloaded{shuffle}{<true>}{<false>}% Check if Shuffle has been added. If so, add spe
265 %\ifhandout % Currently handout mode doesn't allow hints. Putting this code in place in case
266 % \theproblem
267 %\else
268 % \theproblem
269 %\fi
270 }
271
272
273 %%%%%% Configure various problem environment commands
274 \Make@Counter{problem@Depth}
275
276
```

```latex
277
278 %%%% Configure environments start content
279
280 \newcommand{\problemEnvironmentStart}[2]{%
281 % This takes in 2 arguments.
282 % The first is optional and is the old optional argument from existing environments.
283 % This is passed down to the associated problem environment name in case you want a global va
284 % The second argument is mandatory and is the name of the 'problem' environment,
285 % such as problem, question, exercise, etc.
286 % It then configures everything needed at the start of that environment.
287
288 \stepcounter{problem@Depth}% Started a problem, so we've sunk another problem layer.
289 \def\spaceatend{#1}%
290 \begin{trivlist}%
291 \item%
292   [%
293     \hskip\labelsep\sffamily\bfseries
294     #2 \problemNumber% Determine the correct number of the problem, and the format of that nu
295 ]%
296 \slshape
297 }
298
299
300
301 %%%% Configure environments end content
302
303 \newcommand{\problemEnvironmentEnd}{%This configures all the end content for a problem.
304 %
305 % First we need to see if we've dropped fully out of a depth level,
306 % so we can reset that counter back to zero for the next time we enter that depth level.
307 \stepcounter{problem@Depth}
308 \ifcsname c@depth\Roman{problem@Depth}Count\endcsname
309 \expandafter\ifnum\expandafter\value{depth\Roman{problem@Depth}Count}>0
310 \expandafter\setcounter{depth\Roman{problem@Depth}Count}{0}
311 \fi
312 \fi
313
314 \addtocounter{problem@Depth}{-2}% Exited a problem so we've exited a problem layer. Need -2 b
315
316 \par\addvspace{.5ex}\nobreak\noindent\hung %% line at the bottom
317
318 \ifhandout
319 \ifnewpage
320 \newpage
321 \fi
322 \fi
323 \end{trivlist}
324 }
325
326
327
328 %%%% Now populate the old environment names
329 %
330 % Old environments were "problem", "exercise", "exploration", and "question".
331 % Note that you can add content to the start/end code on top of these base code pieces if you
332
333
334 \newenvironment{problem}[1][2in]%
335 {%Env start code
336 \problemEnvironmentStart{#1}{Problem}
337 }
338 {%Env end code
339 \problemEnvironmentEnd
```

```
340 }
341
342 \newenvironment{exercise}[1][2in]%
343 {%Env start code
344 \problemEnvironmentStart{#1}{Exercise}
345 }
346 {%Env end code
347 \problemEnvironmentEnd
348 }
349
350 \newenvironment{exploration}[1][2in]%
351 {%Env start code
352 \problemEnvironmentStart{#1}{Exploration}
353 }
354 {%Env end code
355 \problemEnvironmentEnd
356 }
357
358 \newenvironment{question}[1][2in]%
359 {%Env start code
360 \problemEnvironmentStart{#1}{Question}
361 }
362 {%Env end code
363 \problemEnvironmentEnd
364 }
365 ⟨/classXimera⟩
```

Use an "identification" counter to assign IDs to the various problem-related DOM elements

```
366 ⟨∗htXimera⟩
367 \newcounter{identification}
368 \setcounter{identification}{0}
369
370 \newcommand{\ConfigureQuestionEnv}[2]{%
371 % refstepcounter ensures that labels get updated within these environments
372 \renewenvironment{#1}{\refstepcounter{problem}}{}%
373 \ConfigureEnv{#1}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div role="a
374 }
375
376 \ConfigureQuestionEnv{problem}{problem}
377 \ConfigureQuestionEnv{exercise}{exercise}
378 \ConfigureQuestionEnv{question}{question}
379 \ConfigureQuestionEnv{exploration}{exploration}
380 \ConfigureQuestionEnv{hint}{hint}
381 %%%%\ConfigureQuestionEnv{shuffle}{shuffle}
382 ⟨/htXimera⟩
```

2.4.6 Hints

hint (*env.*) Hint environments can be embedded inside problems.

```
383 ⟨∗classXimera⟩
```

Create a counter that will track how deeply nested the current hint is

```
384 \newcounter{hintLevel}
385 \setcounter{hintLevel}{0}
```

Create an empty shell to renew

```
386 \newenvironment{hint}{}{}
```

Now we renew the environment as needed, this should allow support for any transition code that treats some parts as a "handout" and some parts as non-handout. renewing the environment on the fly is a bit hacky.

```
387 \renewenvironment{hint}
388 {
```

```
389 \ifhandout
390 \setbox0\vbox\bgroup
391 \else
392 \begin{trivlist}\item[\hskip \labelsep\small\slshape\bfseries Hint:\hspace{2ex}]
393 \small\slshape
394 \fi
```

Step up hint level to track the nested level of the hint. This will be used for problem numbering.

```
395 \stepcounter{hintLevel}
396 }
397 {
398 \ifhandout
399 \egroup\ignorespacesafterend
400 \else
401 \end{trivlist}
402 \fi
```

Detract from hint level counter to track hint nested level

```
403 \addtocounter{hintLevel}{-1}
404 }
405
406 \ifhints
407 \renewenvironment{hint}{
408 \begin{trivlist}\item[\hskip \labelsep\small\slshape\bfseries Hint:\hspace{2ex}]
409 \small\slshape}
410 {\end{trivlist}}
411 \fi
412
413 ⟨/classXimera⟩
```

2.4.7 Solution

solution (*env.*) The solution to a problem.

```
414 ⟨∗classXimera⟩
415 %% solution environment
416 \ifhandout % what follows is handout behavior
417 \newenvironment{solution}%
418         {%
419 \setbox0\vbox\bgroup
420         }
421              {%
422 \egroup
423         }
424 \else
425 \newenvironment{solution}%
426         {%
427 \begin{trivlist}
428 \item[\hskip \labelsep\bfseries Solution:\hspace{2ex}]
429         }
430         % %% line at the bottom}
431         {
432 \end{trivlist}
433 \par\addvspace{.5ex}\nobreak\noindent\hung
434         }
435 \fi
436
437
438
439 ⟨/classXimera⟩
```

2.4.8 Code listing environments

code (*env.*) A code answer environment You cannot use Environ with the fancyvrb/listings package

if you want nested environments.

```
440 ⟨*classXimera⟩
441 \DefineVerbatimEnvironment{code}{Verbatim}{numbers=left,frame=lines,label=Code,labelposition=
442 ⟨/classXimera⟩
```

python (*env.*) A python answer environment You cannot use Environ with the fancyvrb/listings package
if you want nested environments

```
443 ⟨*classXimera⟩
444 \DefineVerbatimEnvironment{python}{Verbatim}{numbers=left,frame=lines,label=Python,labelposit
445 ⟨/classXimera⟩
```

javascriptCode (*env.*) A JavaScript answer environment Unfortunately the name `javascript` is already used
for the actual, executed (!) JavaScript interactive. environments

```
446 ⟨*classXimera⟩
447 \DefineVerbatimEnvironment{javascriptCode}{Verbatim}{numbers=left,frame=lines,label=JavaScrip
448 ⟨/classXimera⟩
449 ⟨*cfgXimera⟩
450 \renewenvironment{javascriptCode}{\NoFonts}{\EndNoFonts}
451 \ScriptEnv{javascriptCode}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<d:
452 ⟨/cfgXimera⟩
```

On the web, translate verbatim and lstlisting blocks into `<pre>` elements.

```
453 ⟨*cfgXimera⟩
454 \ConfigureEnv{verbatim}{\HCode{<pre>}}{\HCode{</pre>}}{}{}
455 \ConfigureEnv{lstlisting}{\HCode{<pre>}}{\HCode{</pre>}}{}{}
456 ⟨/cfgXimera⟩
```

2.4.9 Dialogues

dialogue (*env.*) A dialogue between people.

```
457 ⟨*classXimera⟩
458 \newenvironment{dialogue}{%
459     \renewcommand\descriptionlabel[1]{\hspace{\labelsep}\textbf{##1:}}
460     \begin{description}%
461 }{%
462     \end{description}%
463 }
464 ⟨/classXimera⟩
```

On the web, the resulting `<dl>` should have an appropriate `class` set.

```
465 ⟨*htXimera⟩
466 \renewenvironment{dialogue}{\begin{description}}{\end{description}}
467
468 \ConfigureList{dialogue}%
469     {\EndP\HCode{<dl \a:LRdir class="dialogue">}%
470         \PushMacro\end:itm
471 \global\let\end:itm=\empty}
472     {\PopMacro\end:itm \global\let\end:itm \end:itm
473 \EndP\HCode{</dd></dl>}\ShowPar}
474     {\end:itm \global\def\end:itm{\EndP\Tg</dd>}\HCode{<dt
475         class="actor">}\bgroup \bf}
476     {\egroup\EndP\HCode{</dt><dd\Hnewline class="speech">}}
477 ⟨/htXimera⟩
```

2.4.10 Instructor notes

```
478 ⟨*classXimera⟩
479
480 %% instructor intro/instructor notes
481 %%
482 \ifhandout % what follows is handout behavior
483 \ifinstructornotes
484 \newenvironment{instructorIntro}%
485         {%
```

```
486  \begin{trivlist}
487  \item[\hskip \labelsep\bfseries Instructor Introduction:\hspace{2ex}]
488  }
489        % %% line at the bottom}
490        {
491  \end{trivlist}
492  \par\addvspace{.5ex}\nobreak\noindent\hung
493        }
494  \else
495  \newenvironment{instructorIntro}%
496        {%
497  \setbox0\vbox\bgroup
498        }
499        {%If this mysteriously starts breaking
500                      % remove \ignorespacesafterend
501  \egroup\ignorespacesafterend
502        }
503                \fi
504  \else% for handout, so what follows is default
505  \ifinstructornotes
506  \newenvironment{instructorIntro}%
507        {%
508          \setbox0\vbox\bgroup
509        }
510  {%
511    \egroup
512  }
513                \else
514        \newenvironment{instructorIntro}%
515  {%
516    \begin{trivlist}
517    \item[\hskip \labelsep\bfseries Instructor Introduction:\hspace{2ex}]
518  }
519  % %% line at the bottom}
520  {
521    \end{trivlist}
522    \par\addvspace{.5ex}\nobreak\noindent\hung
523  }
524                \fi
525  \fi
526
527
528
529
530  %% instructorNotes environment
531  \ifhandout % what follows is handout behavior
532  \ifinstructornotes
533  \newenvironment{instructorNotes}%
534        {%
535  \begin{trivlist}
536  \item[\hskip \labelsep\bfseries Instructor Notes:\hspace{2ex}]
537        }
538        % %% line at the bottom}
539        {
540  \end{trivlist}
541  \par\addvspace{.5ex}\nobreak\noindent\hung
542        }
543        \else
544  \newenvironment{instructorNotes}%
545        {%
546          \setbox0\vbox\bgroup
547        }
548  {%
```

```
549    \egroup
550  }
551                   \fi
552 \else% for handout, so what follows is default
553 \ifinstructornotes
554 \newenvironment{instructorNotes}%
555        {%
556 \setbox0\vbox\bgroup
557        }
558        {%
559 \egroup
560        }
561        \else
562        \newenvironment{instructorNotes}%
563             {%
564         \begin{trivlist}
565         \item[\hskip \labelsep\bfseries Instructor Notes:\hspace{2ex}]
566             }
567             % %% line at the bottom}
568             {
569         \end{trivlist}
570         \par\addvspace{.5ex}\nobreak\noindent\hung
571             }
572                      \fi
573                          \fi
574
575 ⟨/classXimera⟩
```

2.4.11   Only

prompt (*env.*) The prompt part for mathmode

```
576 ⟨∗classXimera⟩
577 \ifxake
578        \newenvironment{prompt}{}{}
579 \else
580 \ifhandout
581 \NewEnviron{prompt}{}
582 % Currently breaks when put in mathmode!
583 % \newenvironment{prompt}{\suppress}{\endsuppress}
584 \else
585 \newenvironment{prompt}
586     {\bgroup\color{gray!50!black}}
587        {\egroup}
588 \fi
589 \fi
```

onlineOnly (*env.*)   Only display it online

```
590 \ifhandout
591 \NewEnviron{onlineOnly}{
592 \iftikzexport
593 \BODY
594 \else
595 \fi
596 }
597 \else
598 \newenvironment{onlineOnly}
599     {\bgroup\color{red!50!black}}
600 {\egroup}
601 \fi
602
603 \newcommand{\pdfOnly}[1]{\iftikzexport\else #1\fi}
604 ⟨/classXimera⟩
```

2.4.12 Foldable

The package `mdframed` is used to make pretty foldable, but the amsthm/mdframed conflict also messes up the .jax file so we don't load mdframed when performing the xake step. But even the below isn't enough to fix this.

```
605 %\iftikzexport\else\RequirePackage[framemethod=TikZ]{mdframed}\fi
```

foldable (*env.*) Does it fold?

```
606 ⟨*classXimera⟩
607
608 \colorlet{textColor}{black} % since textColor is referenced below
609 \colorlet{background}{white} % since background is referenced below
610
611 % The core environments. Find results in 4ht file.
612 %% pretty-foldable
613 %\iftikzexport
614 \newenvironment{foldable}{%
615 }{%
616 }
617 %\else
618 %\renewmdenv[
619 %  font=\upshape,
620 %  outerlinewidth=3,
621 %  topline=false,
622 %  bottomline=false,
623 %  leftline=true,
624 %  rightline=false,
625 %  leftmargin=0,
626 %  innertopmargin=0pt,
627 %  innerbottommargin=0pt,
628 %  skipbelow=\baselineskip,
629 %  linecolor=textColor!20!white,
630 %  fontcolor=textColor,
631 %  backgroundcolor=background
632 %]{foldable}%
633 %\fi
634
635 %% pretty-expandable
636 %\iftikzexport
637 \newenvironment{expandable}{%
638 }{%
639 }
640 %\else
641 %\newmdenv[
642 %  font=\upshape,
643 %  outerlinewidth=3,
644 %  topline=false,
645 %  bottomline=false,
646 %  leftline=true,
647 %  rightline=false,
648 %  leftmargin=0,
649 %  innertopmargin=0pt,
650 %  innerbottommargin=0pt,
651 %  skipbelow=\baselineskip,
652 %  linecolor=black,
653 %]{expandable}%
654 %\fi
655
656 \newcommand{\unfoldable}[1]{#1}
657
658 ⟨/classXimera⟩
```

On the web, these foldable elements could be HTML5 details and summary.

```
659 ⟨*htXimera⟩
```

660 `\renewenvironment{foldable}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<c`

661

662 `\renewenvironment{expandable}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode`

663

664 `}{\HCode{</div>}\IgnoreIndent}`

665

666 `\renewcommand{\unfoldable}[1]{\HCode{<span class="unfoldable">}#1\HCode{</span>}}`

667 ⟨/htXimera⟩

2.4.13 Leashes

leash (*env.*) Put content inside a scrollable box.

668 ⟨∗classXimera⟩

669

670 `\newenvironment{leash}[1]{%`

671 `}{%`

672 `}`

673

674

675 ⟨/classXimera⟩

676 ⟨∗htXimera⟩

677 `\renewenvironment{leash}[1]{\ifvmode \IgnorePar\fi \EndP\HCode{<div style="overflow: auto; he`

678 ⟨/htXimera⟩

2.5 Document metadata

2.5.1 Metadata

To encourage authors to include relevant parseable metadata in the preamble, we define some currently ignored commands.

\license   In the preamble, use `\license` with an SPDX license expression.

679 ⟨∗classXimera⟩

680 `\newcommand{\license}{\excludecomment}`

681 ⟨/classXimera⟩

\acknowledgement   In the preamble, use `\acknowledgement` to credit others who contributed to the intellectual content beside the author.

682 ⟨∗classXimera⟩

683 `\newcommand{\acknowledgement}{\excludecomment}`

684 ⟨/classXimera⟩

\tag   In the preamble, a `\tag` provides a free-form taxonomy.

685 ⟨∗classXimera⟩

686 `\renewcommand{\tag}{\excludecomment}`

687 ⟨/classXimera⟩

On the HTML side, we mark the file as the appropriate kind of object—either activity or xourse.

688 ⟨∗htXourse⟩

689 `% Mark this as a xourse file`

690 `\Configure{@HEAD}{\HCode{<meta name="description" content="xourse" />\Hnewline}}`

691 ⟨/htXourse⟩

2.5.2 Abstract

abstract (*env.*) Every activity should include a short abstract.

692 ⟨∗classXimera⟩

693 `\let\abstract\relax`

694 `\let\endabstract\relax`

695 `% Use of environ package, may want to find a better way.`

696 `\NewEnviron{abstract}{\protected@xdef\theabstract{\BODY}}`

697 ⟨/classXimera⟩

The abstract has been stored in `\theabstract` and should be emitted as a div, but confusingly I guess `<div class="abstract">` is defined somewhere deeper inside tex4ht, so the code below is probably unnecessary.

2.5.3 Titles and authors

2.5.4 Authors

`\author` Activities have authors. Warn the user if no author is provided.

```
698 ⟨∗classXimera⟩
699 \let\@emptyauthor\@author
700 \def\author#1{\gdef\@author{#1}}
701 \def\@author{\@latex@warning@no@line{No \noexpand\author given}}
702 ⟨/classXimera⟩
```

Include author name in meta tags

```
703 ⟨∗htXimera⟩
704 \Configure{@HEAD}{\HCode{<meta name="author" content="}\@author\HCode{" />\Hnewline}}
705 ⟨/htXimera⟩
```

The `\and` command would emit tabular environments which really should not appear in a meta tag.

```
706 ⟨htXimera | classXimera⟩\def\and{and }
```

2.5.5 Title

`\title` Activities have titles.

```
707 ⟨∗classXimera⟩
708 \let\title\relax
709 \newcommand{\title}[1][]{{\protected@xdef\@pretitle{#1}}\protected@xdef\@title}
710
711 \title{}
712
713 \newcounter{titlenumber}
714 \renewcommand{\thetitlenumber}{\arabic{titlenumber}}
715 %\renewcommand{\thesection}{\arabic{titlenumber}} %% Makes section numbers work
716 \setcounter{titlenumber}{0}
717
718 \newpagestyle{main}{
719 \sethead[\textsl{\ifnumbers\thetitlenumber\hspace{1em}\fi\@title}][][] % even
720 {}{}{\textsl{\ifnumbers\thetitlenumber\hspace{1em}\fi\@title}} % odd
721 \setfoot[\thepage][][] % even
722 {}{}{\thepage} % odd
723 }
724 \pagestyle{main}
```

`\maketitle` In a ximera document, redefine `\maketitle` and put them in a table of contents. The `\phantomsection` is to fix the hrefs.

```
725 \renewcommand\maketitle{%
726   \addtocounter{titlenumber}{1}%
727   {\flushleft\large\bfseries \@pretitle\par\vspace{-1em}}
728   {\flushleft\LARGE\bfseries {\ifnumbers\thetitlenumber\fi}{\ifnumbers\hspace{1em}\else\hspac
729   \phantomsection%
730   \ifnumbers\addcontentsline{toc}{section}{\thetitlenumber~\@title}\else\addcontentsline{toc}
731   \vskip .6em\noindent\textit\theabstract\setcounter{problem}{0}\setcounter{section}{0}\setco
732   \ifnooutcomes\else\let\thefootnote\relax\footnote{Learning outcomes: \theoutcomes}\fi
733   \ifnoauthor\else\let\thefootnote\relax\footnote{Author(s):~\@author}\fi
734   \aftergroup\@afterindentfalse
735   \aftergroup\@afterheading}
736
737 \ifnumbers
738 \setcounter{secnumdepth}{2}
739 \renewcommand{\thesection}{\arabic{titlenumber}.\arabic{section}}
740 \renewcommand{\thesubsection}{\arabic{titlenumber}.\arabic{section}.\arabic{subsection}}
```

```
741 \else
742 \setcounter{secnumdepth}{-2}
743 \fi
744
745 \def\activitystyle{}
746 \newcounter{sectiontitlenumber}
747 %\setcounter{secnumdepth}{0}
748
749 \newcommand\chapterstyle{%
750   \def\activitystyle{activity-chapter}
751   \def\maketitle{%
752     \addtocounter{titlenumber}{1}%
753                  {\flushleft\small\sffamily\bfseries\@pretitle\par\vspace{-1.5em}}%
754                  {\flushleft\LARGE\sffamily\bfseries\thetitlenumber\hspace{1em}\@title \pa
755                  {\vskip .6em\noindent\textit\theabstract\setcounter{problem}{0}\setcounte
756                  \par\vspace{2em}
757                  \phantomsection\addcontentsline{toc}{section}{\textbf{\thetitlenumber\hsp
758 }}
759
760
761 \newcommand\sectionstyle{%
762   \def\activitystyle{activity-section}
763   \def\maketitle{%
764     \addtocounter{section}{1}
765     \setcounter{sectiontitlenumber}{\value{section}}
766     {\flushleft\small\sffamily\bfseries\@pretitle\par\vspace{-1.5em}}%
767     {\flushleft\Large\sffamily\bfseries\thetitlenumber.\thesectiontitlenumber\hspace{1em}\@ti
768     {\vskip .6em\noindent\textit\theabstract\setcounter{subsection}{0}}%
769     \par\vspace{2em}
770     \phantomsection\addcontentsline{toc}{section}{\thetitlenumber.\thesectiontitlenumber\hspa
771 \renewcommand\section{\@startsection{subsection}{2}{\z@}%
772                                     {-3.25ex\@plus -1ex \@minus -.2ex}%
773                                     {1.5ex \@plus .2ex}%
774                                     {\normalfont\large\bfseries}}
775
776 \renewcommand\subsection{\@startsection{subsubsection}{3}{\z@}%
777                                        {-3.25ex\@plus -1ex \@minus -.2ex}%
778                                        {1.5ex \@plus .2ex}%
779                                        {\normalfont\normalsize\bfseries}}
780
781 }}
782
783
784 \iftikzexport%% allows xake to handle \chapterstyle and \sectionstye
785 \renewcommand\chapterstyle{\def\activitystyle{chapter}}
786 \renewcommand\sectionstyle{\def\activitystyle{section}}
787 \else
788 \fi
789
790 ⟨/classXimera⟩
```

Eliminate some formatting that we'll handle later with CSS

```
791 ⟨∗htXimera⟩
792 \renewcommand{\maketitle}{}
793 ⟨/htXimera⟩
```

2.5.6   Learning Outcomes

\outcome   Specify a learning outcome, either at the level of a `problem` or an entire document in the preamble.

```
794 ⟨∗classXimera⟩
795 \def\theoutcomes{}
796
797 \ifdefined\HCode%
```

```
798    \newcommand{\outcome}[1]{}
799  \else%
800    \newwrite\outcomefile
801    \immediate\openout\outcomefile=\jobname.oc
802
803    \newcommand{\outcome}[1]{\edef\theoutcomes{\theoutcomes #1~}%
804    \immediate\write\outcomefile{\unexpanded{\outcome}{#1}}}
805    \fi%
806  ⟨/classXimera⟩
```

These can appear in either the preamble or in problem environments. with pdflatex, we produce the .oc file which includes ALL the outcomes; in the tex4ht world, we just produce spans for the specific outcomes.

```
807  ⟨*cfgXimera⟩
808  \renewcommand{\outcome}[1]{
809    \Configure{@HEAD}{\HCode{<meta name="learning-outcome" content="#1"/>\Hnewline}}
810  }
811  % Sometimes there are no outcomes at all
812  \IfFileExists{\jobname.oc}{\input{\jobname.oc}}{}
813
814  \renewcommand{\outcome}[1]{%
815    \HCode{<span class="learning-outcome">#1</span>}
816  }
817  ⟨/cfgXimera⟩
```

2.5.7 Labels and references

\label Labels and refs both generate anchors. A \label can be referenced from any file in the xourse.

```
818  ⟨*htXimera⟩
819  \let\oldlabel\label
820  \renewcommand{\label}[1]{\oldlabel{#1}\HCode{<a class="ximera-label" id="#1"></a>}}
821  ⟨/htXimera⟩
```

\ref A \ref can connect one TEX file to another if they are in the same xourse.

```
822  ⟨*htXimera⟩
823  \renewcommand{\ref}[1]{\HCode{<a class="reference" href="\##1">#1</a>}}
824  ⟨/htXimera⟩
```

2.6 Images

2.6.1 Images

image (*env.*) Place images inside an image environment. On paper, this centers the image. On the web, this provides additional benefits.

```
825  ⟨*classXimera⟩
826  %\newenvironment{image}[1][]{\begin{center}}{\end{center}}
827  \NewEnviron{image}[1][3in]{%
828    \begin{center}\resizebox{#1}{!}{\BODY}\end{center}% resize and center
829  }
830  ⟨/classXimera⟩
```

\alt Inside an image environment, \alt provides alt-text for assistive technology like screen-readers.

```
831  ⟨*classXimera⟩
832  \newcommand{\alt}[1]{}
833  ⟨/classXimera⟩
```

The image environment doesn't actually work in tex4ht as defined with NewEnviron; so this renewenvironment is needed. image-environment also gets formatted in a well, and when the user clicks on the image, it zooms in.

```
834  ⟨*htXimera⟩
835  \newcounter{imagealt}
836  \setcounter{imagealt}{0}
```

```
837 \renewenvironment{image}[1][]{\stepcounter{imagealt}%
838   \ifvmode \IgnorePar\fi \EndP%
839   \HCode{<div class="image-environment" role="img" aria-labelledby="image-alt-\arabic{imageal
840 }{\HCode{</div>}}
841 \renewcommand{\alt}[1]{\HCode{<div style="display: none;" id="image-alt-\arabic{imagealt}">}
842 ⟨/htXimera⟩
```

Although we accept many formats, SVG is preferred on the web. Since we have a different mechanism for producing `alt` text, we want to ignore tex4ht's own method fo producing alt text.

```
843 ⟨*cfgXimera⟩
844 \DeclareGraphicsExtensions{.jpg,.png,.gif,.svg}
845 \Configure{graphics*}
846 {svg}{
847   {\Configure{Needs}{File: \Gin@base.svg}\Needs{}}
848   \Picture[]{\csname Gin@base\endcsname.svg \csname a:Gin-dim\endcsname}%
849 }
850 ⟨/cfgXimera⟩
```

This is a hack to kill `includegraphics` commands in `\documentclass{standalone}` files

```
851 ⟨*cfgXimera⟩
852 \ifcsname ifstandalone\endcsname
853   \ifstandalone
854     \renewcommand\includegraphics[2][]{}
855   \fi
856 ⟨/cfgXimera⟩
```

PGF sometimes causes trouble, but we simply don't care in tex4ht mode.

```
857 ⟨*htXimera⟩
858 \newcommand{\pgfsyspdfmark}[3]{}
859 ⟨/htXimera⟩
```

2.6.2 TikZ export

We generate SVGs and PNGs for any TikZ images, via the "externalize" feature of TikZ.

Currently TikZ doesn't compile natively into the website because of how the xake bake compilation works. In order to make Tikz work, you need to get the tool `mutool` on the machine that is performing `xake bake`.

```
860 ⟨*classXimera⟩
861 \ifdefined\HCode
862   \tikzexporttrue
863 \fi
864
865 \iftikzexport
866   \usetikzlibrary{external}
867
868   \ifdefined\HCode
869     % in htlatex, just include the svg files
870     \def\pgfsys@imagesuffixlist{.svg}
871
872     \tikzexternalize[prefix=./,mode=graphics if exists]
873   \else
874     % in pdflatex, actually generate the svg files
875     \tikzset{
876       /tikz/external/system call={
877         pdflatex \tikzexternalcheckshellescape
878         -halt-on-error -interaction=batchmode
879         -jobname "\image" "\\PassOptionsToClass{tikzexport}{ximera}\texsource";
880         mutool draw -F svg \image.pdf > \image.svg ;     % mutool adds "1" to filename ????
881         mutool draw -o \image.svg \image.pdf ;
882         mutool draw -r 150 -c rgbalpha -o \image.png \image.pdf ;
883         ebb -x \image.png
884       }
```

```
885      }
886      \tikzexternalize[optimize=false,prefix=./]
887    \fi
888
889    \fi
890
891 ⟨/classXimera⟩
```

2.6.3   XKCD

\xkcd   Reference an XKCD cartoon.

```
892 ⟨*classXimera⟩
893 \newcommand{\xkcd}[1]{#1}
894 ⟨/classXimera⟩
```

On the web, this should be an image linked to the actual XKCD website.

```
895 ⟨*htXimera⟩
896 \renewcommand{\xkcd}[1]{\ifvmode \IgnorePar\fi \EndP\HCode{<img src="https://imgs.xkcd.com/cc
897 ⟨/htXimera⟩
```

2.7   Links

We put hyperref after all other packages becuase that is better.

```
898 ⟨*classXimera⟩
899 % Don't use hyperref when using Tex4ht
900 \ifdefined\HCode
901 \RequirePackage{hyperref}
902 \else
903 \RequirePackage[pdfpagelabels,colorlinks=true,allcolors=blue!30!black]{hyperref}
904 \pdfstringdefDisableCommands{\def\hskip{}}%% quiets warning
905 \fi
906 ⟨/classXimera⟩
```

2.8   Interactives

2.8.1   Including widgets

\includeinteractive   Cognate to `includegraphics` but instead of a graphics file, accepts a `.js` file which will be loaded as an interactive widget.

```
907 ⟨*classXimera⟩
908 \define@key{interactive}{id}{\def\interactive@id{#1}}
909 \setkeys{interactive}{id=}
910 \newcommand{\includeinteractive}[2][]{
911 \setkeys*{interactive}{#1}%
912 \ifthenelse{\equal{\interactive@id}{}}{}{\recordvariable{\interactive@id}}
913 Interactive
914 }
915 ⟨/classXimera⟩
```

```
916 ⟨*htXimera⟩
917 \renewcommand{\includeinteractive}[2][]{\stepcounter{identification}\ifvmode \IgnorePar\fi \H
918 ⟨/htXimera⟩
```

2.8.2   Google Sheet

\googleSheet   googleSheet command. Requires id, width, and height as arguments. optional arguments are gid for sheet ID and range for cell range. command definition

```
919 ⟨*classXimera⟩
920 % Google Spreadsheet link (read only)
921 \newcommand{\googleSheet}[5]{%
922   Google Spreadsheet link: \url{https://docs.google.com/spreadsheets/d/#1}%
923 }
924 ⟨/classXimera⟩
```

```
925 ⟨∗htXimera⟩
926 \renewcommand{\googleSheet}[5]{%
927   \ifthenelse{\equal{#4}{}}%
928     {\HCode{<iframe width="#2px" height="#3px" src="https://docs.google.com/spreadsheets/d/#1
929     {\ifthenelse{\equal{#5}{}}%
930       {\HCode{<iframe width="#2px" height="#3px" src="https://docs.google.com/spreadsheets/d
931       {\HCode{<iframe width="#2px" height="#3px" src="https://docs.google.com/spreadsheets/d
932     }%
933   }%
934 ⟨/htXimera⟩
```

2.8.3   Geogebra

\geogebra   Geogebra command. Requires id, width, and height as arguments.

```
935 ⟨∗classXimera⟩
936 %Geogebra link
937 \newcommand{\geogebra}[3]{Geogebra link: \url{https://www.geogebra.org/m/#1}}
938 ⟨/classXimera⟩
```

Define keys for answer geogebra key=value pairs.

```
939 ⟨∗htXimera⟩
940 \define@key{geogebra}{rc}[true]{\def\geo@rc{#1}}
941 \define@key{geogebra}{sdz}[true]{\def\geo@sdz{#1}}
942 \define@key{geogebra}{smb}[true]{\def\geo@smb{#1}}
943 \define@key{geogebra}{stb}[true]{\def\geo@stb{#1}}
944 \define@key{geogebra}{stbh}[true]{\def\geo@stbh{#1}}
945 \define@key{geogebra}{ld}[true]{\def\geo@ld{#1}}
946 \define@key{geogebra}{sri}[true]{\def\geo@sri{#1}}
947 %set default key values
948 \setkeys{geogebra}{rc=false,sdz=false,smb=false,stb=false,stbh=false,ld=false,sri=false}
949 %command definition
950 \renewcommand{\geogebra}[4][]{%
951   \setkeys{geogebra}{#1}% Set new keys
952   \HCode{<iframe scrolling="no" src="https://www.geogebra.org/material/iframe/id/#2/width/#3/
953 ⟨/htXimera⟩
```

2.8.4   Desmos

\desmos   Desmos command. Requires id, width, and height as arguments.

```
954 ⟨∗classXimera⟩
955 \newcommand{\desmos}[3]{Desmos link: \url{https://www.desmos.com/calculator/#1}}
956 \newcommand{\desmosThreeD}[3]{Desmos3D link: \url{https://www.desmos.com/3d/#1}}
957 ⟨/classXimera⟩
```

```
958 ⟨∗htXimera⟩
959 \renewcommand{\desmos}[3]{\HCode{<iframe src="https://www.desmos.com/calculator/#1" width="10
960 \renewcommand{\desmosThreeD}[3]{\HCode{<iframe src="https://www.desmos.com/3d/#1" width="#2px
961 ⟨/htXimera⟩
```

2.8.5   Graphs

\graph   An embedded graph (in math mode).

```
962 ⟨∗classXimera⟩
963 \newcommand{\graph}[2][]{\text{Graph of $#2$}}
964 ⟨/classXimera⟩
```

```
965 ⟨∗htXimera⟩
966 \renewcommand{\graph}[2][]{\HCode{<div class="graph" data-options="#1">}#2\HCode{</div>}}
967 ⟨/htXimera⟩
```

2.8.6   Video

\youtube   Youtube command. Requires id.

```
968 ⟨∗classXimera⟩
```

```
969 \newcommand{\youtube}[1]{YouTube link: \url{https://www.youtube.com/watch?v=#1}}
970 ⟨/classXimera⟩

971 ⟨∗htXimera⟩
972 \renewcommand{\youtube}[1]{\ifvmode \IgnorePar\fi \EndP\HCode{<div class="video youtube-playe
973 ⟨/htXimera⟩
```

Video commands are also emitted, slightly differently, when placed at top-level in a xourse file.

```
974 ⟨∗htXourse⟩
975 \renewcommand\youtube[1]{%
976 \ifvmode \IgnorePar\fi \EndP\HCode{<a class="youtube" href="https://www.youtube.com/watch?v=#
977 }
978 ⟨/htXourse⟩
```

2.8.7   JavaScript

javascript (*env.*) Code inside a javascript environment is printed on paper, but executed on the web.

```
979 ⟨∗classXimera⟩
980 \DefineVerbatimEnvironment{javascript}{Verbatim}{numbers=left,frame=lines,label=JavaScript,la
981 ⟨/classXimera⟩

982 ⟨∗htXimera⟩
983 % for programming javascript
984 \renewenvironment{javascript}{\NoFonts}{\EndNoFonts}
985 \ScriptEnv{javascript}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div cl
986 ⟨/htXimera⟩
```

\js   Code inside a \js macro is evaluated and replaced with its value.

```
987 ⟨∗classXimera⟩
988 \def\js#1{\mbox{\texttt{\detokenize{#1}}}}
989 ⟨/classXimera⟩

990 ⟨∗htXimera⟩
991 \def\js#1{\stepcounter{identification}\HCode{<span class="inline-javascript" id="javascript\a
992 ⟨/htXimera⟩
```

2.9   SageMath support

Load SageTEX if it exists.

```
993 ⟨∗classXimera⟩
994 \IfFileExists{sagetex.sty}{\RequirePackage{sagetex}}{}
995 ⟨/classXimera⟩
```

sageCell (*env.*)   Create an interactive SageMath widget.

```
996 ⟨∗classXimera⟩
997 \DefineVerbatimEnvironment{sageCell}{Verbatim}{numbers=left,frame=lines,label=SAGE,labelposit
998 ⟨/classXimera⟩

999 ⟨∗htXimera⟩
1000 \renewenvironment{sageCell}{\NoFonts}{\EndNoFonts}
1001 \ScriptEnv{sageCell}{\ifvmode \IgnorePar\fi \EndP\HCode{<div class="sage"><script type="text/
1002 ⟨/htXimera⟩
```

sageOutput (*env.*)   Execute SageMath code and output the result.

```
1003 ⟨∗classXimera⟩
1004 \DefineVerbatimEnvironment{sageOutput}{Verbatim}{numbers=left,frame=lines,label=SAGE-Output,l
1005 ⟨/classXimera⟩

1006 ⟨∗htXimera⟩
1007 \renewenvironment{sageOutput}{\NoFonts}{\EndNoFonts}
1008 \ScriptEnv{sageOutput}{\ifvmode \IgnorePar\fi \EndP\HCode{<div class="sageOutput"><script typ
1009 ⟨/htXimera⟩
```

sageSilent (*env.*)   Execute SageMath code without outputing the result.

```
1010 ⟨∗htXimera⟩
1011 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1012 \renewenvironment{sagesilent}{\NoFonts}{\EndNoFonts}
1013 \ScriptEnv{sagesilent}{\ifvmode \IgnorePar\fi \EndP\HCode{<script type="text/sagemath">}\Html
1014 ⟨/htXimera⟩
```

2.10 Answerables

2.10.1 Answers

\answer A math answer

```
1015 ⟨∗classXimera⟩
1016
1017 \ifdefined\HCode
1018 \newcommand{\recordvariable}[1]{}
1019 \else
1020 \newwrite\idfile
1021 \immediate\openout\idfile=\jobname.ids
1022 \newcommand{\recordvariable}[1]{\ifthenelse{\equal{#1}{}}{}{\immediate\write\idfile{var #1;}}
1023 \fi
```

Determines if answer is shown in handout mode. when `given=true`, show answer in handout mode, show answer in "given box" outside handout mode. When `given=false`, do not show answer in handout mode, show answer outside handout mode

```
1024 \define@key{answer}{given}[true]{\def\ans@given{#1}}
```

Used for setting numeric answer tolerance for online student input.

```
1025 \define@key{answer}{tolerance}{\def\ans@tol{#1}}
```

Used to run dynamic js code on student provided answers. Note: currently pdf outputs the validator code itself.

```
1026 \define@key{answer}{validator}{}
```

Used for assigning a js ID to answer for dynamic code (eg validators).

```
1027 \define@key{answer}{id}{\def\ans@id{#1}}
```

Used to set anticipated input format; eg "string".

```
1028 \define@key{answer}{format}{}
```

Used to hide the answer input box on the web.

```
1029 \define@key{answer}{onlinenoinput}[false]{}
```

Used to add a 'show answer' button to the answer blank.

```
1030 \define@key{answer}{onlineshowanswerbutton}[false]{}
```

Set default values for \answer command `key=value` pairs. Default values are `given = false`.

```
1031 \setkeys{answer}{id=,given=false,onlinenoinput=false,onlineshowanswerbutton=false}
```

Basic code for \answer.

```
1032
1033 % Options for handout
1034 \newcommand{\answerFormatLength}{2cm}
1035
1036 \newcommand{\answerFormatDots}[1]{\ldots\ldots}
1037 \newcommand{\answerFormatLine}[1]{\protect\rule{\answerFormatLength}{0.4pt}}
1038 \newcommand{\answerFormatFlexibleLine}[1]{\protect\rule{\widthof{$#1$}*2}{0.4pt}}
1039 \newcommand{\answerFormatFlexibleBox}[1]{\fbox{\scalebox{2}{\phantom{$#1$}}}}
1040
1041 % options for default (i.e with answers filled in)
1042 \newcommand{\answerFormatPlain}[1]{\ensuremath{#1}}
1043 \newcommand{\answerFormatBlue}[1]{\color{blue}\ensuremath{#1}}
1044 \newcommand{\answerFormatBoxed}[1]{\fbox{\ensuremath{#1}}}
1045 \newcommand{\answerFormatBoxedGiven}[1]{\underset{\scriptstyle\mathrm{given}}{\fbox{\ensurema
1046
1047 % defaults for handout and default mode, and for \answer[given]
```

```
1048 \let\handoutAnswerFormat\answerFormatDots
1049 \let\defaultAnswerFormat\answerFormatBlue
1050 \let\givenAnswerFormat\answerFormatBoxedGiven
1051
1052 \newcommand{\answer}[2][]{%
1053 \ifmmode%
1054 \setkeys{answer}{#1}%
1055 \recordvariable{\ans@id}
1056 \ifthenelse{\boolean{\ans@given}}
1057 {% Start then statement
1058 \ifhandout
1059 #2
1060 \else
1061 \givenAnswerFormat{#2} %% in case the argument helps formatting
1062 \fi
1063 }% End then statement
1064 {% Start else statement
1065 \ifhandout
1066 \handoutAnswerFormat{#2} %% in case the argument helps formatting
1067 \else% show answer in box outside handout mode
1068 \defaultAnswerFormat{#2} %% in case the argument helps formatting
1069 \fi
1070 }% End else statement
1071 \else%
1072 \GenericError{\space\space\space\space}% Throw an error based on... something? -- Jason
1073 {Attempt to use \@backslashchar answer outside of math mode}
1074 {See https://github.com/ximeraProject/ximeraLatex for explanation.}
1075 {Need to use either inline or display math.}%
1076 \fi
1077 }
1078 ⟨/classXimera⟩
```

On the HTML side, `\answer` emits spans—but it is usually just handled directly by MathJax.

```
1079 ⟨∗htXimera⟩
1080 \renewcommand{\answer}[2][false]{\HCode{<span class="answer respondable">}#2\HCode{</span>}}
1081
1082 \def\validator[#1]{\stepcounter{identification}\HCode{<div class="validator" id="validator\ar
1083 \def\endvalidator{\HCode{</div>}}
1084
1085 ⟨/htXimera⟩
```

2.10.2 Multiple choice and the like

multipleChoice (*env.*) Multiple choice

```
1086 ⟨∗classXimera⟩
1087 % Jim: Originally this was \renewcommand{\theenumi}{$(\mathrm{\alph{enumi}})$}
1088 % but that breaks tex4ht because mathmode can only be processed by mathjax.
1089 % so now I made this just italicized.
```

2.10.3 Options

```
1090 \define@key{choice}{value}[]{\def\choice@value{#1}}
```

This flags the answer as the correct answer
```
1091 \define@boolkey{choice}{correct}[true]{\def\choice@correct{#1}}
```

Use an ID to refer to the choice.
```
1092 \define@key{multipleChoice}{id}{\def\mc@id{#1}}
```

`\otherchoice` outputs the item if correct and nothing if incorrect.
```
1093 \define@key{otherchoice}{value}[]{\def\otherchoice@value{#1}}
1094 \define@boolkey{otherchoice}{correct}[true]{\def\otherchoice@correct{#1}}
```

Default key choices for multiple choice options. Default for choice pairs. Default: answers without the option "correct=true" is "incorrect".

1095 `\setkeys{choice}{correct=false,value=}`

Defaults for multipleChoice pairs. Default to no id? – Jason

1096 `\setkeys{multipleChoice}{id=}`

Defaults for otherchoice pairs. Default "otherchoice" to behave like "choice" for error checking.

1097 `\setkeys{otherchoice}{correct=false,value=}`
1098 ⟨/classXimera⟩

2.10.4 Choices

`\choice` Like `\item` but for choice environments. choice command denotes a possible answer choice for the multiple choice question.

```
1099 ⟨∗classXimera⟩
1100 \newcommand{\choice}[2][]{%
1101 \setkeys{choice}{#1}%
1102 \item{#2}
1103 \ifthenelse{\boolean{\choice@correct}}
1104     {% Begin then result
1105     \ifhandout% if it's a handout do nothing.
1106     \else% otherwise place a checkmark when you select the "correct choice"... maybe? -- Jaso
1107         \,\checkmark\,\setkeys{choice}{correct=false}
1108     \fi
1109     }% End then result
1110     {}% Begin/End else result.
1111 }
1112
1113 %Define an expandable version of choice Not really meant to be used outside this package (use
1114 % Is there a reason we can't just always use this as default? -- Jason
1115 \newcommand{\choiceEXP}[2][]{%
1116 \expandafter\setkeys\expandafter{choice}{#1}%
1117 \item{#2}
1118 \ifthenelse{\boolean{\choice@correct}}
1119 {% Begin then result
1120 \ifhandout
1121 \else
1122 \,\checkmark\,\setkeys{choice}{correct=false}
1123 \fi
1124 }% End then result
1125 {}% Begin/End else result.
1126 } %% note all the {} are needed in case the choice has [] in it.
1127
1128 % \otherchoice is the \choice used in wordChoice command.
1129 \newcommand{\otherchoice}[2][]{%
1130 \ignorespaces%
1131 \setkeys{otherchoice}{#1}%
1132 \ifthenelse{\boolean{\otherchoice@correct}}%
1133 {% Start then result
1134 #2\ignorespaces\setkeys{otherchoice}{correct=false}\ignorespaces%
1135 }% End then result
1136 {}% Start/End else result
1137 \ignorespaces%
1138 }%
1139 \newcommand{\inlinechoice}[2][]{%
1140 \setkeys{choice}{#1}%
1141 \iffirstinlinechoice
1142 (\hspace{-.25em}
1143 \firstinlinechoicefalse
1144 \else
1145 /
1146 \fi
1147 #2
1148 \ifthenelse{\boolean{\choice@correct}}%
1149 {% Start then result
```

```
1150 \ifhandout\else\checkmark\ignorespaces\setkeys{choice}{correct=false}\ignorespaces\fi%
1151 }% End then result
1152 {}% Start/End else result
1153 \hspace{-.25em}\ignorespaces%
1154 }
1155
1156 ⟨/classXimera⟩
```

On the HTML side, \choice emits <span>s.

```
1157 ⟨∗htXimera⟩
1158 \newcounter{choiceId}
1159 \renewcommand{\choice}[2][]{%
1160 \setkeys{choice}{correct=false}%
1161 \setkeys{choice}{#1}%
1162 \stepcounter{choiceId}\IgnorePar%
1163 \HCode{<span class="choice }%
1164 \ifthenelse{\boolean{\choice@correct}}{\HCode{correct}}{}
1165 \HCode{" }
1166 \ifthenelse{\equal{\choice@value}{}}{}{\HCode{data-value="\choice@value" }}
1167 \HCode{id="choice\arabic{choiceId}">}%
1168 #2\HCode{</span>}}
1169 \let\inlinechoice\choice
1170 ⟨/htXimera⟩
```

2.10.5 Environment(s)

multipleChoice (*env.*) The environment multipleChoice@ is for internal use only. Wrap \choices in a multipleChoice environment to make a multiple choice question.

```
1171 ⟨∗classXimera⟩
1172 \newenvironment{multipleChoice}[1][]
1173 {% Environment Start Code
1174 \setkeys{multipleChoice}{#1}%
1175 \recordvariable{\mc@id}%
1176 \begin{trivlist}
1177 \item[\hskip \labelsep\small\bfseries Multiple Choice:]\hfil
1178 \begin{enumerate}
1179 }% Note this means that \item has to be the first line after \begin{multipleChoice}.
1180 {% Environment End Code
1181 \end{enumerate}
1182 \end{trivlist}
1183 }
1184
1185 %multipleChoice@ is for internal use only! (used in wordChoice)
1186 %this is simply a wrapper for the sole showing (other)choice.
1187 \newenvironment{multipleChoice@}[1][]{}{})}
1188 ⟨/classXimera⟩
```

On the web, you might also expect these to be "problem environments" but they aren't – they're respondables. You might expect a \setcounter{choiceId}{0} here — that would be wrong, because then the generated IDs would no longer be unique.

```
1189 ⟨∗htXimera⟩
1190 \renewenvironment{multipleChoice}[1][]
1191 {\setkeys{multipleChoice}{#1}%
1192 \stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div class="multiple-choice" ]
1193 \ifthenelse{\equal{\mc@id}{}}{}{\HCode{data-id="\mc@id" }}%
1194 \HCode{id="problem\arabic{identification}">}%
1195 }{\HCode{</div>}\IgnoreIndent}
1196 \ConfigureEnv{multipleChoice}{}{}{}{}
1197 ⟨/htXimera⟩
```

2.11 Word choice

\wordChoice An in-line version of multipleChoice: uses enumitem package note, it is coded as a single

line to avoid unwanted spaces in "given" mode.

```
1198 ⟨∗classXimera⟩
1199 \newcommand{\wordChoice}[1]{%
1200 \let\choicetemp\choice% Assign a "choicetemp" command to duplicate choice.
1201 \ifwordchoicegiven% If wordchoice option is on, we need to juggle around some definitions.
1202 \let\choice\otherchoice%
1203 %\begin{multipleChoice@}% -unnecessary (REMOVE THIS LINE IF THE YEAR IS 2019 or Beyond)
1204 #1
1205 %\end{multipleChoice@}% -unnecessary (REMOVE THIS LINE IF THE YEAR IS 2019 or Beyond)
1206 \else% If it isn't the regular "choice" command should work.
1207 \let\choice\inlinechoice%
1208 \begin{multipleChoice@}%
1209 #1%
1210 \end{multipleChoice@}%
1211 \fi%
1212 \let\choice\choicetemp% Now that choicetmp has been manipulated to what we want, replace choi
1213 }%
1214
1215
1216 ⟨/classXimera⟩
```

This is actually just word choice

```
1217 ⟨∗htXimera⟩
1218 \renewenvironment{multipleChoice@}{\refstepcounter{problem}}{}%
1219 \ConfigureEnv{multipleChoice@}{\stepcounter{identification}\IgnorePar\HCode{<span class="word
1220 ⟨/htXimera⟩
```

2.12 Select all

selectAll (*env.*) A multiple-multiple choice question

```
1221 ⟨∗classXimera⟩
1222 \newenvironment{selectAll}[1][]
1223 {\begin{trivlist}\item[\hskip \labelsep\small\bfseries Select All Correct Answers:]\hfil\begi
1224     {\end{enumerate}\end{trivlist}}
1225 ⟨/classXimera⟩
```

In the future we need this to (optionally) be displayed in the problem, while the actual code lives in the solution. Here is how this could be implemented: Like the title/maketitle commands, the multiple-choice could be stored in \themultiplechoice, flip a boolean, and execute \makemultiplechoice at the \end of the problem. We should also make a command called \showchoices that will show choices in the handout.

On the web, selectAll is handled just like multipleChoice.

```
1226 ⟨∗htXimera⟩
1227 \renewenvironment{selectAll}{\refstepcounter{problem}}{}%
1228 \ConfigureEnv{selectAll}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div
1229 ⟨/htXimera⟩
```

2.12.1 Free response

freeResponse (*env.*) A freeform input box.

```
1230 ⟨∗classXimera⟩
1231 \newboolean{given} %% required for freeResponse
1232 \setboolean{given}{true} %% could be replaced by a key=value pair later if needed
1233
1234 \ifhandout
1235 \newenvironment{freeResponse}[1][false]%
1236 {%
1237 \def\givenatend{\boolean{#1}}
1238 \ifthenelse{\boolean{#1}}
1239 {% Begin then result
1240 \begin{trivlist}
1241 \item
1242 }% End then result
```

27

```
1243 {% Begin else result
1244 \setbox0\vbox\bgroup
1245 }% End else result
1246 % {}% Don't think this is doing anything? -- Jason
1247 }
1248 {%
1249 \ifthenelse{\givenatend}
1250 {% Begin then result
1251 \end{trivlist}
1252 }% End then result
1253 {% Begin else result
1254 \egroup
1255 }% End else result
1256 % {}% Don't think this is doing anything? -- Jason
1257 }
1258 \else
1259 \newenvironment{freeResponse}[1][false]%
1260 {% Environment Beginning Code
1261   \ifthenelse{\boolean{#1}}%% Could probably change this with just putting the (given) in the
1262     {% Begin then result
1263     \begin{trivlist}
1264     \item[\hskip \labelsep\bfseries Free Response (Given):\hspace{2ex}]
1265     }% End then result
1266 {% Begin else result
1267 \begin{trivlist}
1268 \item[\hskip \labelsep\bfseries Free Response:\hspace{2ex}]
1269 }% End else result
1270 }
1271 {% Environment Ending Code
1272 \end{trivlist}
1273 }
1274 \fi
1275
1276 ⟨/classXimera⟩
1277 ⟨∗htXimera⟩
1278
1279 \renewenvironment{freeResponse}{\refstepcounter{problem}}{}%
1280 \ConfigureEnv{freeResponse}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<
1281
1282 ⟨/htXimera⟩
```

2.12.2 Feedback

feedback (*env.*) An initially hidden environment that uncovers itself at an appropriate time. New Valida-
tor rewrite code added by Jason Nowell. Original code orovided by Jim Fowler Validator
is an environment designed to run a custom check on answers (usually) using javascript
code.

Define a placeholder command for validator and feedback.

```
1283 ⟨∗classXimera⟩
1284 \newcommand{\PH@Command}{}
```

Validator should take an argument and detokenize it and display it at the start of the
environment. The original Validator environment had everything framed in an mbox;
presumably to make the text look a bit nicer, although this seems redundant with `texttt`.
It shouldn't cause any harm so I have left it in for now.

```
1285 \newenvironment{validator}[1][]{
1286 \def\PH@Command{#1}% Use PH@Command to hold the content and be a target for "\expandafter" to
1287 \mbox{\texttt{\detokenize\expandafter{\PH@Command}}}% Now expand PH@Command once and then det
1288 }{}
```

First, if it's a handout, we want feedback to eat everything and then disappear entirely.
So we do this:

```
1289 \ifhandout%
```

28

```
1290 \newenvironment{feedback}
1291                 {%
1292 \setbox0\vbox\bgroup
1293         }
1294                 {%
1295 \egroup
1296         }
```

If this isn't a handout, then we want to display the Feedback by using a label, positioned and formated as a \item in a trivlist. It is important that we also detokenize the content of the optional argument, as it is likely to contain javascript or other code that latex won't be able to make sense of.

```
1297 \else
1298 \newenvironment{feedback}[1][attempt]{
1299
1300 \def\PH@Command{#1}% Use PH@Command to hold the content and be a target for "\expandafter" to
1301
1302 \begin{trivlist}% Begin the trivlist to use formating of the "Feedback" label.
1303 \item[\hskip \labelsep\small\slshape\bfseries Feedback% Format the "Feedback" label. Don't fo
1304 (\texttt{\detokenize\expandafter{\PH@Command}}):% Format (and detokenize) the condition for f
1305 \hspace{2ex}]\small\slshape% Insert some space before the actual feedback given.
1306 }{
1307 \end{trivlist}
1308 }
1309
1310 \fi
1311 ⟨/classXimera⟩
```

Feedback environments take an optional parameter (which describes when the feedback is to be provided)

```
1312 ⟨∗htXimera⟩
1313 \def\feedback{\@ifnextchar[{\@feedbackcode}{\@feedbackattempt}}
1314 \def\@feedbackattempt{\@feedbackcode[attempt]}
1315 \def\@feedbackcode[#1]{\stepcounter{identification}%
1316 \ifvmode \IgnorePar\fi \EndP%
1317 \ifthenelse{\equal{#1}{attempt}}{\HCode{<div class="feedback" data-feedback="attempt" id="fee
1318 {\ifthenelse{\equal{#1}{correct}}{\HCode{<div class="feedback" data-feedback="correct" id="fe
1319 {\HCode{<div class="feedback" data-feedback="script" id="feedback\arabic{identification}"><sc
1320 \def\endfeedback{\HCode{</div>}\IgnoreIndent}
1321 ⟨/htXimera⟩
```

2.12.3 Ungraded activities

ungraded (*env.*) The ungraded environment is used to record that certain parts of activities should not be worth points. For example, if you want to use a multipleChoice as a survey question, you can place it inside an ungraded environment. On the LATEX side, the ungraded environment does nothing.

```
1322 ⟨∗classXimera⟩
1323 \newenvironment{ungraded}{}{}
1324 ⟨/classXimera⟩
```

But on the html side, ungraded wraps the activities in a div in order to assign some weight to them for grading.

```
1325 ⟨∗htXimera⟩
1326 \renewenvironment{ungraded}{%
1327 \ifvmode \IgnorePar\fi \EndP\HCode{<div class="ungraded">}\IgnoreIndent%
1328 }{
1329 \ifvmode \IgnorePar\fi \EndP\HCode{</div>}\IgnoreIndent%
1330 }
1331 ⟨/htXimera⟩
```

2.13 Support for the web

2.13.1 MathJax support

When using mathjax, dump all the `\newcommand`s to a `.jax` file.

First, create the `.jax` file.

```
1332 ⟨∗classXimera⟩
1333 \ifdefined\HCode
1334   \else
1335     \newwrite\myfile
1336     \immediate\openout\myfile=\jobname.jax
1337 \fi
1338 ⟨/classXimera⟩
```

From `only.dtx` we must also create `prompt` on the MathJax side.

```
1339 ⟨∗classXimera⟩
1340 \ifdefined\HCode
1341   \else
1342     \immediate\write\myfile{\unexpanded{\newenvironment}{prompt}{}{}}
1343 \fi
1344 ⟨/classXimera⟩
```

Redefine newcommand appropriately.

```
1345 ⟨∗classXimera⟩
1346 \ifdefined\HCode
1347   \else
1348 \let\@oldargdef\@argdef
1349 \long\def\@argdef#1[#2]#3{%
1350 \immediate\write\myfile{\unexpanded{\newcommand}{\unexpanded{#1}}[\unexpanded{#2}]{\unexpande
1351 \@oldargdef#1[#2]{#3}%
1352 }
1353
1354 \let\@OldDeclareMathOperator\DeclareMathOperator
1355 \renewcommand{\DeclareMathOperator}[2]{\@OldDeclareMathOperator{#1}{#2}\immediate\write\myfil
1356
1357 \fi
1358 ⟨/classXimera⟩
```

Include the jax'ed newcommands

```
1359 ⟨∗cfgXimera⟩
1360 % Remove commands that use @
1361 \immediate\write18{sed -i "/@/d" \jobname.jax}
1362 % Replace ##1 with #1 and so forth
1363 \immediate\write18{sed -i "s/\string#\string#\string\\([0-9]\string\\)/\string#\string\\1/g"
1364
1365 \Configure{BVerbatimInput}{}{}{}{}
1366
1367 \Configure{verbatiminput}{}{}{}{}
1368
1369 % Instead of a nonbreaking space, use a standard space
1370 \makeatletter
1371 \def\FV@Space{\space}
1372 \makeatother
1373
1374 % Include the mathjax newcommands in a math/tex script right at the beginning of the body
1375 \Configure{BODY}{%
1376 \HCode{<body>\Hnewline}%
1377 \Tg<div class="preamble">%
1378 \Tg<script type="math/tex">%
1379 \BVerbatimInput{\jobname.jax}%
1380 \Tg</script>%
1381 \IfFileExists{\jobname.ids}{\HCode{<script type="text/javascript">\Hnewline}%
1382 \BVerbatimInput{\jobname.ids}%
1383 \HCode{</script>\Hnewline}%
1384 \Tg</div>%
```

```
1385 }{}
1386 }{%
1387 \HCode{</body>\Hnewline}%
1388 }
```

Now I just need to add a newcommand command which outputs the appropriate new-commands to MathJax; then this should be "good enough" for our purposes.

```
1389 \newtoks\eqtoks
1390 \def\AltMath#1${\eqtoks{#1}%
1391        \HCode{<script type="math/tex">\the\eqtoks</script>}$}
1392 \Configure{$}{}{}{\expandafter\AltMath}
1393
1394 \def\AltlMathI#1\){\eqtoks{#1}%
1395        \HCode{<script type="math/tex">\the\eqtoks</script>}\)}
1396 \Configure{()}{\AltlMathI}{}
1397
1398 \def\AltlDisplay#1\]{\eqtoks{#1}%
1399        \HCode{<script type="math/tex; mode=display">\the\eqtoks</script>}\]}
1400 \Configure{[]}{\AltlDisplay}{}
1401
1402 \def\AltlDisplayI#1$${\eqtoks{#1}%
1403        \HCode{<script type="math/tex; mode=display">\the\eqtoks</script>}$$}
1404 \Configure{$$}{}{}{\expandafter\AltlDisplayI}
```

Need to turn off htmlpar too, as expained in http://tex.stackexchange.com/questions/204930/vertical-spaces-in-htlatex-scriptenv

```
1405 \newcommand\VerbMath[1]{%
1406 \renewenvironment{#1}{\NoFonts}{\EndNoFonts}
1407 \ScriptEnv{#1}{\ifvmode \IgnorePar\fi \EndP\HCode{<script type="math/tex; mode=display"> \str
1408 }
```

This is a fix for the LAODE book, which uses matlabEquation as if it were an equation

```
1409 \ScriptEnv{matlabEquation}{\ifvmode \IgnorePar\fi \EndP\HCode{<script type="math/tex; mode=di
1410
1411 \VerbMath{equation}
1412 \VerbMath{equation*}
1413 \VerbMath{align}
1414 \VerbMath{align*}
1415 \VerbMath{alignat}
1416 \VerbMath{alignat*}
1417 \VerbMath{eqnarray}
1418 \VerbMath{eqnarray*}
1419
1420 ⟨/cfgXimera⟩
```

2.13.2   Semantic HTML

\textbf Using \textbf emits a <strong> tag.

```
1421 ⟨*cfgXimera⟩
1422 \Configure{textbf}{\ifvmode\ShowPar\fi\HCode{<strong>}}{\HCode{</strong>}}
1423 ⟨/cfgXimera⟩
```

\textit Using \textit or similar emits an <em> tag.

```
1424 ⟨*cfgXimera⟩
1425 \Configure{textit}{\ifvmode\ShowPar\fi\HCode{<em>}}{\HCode{</em>}}
1426 \Configure{emph}{\ifvmode\ShowPar\fi\HCode{<em>}}{\HCode{</em>}}
1427 ⟨/cfgXimera⟩
```

\texttt Using \texttt emits a <code> tag.

```
1428 ⟨*cfgXimera⟩
1429 \Configure{texttt}{\ifvmode\ShowPar\fi\HCode{<code>}}{\HCode{</code>}}
1430 ⟨/cfgXimera⟩
```

2.14   Tools

2.14.1   Suppress

suppress (*env.*)   The suppress environment is a good way to suppress output without commenting it. This way we can avoid many of the places we use environ package and this should also avoid most of the verbatim conflicts. This is code adapted from `syntonly.sty`.

```
1431 ⟨*classXimera⟩
1432 \font\dummyft@=dummy \relax
1433 \def\suppress{%
1434   \begingroup\par
1435   \parskip\z@
1436   \offinterlineskip
1437   \baselineskip=\z@skip
1438   \lineskip=\z@skip
1439   \lineskiplimit=\maxdimen
1440   \dummyft@
1441   \count@\sixt@@n
1442   \loop\ifnum\count@ >\z@
1443     \advance\count@\m@ne
1444     \textfont\count@\dummyft@
1445     \scriptfont\count@\dummyft@
1446     \scriptscriptfont\count@\dummyft@
1447   \repeat
1448   \let\selectfont\relax
1449   \let\mathversion\@gobble
1450   \let\getanddefine@fonts\@gobbletwo
1451   \tracinglostchars\z@
1452   \frenchspacing
1453   \hbadness\@M}
1454 \def\endsuppress{\par\endgroup}
1455 ⟨/classXimera⟩
```

2.14.2   The End

It seems that some of the files need to conclude with something or another.

```
1456 ⟨*htXimera⟩
1457 \Hinput{ximera}
1458 ⟨/htXimera⟩

1459 ⟨*htXourse⟩
1460 \Hinput{xourse}
1461 ⟨/htXourse⟩

1462 ⟨*cfgXimera⟩
1463 \begin{document}
1464 \EndPreamble
1465 ⟨/cfgXimera⟩
```

3   xourse.cls

```
1466 ⟨*classXourse⟩
```

notoc   The default behavior of the class is to provide a table of contents listing all activities in the course. This option will supress this table of contents.

```
1467 \newif\ifnotoc
1468 \notocfalse
1469 \DeclareOption{notoc}{\notoctrue}
```

nonewpage   The default behavior of the class is to start each activity on a new page. This option will start activities without making a new page.

```
1470 \newif\ifnonewpage
1471 \nonewpagefalse
1472 \DeclareOption{nonewpage}{\nonewpagetrue}
```

```
1473 \DeclareOption*{\PassOptionsToClass{\CurrentOption}{ximera}}
1474 \ProcessOptions\relax
1475 \LoadClass{ximera}
1476 %    \begin{macrocode}
1477 ⟨/classXourse⟩
```

3.1 Activities

The core of the xourse system. It works by redefining the document environment, thus making the \begin and \end{document} of the subfile 'transparent' to the inclusion. The redefinition of \documentclass is analogous, just having a required and an optional arguments which mean nothing to \subfile.

```
1478 ⟨∗classXourse⟩
1479 \newcommand{\skip@preamble}{%
1480     \let\document\relax\let\enddocument\relax%
1481     \newenvironment{document}{\let\input\otherinput}{}%
1482     \renewcommand{\documentclass}[2][subfiles]{}}
```

Note that the new command \subfile calls for \skip@preamble *within a group*. The changes to document and \documentclass are undone after the inclusion of the subfile.

Numbering starts a page too soon without this:

```
1483 \let\otherinput\input
```

Store usual \maketitle as \othermaketitle

```
1484 \let\othermaketitle\maketitle
```

\maketitle In a xourse file, \maketitle is redefined to give course packet title page and toc.

```
1485 \renewcommand{\maketitle}{ %
1486 \pagestyle{empty}
1487 \begin{center}
1488 ~\\ %puts space at top of page to move title down.
1489 \vskip .25\textheight
1490 \hrulefill\\
1491 \vskip 1em
1492 \bfseries{\Huge \@title} \\
1493 \hrulefill\\
1494 \vskip 3em
1495 {\Large \@author}
1496 \vskip 2em
1497 {\large \@date}
1498 \end{center}
1499 \clearpage
```

When notoc option is used, we do not include a table of contents. Otherwise we include a table of contents in every course packet.

```
1500 \ifnotoc
1501 \else
1502   \tableofcontents\clearpage
1503   \clearpage
1504 \fi
```

Switch to main pagestyle, just like a document with documentclass ximera.

```
1505 \pagestyle{main}
```

Renew maketitle to usual definition.

```
1506 \let\maketitle\othermaketitle
```

And we finish with our redefinition of \maketitle.

```
1507 }
1508 \relax
1509 ⟨/classXourse⟩
```

3.1.1 Regular activities

\activity Documents included with \activity will be included in the body of the xourse docu-
ment. Any \input commands within included ximera documents will be ignored. Any
\usepackage commands within included ximera documents will cause an error. Overlap-
ping \newcommand definitions within multiple ximera documents included simultaneously
will cause an error. The \activity command inputs the file name provided without
\documentclass, without \begin{document}/\end{document} and without any inputs
in the preamble of the included file.

```
1510 ⟨∗classXourse⟩
1511 \ifnonewpage
1512 \newcommand{\activity}[2][]{%
1513 \setkeys{activity}{#1}
1514   \renewcommand{\input}[1]{}
1515   \begingroup\skip@preamble\otherinput{#2}\endgroup\par\vspace{\topsep}
1516   \let\input\otherinput}
1517 \else
1518 \newcommand{\activity}[2][]{%
1519 \setkeys{activity}{#1}
1520   \renewcommand{\input}[1]{}
1521   \begingroup\skip@preamble\otherinput{#2}\endgroup\clearpage
1522   \let\input\otherinput}
1523 \fi
1524 \relax
1525 ⟨/classXourse⟩
```

```
1526 ⟨∗htXourse⟩
1527 \renewcommand\activity[2][]{%
1528 \ifvmode \IgnorePar\fi \EndP\HCode{<a class="activity card \activitystyle" href="#2" data-opt
1529 }
1530 ⟨/htXourse⟩
```

When running xake, we can just ignore activities

```
1531 ⟨∗classXourse⟩
1532 \ifxake
1533 \renewcommand\activity[2][]{}
1534 \fi
1535 ⟨/classXourse⟩
```

3.1.2 Practice activities

\practice Like \activity but not expecting a title.

```
1536 ⟨∗classXourse⟩
1537 \ifhandout
1538 \newcommand{\practice}[2][]{
1539 \setkeys{practice}{#1}%!!!!!
1540   \renewcommand{\input}[1]{}
1541   \begingroup\skip@preamble\otherinput{#2}\endgroup
1542   \let\input\otherinput}
1543 \else
1544 \newcommand{\practice}[2][]{\texttt{\detokenize{#2}}%% gives file name for practice
1545 \setkeys{practice}{#1}%!!!!!
1546   \renewcommand{\input}[1]{}
1547   \begingroup\skip@preamble\otherinput{#2}\endgroup
1548   \let\input\otherinput}
1549 \fi
1550 \relax
1551 ⟨/classXourse⟩
```

The practice environment does nothing, but will eventually produce exercises at the
end of an activity

```
1552 ⟨∗classXourse⟩
1553 \ifxake
1554 \renewcommand\practice[2][]{}
```

```
1555 \fi
1556 ⟨/classXourse⟩
```

I suppose it is reasonable for practice cards to NOT have an activitystyle, since the activitystyle is basically PRACTICE.

```
1557 ⟨*htXourse⟩
1558 \renewcommand\practice[2][]{%
1559   \ifvmode\IgnorePar\fi\EndP%
1560   \HCode{<a class="activity card practice" href="#2" data-options="#1">#2</a>}%
1561   \IgnoreIndent%
1562 }
1563 ⟨/htXourse⟩
```

3.2   Sectioning

\section Makes the table of contents look a bit better. This can be redefined in the preamble if you do not like the appearance. The name of a section inside an activity.

```
1564 ⟨*classXourse⟩
1565 \renewcommand*\l@section{\@dottedtocline{1}{1.5em}{4.2em}}
1566 ⟨/classXourse⟩
```

\subsection The name of a subsection inside an activity.

```
1567 ⟨*classXourse⟩
1568 \renewcommand*\l@subsection{\@dottedtocline{2}{3.8em}{4.2em}}
1569 ⟨/classXourse⟩
```

\part Xourse files can have parts. The name of a large part of a xourse.

```
1570 ⟨*htXourse⟩
1571 \newcounter{ximera@part}
1572 \setcounter{ximera@part}{0}
1573 \renewcommand\part[1]{%
1574 \stepcounter{ximera@part}%
1575 \ifvmode \IgnorePar\fi \EndP%
1576 %\HCode{<h1 id="part\arabic{ximera@part}" class="card part">}#1\HCode{</h1>}% makes cards dis
1577 \HCode{<h1 id="part\arabic{ximera@part}" class="card part">#1</h1>}%
1578 \IgnoreIndent%
1579 }
1580 ⟨/htXourse⟩
```

\paragraph Paragraph commands emit spans. A small heading.

```
1581 ⟨*cfgXimera⟩
1582 \renewcommand{\paragraph}[1]{%
1583   \HCode{<span class="paragraphHead">}%
1584   #1%
1585   \HCode{</span>}\par\IgnorePar}
1586 ⟨/cfgXimera⟩
```

\subparagraph An even smaller heading.

```
1587 ⟨*cfgXimera⟩
1588 \renewcommand{\subparagraph}[1]{%
1589   \HCode{<span class="subparagraphHead">}%
1590   #1%
1591   \HCode{</span>}\par\IgnorePar}
1592 ⟨/cfgXimera⟩
```

3.3   Grading by points

graded (*env.*) The `graded` environment does nothing in latex, but in html, it wraps the activities in a div in order to assign some weight to them for grading.

```
1593 ⟨*classXourse⟩
1594 \newenvironment{graded}[1]{}{}
1595 ⟨/classXourse⟩
```

So indeed this environment in html wraps the activities in a div in order to assign some number of points to them.

```
1596 ⟨*htXourse⟩
1597 \renewenvironment{graded}[1]{%
1598 \ifvmode \IgnorePar\fi \EndP\HCode{<div class="graded" data-weight="#1">}\IgnoreIndent%
1599 }{
1600 \ifvmode \IgnorePar\fi \EndP\HCode{</div>}\IgnoreIndent%
1601 }
1602 ⟨/htXourse⟩
```

3.4   Logos

\logo A logo for the xourse.

```
1603 ⟨*classXourse⟩
1604 \newcommand*{\logo}[1]{%
1605   \ifx\@onlypreamble\@notprerr
1606     \ClassError{xourse}{logo can only be used in the preamble}
1607       {Move your logo command to the preamble}
1608   \else %
1609     \IfFileExists{#1}%
1610       {\gdef\xourse@logo{#1}}%
1611       {\ClassError{xourse}{logo file does not exist}
1612         {To use logo, make sure that the referenced image file exists}}%
1613   \fi%
1614 }
1615
1616 ⟨/classXourse⟩
```

The xourse logo is an `og:image` in the opengraph taxonomy.

```
1617 ⟨*htXourse⟩
1618 \Configure{@HEAD}{%
1619   \HCode{<meta name="og:image" content="}%
1620 \ifdefined\xourse@logo%
1621   \xourse@logo%
1622 \fi%
1623 \HCode{" />\Hnewline}}%
1624 ⟨/htXourse⟩
```