ximera — Simultaneously write print and online interactive materials.*

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

Released 2024/05/12

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

Ximera, pronounced "chimera," (**X**imera: **I**nteractive, **M**athematics, **ER**esources, for **A**ll) is an open-source platform that provides tools for authoring and publishing (PDF and Online), open-source, interactive educational content, such as textbooks, assessments, and online courses. The Ximera document class provides the following features:

Formatting for different domains The Ximera document class provides built-in support for formatting documents in both PDF and online formats, which can be a big time-saver for authors. Additionally, it allows for the simultaneous creation of solution manuals and teaching editions, which can be especially useful for educators.

Compiling individually or as a whole With the Ximera document class, authors can easily compile individual documents or an entire collection of documents. This flexibility can be helpful when making changes to specific documents without having to re-compile the entire collection. Moreover, this allows an author to share large portions of a text with another, with minimal changes.

Interactive content The Ximera document class allows for the inclusion of interactive content, such as answer boxes that are validated by a client-side computer algebra system. Additionally, it allows for the embedding of YouTube videos, Desmos graphs, and GeoGebra interactives.

All content displayed By default, the Ximera document class displays all content to the author. This means the author see what the students see, along with answers and solutions, and links (that can be checked) to various interactive elements (when deployed, the interactive elements are truly embedded). This can be especially helpful for catching errors or inconsistencies in the content.

Online examples can be found at

https://go.osu.edu/ximera-examples

---

*This file describes version v1.5.1, last revised 2024/05/12.

2 ximera.cls

1 ⟨∗classXimera⟩

2 `\newif\ifnumberedProblems`
3 `\numberedProblemsfalse% Default to no numbers, as that was previous behavior.`
4 `\DeclareOption{onlineProblemNumbers}{\numberedProblemstrue}`

5 ⟨/classXimera⟩

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.

6 ⟨∗classXimera⟩

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."

7 `\newif\ifhandout`
8 `\handoutfalse`
9 `\DeclareOption{handout}{\handouttrue}`

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

10 `\newif\ifnoauthor`
11 `\noauthorfalse`
12 `\DeclareOption{noauthor}{\noauthortrue}`

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.

13 `\newif\ifnooutcomes`
14 `\nooutcomesfalse`
15 `\DeclareOption{nooutcomes}{\nooutcomestrue}`

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

16 `\newif\ifinstructornotes`
17 `\instructornotesfalse`
18 `\DeclareOption{instructornotes}{\instructornotestrue}`

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

19 `\DeclareOption{noinstructornotes}{\instructornotestrue}`

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

20 `\newif\ifhints`
21 `\hintsfalse`
22 `\DeclareOption{hints}{\hintstrue}`

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

23 `\newif\ifnewpage`
24 `\newpagefalse`
25 `\DeclareOption{newpage}{\newpagetrue}`

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

26 `\newif\ifnumbers`
27 `\numbersfalse`
28 `\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.

```
29 \newif\ifwordchoicegiven
30 \wordchoicegivenfalse
31 \DeclareOption{wordchoicegiven}{\wordchoicegiventrue}
32 \newif\iffirstinlinechoice% Support for other wordchoice command contents.
33 \firstinlinechoicetrue

34
35 \newif\ifxake
36 \xakefalse
37 \DeclareOption{xake}{\xaketrue}
38
39 \newif\iftikzexport
40 \tikzexportfalse
41 \DeclareOption{tikzexport}{%
42   \tikzexporttrue%
43   \handoutfalse%
44   \numbersfalse%
45   \newpagefalse%
46   \hintsfalse%
47   \nooutcomesfalse%
48 }
49
50 \DeclareOption*{%
51   \PassOptionsToClass{\CurrentOption}{article}%
52 }
53 \ProcessOptions\relax
54 \LoadClass{article}
55
56 \ifdefined\HCode
57   \xaketrue%
58   \tikzexporttrue%
59   \handoutfalse%
60   \numbersfalse%
61   \newpagefalse%
62   \hintsfalse%
63   \nooutcomesfalse%
64 \fi
65 ⟨/classXimera⟩
66 ⟨∗classXimera⟩
```

2.2 Loading packages

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

```
67 \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.

```
68 \RequirePackage[inline]{enumitem}
69 \RequirePackage[pagestyles]{titlesec}
70 \RequirePackage{titletoc}
71 \RequirePackage{titling}
72 \RequirePackage{url}
73 \RequirePackage[table]{xcolor}
74 \RequirePackage{tikz}
75 \RequirePackage{pgfplots}
76 \usepgfplotslibrary{groupplots}
77 \usetikzlibrary{calc}
78 \RequirePackage{fancyvrb}
```

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

```
79 \RequirePackage{forloop}
```

Now we load even more packages.

```
80 \RequirePackage{environ}% Included to allow saving of environment contents. This does *not* 
81 \RequirePackage{amssymb}% Included to have access to math typeset.
82 \RequirePackage{amsmath}% Included to have access to math typeset.
83 \RequirePackage{amsthm}%  Included to have access to math typeset.
84 \RequirePackage{xifthen}% http://ctan.org/pkg/xifthen
85 \RequirePackage{multido}% http://ctan.org/pkg/multido
86 \RequirePackage{listings} %% is this required???
87
88 \RequirePackage{xkeyval}
89
90 \RequirePackage{currfile}
91 \RequirePackage{comment}
92 ⟨/classXimera⟩
```

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

```
93 ⟨*classXimera⟩
94 \RequirePackage{gettitlestring}
95 \RequirePackage{nameref}
96 \RequirePackage{epstopdf}
97 ⟨/classXimera⟩
```

2.3   Page setup

We want non-indented spaced-out paragraphs.

```
98 ⟨*classXimera⟩
99 \setlength{\parindent}{0pt}
100 \setlength{\parskip}{5pt}
101 ⟨/classXimera⟩
```

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

```
102 ⟨*classXimera⟩
103 \oddsidemargin 62pt
104 \evensidemargin 62pt
105 \textwidth 345pt
106 \headheight 14pt
107 ⟨/classXimera⟩
```

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

```
108 ⟨*cfgXimera⟩
109 \Preamble{xhtml,mathjax}
110
111 % We don't want to translate font suggestions with ugly wrappers like
112 % <span class="cmti-10"> for italic text
113 \NoFonts
114
115 % Don't output xml version tag
116 % \Configure{VERSION}{}
117
118 % Output HTML5 doctype instead of the default for HTML4
119 % \Configure{DOCTYPE}{\HCode{<!doctype html>\Hnewline}}
120
121 % Custom page opening
122 % \Configure{HTML}{\HCode{<html lang="en">\Hnewline}}{\HCode{\Hnewline</html>}}
123
124 % Reset <head>, aka delete all default boilerplate; alternatively set up new content
125 % \Configure{@HEAD}{\HCode{<meta name="generator" content="TeX4ht (http://www.cse.ohio-state.
126 \Configure{@HEAD}{\HCode{<meta name="ximera" content="version 2.5.1" />\Hnewline}}
127 \Configure{@HEAD}{\HCode{<link href="https://ximera.osu.edu/public/stylesheets/standalone.css
128 \Configure{@HEAD}{\HCode{<script type="text/javascript" async src="https://ximera.osu.edu/pub
129
```

```
130 % OVERWRITE css in ximera-server (to be removed whenever this has been fixed in the server;
131 \catcode`\%=11
132 \Configure{@BODY}{\HCode{<style>
133 .activity-body pre {
134     white-space: pre;
135     background-color: lightgray;
136 }
137 .xmyoutube {
138     aspect-ratio: 16/9;
139     min-width: 75%;
140 }
141 .image-environment img {
142     width: unset;
143 }
144 </style>\Hnewline}}
145 \catcode`\%=14
146
147 ⟨/cfgXimera⟩
```

Disable certain ligatures in HTML.

```
148 ⟨∗htXimera⟩
149 \usepackage{microtype}
150 \DisableLigatures[f]{encoding=*}
151 ⟨/htXimera⟩
```

I am not sure what this does.

```
152 ⟨∗htXimera⟩
153 \NewEnviron{html}{\HCode{\BODY}}
154 ⟨/htXimera⟩
```

2.4 Structure

2.4.1 Macros

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

```
155 ⟨∗classXimera⟩
156 \everymath{\displaystyle}
157 ⟨/classXimera⟩
```

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

```
158 ⟨∗classXimera⟩
159 \let\prelim\lim
160 \renewcommand{\lim}{\displaystyle\prelim}
161 ⟨/classXimera⟩
```

2.4.2 Theorem and theorem-like environments

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

```
162 ⟨∗htXimera⟩
163 \newcommand{\ConfigureTheoremEnv}[1]{%
164 \renewenvironment{#1}[1][]{\refstepcounter{problem}%
165 \ifthenelse{\equal{##1}{}}{}{%
166   \HCode{<span class="theorem-like-title">}##1\HCode{</span>}%
167 }}{}
168 \ConfigureEnv{#1}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div class=
169 }
170 ⟨/htXimera⟩
171 ⟨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

```
172 ⟨classXimera⟩        \newtheorem{theorem}{\GetTranslation{theorem}}
173 ⟨htXimera⟩           \ConfigureTheoremEnv{theorem}
```

algorithm (*env.*)   Algorithm

174 ⟨classXimera⟩   \newtheorem{algorithm}{\GetTranslation{algorithm}}
175 ⟨htXimera⟩   \ConfigureTheoremEnv{algorithm}

axiom (*env.*)   Axiom

176 ⟨classXimera⟩   \newtheorem{axiom}{\GetTranslation{axiom}}
177 ⟨htXimera⟩   \ConfigureTheoremEnv{axiom}

claim (*env.*)   Claim

178 ⟨classXimera⟩   \newtheorem{claim}{\GetTranslation{claim}}
179 ⟨htXimera⟩   \ConfigureTheoremEnv{claim}

conclusion (*env.*)   Conclusion

180 ⟨classXimera⟩   \newtheorem{conclusion}{\GetTranslation{conclusion}}
181 ⟨htXimera⟩   \ConfigureTheoremEnv{conclusion}

condition (*env.*)   Condition

182 ⟨classXimera⟩   \newtheorem{condition}{\GetTranslation{condition}}
183 ⟨htXimera⟩   \ConfigureTheoremEnv{condition}

conjecture (*env.*)   Conjecture

184 ⟨classXimera⟩   \newtheorem{conjecture}{\GetTranslation{conjecture}}
185 ⟨htXimera⟩   \ConfigureTheoremEnv{conjecture}

corollary (*env.*)   Corollary

186 ⟨classXimera⟩   \newtheorem{corollary}{\GetTranslation{corollary}}
187 ⟨htXimera⟩   \ConfigureTheoremEnv{corollary}

criterion (*env.*)   Criterion

188 ⟨classXimera⟩   \newtheorem{criterion}{\GetTranslation{criterion}}
189 ⟨htXimera⟩   \ConfigureTheoremEnv{criterion}

definition (*env.*)   Definition

190 ⟨classXimera⟩   \newtheorem{definition}{\GetTranslation{definition}}
191 ⟨htXimera⟩   \ConfigureTheoremEnv{definition}

example (*env.*)   Example

192 ⟨classXimera⟩   \newtheorem{example}{\GetTranslation{example}}
193 ⟨htXimera⟩   \ConfigureTheoremEnv{example}

explanation (*env.*)   Explanation

194 ⟨classXimera⟩   \newtheorem*{explanation}{\GetTranslation{explanation}}
195 ⟨htXimera⟩   \ConfigureTheoremEnv{explanation}

fact (*env.*)   Fact

196 ⟨classXimera⟩   \newtheorem{fact}{\GetTranslation{fact}}
197 ⟨htXimera⟩   \ConfigureTheoremEnv{fact}

lemma (*env.*)   Lemma

198 ⟨classXimera⟩   \newtheorem{lemma}{\GetTranslation{lemma}}
199 ⟨htXimera⟩   \ConfigureTheoremEnv{lemma}

formula (*env.*)   Formula

200 ⟨classXimera⟩   \newtheorem{formula}{\GetTranslation{formula}}
201 ⟨htXimera⟩   \ConfigureTheoremEnv{formula}

idea (*env.*)   Idea

202 ⟨classXimera⟩   \newtheorem{idea}{\GetTranslation{idea}}
203 ⟨htXimera⟩   \ConfigureTheoremEnv{idea}

notation (*env.*)   Notation

204 ⟨classXimera⟩   \newtheorem{notation}{\GetTranslation{notation}}
205 ⟨htXimera⟩   \ConfigureTheoremEnv{notation}

model (*env.*)   Model

206 ⟨classXimera⟩   \newtheorem{model}{\GetTranslation{model}}
207 ⟨htXimera⟩   \ConfigureTheoremEnv{model}

observation (*env.*)   Observation

208 ⟨classXimera⟩   \newtheorem{observation}{\GetTranslation{observation}}
209 ⟨htXimera⟩   \ConfigureTheoremEnv{observation}

proposition (*env.*)	Proposition	
	210 ⟨classXimera⟩	\newtheorem{proposition}{\GetTranslation{proposition}}
	211 ⟨htXimera⟩	\ConfigureTheoremEnv{proposition}
paradox (*env.*)	Paradox	
	212 ⟨classXimera⟩	\newtheorem{paradox}{\GetTranslation{paradox}}
	213 ⟨htXimera⟩	\ConfigureTheoremEnv{paradox}
procedure (*env.*)	Procedure	
	214 ⟨classXimera⟩	\newtheorem{procedure}{\GetTranslation{procedure}}
	215 ⟨htXimera⟩	\ConfigureTheoremEnv{procedure}
remark (*env.*)	Remark	
	216 ⟨classXimera⟩	\newtheorem{remark}{\GetTranslation{remark}}
	217 ⟨htXimera⟩	\ConfigureTheoremEnv{remark}
summary (*env.*)	Summary	
	218 ⟨classXimera⟩	\newtheorem{summary}{\GetTranslation{summary}}
	219 ⟨htXimera⟩	\ConfigureTheoremEnv{summary}
template (*env.*)	Template	
	220 ⟨classXimera⟩	\newtheorem{template}{\GetTranslation{template}}
	221 ⟨htXimera⟩	\ConfigureTheoremEnv{template}
warning (*env.*)	Warning	
	222 ⟨classXimera⟩	\newtheorem{warning}{\GetTranslation{warning}}
	223 ⟨htXimera⟩	\ConfigureTheoremEnv{warning}

2.4.3 Enumerate fixes

Make enumerate use a letter

```
224 ⟨*classXimera⟩
225 \renewcommand{\theenumi}{\textup{(\alph{enumi})}}
226 \renewcommand{\labelenumi}{\theenumi}
227 \renewcommand{\theenumii}{\textup{(\roman{enumii})}}
228 \renewcommand{\labelenumii}{\theenumii}
229 ⟨/classXimera⟩
```

2.4.4 Proofs

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

```
230 ⟨*classXimera⟩
231 \renewcommand{\qedsymbol}{$\blacksquare$}
232 \renewenvironment{proof}[1][\proofname]
233   {\begin{trivlist}\item[\hskip \labelsep \itshape \bfseries #1{}\hspace{2ex}]]
234 {\qed\end{trivlist}}
235 ⟨/classXimera⟩
236 ⟨*htXimera⟩
237     % Mmm, (why) do we want/need this ...?
238     \ConfigureTheoremEnv{proof}
239 \ConfigureEnv{proof}{\ifvmode\IgnorePar\fi\EndP\HCode{<div class="proof">}}
240 \ConfigureList{trivlist}{\ifvmode\IgnorePar\fi\EndP}{}{}{}
241 }{\ifvmode\IgnorePar\fi\EndP\HCode{</div>}}{}{}
242 ⟨/htXimera⟩
```

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

```
243 ⟨*classXimera⟩
```

```
244 \newcommand{\hang}{% top theorem decoration
245   \begingroup%
246   \setlength{\unitlength}{.005\linewidth}% \linewidth/200
247 \begin{picture}(0,0)(1.5,0)%
248   \linethickness{1pt} \color{black!50}%
249   \put(-3,2){\line(1,0){206}}% Top line
250   \multido{\iA=2+-1,\iB=50+-10}{5}{% Top hangs
251 \color{black!\iB}%
252 \put(-3,\iA){\line(0,-1){1}}% Top left hang
253 %\put(203,\iA){\line(0,-1){1}}% Top right hang
254   }%
255 \end{picture}%
256   \endgroup%
257 }%
258 \newcommand{\hung}{% bottom theorem decoration
259   \nobreak
260   \begingroup%
261 \setlength{\unitlength}{.005\linewidth}% \linewidth/200
262 \begin{picture}(0,0)(1.5,0)%
263   \linethickness{1pt} \color{black!50}%
264   \put(60,0){\line(1,0){143}}% Bottom line
265   \multido{\iA=0+1,\iB=50+-10}{5}{% Bottom hangs
266 \color{black!\iB}%
267 %\put(-3,\iA){\line(0,1){1}}% Bottom left hang
268 \put(203,\iA){\line(0,1){1}}% Bottom right hang
269 \put(\iB,0){\line(60,0){10}}% Left fade out
270   }%
271 \end{picture}%
272   \endgroup%
273 }%
```

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.

```
274 \MakeCounter{Iteration@probCnt}
275 \MakeCounter{problem}
276 \newcommand{\problemNumber}{
277 % First we determine if we have a counter for this question depth level.
278 \ifcsname c@depth\Roman{problem@Depth}Count\endcsname% Check to see if counter exists
279 %If so, do nothing.
280   \else
281 %If not, create it.
282 \expandafter\newcounter{depth\Roman{problem@Depth}Count}
283 \expandafter\setcounter{depth\Roman{problem@Depth}Count}{0}
284 \fi
285
286 \expandafter\stepcounter{depth\Roman{problem@Depth}Count}
287 \arabic{depthICount}% The first problem depth, what use to be |\theproblem|.
288
289 \forloop{Iteration@probCnt}{2}{\arabic{Iteration@probCnt} < \numexpr \value{problem@Depth} +
290 .\expandafter\arabic{depth\Roman{Iteration@probCnt}Count}% Get the problem number of the next
291 }
292 }
293 %%%%% Configure various problem environment commands
294 \Make@Counter{problem@Depth}
295 %%%% Configure environments start content
296 \newcommand{\problemEnvironmentStart}[2]{%
297 \stepcounter{problem@Depth}% Started a problem, so we've sunk another problem layer.
298 \def\spaceatend{#1}%
299 \begin{trivlist}%
300 \item[\hskip\labelsep\sffamily\bfseries\GetTranslation{#2} \problemNumber% Determine the cor
301 ]%
302 \slshape
303 }
```

```
304 %%%% Configure environments end content %%%%%
305 \newcommand{\problemEnvironmentEnd}{%This configures all the end content for a problem.
306 \stepcounter{problem@Depth}
307 \ifcsname c@depth\Roman{problem@Depth}Count\endcsname
308 \expandafter\ifnum\expandafter\value{depth\Roman{problem@Depth}Count}>0
309 \expandafter\setcounter{depth\Roman{problem@Depth}Count}{0}
310 \fi
311 \fi
312 \addtocounter{problem@Depth}{-2}% Exited a problem so we've exited a problem layer. Need -2 b
313 \ifhandout
314 \ifnewpage
315 \newpage
316 \fi
317 \fi
318 \end{trivlist}
319 }
320 %% Add a simple command that handles all the problem creation aspects:
321 \newcommand{\createProblemEnv}[2]{% This is a nice command to define a new problem-like envir
322 \newenvironment{#1}[1][2in]%
323 {%Env start code
324 \problemEnvironmentStart{#1}{#2}
325 }
326 {%Env end code
327 \problemEnvironmentEnd
328 }
329 }
330
331 %%%% Now populate the old environment names
332 %
333 % Old environments were "problem", "exercise", "exploration", and "question".
334 % Note that you can add content to the start/end code on top of these base code pieces if you
335 %
336 % These definitions will be overwritten in ximera.4ht !
337
338 \createProblemEnv{problem}{Problem}
339 \createProblemEnv{exercise}{Exercise}
340 \createProblemEnv{exploration}{Exploration}
341 \createProblemEnv{question}{Question}

342 ⟨/classXimera⟩

343 ⟨*htXimera⟩

344 \newcounter{identification}
345 \setcounter{identification}{0}
346 \newcommand{\ConfigureQuestionEnv}[2]{%
347 \renewenvironment{#1}{
348   }
349   {
350   }%
351   \ConfigureEnv{#1}
352   {
353 %    \ifnumberedProblems% The code below is all to generate online problem numbering if optio
354 %    \stepcounter{problem@Depth}% Started a problem, so we've sunk another problem layer.
355 %    \ifcsname c@depth\Roman{problem@Depth}Count\endcsname
356 %    \else
357 %      \expandafter\newcounter{depth\Roman{problem@Depth}Count}
358 %      \expandafter\setcounter{depth\Roman{problem@Depth}Count}{0}
359 %    \fi
360 %    \expandafter\stepcounter{depth\Roman{problem@Depth}Count}
361 %    \def\problemNumDisp{
362 %    \arabic{depthICount}% Top Level Problem Number: X.1.1.1.1 Number.
363 %    \ifcsname c@depthIICount\endcsname\ifnum\value{problem@Depth}>1 .\arabic{depthIICount}\f
364 %    \ifcsname c@depthIIICount\endcsname\ifnum\value{problem@Depth}>2 .\arabic{depthIIICount]
365 %    \ifcsname c@depthIVCount\endcsname\ifnum\value{problem@Depth}>3 .\arabic{depthIVCount}\f
366 %    \ifcsname c@depthVCount\endcsname\ifnum\value{problem@Depth}>4 .\arabic{depthVCount}\fi%
```

9

```
367 %    \fi\fi\fi\fi
368 %        }
369 %  \else
370     \def\problemNumDisp{}% Otherwise don't display a problem number.
371 %  \fi
372     \stepcounter{identification}
373     \ifvmode
374     \IgnorePar
375     \fi
376 \EndP
377 \HCode{<div role="article" class="problem-environment #1" id="problem\arabic{identification}'
378 }
379 {
380 \stepcounter{problem@Depth}
381 \ifcsname c@depth\Roman{problem@Depth}Count\endcsname
382 \expandafter\ifnum\expandafter\value{depth\Roman{problem@Depth}Count}>0
383 \expandafter\setcounter{depth\Roman{problem@Depth}Count}{0}
384 \fi
385 \fi
386 \addtocounter{problem@Depth}{-2}% Exited a problem so we've exited a problem layer. Need -2 k
387 \ifvmode
388 \IgnorePar
389 \fi
390 \EndP
391 \HCode{</div>}\IgnoreIndent
392 }{}{}%
393 }
394
395 \ConfigureQuestionEnv{problem}{Problem}
396 \ConfigureQuestionEnv{exercise}{Exercise}
397 \ConfigureQuestionEnv{question}{Question}
398 \ConfigureQuestionEnv{exploration}{Exploration}
399
400 \ifdefined\xmNotHintAsExpandable
401    \ConfigureQuestionEnv{hint}{hint} % 2024: hint is no longer a 'question-environment'.
402 \fi
403 ⟨/htXimera⟩
```

2.4.6 Hints

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

```
404 ⟨*classXimera⟩
```

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

```
405 \newcounter{hintLevel}
406 \setcounter{hintLevel}{0}
```

Create an empty shell to renew

```
407 \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.

```
408 \renewenvironment{hint}
409 {
410   \ifhandout
411     \setbox0\vbox\bgroup
412   \else
413     \begin{trivlist}\item[\hskip \labelsep\small\slshape\bfseries \GetTranslation{hint}:\hspa
414     \small\slshape
415   \fi
416   \stepcounter{hintLevel}
417 }
418 {
```

```
419   \ifhandout
420     \egroup\ignorespacesafterend
421   \else
422     \end{trivlist}
423   \fi
424   \addtocounter{hintLevel}{-1}
425 }
426
427 \ifhints
428 \renewenvironment{hint}{
429   \begin{trivlist}\item[\hskip \labelsep\small\slshape\bfseries \GetTranslation{hint}:\hspace
430   \small\slshape
431 }
432 {
433   \end{trivlist}
434 }
435 \fi
436
437 ⟨/classXimera⟩
```

2.4.7 Solution

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

```
438 ⟨∗classXimera⟩
439 %% solution environment
440 \ifhandout % what follows is handout behavior
441 \newenvironment{solution}%
442          {%
443     \setbox0\vbox\bgroup
444          }
445                  {%
446     \egroup
447          }
448 \else
449 \newenvironment{solution}%
450          {%
451     \begin{trivlist}
452     \item[\hskip \labelsep\bfseries \GetTranslation{Solution}:\hspace{2ex}]
453          }
454          % %% line at the bottom}
455          {
456     \end{trivlist}
457     % (202410: no longer \par\addvspace{.5ex}\nobreak\noindent\hung
458          }
459 \fi
460
461
462
463 ⟨/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.

```
464 ⟨∗classXimera⟩
465 \DefineVerbatimEnvironment{code}{Verbatim}{numbers=left,frame=lines,label=Code,labelposition=
466 ⟨/classXimera⟩
```

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

```
467 ⟨∗classXimera⟩
468 \DefineVerbatimEnvironment{python}{Verbatim}{numbers=left,frame=lines,label=Python,labelposit
469 ⟨/classXimera⟩
```

11

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

```
470 ⟨*classXimera⟩
471 \DefineVerbatimEnvironment{javascriptCode}{Verbatim}{numbers=left,frame=lines,label=JavaScri
472 ⟨/classXimera⟩
473 ⟨*cfgXimera⟩
474 \renewenvironment{javascriptCode}{\NoFonts}{\EndNoFonts}
475 \ScriptEnv{javascriptCode}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<di
476 ⟨/cfgXimera⟩
```

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

```
477 %%%<*cfgXimera>
478 %%\ConfigureEnv{verbatim}{\ifvmode\IgnorePar\fi\EndP\HCode{<pre style="white-space: pre; back
479 %%\ConfigureEnv{lstlisting}{\ifvmode\IgnorePar\fi\EndP\HCode{<pre>}}{\ifvmode\IgnorePar\fi\En
480 %%%</cfgXimera>
481 %%
```

2.4.9 Dialogues

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

```
482 ⟨*classXimera⟩
483 \newenvironment{dialogue}{%
484     \renewcommand\descriptionlabel[1]{\hspace{\labelsep}\textbf{##1:}}
485     \begin{description}%
486 }{%
487     \end{description}%
488 }
489 ⟨/classXimera⟩
```

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

```
490 ⟨*htXimera⟩
491 \renewenvironment{dialogue}{\begin{description}}{\end{description}}
492
493 \ConfigureList{dialogue}%
494     {\EndP\HCode{<dl \a:LRdir class="dialogue">}%
495         \PushMacro\end:itm
496 \global\let\end:itm=\empty}
497     {\PopMacro\end:itm \global\let\end:itm \end:itm
498 \EndP\HCode{</dd></dl>}\ShowPar}
499     {\end:itm \global\def\end:itm{\EndP\Tg</dd>}\HCode{<dt
500         class="actor">}\bgroup \bf}
501     {\egroup\EndP\HCode{</dt><dd\Hnewline class="speech">}}
502 ⟨/htXimera⟩
```

2.4.10 Instructor notes

```
503 ⟨*classXimera⟩
504
505 %% instructor intro/instructor notes
506 %%
507 \ifhandout % what follows is handout behavior
508 \ifinstructornotes
509 \newenvironment{instructorIntro}%
510         {%
511     \begin{trivlist}
512     \item[\hskip \labelsep\bfseries \GetTranslation{Instructor Introduction}:\hspace{2ex}]
513     }
514         % %% line at the bottom}
515         {
516     \end{trivlist}
517     \par\addvspace{.5ex}\nobreak\noindent\hung
518         }
519 \else
520 \newenvironment{instructorIntro}%
```

```
521              {%
522    \setbox0\vbox\bgroup
523              }
524              {%If this mysteriously starts breaking
525                         % remove \ignorespacesafterend
526    \egroup\ignorespacesafterend
527              }
528                       \fi
529  \else% for handout, so what follows is default
530  \ifinstructornotes
531    \newenvironment{instructorIntro}%
532            {%
533                \setbox0\vbox\bgroup
534            }
535      {%
536      \egroup
537      }
538                   \else
539            \newenvironment{instructorIntro}%
540      {%
541      \begin{trivlist}
542      \item[\hskip \labelsep\bfseries \GetTranslation{Instructor Introduction}:\hspace{2ex}]
543      }
544      % %% line at the bottom}
545      {
546      \end{trivlist}
547      \par\addvspace{.5ex}\nobreak\noindent\hung
548      }
549                   \fi
550  \fi
551
552
553
554
555  %% instructorNotes environment
556  \ifhandout % what follows is handout behavior
557  \ifinstructornotes
558  \newenvironment{instructorNotes}%
559          {%
560      \begin{trivlist}
561      \item[\hskip \labelsep\bfseries \GetTranslation{Instructor Notes}:\hspace{2ex}]
562          }
563          % %% line at the bottom}
564          {
565      \end{trivlist}
566      \par\addvspace{.5ex}\nobreak\noindent\hung
567          }
568          \else
569    \newenvironment{instructorNotes}%
570            {%
571                \setbox0\vbox\bgroup
572            }
573      {%
574        \egroup
575      }
576                   \fi
577  \else% for handout, so what follows is default
578  \ifinstructornotes
579  \newenvironment{instructorNotes}%
580          {%
581      \setbox0\vbox\bgroup
582          }
583          {%
```

```
584     \egroup
585         }
586         \else
587         \newenvironment{instructorNotes}%
588             {%
589         \begin{trivlist}
590         \item[\hskip \labelsep\bfseries \GetTranslation{Instructor Notes}:\hspace{2ex}]
591             }
592         % %% line at the bottom}
593             {
594         \end{trivlist}
595         \par\addvspace{.5ex}\nobreak\noindent\hung
596             }
597                 \fi
598                     \fi
599
600 ⟨/classXimera⟩
```

2.4.11 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.

```
601 %\iftikzexport\else\RequirePackage[framemethod=TikZ]{mdframed}\fi
```

foldable (*env.*) Does it fold?

```
602 ⟨*classXimera⟩
603
604 \colorlet{textColor}{black} % since textColor is referenced below
605 \colorlet{background}{white} % since background is referenced below
606
607 % The core environments. Find results in 4ht file.
608 %% pretty-foldable
609 %\iftikzexport
610 \newenvironment{foldable}{%
611 }{%
612 }
613 %\else
614 %\renewmdenv[
615 %   font=\upshape,
616 %   outerlinewidth=3,
617 %   topline=false,
618 %   bottomline=false,
619 %   leftline=true,
620 %   rightline=false,
621 %   leftmargin=0,
622 %   innertopmargin=0pt,
623 %   innerbottommargin=0pt,
624 %   skipbelow=\baselineskip,
625 %   linecolor=textColor!20!white,
626 %   fontcolor=textColor,
627 %   backgroundcolor=background
628 %]{foldable}%
629 %\fi
630
631 %% pretty-expandable
632 %\iftikzexport
633 %% Overwritten in .4ht, but probably also in accordion!
634 \ifdefined\xmNotExpandableAsAccordion
635 \newenvironment{expandable}{}{}
636 \else
637 \newenvironment{expandable}[2]{}{}
638 \fi
639 %\else
```

```
640 %\newmdenv[
641 %  font=\upshape,
642 %  outerlinewidth=3,
643 %  topline=false,
644 %  bottomline=false,
645 %  leftline=true,
646 %  rightline=false,
647 %  leftmargin=0,
648 %  innertopmargin=0pt,
649 %  innerbottommargin=0pt,
650 %  skipbelow=\baselineskip,
651 %  linecolor=black,
652 %]{expandable}%
653 %\fi
654
655 \newcommand{\unfoldable}[1]{#1}
656
657 ⟨/classXimera⟩
```

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

```
658 ⟨*htXimera⟩
659 \renewenvironment{foldable}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<d
660
661 \ifdefined\xmNotExpandableAsAccordion
662 \renewenvironment{expandable}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode
663 \fi
664
665 \renewcommand{\unfoldable}[1]{\HCode{<span class="unfoldable">}#1\HCode{</span>}}
666 ⟨/htXimera⟩
```

2.4.12   Leashes

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

```
667 ⟨*classXimera⟩
668
669 \newenvironment{leash}[1]{%
670 }{%
671 }
672
673
674 ⟨/classXimera⟩

675 ⟨*htXimera⟩
676 \renewenvironment{leash}[1]{\ifvmode \IgnorePar\fi \EndP\HCode{<div style="overflow: auto; he
677 ⟨/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.

```
678 ⟨*classXimera⟩
679 \newcommand{\license}{\excludecomment}
680 ⟨/classXimera⟩
```

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

```
681 ⟨*classXimera⟩
682 \newcommand{\acknowledgement}{\excludecomment}
683 ⟨/classXimera⟩
```

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

```
684 ⟨*classXimera⟩
```

15

```
685 \renewcommand{\tag}{\excludecomment}
686 ⟨/classXimera⟩
```

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

```
687 ⟨*htXourse⟩
688 % Mark this as a xourse file
689 \Configure{@HEAD}{\HCode{<meta name="description" content="xourse" />\Hnewline}}
690 ⟨/htXourse⟩
```

2.5.2 Abstract

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

```
691 ⟨*classXimera⟩
692 \let\abstract\relax
693 \let\endabstract\relax
694 % Use of environ package, may want to find a better way.
695 % see the messing around with \theabstract in title.dtx ... Is this really needed/wanted?
696 \NewEnviron{abstract}{\protected@xdef\theabstract{\BODY}}
697 ⟨/classXimera⟩
```

The abstract has been stored in **\theabstract** and should be emitted as a div. The code below is required for the abstract to show online.

```
698 ⟨*cfgXimera⟩
699 \ifvmode\IgnorePar\fi\EndP
700 \ConfigureEnv{abstract}{\ifvmode\IgnorePar\fi\EndP\HCode{\Hnewline<div class="abstract">}\par
701 ⟨/cfgXimera⟩
```

```
702 ⟨*htXimera⟩
703 \RenewEnviron{abstract}{\BODY}
704 ⟨*htXimera⟩
```

2.5.3 Titles and authors

2.5.4 Authors

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

```
705 ⟨*classXimera⟩
706 \let\@emptyauthor\@author
707 \def\author#1{\gdef\@author{#1}}
708 \def\@author{\@latex@warning@no@line{No \noexpand\author given}}
709 ⟨/classXimera⟩
```

Include author name in meta tags

```
710 ⟨*htXimera⟩
711 \Configure{@HEAD}{\HCode{<meta name="author" content="}\@author\HCode{" />\Hnewline}}
712 ⟨/htXimera⟩
```

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

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

2.5.5 Title

\title Activities have titles.

```
714 ⟨*classXimera⟩
715 \let\title\relax
716 \newcommand{\title}[1][]{{\protected@xdef\@pretitle{#1}}\protected@xdef\@title}
717
718 \title{}
719
720 \newcounter{titlenumber}
721 \renewcommand{\thetitlenumber}{\arabic{titlenumber}}
722 %\renewcommand{\thesection}{\arabic{titlenumber}} %% Makes section numbers work
723 \setcounter{titlenumber}{0}
```

```
724
725 \newpagestyle{main}{
726 \sethead[\textsl{\ifnumbers\thetitlenumber\hspace{1em}\fi\@title}][][] % even
727 {}{}{\textsl{\ifnumbers\thetitlenumber\hspace{1em}\fi\@title}} % odd
728 \setfoot[\thepage][][] % even
729 {}{}{\thepage} % odd
730 }
731 \pagestyle{main}
```

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

```
732 \renewcommand\maketitle{%
733   \addtocounter{titlenumber}{1}%
734   {\flushleft\large\bfseries \@pretitle\par\vspace{-1em}}
735   {\flushleft\LARGE\bfseries {\ifnumbers\thetitlenumber\fi}{\ifnumbers\hspace{1em}\else\hspac
736   \phantomsection%
737   \ifnumbers\addcontentsline{toc}{section}{\thetitlenumber~\@title}\else\addcontentsline{toc]
738   \vskip .6em\noindent\textit\theabstract\setcounter{problem}{0}\setcounter{section}{0}\setco
739   %\ifnooutcomes\else\let\thefootnote\relax\footnote{Learning outcomes: \theoutcomes}\fi% Dep
740   \ifnoauthor\else\let\thefootnote\relax\footnote{Author(s):~\@author}\fi
741   \aftergroup\@afterindentfalse
742   \aftergroup\@afterheading}
743
744 \ifnumbers
745 \setcounter{secnumdepth}{2}
746 \renewcommand{\thesection}{\arabic{titlenumber}.\arabic{section}}
747 \renewcommand{\thesubsection}{\arabic{titlenumber}.\arabic{section}.\arabic{subsection}}
748 \else
749 \setcounter{secnumdepth}{-2}
750 \fi
751
752 \def\activitystyle{}
753 \newcounter{sectiontitlenumber}
754 \setcounter{secnumdepth}{2}
755 \setcounter{tocdepth}{2}
756 \newcommand\chapterstyle{%
757   \def\activitystyle{activity-chapter}
758   \def\maketitle{%
759     \addtocounter{titlenumber}{1}%
760                   {\flushleft\small\sffamily\bfseries\@pretitle\par\vspace{-1.5em}}%
761                   {\flushleft\LARGE\sffamily\bfseries\thetitlenumber\hspace{1em}\@title \pa
762                   {\vskip .6em\noindent\textit\theabstract\setcounter{problem}{0}\setcounte
763                   \par\vspace{2em}
764                   \phantomsection\addcontentsline{toc}{section}{\textbf{\thetitlenumber\hsp
765 }}
766
767
768 \newcommand\sectionstyle{%
769   \def\activitystyle{activity-section}
770   \def\maketitle{%
771     \addtocounter{section}{1}
772     \setcounter{sectiontitlenumber}{\value{section}}
773     {\flushleft\small\sffamily\bfseries\@pretitle\par\vspace{-1.5em}}%
774     {\flushleft\Large\sffamily\bfseries\thetitlenumber.\thesectiontitlenumber\hspace{1em}\@ti
775     {\vskip .6em\noindent\textit\theabstract\setcounter{subsection}{0}}%
776     \par\vspace{2em}
777     \phantomsection\addcontentsline{toc}{section}{\thetitlenumber.\thesectiontitlenumber\hspa
778   \renewcommand\section{\@startsection{subsection}{2}{\z@}%
779                                       {-3.25ex\@plus -1ex \@minus -.2ex}%
780                                       {1.5ex \@plus .2ex}%
781                                       {\normalfont\large\bfseries}}
782
783   \renewcommand\subsection{\@startsection{subsubsection}{3}{\z@}%
784                                         {-3.25ex\@plus -1ex \@minus -.2ex}%
```

17

```
785                                                    {1.5ex \@plus .2ex}%
786                                                    {\normalfont\normalsize\bfseries}}
787
788  }}
789
790
791  \iftikzexport%% allows xake to handle \chapterstyle and \sectionstye
792  \renewcommand\chapterstyle{\def\activitystyle{chapter}}
793  \renewcommand\sectionstyle{\def\activitystyle{section}}
794  \else
795  \fi
796
797  ⟨/classXimera⟩
```

Eliminate some formatting that we'll handle later with CSS

```
798  ⟨∗htXimera⟩
799  \renewcommand{\maketitle}{}
800  ⟨/htXimera⟩
```

2.5.6 Only in HTML or PDF

Ximera provides several techniques to display some content only in the PDF, or only online. The `prompt` environment can be used to hide the data-entry part of a problem from the PDF: it's contents only get displayed online.

The lower level commands `\pdfOnly` and `\htmlOnly` also limit the output to either PDF or online, similarly to the environments `onlyPdf` and `onlyHtml`.

If `\xmPrintHtmlOnlyAlsoInPdf` is set, the online/html only things are printed in the PDF anyway (e.g. for review).

Unfortunately it is not possible in LaTeX to have a command and an environment with the same name. We opted for the above (confusing...) names.

For backward compatibility, the deprecated environment `onlineOnly` is identical to `onlyHtml`.

For more advanced usage also commands `\ifonline` and `ifonlineTF` are provided.

The technique used to distinguish between the PDF-version and the online HTML-version is always the existence of the TeX4ht macro `\HCode`. Older distinctions such as `\ifxake`, `ifhandout` or `\iftikzexport` should no longer be used for this purpose.

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

```
801  ⟨∗classXimera⟩
802  \ifxake
803          \newenvironment{prompt}{}{}
804  \else
805  \ifhandout
806  \NewEnviron{prompt}{}
807      % Breaks when put in mathmode ?
808      % \newenvironment{prompt}{\suppress}{\endsuppress}
809  \else
810  \newenvironment{prompt}{\bgroup\color{gray!50!black}}{\egroup}
811  \fi
812  \fi
```

onlyHtml (*env.*) Only display online
onlyPdf (*env.*) Only display in the PDF
onlineOnly (*env.*) Only display online (deprecated: use onlyHtml instead)

```
813  \ifdefined\HCode
814  \newenvironment{onlyPdf}{\setbox0\vbox\bgroup}{\egroup}
815  \newenvironment{onlyHtml}{\bgroup}{\egroup}
816  \newenvironment{onlineOnly}{\bgroup}{\egroup}
817  \else
818  \newenvironment{onlyPdf}{\bgroup}{\egroup}
819  \ifdefined\xmPrintHtmlOnlyAlsoInPdf
820  \newenvironment{onlyHtml}{\bgroup\color{red!50!black}}{\egroup}
821  \newenvironment{onlineOnly}{\bgroup\color{red!50!black}}{\egroup}
822  \else
```

```
823   \newenvironment{onlyHtml}{\setbox0\vbox\bgroup}{\egroup}
824   \newenvironment{onlineOnly}{\setbox0\vbox\bgroup}{\egroup}
825 \fi
826 \fi
827
```

\htmlOnly Only display online
\pdfOnly Only display in the PDF

```
828
829 \ifdefined\HCode
830   \newcommand{\pdfOnly}[1]{}
831   \newcommand{\htmlOnly}[1]{#1}
832 \else
833   \ifdefined\xmPrintHtmlOnlyAlsoInPdf
834     \newcommand{\pdfOnly}[1]{#1}
835     \newcommand{\htmlOnly}[1]{\bgroup\color{red!50!black}#1\egroup}
836   \else
837     \newcommand{\pdfOnly}[1]{#1}
838     \newcommand{\htmlOnly}[1]{}
839   \fi
840 \fi
841
```

\ifonline Only execute online (ie in HTML version)
\ifonlineTF Different output online vs PDF

```
842 % An alternatife for \pdfOnly/\begin{htmlOnly} :
843 % Usage: Hello \ifonlineTF{online reader}{PDF reader}
844 \providecommand{\ifonlineTF}[2]{\htmlOnly{#1}\pdfOnly{#2}}
845 \newif{\ifonline}
846 \ifdefined\HCode
847 \onlinetrue
848 \else
849 \onlinefalse
850 \fi
851 ⟨/classXimera⟩
```

2.5.7 Learning Outcomes

```
852 ⟨∗classXimera⟩
853 \newcommand{\preOutcomeLine}{\item }
854 \newcommand{\postOutcomeLine}{}
855 \newcommand{\preOutcomeBlock}{After completing this content, students should be able to: \beg
856 \newcommand{\postOutcomeBlock}{\end{itemize} So go forth and learn!}
857
858 \newcommand{\outcomeHeader}{Goals for this Section}
859 \htmlOnly{
860   \newcommand{\outcomeBlock}{\ifvmode\IgnorePar\fi\EndP\HCode{<div class="outcomeHead">} \out
861 }
862
863
864 \newwrite\outcomefile
865 \immediate\openout\outcomefile=\jobname.oc
866 \newcommand{\outcome}[1]{%
867   \immediate\write\outcomefile{\expandafter\unexpanded\expandafter{\preOutcomeLine #1} \expan
868 }
869
870 \newcommand{\displayOutcomes}[1][]{%
871   \immediate\closeout\outcomefile
872   \IfFileExists{\currfiledir\currfilebase.oc}{
873     \htmlOnly{\outcomeBlock}
874       \expandafter\preOutcomeBlock
875       \input{\currfiledir\currfilebase.oc}
876       \postOutcomeBlock
877       \htmlOnly{\ifvmode\IgnorePar\fi\EndP\HCode{</div>}}
```

```
878      }
879      {
880      \IfFileExists{\currfilebase.oc}{
881        \htmlOnly{\outcomeBlock}
882          \expandafter\preOutcomeBlock
883          \input{\currfilebase.oc}
884          \postOutcomeBlock
885          \htmlOnly{\ifvmode\IgnorePar\fi\EndP\HCode{</div>}}
886      }
887      {
888        No outcome file found.
889      }
890    }
891  }
892 %
893 ⟨/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.

```
894 ⟨*cfgXimera⟩
895 \renewcommand{\outcome}[1]{
896   \Configure{@HEAD}{\HCode{<meta name="learning-outcome" content="#1"/>\Hnewline}}
897 }
898 % Sometimes there are no outcomes at all
899 \IfFileExists{\jobname.oc}{\input{\jobname.oc}}{}
900
901 \renewcommand{\outcome}[1]{%
902   \HCode{<span class="learning-outcome">#1</span>}
903 }
904 ⟨/cfgXimera⟩
```

2.5.8 Labels and references

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

```
905 ⟨*htXimera⟩
906 \let\oldlabel\label
907 \renewcommand{\label}[1]{\oldlabel{#1}\HCode{<a class="ximera-label" id="#1"></a>}}
908 ⟨/htXimera⟩
```

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

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

2.6 Images

2.6.1 Images

image (*env.*)   Place images inside an image environment. On paper, this centers the image. On the
\xmDefaultGraphicsPath   web, this provides additional benefits. Base graphicspath, deafult '/xmPictures'. Can only be changed BEFORE loading ximera.cls!

```
912 ⟨*classXimera⟩
913 % Provide a default graphicspath
914 % (somewhat tricky: an activity can be included in a xourse in a wildly different path !)
915 % Suggested convention: put all images in i /pictures folder in the root of your project
916 \providecommand{\xmDefaultGraphicsPath}{/xmPictures}
917 \graphicspath{  %% When looking for images,
918 {./}              %% look here first,
919 {.\xmDefaultGraphicsPath/}   %% then look for a pictures folder,
920 {..\xmDefaultGraphicsPath/}   %% then look for a pictures folder,
921 {../..\xmDefaultGraphicsPath/}   %% then look for a pictures folder,
922 {../../..\xmDefaultGraphicsPath/}   %% then look for a pictures folder,
923 }
```

```
924 %\newenvironment{image}[1][]{\begin{center}}{\end{center}}
925 \NewEnviron{image}[1][3in]{%
926   \begin{center}\resizebox{#1}{!}{\BODY}\end{center}% resize and center
927 }
928 ⟨/classXimera⟩
```

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

```
929 ⟨∗classXimera⟩
930 \newcommand{\alt}[1]{}
931 ⟨/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.

```
932 ⟨∗htXimera⟩
933 \newcounter{imagealt}
934 \setcounter{imagealt}{0}
935 \renewenvironment{image}[1][]{\stepcounter{imagealt}%
936   \ifvmode \IgnorePar\fi \EndP%
937   \HCode{<div class="image-environment" role="img" aria-labelledby="image-alt-\arabic{imageal
938 }{\HCode{</div>}}
939 \renewcommand{\alt}[1]{\HCode{<div style="display: none;" id="image-alt-\arabic{imagealt}">}#
940 ⟨/htXimera⟩
941 ⟨∗cfgXimera⟩
942 %%  Although we accept many formats, SVG is preferred on the web.
943 %%  Since we have a different mechanism for producing |alt| text, we
944 %%  want to ignore tex4ht's own method fo producing alt text.
945 %%  2024: is now in TeX4ht ...
946 % \DeclareGraphicsExtensions{.jpg,.png,.gif,.svg}
947 % \Configure{graphics*}
948 % {svg}{
949 %   {\Configure{Needs}{File: \Gin@base.svg}\Needs{}}
950 %   \Picture[]{\csname Gin@base\endcsname.svg \csname a:Gin-dim\endcsname}%
951 % }
952 ⟨/cfgXimera⟩
```

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

```
953 ⟨∗cfgXimera⟩
954 \ifcsname ifstandalone\endcsname
955   \ifstandalone
956     \renewcommand\includegraphics[2][]{}
957   \fi
958 ⟨/cfgXimera⟩
```

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

```
959 ⟨∗htXimera⟩
960 \providecommand{\pgfsyspdfmark}[3]{}
961 ⟨/htXimera⟩
```

2.6.2 TikZ export

2024: We DON NOT ANYMORE generate SVGs and PNGs for any TikZ images, via the "externalize" feature of TikZ.

Previously TikZ didn'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`.

```
962 ⟨∗classXimera⟩
963 % everything skipped, assumle TeX4ht does the jjb now
964 \ifdefined\reallyneverever
965
966 \ifdefined\HCode
967   \tikzexporttrue
```

```
968  \fi
969
970  \iftikzexport
971    \usetikzlibrary{external}
972
973    \ifdefined\HCode
974      % in htlatex, just include the svg files
975      \def\pgfsys@imagesuffixlist{.svg}
976
977      \tikzexternalize[prefix=./,mode=graphics if exists]
978    \else
979      % in pdflatex, actually generate the svg files
980      \tikzset{
981        /tikz/external/system call={
982          pdflatex \tikzexternalcheckshellescape
983          -halt-on-error -interaction=batchmode
984          -jobname "\image" "\\PassOptionsToClass{tikzexport}{ximera}\texsource";
985          mutool draw -F svg \image.pdf > \image.svg ;      % mutool adds "1" to filename ????
986          mutool draw -o \image.svg \image.pdf ;
987          mutool draw -r 150 -c rgbalpha -o \image.png \image.pdf ;
988          ebb -x \image.png
989        }
990      }
991      \tikzexternalize[optimize=false,prefix=./]
992    \fi
993
994  \fi
995 \fi
996 ⟨/classXimera⟩
```

2.6.3 XKCD

\xkcd Reference an XKCD cartoon.

```
997  ⟨*classXimera⟩
998  \newcommand{\xkcd}[1]{#1}
999  ⟨/classXimera⟩
```

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

```
1000 ⟨*htXimera⟩
1001 \renewcommand{\xkcd}[1]{\ifvmode \IgnorePar\fi \EndP\HCode{<img src="https://imgs.xkcd.com/co
1002 ⟨/htXimera⟩
```

2.7   Links

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

```
1003 ⟨*classXimera⟩
1004 % Don't use hyperref when using Tex4ht
1005 \ifdefined\HCode
1006 \RequirePackage{hyperref}
1007 \else
1008 \RequirePackage[pdfpagelabels,colorlinks=true,allcolors=blue!30!black]{hyperref}
1009 \pdfstringdefDisableCommands{\def\hskip{}}%% quiets warning
1010 \fi
1011 ⟨/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.

```
1012 ⟨*classXimera⟩
1013 \define@key{interactive}{id}{\def\interactive@id{#1}}
```

```
1014 \setkeys{interactive}{id=}
1015 \newcommand{\includeinteractive}[2][]{
1016 \setkeys*{interactive}{#1}%
1017 \ifthenelse{\equal{\interactive@id}{}}{}{\recordvariable{\interactive@id}}
1018 Interactive
1019 }
1020 ⟨/classXimera⟩

1021 ⟨∗htXimera⟩
1022 \renewcommand{\includeinteractive}[2][]{\stepcounter{identification}\ifvmode \IgnorePar\fi \I
1023 ⟨/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

```
1024 ⟨∗classXimera⟩
1025 % Google Spreadsheet link (read only)
1026 \newcommand{\googleSheet}[5]{%
1027   Google Spreadsheet link: \url{https://docs.google.com/spreadsheets/d/#1}%
1028 }
1029 ⟨/classXimera⟩

1030 ⟨∗htXimera⟩
1031 \renewcommand{\googleSheet}[5]{%
1032   \ifthenelse{\equal{#4}{}}%
1033     {\HCode{<iframe width="#2px" height="#3px" src="https://docs.google.com/spreadsheets/d/#
1034     {\ifthenelse{\equal{#5}{}}%
1035       {\HCode{<iframe width="#2px" height="#3px" src="https://docs.google.com/spreadsheets/
1036       {\HCode{<iframe width="#2px" height="#3px" src="https://docs.google.com/spreadsheets/
1037     }%
1038   }%
1039 ⟨/htXimera⟩
```

2.8.3   Geogebra

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

```
1040 ⟨∗classXimera⟩
1041 %Geogebra link
1042 \newcommand{\geogebra}[3]{GeoGebra link: \url{https://www.geogebra.org/m/#1}}
1043 ⟨/classXimera⟩
```

Define keys for answer geogebra key=value pairs.

```
1044 ⟨∗htXimera⟩
1045 \define@key{geogebra}{rc}[true]{\def\geo@rc{#1}}
1046 \define@key{geogebra}{sdz}[true]{\def\geo@sdz{#1}}
1047 \define@key{geogebra}{smb}[true]{\def\geo@smb{#1}}
1048 \define@key{geogebra}{stb}[true]{\def\geo@stb{#1}}
1049 \define@key{geogebra}{stbh}[true]{\def\geo@stbh{#1}}
1050 \define@key{geogebra}{ld}[true]{\def\geo@ld{#1}}
1051 \define@key{geogebra}{sri}[true]{\def\geo@sri{#1}}
1052 %set default key values
1053 \setkeys{geogebra}{rc=false,sdz=false,smb=false,stb=false,stbh=false,ld=false,sri=false}
1054 %command definition
1055 \renewcommand{\geogebra}[4][]{%
1056   \setkeys{geogebra}{#1}% Set new keys
1057   \HCode{<iframe scrolling="no" src="https://www.geogebra.org/material/iframe/id/#2/width/#3/
1058 ⟨/htXimera⟩
```

2.8.4   Desmos

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

```
1059 ⟨∗classXimera⟩
1060 \newcommand{\desmos}[3]{Desmos link: \url{https://www.desmos.com/calculator/#1}}
```

1061 ⟨classXimera⟩ `\newcommand{\desmosThreeD}[3]{Desmos3D link: \url{https://www.desmos.com/3d/#1}}`
1062 ⟨/classXimera⟩

1063 ⟨∗htXimera⟩
1064 `\catcode'\%=11`
1065 `\renewcommand{\desmos}[3]{\HCode{<iframe src="https://www.desmos.com/calculator/#1" width="10`
1066 `\catcode'\%=14`
1067 `\renewcommand{\desmosThreeD}[3]{\HCode{<iframe src="https://www.desmos.com/3d/#1" width="#2p`
1068 ⟨/htXimera⟩

2.8.5 Graphs

\graph An embedded graph (in math mode).

1069 ⟨∗classXimera⟩
1070 `\newcommand{\graph}[2][]{\text{Graph of $#2$}}`
1071 ⟨/classXimera⟩

1072 ⟨∗htXimera⟩
1073 `\renewcommand{\graph}[2][]{\HCode{<div class="graph" data-options="#1">}#2\HCode{</div>}}`
1074 ⟨/htXimera⟩

2.8.6 Video

\youtube Youtube command. Requires id.

1075 ⟨∗classXimera⟩
1076 `\newcommand{\youtube}[1]{YouTube link: \url{https://www.youtube.com/watch?v=#1}}`
1077 ⟨/classXimera⟩

1078 ⟨∗htXimera⟩
1079 `%% \renewcommand{\youtube}[1]{\ifvmode \IgnorePar\fi \EndP\HCode{<div class="video youtube-pl`
1080 `% Fixes no-youtube-when-no-cookies-accepted. Class xmyoutube allows for css customization.`
1081 `\renewcommand{\youtube}[1]{\ifvmode \IgnorePar\fi \EndP\HCode{<iframe class="xmyoutube" src="`
1082
1083 ⟨/htXimera⟩

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

1084 ⟨∗htXourse⟩
1085 `\renewcommand\youtube[1]{%`
1086 `\ifvmode \IgnorePar\fi \EndP\HCode{<a class="youtube" href="https://www.youtube.com/watch?v=#`
1087 `}`
1088 ⟨/htXourse⟩

2.8.7 JavaScript

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

1089 ⟨∗classXimera⟩
1090 `\DefineVerbatimEnvironment{javascript}{Verbatim}{numbers=left,frame=lines,label=JavaScript,la`
1091 ⟨/classXimera⟩

1092 ⟨∗htXimera⟩
1093 `% for programming javascript`
1094 `\renewenvironment{javascript}{\NoFonts}{\EndNoFonts}`
1095 `\ScriptEnv{javascript}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div cl`
1096 ⟨/htXimera⟩

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

1097 ⟨∗classXimera⟩
1098 `\def\js#1{\mbox{\texttt{\detokenize{#1}}}}`
1099 ⟨/classXimera⟩

1100 ⟨∗htXimera⟩
1101 `\def\js#1{\stepcounter{identification}\HCode{<span class="inline-javascript" id="javascript\a`
1102 ⟨/htXimera⟩

24

2.9 SageMath support

Load SageTeX if it exists.

```
1103 ⟨*classXimera⟩
1104 \IfFileExists{sagetex.sty}{\RequirePackage{sagetex}}{}
1105 ⟨/classXimera⟩
```

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

```
1106 ⟨*classXimera⟩
1107 \DefineVerbatimEnvironment{sageCell}{Verbatim}{numbers=left,frame=lines,label=SAGE,labelposi
1108 ⟨/classXimera⟩

1109 ⟨*htXimera⟩
1110 \renewenvironment{sageCell}{\NoFonts}{\EndNoFonts}
1111 \ScriptEnv{sageCell}{\ifvmode \IgnorePar\fi \EndP\HCode{<div class="sage"><script type="text/
1112 ⟨/htXimera⟩
```

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

```
1113 ⟨*classXimera⟩
1114 \DefineVerbatimEnvironment{sageOutput}{Verbatim}{numbers=left,frame=lines,label=SAGE-Output,l
1115 ⟨/classXimera⟩

1116 ⟨*htXimera⟩
1117 \renewenvironment{sageOutput}{\NoFonts}{\EndNoFonts}
1118 \ScriptEnv{sageOutput}{\ifvmode \IgnorePar\fi \EndP\HCode{<div class="sageOutput"><script typ
1119 ⟨/htXimera⟩
```

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

```
1120 ⟨*htXimera⟩
1121 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1122 \ifdefined\sagesilent
1123   \renewenvironment{sagesilent}{\NoFonts}{\EndNoFonts}
1124 \fi
1125 \ScriptEnv{sagesilent}{\ifvmode \IgnorePar\fi \EndP\HCode{<script type="text/sagemath">}\Html
1126 ⟨/htXimera⟩
```

2.10 Answerables

2.10.1 Answers

\answer A math answer

```
1127 ⟨*classXimera⟩
1128
1129 \ifdefined\HCode
1130 \newcommand{\recordvariable}[1]{}
1131 \else
1132 \newwrite\idfile
1133 \immediate\openout\idfile=\jobname.ids
1134 \newcommand{\recordvariable}[1]{\ifthenelse{\equal{#1}{}}{}{\immediate\write\idfile{var #1;}}
1135 \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

```
1136 \define@key{answer}{given}[true]{\def\ans@given{#1}}
```

Used for setting numeric answer tolerance for online student input.

```
1137 \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.

```
1138 \define@key{answer}{validator}{}
```

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

```
1139 \define@key{answer}{id}{\def\ans@id{#1}}
```

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

```
1140 \define@key{answer}{format}{}
```

Used to hide the answer input box on the web.

```
1141 \define@key{answer}{onlinenoinput}[false]{}
```

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

```
1142 \define@key{answer}{onlineshowanswerbutton}[false]{}
```

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

```
1143 \setkeys{answer}{id=,given=false,onlinenoinput=false,onlineshowanswerbutton=false}
```

Basic code for \answer.

```
1144
1145 % Options for handout
1146 \newcommand{\answerFormatLength}{2cm}
1147
1148 \newcommand{\answerFormatDots}[1]{\ldots\ldots}
1149 \newcommand{\answerFormatLine}[1]{\protect\rule{\answerFormatLength}{0.4pt}}
1150 \newcommand{\answerFormatFlexibleLine}[1]{\protect\rule{\widthof{$#1$}*2}{0.4pt}}
1151 \newcommand{\answerFormatFlexibleBox}[1]{\fbox{\scalebox{2}{\phantom{$#1$}}}}
1152
1153 % options for default (i.e with answers filled in)
1154 \newcommand{\answerFormatPlain}[1]{\ensuremath{#1}}
1155 \newcommand{\answerFormatBlue}[1]{\color{blue}\ensuremath{#1}}
1156 \newcommand{\answerFormatBoxed}[1]{\fbox{\ensuremath{#1}}}
1157 \newcommand{\answerFormatBoxedGiven}[1]{\underset{\scriptstyle\mathrm{given}}{\fbox{\ensurema
1158
1159 % defaults for handout and default mode, and for \answer[given]
1160 \let\handoutAnswerFormat\answerFormatDots
1161 \let\defaultAnswerFormat\answerFormatBlue
1162 \let\givenAnswerFormat\answerFormatBoxedGiven
1163
1164 \newcommand{\answer}[2][]{%
1165  \ifmmode%
1166   \setkeys{answer}{#1}%
1167   \recordvariable{\ans@id}
1168   \ifthenelse{\boolean{\ans@given}}
1169    {% Start then statement
1170    \ifhandout
1171     #2
1172    \else
1173     \givenAnswerFormat{#2} %% in case the argument helps formatting
1174    \fi
1175    }% End then statement
1176    {% Start else statement
1177    \ifhandout
1178     \handoutAnswerFormat{#2} %% in case the argument helps formatting
1179    \else% show answer in box outside handout mode
1180     \defaultAnswerFormat{#2} %% in case the argument helps formatting
1181    \fi
1182    }% End else statement
1183  \else%
1184   \GenericError{\space\space\space\space}% Throw an error based on... something? -- Jason
1185   {Attempt to use \@backslashchar answer outside of math mode}
1186   {See https://github.com/ximeraProject/ximeraLatex for explanation.}
1187   {Need to use either inline or display math.}%
1188  \fi
1189 }
1190 ⟨/classXimera⟩
```

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

```
1191 ⟨∗htXimera⟩
1192 \renewcommand{\answer}[2][false]{\HCode{<span class="answer respondable">}#2\HCode{</span>}}
1193
1194 \def\validator[#1]{\stepcounter{identification}\HCode{<div class="validator" id="validator\a
1195 \def\endvalidator{\HCode{</div>}}
```

```
1196
1197 ⟨/htXimera⟩
```

2.10.2 Multiple choice and the like

multipleChoice (*env.*) Multiple choice

```
1198 ⟨∗classXimera⟩
1199 % Jim: Originally this was \renewcommand{\theenumi}{$(\mathrm{\alph{enumi}})$}
1200 % but that breaks tex4ht because mathmode can only be processed by mathjax.
1201 % so now I made this just italicized.
```

2.10.3 Options

```
1202 \define@key{choice}{value}[]{\def\choice@value{#1}}
```

This flags the answer as the correct answer
```
1203 \define@boolkey{choice}{correct}[true]{\def\choice@correct{#1}}
```

Use an ID to refer to the choice.
```
1204 \define@key{multipleChoice}{id}{\def\mc@id{#1}}
```

\otherchoice outputs the item if correct and nothing if incorrect.
```
1205 \define@key{otherchoice}{value}[]{\def\otherchoice@value{#1}}
1206 \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".
```
1207 \setkeys{choice}{correct=false,value=}
```

Defaults for multipleChoice pairs. Default to no id? – Jason
```
1208 \setkeys{multipleChoice}{id=}
```

Defaults for otherchoice pairs. Default "otherchoice" to behave like "choice" for error checking.
```
1209 \setkeys{otherchoice}{correct=false,value=}
1210 ⟨/classXimera⟩
```

2.10.4 Choices

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

```
1211 ⟨∗classXimera⟩
1212 \newcommand{\choice}[2][]{%
1213 \setkeys{choice}{#1}%
1214 \item{#2}
1215 \ifthenelse{\boolean{\choice@correct}}
1216     {% Begin then result
1217     \ifhandout% if it's a handout do nothing.
1218     \else% otherwise place a checkmark when you select the "correct choice"... maybe? -- Jaso
1219         \,\checkmark\,\setkeys{choice}{correct=false}
1220     \fi
1221     }% End then result
1222     {}% Begin/End else result.
1223 }
1224
1225 %Define an expandable version of choice Not really meant to be used outside this package (use
1226 % Is there a reason we can't just always use this as default? -- Jason
1227 \newcommand{\choiceEXP}[2][]{%
1228 \expandafter\setkeys\expandafter{choice}{#1}%
1229 \item{#2}
1230 \ifthenelse{\boolean{\choice@correct}}
1231   {% Begin then result
1232   \ifhandout
1233   \else
1234   \,\checkmark\,\setkeys{choice}{correct=false}
1235   \fi
1236   }% End then result
1237   {}% Begin/End else result.
```

```
1238  } %% note all the {} are needed in case the choice has [] in it.
1239
1240  % \otherchoice is the \choice used in wordChoice command.
1241  \newcommand{\otherchoice}[2][]{%
1242  \ignorespaces%
1243  \setkeys{otherchoice}{#1}%
1244  \ifthenelse{\boolean{\otherchoice@correct}}%
1245  {% Start then result
1246  #2\ignorespaces\setkeys{otherchoice}{correct=false}\ignorespaces%
1247  }% End then result
1248  {}% Start/End else result
1249  \ignorespaces%
1250  }%
1251  \newcommand{\inlinechoice}[2][]{%
1252  \setkeys{choice}{#1}%
1253  \iffirstinlinechoice
1254  (\hspace{-.25em}
1255  \firstinlinechoicefalse
1256  \else
1257  /
1258  \fi
1259  #2
1260  \ifthenelse{\boolean{\choice@correct}}%
1261  {% Start then result
1262  \ifhandout\else\checkmark\ignorespaces\setkeys{choice}{correct=false}\ignorespaces\fi%
1263  }% End then result
1264  {}% Start/End else result
1265  \hspace{-.25em}\ignorespaces%
1266  }
1267
1268  ⟨/classXimera⟩
```

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

```
1269  ⟨∗htXimera⟩
1270  \newcounter{choiceId}
1271  \renewcommand{\choice}[2][]{%
1272  \setkeys{choice}{correct=false}%
1273  \setkeys{choice}{#1}%
1274  \stepcounter{choiceId}\IgnorePar%
1275  \HCode{<span class="choice }%
1276  \ifthenelse{\boolean{\choice@correct}}{\HCode{correct}}{}
1277  \HCode{" }
1278  \ifthenelse{\equal{\choice@value}{}}{}{\HCode{data-value="\choice@value" }}
1279  \HCode{id="choice\arabic{choiceId}">}%
1280  #2\HCode{</span>}}
1281  \let\inlinechoice\choice
1282  ⟨/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.

```
1283  ⟨∗classXimera⟩
1284  \newenvironment{multipleChoice}[1][]
1285  {% Environment Start Code
1286  \setkeys{multipleChoice}{#1}%
1287  \recordvariable{\mc@id}%
1288  \begin{trivlist}
1289   \item[\hskip \labelsep\small\bfseries \GetTranslation{Multiple Choice}:]\hfil
1290   \begin{enumerate}
1291  }% Note this means that \item has to be the first line after \begin{multipleChoice}.
1292  {% Environment End Code
1293   \end{enumerate}
1294  \end{trivlist}
```

```
1295  }
1296
1297  %multipleChoice@ is for internal use only! (used in wordChoice)
1298  %this is simply a wrapper for the sole showing (other)choice.
1299  \newenvironment{multipleChoice@}[1][]{}{})}
1300  ⟨/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.

```
1301  ⟨∗htXimera⟩
1302  \renewenvironment{multipleChoice}[1][]
1303  {\setkeys{multipleChoice}{#1}%
1304  \stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div class="multiple-choice" }
1305  \ifthenelse{\equal{\mc@id}{}}{}{\HCode{data-id="\mc@id" }}%
1306  \HCode{id="problem\arabic{identification}" titletext=" \GetTranslation{Multiple Choice}">}%
1307  }{\HCode{</div>}\IgnoreIndent}
1308  \ConfigureEnv{multipleChoice}{}{}{}{}
1309  ⟨/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.

```
1310  ⟨∗classXimera⟩
1311  \newcommand{\wordChoice}[1]{%
1312  \let\choicetemp\choice% Assign a "choicetemp" command to duplicate choice.
1313  \ifwordchoicegiven% If wordchoice option is on, we need to juggle around some definitions.
1314  \let\choice\otherchoice%
1315  %\begin{multipleChoice@}% -unnecessary (REMOVE THIS LINE IF THE YEAR IS 2019 or Beyond)
1316  #1
1317  %\end{multipleChoice@}% -unnecessary (REMOVE THIS LINE IF THE YEAR IS 2019 or Beyond)
1318  \else% If it isn't the regular "choice" command should work.
1319  \let\choice\inlinechoice%
1320  \begin{multipleChoice@}%
1321  #1%
1322  \end{multipleChoice@}%
1323  \fi%
1324  \let\choice\choicetemp% Now that choicetmp has been manipulated to what we want, replace cho
1325  }%
1326
1327
1328  ⟨/classXimera⟩
```

This is actually just word choice

```
1329  ⟨∗htXimera⟩
1330  \renewenvironment{multipleChoice@}{\refstepcounter{problem}}{}%
1331  \ConfigureEnv{multipleChoice@}{\stepcounter{identification}\IgnorePar\HCode{<span class="wor
1332  ⟨/htXimera⟩
```

2.12   Select all

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

```
1333  ⟨∗classXimera⟩
1334  \newenvironment{selectAll}[1][]
1335  {\begin{trivlist}\item[\hskip \labelsep\small\bfseries \GetTranslation{Select All Correct Ans
1336      {\end{enumerate}\end{trivlist}}
1337  ⟨/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`.

```
1338 ⟨∗htXimera⟩
1339 \renewenvironment{selectAll}{\refstepcounter{problem}}{}%
1340 \ConfigureEnv{selectAll}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div
1341 ⟨/htXimera⟩
```

2.12.1 Free response

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

```
1342 ⟨∗classXimera⟩
1343 \newboolean{given} %% required for freeResponse
1344 \setboolean{given}{true} %% could be replaced by a key=value pair later if needed
1345
1346 \ifhandout
1347 \newenvironment{freeResponse}[1][false]%
1348   {%
1349   \def\givenatend{\boolean{#1}}
1350   \ifthenelse{\boolean{#1}}
1351   {% Begin then result
1352   \begin{trivlist}
1353    \item
1354   }% End then result
1355   {% Begin else result
1356   \setbox0\vbox\bgroup
1357   }% End else result
1358 % {}% Don't think this is doing anything? -- Jason
1359   }
1360   {%
1361   \ifthenelse{\givenatend}
1362   {% Begin then result
1363   \end{trivlist}
1364   }% End then result
1365   {% Begin else result
1366   \egroup
1367   }% End else result
1368 % {}% Don't think this is doing anything? -- Jason
1369   }
1370 \else
1371 \newenvironment{freeResponse}[1][false]%
1372   {% Environment Beginning Code
1373    \ifthenelse{\boolean{#1}}%% Could probably change this with just putting the (given) in t
1374     {% Begin then result
1375     \begin{trivlist}
1376      \item[\hskip \labelsep\bfseries \GetTranslation{Free Response (Given)}:\hspace{2ex}]
1377     }% End then result
1378   {% Begin else result
1379   \begin{trivlist}
1380    \item[\hskip \labelsep\bfseries \GetTranslation{Free Response}:\hspace{2ex}]
1381   }% End else result
1382   }
1383   {% Environment Ending Code
1384    \end{trivlist}
1385   }
1386 \fi
1387
1388 ⟨/classXimera⟩

1389 ⟨∗htXimera⟩
1390
1391 \renewenvironment{freeResponse}{\refstepcounter{problem}}{}%
1392 \ConfigureEnv{freeResponse}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<
1393
1394 ⟨/htXimera⟩
```

2.12.2 Feedback

feedback (*env.*) An initially hidden environment that uncovers itself at an appropriate time. New Validator 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.

```
1395 ⟨∗classXimera⟩
1396 \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.

```
1397 \newenvironment{validator}[1][]{
1398 \def\PH@Command{#1}% Use PH@Command to hold the content and be a target for "\expandafter" t
1399 \mbox{\texttt{\detokenize\expandafter{\PH@Command}}}% Now expand PH@Command once and then de
1400 }{}
```

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

```
1401 \ifhandout%
1402 \newenvironment{feedback}
1403              {%
1404     \setbox0\vbox\bgroup
1405         }
1406              {%
1407     \egroup
1408         }
```

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.

```
1409 \else
1410 \newenvironment{feedback}[1][attempt]{
1411
1412 \edef\PH@Command{\GetTranslation{#1}}% Use PH@Command to hold the content and be a target fo
1413
1414 \begin{trivlist}% Begin the trivlist to use formating of the "Feedback" label.
1415 \item[\hskip \labelsep\small\slshape\bfseries \GetTranslation{feedback}% Format the "Feedbac
1416 (\texttt{\expandafter\detokenize\expandafter{\PH@Command}}):% Format (and detokenize) the co
1417 \hspace{2ex}]\small\slshape% Insert some space before the actual feedback given.
1418 }{
1419 \end{trivlist}
1420 }
1421
1422 \fi
1423 ⟨/classXimera⟩
```

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

```
1424 ⟨∗htXimera⟩
1425 \def\feedback{\@ifnextchar[{\@feedbackcode}{\@feedbackattempt}}
1426 \def\@feedbackattempt{\@feedbackcode[attempt]}
1427 \def\@feedbackcode[#1]{\stepcounter{identification}%
1428 \ifvmode \IgnorePar\fi \EndP%
1429 \ifthenelse{\equal{#1}{attempt}}{\HCode{<div class="feedback" data-feedback="attempt" id="fe
1430 {\ifthenelse{\equal{#1}{correct}}{\HCode{<div class="feedback" data-feedback="correct" id="fe
1431 {\HCode{<div class="feedback" data-feedback="script" id="feedback\arabic{identification}" ti
1432 \def\endfeedback{\HCode{</div>}\IgnoreIndent}
1433 ⟨/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.

```
1434 ⟨*classXimera⟩
1435 \newenvironment{ungraded}{}{}
1436 ⟨/classXimera⟩
```

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

```
1437 ⟨*htXimera⟩
1438 \renewenvironment{ungraded}{%
1439 \ifvmode \IgnorePar\fi \EndP\HCode{<div class="ungraded">}\IgnoreIndent%
1440 }{
1441 \ifvmode \IgnorePar\fi \EndP\HCode{</div>}\IgnoreIndent%
1442 }
1443 ⟨/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. Redefine newcommand appropriately.

```
1444 ⟨*classXimera⟩
1445 %% Pre-202412: .jax file written in non-\HCode, and in a next run inserted by ximera.cfg in
1446 %% Post-202501: .mjax file written only in \HCode, and in luaxake post-processing inserted in
1447 %%    ( used luaxake rather than sed ...)
1448 \newwrite\myfile
1449 \ifdefined\HCode
1450 \immediate\openout\myfile=\jobname.xmjax
1451
1452 %% From |only.dtx| we must also create |prompt| on the MathJax side.
1453 \immediate\write\myfile{\unexpanded{\newenvironment}{prompt}{}{}}
1454
1455 %% Write all newcommands to .xmjax file, that will be included in the .html via luaxake
1456 \let\@oldargdef\@argdef
1457 \long\def\@argdef#1[#2]#3{%
1458 \immediate\write\myfile{\unexpanded{\newcommand}{\unexpanded{#1}}[\unexpanded{#2}]{\unexpand
1459 \@oldargdef#1[#2]{#3}%
1460 }
1461
1462 %% Same for \DeclareMathOperator
1463 \let\@OldDeclareMathOperator\DeclareMathOperator
1464 \renewcommand{\DeclareMathOperator}[2]{\@OldDeclareMathOperator{#1}{#2}\immediate\write\myfil
1465
1466 \fi
1467
1468
1469 ⟨/classXimera⟩
```

Include the jax'ed newcommands (pre-202412 versions ....)

```
1470 ⟨*cfgXimera⟩
1471
1472 % 202501: removed sed-manipulation of .jax file; see luaxake now
1473
1474 \Configure{BVerbatimInput}{}{}{}{}
1475
1476 \Configure{verbatiminput}{}{}{}{}
1477
1478 % Instead of a nonbreaking space, use a standard space
1479 \makeatletter
```

```
1480 \def\FV@Space{\space}
1481 \makeatother
1482
1483 % Include the (problem-?) .ids in a text/javascript script right at the beginning of the body
1484 \Configure{BODY}{%
1485 \HCode{<body>\Hnewline}%
1486 \Tg<div class="preamble">%
1487 %% 202501: removed .jax inclusion  (see luaxake)
1488
1489 %% Include the .ids file
1490 \IfFileExists{\jobname.ids}{\HCode{<script type="text/javascript">\Hnewline}%
1491 \BVerbatimInput{\jobname.ids}%
1492 \HCode{</script>\Hnewline}%
1493 }{}
1494 \Tg</div>%
1495 }{%
1496 \ifvmode\IgnorePar\fi\EndP\HCode{</body>\Hnewline}%
1497 }
1498
1499 % 202501: removed 'prevent spaces as in  "\begin {align}": this is done in luaxake now
1500
1501 % This is a fix for the LAODE book, which uses matlabEquation as if it were an equation
1502 \ScriptEnv{matlabEquation}{\ifvmode \IgnorePar\fi \EndP\HCode{<script type="math/tex; mode=di
1503
1504 ⟨/cfgXimera⟩
```

2.13.2   Semantic HTML

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

```
1505 ⟨*cfgXimera⟩
1506 \Configure{textbf}{\ifvmode\ShowPar\fi\HCode{<strong>}}{\HCode{</strong>}}
1507 ⟨/cfgXimera⟩
```

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

```
1508 ⟨*cfgXimera⟩
1509 \Configure{textit}{\ifvmode\ShowPar\fi\HCode{<em>}}{\HCode{</em>}}
1510 \Configure{emph}{\ifvmode\ShowPar\fi\HCode{<em>}}{\HCode{</em>}}
1511 ⟨/cfgXimera⟩
```

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

```
1512 ⟨*cfgXimera⟩
1513 \Configure{texttt}{\ifvmode\ShowPar\fi\HCode{<code>}}{\HCode{</code>}}
1514 ⟨/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.

```
1515 ⟨*classXimera⟩
1516 \font\dummyft@=dummy \relax
1517 \def\suppress{%
1518    \begingroup\par
1519    \parskip\z@
1520    \offinterlineskip
1521    \baselineskip=\z@skip
1522    \lineskip=\z@skip
1523    \lineskiplimit=\maxdimen
1524    \dummyft@
1525    \count@\sixt@@n
1526    \loop\ifnum\count@ >\z@
1527       \advance\count@\m@ne
```

```
1528        \textfont\count@\dummyft@
1529        \scriptfont\count@\dummyft@
1530        \scriptscriptfont\count@\dummyft@
1531      \repeat
1532      \let\selectfont\relax
1533      \let\mathversion\@gobble
1534      \let\getanddefine@fonts\@gobbletwo
1535      \tracinglostchars\z@
1536      \frenchspacing
1537      \hbadness\@M}
1538 \def\endsuppress{\par\endgroup}
```
1539 ⟨/classXimera⟩

2.14.2   The End

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

1540 ⟨∗htXimera⟩
```
1541 \Hinput{ximera}
```
1542 ⟨/htXimera⟩

1543 ⟨∗htXourse⟩
```
1544 \Hinput{xourse}
```
1545 ⟨/htXourse⟩

1546 ⟨∗cfgXimera⟩
```
1547 \begin{document}
1548 \EndPreamble
```
1549 ⟨/cfgXimera⟩

3   xourse.cls

1550 ⟨∗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.

```
1551 \newif\ifnotoc
1552 \notocfalse
1553 \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.

```
1554 \newif\ifnonewpage
1555 \nonewpagefalse
1556 \DeclareOption{nonewpage}{\nonewpagetrue}

1557 \DeclareOption*{\PassOptionsToClass{\CurrentOption}{ximera}}
1558 \ProcessOptions\relax
1559 \LoadClass{ximera}
1560 %    \begin{macrocode}
```
1561 ⟨/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.

1562 ⟨∗classXourse⟩
```
1563 \newcommand{\skip@preamble}{%
1564      \let\document\relax\let\enddocument\relax%
1565      \newenvironment{document}{\let\input\otherinput}{}%
1566      \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:

```
1567 \let\otherinput\input
```

Store usual \maketitle as \othermaketitle

```
1568 \let\othermaketitle\maketitle
```

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

```
1569 \renewcommand{\maketitle}{ %
1570 \pagestyle{empty}
1571 \begin{center}
1572 ~\\ %puts space at top of page to move title down.
1573 \vskip .25\textheight
1574 \hrulefill\\
1575 \vskip 1em
1576 \bfseries{\Huge \@title} \\
1577 \hrulefill\\
1578 \vskip 3em
1579 {\Large \@author}
1580 \vskip 2em
1581 {\large \@date}
1582 \end{center}
1583 \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.

```
1584 \ifnotoc
1585 \else
1586   \tableofcontents\clearpage
1587   \clearpage
1588 \fi
```

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

```
1589 \pagestyle{main}
```

Renew maketitle to usual definition.

```
1590 \let\maketitle\othermaketitle
```

And we finish with our redefinition of \maketitle.

```
1591 }
1592 \relax
1593 ⟨/classXourse⟩
```

3.1.1   Regular activities

\activity Documents included with \activity will be included in the body of the xourse document. Any \input commands within included ximera documents will be ignored. Any \usepackage commands within included ximera documents will cause an error. Overlapping \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.

```
1594 ⟨*classXourse⟩
1595 \ifnonewpage
1596 \newcommand{\activity}[2][]{%
1597   \setkeys{activity}{#1}
1598   \renewcommand{\input}[1]{}
1599   \begingroup\skip@preamble\otherinput{#2}\endgroup\par\vspace{\topsep}
1600   \let\input\otherinput}
1601 \else
1602 \newcommand{\activity}[2][]{%
1603   \setkeys{activity}{#1}
1604   \renewcommand{\input}[1]{}
1605   \begingroup\skip@preamble\otherinput{#2}\endgroup\clearpage
```

```
1606    \let\input\otherinput}
1607 \fi
1608 \relax
1609 ⟨/classXourse⟩

1610 ⟨∗htXourse⟩
1611 \renewcommand\activity[2][]{%
1612 \ifvmode \IgnorePar\fi \EndP\HCode{<a class="activity card \activitystyle" href="#2" data-op
1613 }
1614 ⟨/htXourse⟩
```

When running xake, we can just ignore activities

```
1615 ⟨∗classXourse⟩
1616 \ifxake
1617 \renewcommand\activity[2][]{}
1618 \fi
1619 ⟨/classXourse⟩
```

3.1.2   Practice activities

\practice Like \activity but not expecting a title.

```
1620 ⟨∗classXourse⟩
1621 \ifhandout
1622 \newcommand{\practice}[2][]{
1623  \setkeys{practice}{#1}%!!!!!
1624    \renewcommand{\input}[1]{}
1625    \begingroup\skip@preamble\otherinput{#2}\endgroup
1626    \let\input\otherinput}
1627 \else
1628 \newcommand{\practice}[2][]{\texttt{\detokenize{#2}}%% gives file name for practice
1629  \setkeys{practice}{#1}%!!!!!
1630    \renewcommand{\input}[1]{}
1631    \begingroup\skip@preamble\otherinput{#2}\endgroup
1632    \let\input\otherinput}
1633 \fi
1634 \relax
1635 ⟨/classXourse⟩
```

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

```
1636 ⟨∗classXourse⟩
1637 \ifxake
1638 \renewcommand\practice[2][]{}
1639 \fi
1640 ⟨/classXourse⟩
```

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

```
1641 ⟨∗htXourse⟩
1642 \renewcommand\practice[2][]{%
1643    \ifvmode\IgnorePar\fi\EndP%
1644    \HCode{<a class="activity card practice" href="#2" data-options="#1">#2</a>}%
1645    \IgnoreIndent%
1646 }
1647 ⟨/htXourse⟩
```

3.2   Sectioning

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

```
1648 ⟨∗classXourse⟩
1649 \renewcommand*\l@section{\@dottedtocline{1}{1.5em}{4.2em}}
1650 ⟨/classXourse⟩
```

\subsection The name of a subsection inside an activity.

```
1651 ⟨*classXourse⟩
1652 \renewcommand*\l@subsection{\@dottedtocline{2}{3.8em}{4.2em}}
1653 ⟨/classXourse⟩
```

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

```
1654 ⟨*htXourse⟩
1655 \newcounter{ximera@part}
1656 \setcounter{ximera@part}{0}
1657 \renewcommand\part[1]{%
1658 \stepcounter{ximera@part}%
1659 \ifvmode \IgnorePar\fi \EndP%
1660 %\HCode{<h1 id="part\arabic{ximera@part}" class="card part">}#1\HCode{</h1>}% makes cards dis
1661 \HCode{<h1 id="part\arabic{ximera@part}" class="card part">#1</h1>}%
1662 \IgnoreIndent%
1663 }
1664 ⟨/htXourse⟩
```

\paragraph Paragraph commands emit spans. A small heading.

```
1665 ⟨*cfgXimera⟩
1666 \renewcommand{\paragraph}[1]{%
1667   \HCode{<span class="paragraphHead">}%
1668   #1%
1669   \HCode{</span>}\par\IgnorePar}
1670 ⟨/cfgXimera⟩
```

\subparagraph An even smaller heading.

```
1671 ⟨*cfgXimera⟩
1672 \renewcommand{\subparagraph}[1]{%
1673   \HCode{<span class="subparagraphHead">}%
1674   #1%
1675   \HCode{</span>}\par\IgnorePar}
1676 ⟨/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.

```
1677 ⟨*classXourse⟩
1678 \newenvironment{graded}[1]{}{}
1679 ⟨/classXourse⟩
```

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

```
1680 ⟨*htXourse⟩
1681 \renewenvironment{graded}[1]{%
1682 \ifvmode \IgnorePar\fi \EndP\HCode{<div class="graded" data-weight="#1">}\IgnoreIndent%
1683 }{
1684 \ifvmode \IgnorePar\fi \EndP\HCode{</div>}\IgnoreIndent%
1685 }
1686 ⟨/htXourse⟩
```

3.4   Logos

\logo A logo for the xourse.

```
1687 ⟨*classXourse⟩
1688 \newcommand*{\logo}[1]{%
1689   \ifx\@onlypreamble\@notprerr
1690     \ClassError{xourse}{logo can only be used in the preamble}
1691       {Move your logo command to the preamble}
1692   \else %
1693     \IfFileExists{#1}%
1694       {\gdef\xourse@logo{#1}}%
1695       {\ClassError{xourse}{logo file does not exist}
```

```
1696            {To use logo, make sure that the referenced image file exists}}%
1697    \fi%
1698 }
1699
1700 ⟨/classXourse⟩
```

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

```
1701 ⟨∗htXourse⟩
1702 \Configure{@HEAD}{%
1703    \HCode{<meta name="og:image" content="}%
1704 \ifdefined\xourse@logo%
1705    \xourse@logo%
1706 \fi%
1707 \HCode{" />\Hnewline}}%
1708 ⟨/htXourse⟩
```