

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 hand-outs, 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, **E**Resources, 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 sees 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

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 suppress such content and generate a reasonable printable “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 suppress 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 suppress the listing of the learning outcomes.

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

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

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

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

```
14 \DeclareOption{noinstructornotes}{\instructornotetrue}
```

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* p
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 \end{classXimera}

```

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

```

88 \begin{classXimera}
89 \RequirePackage{getttitlestring}
90 \RequirePackage{nameref}
91 \RequirePackage{epstopdf}
92 \end{classXimera}

```

2.3 Page setup

We want non-indented spaced-out paragraphs.

```

93 \begin{classXimera}
94 \setlength{\parindent}{0pt}
95 \setlength{\parskip}{5pt}
96 \end{classXimera}

```

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

```

97 \begin{classXimera}
98 \oddsidemargin 62pt
99 \evensidemargin 62pt
100 \textwidth 345pt
101 \headheight 14pt
102 \end{classXimera}

```

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

```

103 \begin{cfgXimera}
104 \Preamble{xhtml,mathjax}
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.edu/~jim/TeX4ht/)>\Hnewline}}
121 \Configure{@HEAD}{\HCode{<meta name="ximera" content="version 2.0.1" />\Hnewline}}
122 \Configure{@HEAD}{\HCode{<link href="https://ximera.osu.edu/public/stylesheets/standalone.css" />\Hnewline}}
123 \Configure{@HEAD}{\HCode{<script type="text/javascript" async src="https://ximera.osu.edu/public/js/ximera.js">\Hnewline}}
124
125 % OVERWRITE css in ximera-server (to be removed whenever this has been fixed in the server; see https://github.com/ximera/ximera-server/issues/11)
126 \catcode'\% =11
127 \Configure{@BODY}{\HCode{<style>
128 .activity-body pre {
129     white-space: pre;
130     background-color: lightgray;
131 }
132 .xmyoutube {
133     aspect-ratio: 16/9;
134     min-width: 75%;

```

```

135 }
136 .image-environment img {
137     width: unset;
138 }
139 </style>\Hnewline}}
140 \catcode'\%=14
141
142 </cfgXimera>

```

Disable certain ligatures in HTML.

```

143 <*htXimera>
144 \usepackage{microtype}
145 \DisableLigatures[f]{encoding=*}
146 </htXimera>

```

I am not sure what this does.

```

147 <*htXimera>
148 \NewEnviron{html}{\HCode{\BODY}}
149 </htXimera>

```

2.4 Structure

2.4.1 Macros

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

```

150 <*classXimera>
151 \everymath{\displaystyle}
152 </classXimera>

```

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

```

153 <*classXimera>
154 \let\prelim\lim
155 \renewcommand{\lim}{\displaystyle\prelim}
156 </classXimera>

```

2.4.2 Theorem and theorem-like environments

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

```

157 <*htXimera>
158 \newcommand{\ConfigureTheoremEnv}[1]{%
159 \renewenvironment{#1}[1][1]{\refstepcounter{problem}%
160 \ifthenelse{\equal{##1}{}}{}{}%
161 \HCode{<span class="theorem-like-title">}##1\HCode{</span>}}%
162 }{}
163 \ConfigureEnv{#1}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div class=
164 }
165 </htXimera>
166 <classXimera>\theoremstyle{definition} % No italic (because this makes also text in TikZ italic

```

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

theorem (<i>env.</i>)	Theorem
	<pre> 167 <classXimera> \newtheorem{theorem}{Theorem} 168 <htXimera> \ConfigureTheoremEnv{theorem} </pre>
algorithm (<i>env.</i>)	Algorithm
	<pre> 169 <classXimera> \newtheorem{algorithm}{Algorithm} 170 <htXimera> \ConfigureTheoremEnv{algorithm} </pre>
axiom (<i>env.</i>)	Axiom
	<pre> 171 <classXimera> \newtheorem{axiom}{Axiom} 172 <htXimera> \ConfigureTheoremEnv{axiom} </pre>
claim (<i>env.</i>)	Claim
	<pre> 173 <classXimera> \newtheorem{claim}{Claim} 174 <htXimera> \ConfigureTheoremEnv{claim} </pre>

<code>conclusion (env.)</code>	Conclusion	
	175 <code>\classXimera</code>	<code>\newtheorem{conclusion}{Conclusion}</code>
	176 <code>\htXimera</code>	<code>\ConfigureTheoremEnv{conclusion}</code>
<code>condition (env.)</code>	Condition	
	177 <code>\classXimera</code>	<code>\newtheorem{condition}{Condition}</code>
	178 <code>\htXimera</code>	<code>\ConfigureTheoremEnv{condition}</code>
<code>conjecture (env.)</code>	Conjecture	
	179 <code>\classXimera</code>	<code>\newtheorem{conjecture}{Conjecture}</code>
	180 <code>\htXimera</code>	<code>\ConfigureTheoremEnv{conjecture}</code>
<code>corollary (env.)</code>	Corollary	
	181 <code>\classXimera</code>	<code>\newtheorem{corollary}{Corollary}</code>
	182 <code>\htXimera</code>	<code>\ConfigureTheoremEnv{corollary}</code>
<code>criterion (env.)</code>	Criterion	
	183 <code>\classXimera</code>	<code>\newtheorem{criterion}{Criterion}</code>
	184 <code>\htXimera</code>	<code>\ConfigureTheoremEnv{criterion}</code>
<code>definition (env.)</code>	Definition	
	185 <code>\classXimera</code>	<code>\newtheorem{definition}{Definition}</code>
	186 <code>\htXimera</code>	<code>\ConfigureTheoremEnv{definition}</code>
<code>example (env.)</code>	Example	
	187 <code>\classXimera</code>	<code>\newtheorem{example}{Example}</code>
	188 <code>\htXimera</code>	<code>\ConfigureTheoremEnv{example}</code>
<code>explanation (env.)</code>	Explanation	
	189 <code>\classXimera</code>	<code>\newtheorem*{explanation}{Explanation}</code>
	190 <code>\htXimera</code>	<code>\ConfigureTheoremEnv{explanation}</code>
<code>fact (env.)</code>	Fact	
	191 <code>\classXimera</code>	<code>\newtheorem{fact}{Fact}</code>
	192 <code>\htXimera</code>	<code>\ConfigureTheoremEnv{fact}</code>
<code>lemma (env.)</code>	Lemma	
	193 <code>\classXimera</code>	<code>\newtheorem{lemma}{Lemma}</code>
	194 <code>\htXimera</code>	<code>\ConfigureTheoremEnv{lemma}</code>
<code>formula (env.)</code>	Formula	
	195 <code>\classXimera</code>	<code>\newtheorem{formula}{Formula}</code>
	196 <code>\htXimera</code>	<code>\ConfigureTheoremEnv{formula}</code>
<code>idea (env.)</code>	Idea	
	197 <code>\classXimera</code>	<code>\newtheorem{idea}{Idea}</code>
	198 <code>\htXimera</code>	<code>\ConfigureTheoremEnv{idea}</code>
<code>notation (env.)</code>	Notation	
	199 <code>\classXimera</code>	<code>\newtheorem{notation}{Notation}</code>
	200 <code>\htXimera</code>	<code>\ConfigureTheoremEnv{notation}</code>
<code>model (env.)</code>	Model	
	201 <code>\classXimera</code>	<code>\newtheorem{model}{Model}</code>
	202 <code>\htXimera</code>	<code>\ConfigureTheoremEnv{model}</code>
<code>observation (env.)</code>	Observation	
	203 <code>\classXimera</code>	<code>\newtheorem{observation}{Observation}</code>
	204 <code>\htXimera</code>	<code>\ConfigureTheoremEnv{observation}</code>
<code>proposition (env.)</code>	Proposition	
	205 <code>\classXimera</code>	<code>\newtheorem{proposition}{Proposition}</code>
	206 <code>\htXimera</code>	<code>\ConfigureTheoremEnv{proposition}</code>
<code>paradox (env.)</code>	Paradox	
	207 <code>\classXimera</code>	<code>\newtheorem{paradox}{Paradox}</code>
	208 <code>\htXimera</code>	<code>\ConfigureTheoremEnv{paradox}</code>
<code>procedure (env.)</code>	Procedure	
	209 <code>\classXimera</code>	<code>\newtheorem{procedure}{Procedure}</code>
	210 <code>\htXimera</code>	<code>\ConfigureTheoremEnv{procedure}</code>

remark (<i>env.</i>)	Remark
211	<code><classXimera> \newtheorem{remark}{Remark}</code>
212	<code><htXimera> \ConfigureTheoremEnv{remark}</code>
summary (<i>env.</i>)	Summary
213	<code><classXimera> \newtheorem{summary}{Summary}</code>
214	<code><htXimera> \ConfigureTheoremEnv{summary}</code>
template (<i>env.</i>)	Template
215	<code><classXimera> \newtheorem{template}{Template}</code>
216	<code><htXimera> \ConfigureTheoremEnv{template}</code>
warning (<i>env.</i>)	Warning
217	<code><classXimera> \newtheorem{warning}{Warning}</code>
218	<code><htXimera> \ConfigureTheoremEnv{warning}</code>

2.4.3 Enumerate fixes

Make enumerate use a letter

```

219 <*classXimera>
220 \renewcommand{\theenumi}{\textup{(\alph{enumi})}}
221 \renewcommand{\labelenumi}{\theenumi}
222 \renewcommand{\theenumii}{\textup{(\roman{enumii})}}
223 \renewcommand{\labelenumii}{\theenumii}
224 </classXimera>

```

2.4.4 Proofs

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

```

225 <*classXimera>
226 \renewcommand{\qedsymbol}{\blacksquare$}
227 \renewenvironment{proof}[1][\proofname]
228 {
\begin{trivlist}
\item[\hskip \labelsep \itshape \bfseries #1\hspace{2ex}}
229 {\qed\end{trivlist}}
230 </classXimera>
231 <*htXimera>
232 % Mmm, (why) do we want/need this ...?
233 \ConfigureTheoremEnv{proof}
234 \ConfigureEnv{proof}{\ifvmode\IgnorePar\fi\EndP\HCode{<div class="proof">}
235 \ConfigureList{trivlist}{\ifvmode\IgnorePar\fi\EndP}{\}{\}}
236 {\ifvmode\IgnorePar\fi\EndP\HCode{</div>}}{\}{\}}
237 </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>

```

238 <*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'.

```

239 \providecommand{\latexProblemContent}[1]{#1}
240 % Iterate count for problem counts.
241 \Make@Counter{Iteration@probCnt}

242 \newcommand{\hang}{% top theorem decoration
243 \begingroup%
244 \setlength{\unitlength}{.005\linewidth}% \linewidth/200
245 \begin{picture}(0,0)(1.5,0)%
246 \linethickness{1pt} \color{black!50}%
247 \put(-3,2){\line(1,0){206}}% Top line
248 \multido{\iA=2+-1,\iB=50+-10}{5}{% Top hangs

```

```

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

```

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.

```

272 \MakeCounter{problem}
273 \newcommand{\problemNumber}{%
274     % First we determine if we have a counter for this question depth level.
275     \ifcsname c@depth\Roman{problem@Depth}Count\endcsname% Check to see if counter exists
276     %If so, do nothing.
277     \else
278     %If not, create it.
279     \expandafter\newcounter{depth\Roman{problem@Depth}Count}
280     \expandafter\setcounter{depth\Roman{problem@Depth}Count}{0}
281     \fi
282
283     \expandafter\stepcounter{depth\Roman{problem@Depth}Count}
284     \arabic{depth\Roman{problem@Depth}Count}% The first problem depth, what use to be |\theproblem|.
285
286     \forloop{Iteration@probCnt}{2}{\arabic{Iteration@probCnt} < \numexpr \value{problem@Depth} +
287         .\expandafter\arabic{depth\Roman{Iteration@probCnt}Count}}% Get the problem number of the
288 }
289 %\@ifpackageloaded{shuffle}{<true>}{<false>}% Check if Shuffle has been added. If so, add sp
290 %\ifhandout % Currently handout mode doesn't allow hints. Putting this code in place in case
291 % \theproblem
292 %\else
293 % \theproblem
294 %\fi
295 }
296
297
298 %%%% Configure various problem environment commands
299 \Make@Counter{problem@Depth}
300
301
302
303 %%%% Configure environments start content
304
305 \newcommand{\problemEnvironmentStart}[2]{%
306     % This takes in 2 arguments.
307     % The first is optional and is the old optional argument from existing environments.
308     % This is passed down to the associated problem environment name in case you want a global va

```



```

309 % The second argument is mandatory and is the name of the 'problem' environment,
310 % such as problem, question, exercise, etc.
311 % It then configures everything needed at the start of that environment.
312
313 \stepcounter{problem@Depth}% Started a problem, so we've sunk another problem layer.
314 \def\spaceatend{#1}%
315 \begin{trivlist}%
316 \item%
317   [%
318     \hskip\labelsep\sffamily\bfseries
319     #2 \problemNumber% Determine the correct number of the problem, and the format of that number.
320   ]%
321 \slshape
322 }
323
324
325
326 %%%% Configure environments end content
327
328 \newcommand{\problemEnvironmentEnd}{%This configures all the end content for a problem.
329 %
330 % First we need to see if we've dropped fully out of a depth level,
331 % so we can reset that counter back to zero for the next time we enter that depth level.
332 \stepcounter{problem@Depth}
333 \ifcsname c@depth\Roman{problem@Depth}Count\endcsname
334 \expandafter\ifnum\expandafter\value{depth\Roman{problem@Depth}Count}>0
335 \expandafter\setcounter{depth\Roman{problem@Depth}Count}{0}
336 \fi
337 \fi
338
339 \addtocounter{problem@Depth}{-2}% Exited a problem so we've exited a problem layer. Need -2 because we incremented twice.
340
341 % 202410: no longer \par\addvspace{.5ex}\nobreak\noindent\hung %% line at the bottom
342
343 \ifhandout
344 \ifnewpage
345 \newpage
346 \fi
347 \fi
348 \end{trivlist}
349 }
350
351
352
353 %%%% Now populate the old environment names
354 %
355 % Old environments were "problem", "exercise", "exploration", and "question".
356 % Note that you can add content to the start/end code on top of these base code pieces if you want.
357 %
358 % These definitions will be overwritten in ximera.4ht !
359
360
361 \newenvironment{problem}[1][2in]%
362 {%Env start code
363 \problemEnvironmentStart{#1}{Problem}
364 }
365 {%Env end code
366 \problemEnvironmentEnd
367 }
368
369 \newenvironment{exercise}[1][2in]%
370 {%Env start code
371 \problemEnvironmentStart{#1}{Exercise}

```

```

372 }
373 {%Env end code
374 \problemEnvironmentEnd
375 }
376
377 \newenvironment{exploration}[1][2in]%
378 {%Env start code
379 \problemEnvironmentStart{#1}{Exploration}
380 }
381 {%Env end code
382 \problemEnvironmentEnd
383 }
384
385 \newenvironment{question}[1][2in]%
386 {%Env start code
387 \problemEnvironmentStart{#1}{Question}
388 }
389 {%Env end code
390 \problemEnvironmentEnd
391 }
392 \end{classXimera}

```

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

```

393 \begin{Ximera}
394 \newcounter{identification}
395 \setcounter{identification}{0}
396
397 % 2024: should perhaps better have been called \ConfigureProblemEnv ...??
398 \newcommand{\ConfigureQuestionEnv}[2]{%
399 % refstepcounter ensures that labels get updated within these environments
400 \renewenvironment{#1}{\refstepcounter{problem}}{}%
401 \ConfigureEnv{#1}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div role="
402 }
403
404 \ConfigureQuestionEnv{problem}{problem}
405 \ConfigureQuestionEnv{exercise}{exercise}
406 \ConfigureQuestionEnv{question}{question}
407 \ConfigureQuestionEnv{exploration}{exploration}
408
409 \ifdefined\xmNotHintAsExpandable
410 \ConfigureQuestionEnv{hint}{hint} % 2024: hint is no longer a 'question-environment'.
411 \fi
412 %%%\ConfigureQuestionEnv{shuffle}{shuffle}
413 \end{Ximera}

```

2.4.6 Hints

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

```
414 \begin{classXimera}
```

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

```
415 \newcounter{hintLevel}
416 \setcounter{hintLevel}{0}
```

Create an empty shell to renew

```
417 \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.

```
418 \renewenvironment{hint}
419 {
420 \ifhandout
```

```

421 \setbox0\vbox\bgroup
422 \else
423 \begin{trivlist}\item[\hskip \labelsep\small\slshape\bfseries Hint:\hspace{2ex}]
424 \small\slshape
425 \fi

```

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

```

426 \stepcounter{hintLevel}
427 }
428 {
429 \ifhandout
430 \egroup\ignorespacesafterend
431 \else
432 \end{trivlist}
433 \fi

```

Detract from hint level counter to track hint nested level

```

434 \addtocounter{hintLevel}{-1}
435 }
436
437 \ifhints
438 \renewenvironment{hint}{
439 \begin{trivlist}\item[\hskip \labelsep\small\slshape\bfseries Hint:\hspace{2ex}]
440 \small\slshape
441 {\end{trivlist}}
442 \fi
443
444 \end{classXimera}

```

2.4.7 Solution

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

```

445 \begin{classXimera}
446 %% solution environment
447 \ifhandout % what follows is handout behavior
448 \newenvironment{solution}%
449     {%
450     \setbox0\vbox\bgroup
451     }
452     {%
453     \egroup
454     }
455 \else
456 \newenvironment{solution}%
457     {%
458     \begin{trivlist}
459     \item[\hskip \labelsep\bfseries Solution:\hspace{2ex}]
460     }
461     % %% line at the bottom}
462     {
463     \end{trivlist}
464     % (202410: no longer \par\addvspace{.5ex}\nobreak\noindent\hung
465     }
466 \fi
467
468
469
470 \end{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.

```
471 \classXimera
472 \DefineVerbatimEnvironment{code}{Verbatim}{numbers=left,frame=lines,label=Code,labelposition=
473 \end{code}}
474 \end{classXimera}
```

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

```
474 \classXimera
475 \DefineVerbatimEnvironment{python}{Verbatim}{numbers=left,frame=lines,label=Python,labelposition=
476 \end{python}}
477 \end{classXimera}
```

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

```
477 \classXimera
478 \DefineVerbatimEnvironment{javascriptCode}{Verbatim}{numbers=left,frame=lines,label=JavaScript
479 \end{javascriptCode}}
480 \classXimera
481 \renewenvironment{javascriptCode}{\NoFonts}{\EndNoFonts}
482 \ScriptEnv{javascriptCode}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<d
483 \end{javascriptCode}}
484 \end{classXimera}
```

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

```
484 %\classXimera
485 %\ConfigureEnv{verbatim}{\ifvmode\IgnorePar\fi\EndP\HCode{<pre style="white-space: pre; backgroun
486 %\ConfigureEnv{lstlisting}{\ifvmode\IgnorePar\fi\EndP\HCode{<pre>}}{\ifvmode\IgnorePar\fi\EndP
487 %\end{classXimera}
488 %
```

2.4.9 Dialogues

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

```
489 \classXimera
490 \newenvironment{dialogue}{%
491 \renewcommand\descriptionlabel[1]{\hspace{\labelsep}\textbf{##1:}}
492 \begin{description}%
493 }{%
494 \end{description}%
495 }
496 \end{classXimera}
```

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

```
497 \classXimera
498 \renewenvironment{dialogue}{\begin{description}}{\end{description}}
499
500 \ConfigureList{dialogue}%
501 {\EndP\HCode{<dl \a:LRdir class="dialogue">}}%
502 \PushMacro\end:itm
503 \global\let\end:itm=\empty
504 {\PopMacro\end:itm \global\let\end:itm \end:itm \end:itm
505 \EndP\HCode{</dd></dl>}\ShowPar}
506 {\end:itm \global\def\end:itm{\EndP\Tg</dd>}\HCode{<dt
507 class="actor">}\bgroup \bf}
508 {\egroup\EndP\HCode{</dt><dd\Hnewline class="speech">}}
509 \end{classXimera}
```

2.4.10 Instructor notes

```
510 \classXimera
511
512 %\instructor intro/instructor notes
513 %
514 \ifhandout % what follows is handout behavior
515 \ifinstructornotes
516 \newenvironment{instructorIntro}%
```

```

517         {%
518         \begin{trivlist}
519         \item[\hskip \labelsep\bfseries Instructor Introduction:\hspace{2ex}]
520         }
521         %% line at the bottom}
522         {
523         \end{trivlist}
524         \par\addvspace{.5ex}\nobreak\noindent\hung
525         }
526     \else
527     \newenvironment{instructorIntro}{%
528         {%
529         \setbox0\vbox\bgroup
530         }
531         {%If this mysteriously starts breaking
532         % remove \ignorespacesafterend
533         \egroup\ignorespacesafterend
534         }
535         \fi
536     \else% for handout, so what follows is default
537     \ifinstructornotes
538     \newenvironment{instructorIntro}{%
539         {%
540         \setbox0\vbox\bgroup
541         }
542         {%
543         \egroup
544         }
545         \else
546         \newenvironment{instructorIntro}{%
547         {%
548         \begin{trivlist}
549         \item[\hskip \labelsep\bfseries Instructor Introduction:\hspace{2ex}]
550         }
551         %% line at the bottom}
552         {
553         \end{trivlist}
554         \par\addvspace{.5ex}\nobreak\noindent\hung
555         }
556         \fi
557     \fi
558
559
560
561
562 %% instructorNotes environment
563 \ifhandout % what follows is handout behavior
564 \ifinstructornotes
565 \newenvironment{instructorNotes}{%
566     {%
567     \begin{trivlist}
568     \item[\hskip \labelsep\bfseries Instructor Notes:\hspace{2ex}]
569     }
570     %% line at the bottom}
571     {
572     \end{trivlist}
573     \par\addvspace{.5ex}\nobreak\noindent\hung
574     }
575     \else
576     \newenvironment{instructorNotes}{%
577         {%
578         \setbox0\vbox\bgroup
579         }

```

```

580   {%
581   \egroup
582   }
583   \fi
584 \else% for handout, so what follows is default
585 \ifinstructornotes
586 \newenvironment{instructorNotes}%
587   {%
588   \setbox0\vbox\bgroup
589   }
590   {%
591   \egroup
592   }
593   \else
594   \newenvironment{instructorNotes}%
595     {%
596     \begin{trivlist}
597     \item[\hskip \labelsep\bfseries Instructor Notes:\hspace{2ex}]
598     }
599     %% line at the bottom}
600     {
601     \end{trivlist}
602     \par\addvspace{.5ex}\nobreak\noindent\hung
603     }
604     \fi
605     \fi
606
607 \end{classXimera}

```

2.4.11 Only

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

```

608 \end{classXimera}
609 \ifxake
610   \newenvironment{prompt}{}{}
611 \else
612 \ifhandout
613   \NewEnviron{prompt}{}
614 % Currently breaks when put in mathmode!
615 % \newenvironment{prompt}{\suppress}{\endsuppress}
616 \else
617   \newenvironment{prompt}
618     {\bgroup\color{gray!50!black}}
619     {\egroup}
620 \fi
621 \fi

```

onlineOnly (*env.*) Only display it online

```

622 \ifhandout
623   \NewEnviron{onlineOnly}{
624     \iftikzexport
625       \BODY
626     \else
627       \fi
628   }
629 \else
630   \newenvironment{onlineOnly}
631     {\bgroup\color{red!50!black}}
632     {\egroup}
633 \fi
634
635 \newcommand{\pdfOnly}[1]{\iftikzexport\else #1\fi}
636 \end{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.

```

637 %\iftikzexport\else\RequirePackage[framemethod=TikZ]{mdframed}\fi
foldable (env.) Does it fold?
638 \classXimera>
639
640 \colorlet{textColor}{black} % since textColor is referenced below
641 \colorlet{background}{white} % since background is referenced below
642
643 % The core environments. Find results in 4ht file.
644 %% pretty-foldable
645 %\iftikzexport
646 \newenvironment{foldable}{%
647 }{%
648 }
649 %\else
650 %\renewmdenv[
651 %   font=\upshape,
652 %   outerlinewidth=3,
653 %   topline=false,
654 %   bottomline=false,
655 %   leftline=true,
656 %   rightline=false,
657 %   leftmargin=0,
658 %   innertopmargin=0pt,
659 %   innerbottommargin=0pt,
660 %   skipbelow=\baselineskip,
661 %   linecolor=textColor!20!white,
662 %   fontcolor=textColor,
663 %   backgroundcolor=background
664 %]{foldable}%
665 %\fi
666
667 %% pretty-expandable
668 %\iftikzexport
669 %% Overwritten in .4ht, but probably also in accordion!
670 \ifdefined\xmNotExpandableAsAccordion
671 \newenvironment{expandable}{}{}
672 \else
673 \newenvironment{expandable}[2]{}{}
674 \fi
675 %\else
676 %\newmdenv[
677 %   font=\upshape,
678 %   outerlinewidth=3,
679 %   topline=false,
680 %   bottomline=false,
681 %   leftline=true,
682 %   rightline=false,
683 %   leftmargin=0,
684 %   innertopmargin=0pt,
685 %   innerbottommargin=0pt,
686 %   skipbelow=\baselineskip,
687 %   linecolor=black,
688 %]{expandable}%
689 %\fi
690
691 \newcommand{\unfoldable}[1]{#1}
692
693 \endclassXimera

```

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

```

694 <*htXimera>
695 \renewenvironment{foldable}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<
696
697 \ifdefined\xmNotExpandableAsAccordion
698 \renewenvironment{expandable}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{
699
700 }{\HCode{</div>}\IgnoreIndent}
701 \fi
702
703 \renewcommand{\unfoldable}[1]{\HCode{<span class="unfoldable">}#1\HCode{</span>}}
704 </htXimera>

```

2.4.13 Leashes

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

```

705 <*classXimera>
706
707 \newenvironment{leash}[1]{%
708 }{%
709 }
710
711
712 </classXimera>
713 <*htXimera>
714 \renewenvironment{leash}[1]{\ifvmode \IgnorePar\fi \EndP\HCode{<div style="overflow: auto; h
715 </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.

```

716 <*classXimera>
717 \newcommand{\license}{\excludecomment}
718 </classXimera>

```

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

```

719 <*classXimera>
720 \newcommand{\acknowledgement}{\excludecomment}
721 </classXimera>

```

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

```

722 <*classXimera>
723 \renewcommand{\tag}{\excludecomment}
724 </classXimera>

```

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

```

725 <*htXourse>
726 % Mark this as a xourse file
727 \Configure{@HEAD}{\HCode{<meta name="description" content="xourse" />\Hnewline}}
728 </htXourse>

```

2.5.2 Abstract

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

```

729 <*classXimera>
730 \let\abstract\relax
731 \let\endabstract\relax

```



```

732 % Use of environ package, may want to find a better way.
733 % see the messing around with \theabstract in title.dtx ... Is this really needed/wanted?
734 \NewEnviron{abstract}{\protected@xdef\theabstract{\BODY}}
735 \endclassXimera

```

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.

```

736 \begin{classXimera}
737 \ConfigureEnv{abstract}{\HCode{\Hnewline<div class="abstract">}}{\HCode{\Hnewline</div>}}{}{}
738 \end{classXimera}

739 \begin{htXimera}
740 \RenewEnviron{abstract}{\BODY}
741 \end{htXimera}

```

2.5.3 Titles and authors

2.5.4 Authors

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

```

742 \begin{classXimera}
743 \let\emptyauthor\@author
744 \def\author#1{\gdef\@author{#1}}
745 \def\@author{\@latex@warning@no@line{No \noexpand\author given}}
746 \end{classXimera}

```

Include author name in meta tags

```

747 \begin{htXimera}
748 \Configure{@HEAD}{\HCode{<meta name="author" content="}\@author\HCode{" />\Hnewline}}
749 \end{htXimera}

```

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

```

750 \begin{htXimera} \begin{classXimera} \def\and{and }

```

2.5.5 Title

`\title` Activities have titles.

```

751 \begin{classXimera}
752 \let\title\relax
753 \newcommand{\title}[1] []{{\protected@xdef\@prettitle{#1}}\protected@xdef\@title{#1}}
754
755 \title{}
756
757 \newcounter{titlenumber}
758 \renewcommand{\thetitlenumber}{\arabic{titlenumber}}
759 %\renewcommand{\thesection}{\arabic{titlenumber}} %% Makes section numbers work
760 \setcounter{titlenumber}{0}
761
762 \newpagestyle{main}{
763 \sethead[\textsl{\ifnumbers\thetitlenumber\hspace{1em}\fi\@title}] [] [] % even
764 {}{}{\textsl{\ifnumbers\thetitlenumber\hspace{1em}\fi\@title}} % odd
765 \setfoot[\thepage] [] [] % even
766 {}{}{\thepage} % odd
767 }
768 \pagestyle{main}

```

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

```

769 \renewcommand\maketitle{%
770 \addtocounter{titlenumber}{1}%
771 {\flushleft\large\bfseries \@prettitle\par\vspace{-1em}}
772 {\flushleft\large\bfseries {\ifnumbers\thetitlenumber\fi}{\ifnumbers\hspace{1em}\else\hspace{1em}}}
773 \phantomsection%
774 \ifnumbers\addcontentsline{toc}{section}{\thetitlenumber~\@title}\else\addcontentsline{toc}{section}{\@title}

```

```

775 \vskip .6em\noindent\textit\theabstract\setcounter{problem}{0}\setcounter{section}{0}\setc
776 \ifnooutcomes\else\let\thefootnote\relax\footnote{Learning outcomes: \theoutcomes}\fi
777 \ifnoauthor\else\let\thefootnote\relax\footnote{Author(s):~\@author}\fi
778 \aftergroup\@afterindentfalse
779 \aftergroup\@afterheading}
780
781 \ifnumbers
782 \setcounter{secnumdepth}{2}
783 \renewcommand{\thesection}{\arabic{titlenumber}.\arabic{section}}
784 \renewcommand{\thesubsection}{\arabic{titlenumber}.\arabic{section}.\arabic{subsection}}
785 \else
786 \setcounter{secnumdepth}{-2}
787 \fi
788
789 \def\activitystyle{}
790 \newcounter{sectiontitlenumber}
791 \setcounter{secnumdepth}{2}
792 \setcounter{tocdepth}{2}
793 \newcommand\chapterstyle{%
794   \def\activitystyle{activity-chapter}
795   \def\maketitle{%
796     \addtocounter{titlenumber}{1}%
797     {\flushleft\small\sffamily\bfseries\@pretitle\par\vspace{-1.5em}}%
798     {\flushleft\LARGE\sffamily\bfseries\thetitlenumber\hspace{1em}\@title \par}
799     {\vskip .6em\noindent\textit\theabstract\setcounter{problem}{0}\setcount
800     \par\vspace{2em}
801     \phantomsection\addcontentsline{toc}{section}{\textbf{\thetitlenumber\hspace{1em}\@title}}
802   }}
803
804
805 \newcommand\sectionstyle{%
806   \def\activitystyle{activity-section}
807   \def\maketitle{%
808     \addtocounter{section}{1}
809     \setcounter{sectiontitlenumber}{\value{section}}
810     {\flushleft\small\sffamily\bfseries\@pretitle\par\vspace{-1.5em}}%
811     {\flushleft\Large\sffamily\bfseries\thetitlenumber.\thesectiontitlenumber\hspace{1em}\@title}
812     {\vskip .6em\noindent\textit\theabstract\setcounter{subsection}{0}}%
813     \par\vspace{2em}
814     \phantomsection\addcontentsline{toc}{section}{\thetitlenumber.\thesectiontitlenumber\hspace{1em}\@title}
815   \renewcommand\section{\@startsection{subsection}{2}{\z@}%
816     {-3.25ex\@plus -1ex \@minus -.2ex}%
817     {1.5ex \@plus .2ex}%
818     {\normalfont\large\bfseries}}
819
820   \renewcommand\subsection{\@startsection{subsubsection}{3}{\z@}%
821     {-3.25ex\@plus -1ex \@minus -.2ex}%
822     {1.5ex \@plus .2ex}%
823     {\normalfont\normalsize\bfseries}}
824
825   }}
826
827
828 \iftikzexport% allows xake to handle \chapterstyle and \sectionstyle
829 \renewcommand\chapterstyle{\def\activitystyle{chapter}}
830 \renewcommand\sectionstyle{\def\activitystyle{section}}
831 \else
832 \fi
833
834 </classXimera>

```

Eliminate some formatting that we'll handle later with CSS

```

835 <htXimera>
836 \renewcommand{\maketitle}{}

```

```
837 </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.

```
838 <*classXimera>
839 \def\theoutcomes{}
840
841 \ifdefined\HCode%
842   \newcommand{\outcome}[1]{%
843     \else%
844     \newwrite\outcomefile
845     \immediate\openout\outcomefile=\jobname.oc
846
847     \newcommand{\outcome}[1]{\edef\theoutcomes{\theoutcomes #1~}%
848       \immediate\write\outcomefile{\unexpanded{\outcome}{#1}}}
849   \fi%
850 </classXimera>
```

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

```
851 <*cfgXimera>
852 \renewcommand{\outcome}[1]{
853   \Configure{@HEAD}{\HCode{<meta name="learning-outcome" content="#1"/>\Hnewline}}
854 }
855 % Sometimes there are no outcomes at all
856 \IfFileExists{\jobname.oc}{\input{\jobname.oc}}{}
857
858 \renewcommand{\outcome}[1]{%
859   \HCode{<span class="learning-outcome">#1</span>}}
860 }
861 </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.

```
862 <*htXimera>
863 \let\oldlabel\label
864 \renewcommand{\label}[1]{\oldlabel{#1}\HCode{<a class="ximera-label" id="#1"></a>}}
865 </htXimera>
```

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

```
866 <*htXimera>
867 \renewcommand{\ref}[1]{\HCode{<a class="reference" href="\##1">#1</a>}}
868 </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. Base graphicspath, default `'/xmPictures'`. Can only be changed BEFORE loading ximera.cls!

```
869 <*classXimera>
870 % Provide a default graphicspath
871 % (somewhat tricky: an activity can be included in a xourse in a wildly different path !)
872 % Suggested convention: put all images in i /pictures folder in the root of your project
873 \providecommand\xmDefaultGraphicsPath{/xmPictures}
874 \graphicspath{ %% When looking for images,
875 {./}           %% look here first,
```

```

876 {\xmDefaultGraphicsPath/} %% then look for a pictures folder,
877 {\xmDefaultGraphicsPath/} %% then look for a pictures folder,
878 {\xmDefaultGraphicsPath/} %% then look for a pictures folder,
879 {\xmDefaultGraphicsPath/} %% then look for a pictures folder,
880 }
881 \newenvironment{image}[1][\begin{center}]{\end{center}}
882 \NewEnviron{image}[1][3in]{%
883   \begin{center}\resizebox{#1}{!}{\BODY}\end{center}% resize and center
884 }
885 \end{classXimera}

```

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

```

886 \begin{classXimera}
887 \newcommand{\alt}[1]{%
888 }
\end{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.

```

889 \begin{htXimera}
890 \newcounter{imagealt}
891 \setcounter{imagealt}{0}
892 \renewenvironment{image}[1][\stepcounter{imagealt}%
893   \ifvmode \IgnorePar\fi \EndP%
894   \HCode{<div class="image-environment" role="img" aria-labelledby="image-alt-\arabic{imagealt}>}
895   \HCode{</div>}}
896 \renewcommand{\alt}[1]{\HCode{<div style="display: none;" id="image-alt-\arabic{imagealt}>}>}
897 \end{htXimera}
898 \begin{cfgXimera}
899 %% Although we accept many formats, SVG is preferred on the web.
900 %% Since we have a different mechanism for producing |alt| text, we
901 %% want to ignore tex4ht's own method fo producing alt text.
902 %% 2024: is now in TeX4ht ...
903 % \DeclareGraphicsExtensions{.jpg,.png,.gif,.svg}
904 % \Configure{graphics*}
905 % {svg}{
906 %   \Configure{Needs}{File: \Gin@base.svg}\Needs{}}
907 %   \Picture[]{\csname Gin@base\endcsname.svg \csname a:Gin-dim\endcsname}%
908 % }
909 \end{cfgXimera}

```

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

```

910 \begin{cfgXimera}
911 \ifcsname ifstandalone\endcsname
912   \ifstandalone
913     \renewcommand\includegraphics[2][{}]{%
914       \fi
915     }
\end{cfgXimera}

```

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

```

916 \begin{htXimera}
917 \providecommand{\pgfsyspdfmark}[3]{%
918 }
\end{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.

```

919 \begin{classXimera}

```

```

920 % everything skipped, assume TeX4ht does the job now
921 \ifdefined\reallyneverever
922
923 \ifdefined\HCode
924   \tikzexporttrue
925 \fi
926
927 \iftikzexport
928   \usetikzlibrary{external}
929
930   \ifdefined\HCode
931     % in htlatex, just include the svg files
932     \def\pgfsys@imagesuffixlist{.svg}
933
934     \tikzexternalize[prefix=./,mode=graphics if exists]
935   \else
936     % in pdflatex, actually generate the svg files
937     \tikzset{
938       /tikz/external/system call={
939         pdflatex \tikzexternalcheckshellescape
940         -halt-on-error -interaction=batchmode
941         -jobname "\image" "\PassOptionsToClass{tikzexport}{ximera}\texsource";
942         mutool draw -F svg \image.pdf > \image.svg ;      % mutool adds "1" to filename ???
943         mutool draw -o \image.svg \image.pdf ;
944         mutool draw -r 150 -c rgbalpha -o \image.png \image.pdf ;
945         ebb -x \image.png
946       }
947     }
948     \tikzexternalize[optimize=false,prefix=./]
949   \fi
950
951 \fi
952 \fi
953 \end{classXimera}

```

2.6.3 XKCD

\xkcd Reference an XKCD cartoon.

```

954 \begin{classXimera}
955 \newcommand{\xkcd}[1]{#1}
956 \end{classXimera}

```

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

```

957 \begin{htXimera}
958 \renewcommand{\xkcd}[1]{\ifvmode \IgnorePar\fi \EndP\HCode{<script type="text/
1069 \endhtXimera

```

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

```

1070 \classXimera
1071 \DefineVerbatimEnvironment{sageOutput}{Verbatim}{numbers=left,frame=lines,label=SAGE-Output,
1072 \endclassXimera

1073 \htXimera
1074 \renewenvironment{sageOutput}{\NoFonts}{\EndNoFonts}
1075 \ScriptEnv{sageOutput}{\ifvmode \IgnorePar\fi \EndP\HCode{<div class="sageOutput"><script typ
1076 \endhtXimera

```

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

```

1077 \htXimera
1078 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1079 \ifdefined\sagesilent
1080 \renewenvironment{sagesilent}{\NoFonts}{\EndNoFonts}
1081 \fi
1082 \ScriptEnv{sagesilent}{\ifvmode \IgnorePar\fi \EndP\HCode{<script type="text/sagemath">}\Htm
1083 \endhtXimera

```

2.10 Answerables

2.10.1 Answers

`\answer` A math answer

```

1084 \classXimera
1085
1086 \ifdefined\HCode
1087 \newcommand{\recordvariable}[1]{}
1088 \else
1089 \newwrite\idfile
1090 \immediate\openout\idfile=\jobname.ids
1091 \newcommand{\recordvariable}[1]{\ifthenelse{\equal{#1}{}}{\immediate\write\idfile{var #1}}{}
1092 \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

```

1093 \define@key{answer}{given}[true]{\def\ans@given{#1}}

```


Used for setting numeric answer tolerance for online student input.

```
1094 \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.

```
1095 \define@key{answer}{validator}{}
```

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

```
1096 \define@key{answer}{id}{\def\ans@id{#1}}
```

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

```
1097 \define@key{answer}{format}{}
```

Used to hide the answer input box on the web.

```
1098 \define@key{answer}{onlinenoinput}[false]{}
```

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

```
1099 \define@key{answer}{onlineshowanswerbutton}[false]{}
```

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

```
1100 \setkeys{answer}{id=,given=false,onlinenoinput=false,onlineshowanswerbutton=false}
```

Basic code for \answer.

```
1101
```

```
1102 % Options for handout
```

```
1103 \newcommand{\answerFormatLength}{2cm}
```

```
1104
```

```
1105 \newcommand{\answerFormatDots}[1]{\ldots\ldots}
```

```
1106 \newcommand{\answerFormatLine}[1]{\protect\rule{\answerFormatLength}{0.4pt}}
```

```
1107 \newcommand{\answerFormatFlexibleLine}[1]{\protect\rule{\widthof{${#1$}*2}{0.4pt}}
```

```
1108 \newcommand{\answerFormatFlexibleBox}[1]{\fbox{\scalebox{2}{\phantom{${#1$}}}}
```

```
1109
```

```
1110 % options for default (i.e with answers filled in)
```

```
1111 \newcommand{\answerFormatPlain}[1]{\ensuremath{#1}}
```

```
1112 \newcommand{\answerFormatBlue}[1]{\color{blue}\ensuremath{#1}}
```

```
1113 \newcommand{\answerFormatBoxed}[1]{\fbox{\ensuremath{#1}}}
```

```
1114 \newcommand{\answerFormatBoxedGiven}[1]{\underset{\scriptstyle\mathrm{given}}{\fbox{\ensuremath{#1}}}}
```

```
1115
```

```
1116 % defaults for handout and default mode, and for \answer[given]
```

```
1117 \let\handoutAnswerFormat\answerFormatDots
```

```
1118 \let\defaultAnswerFormat\answerFormatBlue
```

```
1119 \let\givenAnswerFormat\answerFormatBoxedGiven
```

```
1120
```

```
1121 \newcommand{\answer}[2][{}]{%
```

```
1122 \ifmode%
```

```
1123 \setkeys{answer}{#1}%
```

```
1124 \recordvariable{\ans@id}
```

```
1125 \ifthenelse{\boolean{\ans@given}}{
```

```
1126 {% Start then statement
```

```
1127 \ifhandout
```

```
1128 #2
```

```
1129 \else
```

```
1130 \givenAnswerFormat{#2} %% in case the argument helps formatting
```

```
1131 \fi
```

```
1132 }% End then statement
```

```
1133 {% Start else statement
```

```
1134 \ifhandout
```

```
1135 \handoutAnswerFormat{#2} %% in case the argument helps formatting
```

```
1136 \else% show answer in box outside handout mode
```

```
1137 \defaultAnswerFormat{#2} %% in case the argument helps formatting
```

```
1138 \fi
```

```
1139 }% End else statement
```

```
1140 \else%
```

```
1141 \GenericError{\space\space\space\space}% Throw an error based on... something? -- Jason
```

```
1142 {Attempt to use \@backslashchar answer outside of math mode}
```

```
1143 {See https://github.com/ximeraProject/ximeraLatex for explanation.}
```

```

1144 {Need to use either inline or display math.}%
1145 \fi
1146 }
1147 \endclassXimera

```

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

```

1148 \beginXimera
1149 \renewcommand{\answer}[2][false]{\HCode{<span class="answer_respondable">}#2\HCode{</span>}}
1150
1151 \def\validator[#1]{\stepcounter{identification}\HCode{<div class="validator" id="validator\answer">}}
1152 \def\endvalidator{\HCode{</div>}}
1153
1154 \endXimera

```

2.10.2 Multiple choice and the like

`multipleChoice` (*env.*) Multiple choice

```

1155 \beginXimera
1156 % Jim: Originally this was \renewcommand{\theenumi}{$(\mathrm{\alph{enumi}})$}
1157 % but that breaks tex4ht because mathmode can only be processed by mathjax.
1158 % so now I made this just italicized.

```

2.10.3 Options

```

1159 \define@key{choice}{value}[]{\def\choice@value{#1}}

```

This flags the answer as the correct answer

```

1160 \define@boolkey{choice}{correct}[true]{\def\choice@correct{#1}}

```

Use an ID to refer to the choice.

```

1161 \define@key{multipleChoice}{id}{\def\mc@id{#1}}

```

`\otherchoice` outputs the item if correct and nothing if incorrect.

```

1162 \define@key{otherchoice}{value}[]{\def\otherchoice@value{#1}}

```

```

1163 \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".

```

1164 \setkeys{choice}{correct=false,value=}

```

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

```

1165 \setkeys{multipleChoice}{id=}

```

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

```

1166 \setkeys{otherchoice}{correct=false,value=}

```

```

1167 \endclassXimera

```

2.10.4 Choices

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

```

1168 \beginXimera
1169 \newcommand{\choice}[2][]{%
1170 \setkeys{choice}{#1}%
1171 \item{#2}
1172 \ifthenelse{\boolean{\choice@correct}}{
1173   {% Begin then result
1174     \ifhandout% if it's a handout do nothing.
1175     \else% otherwise place a checkmark when you select the "correct choice"... maybe? -- Jason
1176       \, \checkmark \, \setkeys{choice}{correct=false}
1177     \fi
1178   }% End then result
1179   {% Begin/End else result.
1180 }
1181
1182 %Define an expandable version of choice Not really meant to be used outside this package (use

```

```

1183 % Is there a reason we can't just always use this as default? -- Jason
1184 \newcommand{\choiceEXP}[2][]{%
1185   \expandafter\setkeys\expandafter{\choice}{#1}%
1186   \item{#2}
1187   \ifthenelse{\boolean{\choice@correct}}{
1188     {% Begin then result
1189       \ifhandout
1190       \else
1191         \, \checkmark \, \setkeys{\choice}{correct=false}
1192       \fi
1193     }% End then result
1194   }{% Begin/End else result.
1195 } %% note all the {} are needed in case the choice has [] in it.
1196
1197 % \otherchoice is the \choice used in wordChoice command.
1198 \newcommand{\otherchoice}[2][]{%
1199   \ignorespaces%
1200   \setkeys{\otherchoice}{#1}%
1201   \ifthenelse{\boolean{\otherchoice@correct}}{
1202     {% Start then result
1203       #2\ignorespaces\setkeys{\otherchoice}{correct=false}\ignorespaces%
1204     }% End then result
1205   }{% Start/End else result
1206     \ignorespaces%
1207   }%
1208   \newcommand{\inlinechoice}[2][]{%
1209     \setkeys{\choice}{#1}%
1210     \iffirstinlinechoice
1211     (\hspace{-.25em}
1212     \firstinlinechoicefalse
1213     \else
1214     /
1215     \fi
1216     #2
1217     \ifthenelse{\boolean{\choice@correct}}{
1218       {% Start then result
1219         \ifhandout\else\checkmark\ignorespaces\setkeys{\choice}{correct=false}\ignorespaces\fi
1220       }% End then result
1221     }{% Start/End else result
1222       \hspace{-.25em}\ignorespaces%
1223     }
1224
1225   \</classXimera>

```

On the HTML side, \choice emits s.

```

1226 \*htXimera>
1227 \newcounter{choiceId}
1228 \renewcommand{\choice}[2][]{%
1229   \setkeys{\choice}{correct=false}%
1230   \setkeys{\choice}{#1}%
1231   \stepcounter{choiceId}\IgnorePar%
1232   \HCode{<span class="choice }%
1233   \ifthenelse{\boolean{\choice@correct}}{\HCode{correct}}{}
1234   \HCode{" }
1235   \ifthenelse{\equal{\choice@value}{}}{\HCode{data-value="\choice@value" }}
1236   \HCode{id="choice\arabic{choiceId}">}%
1237   #2\HCode{</span>}}
1238 \let\inlinechoice\choice
1239 \</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.

```

1240 <*classXimera>
1241 \newenvironment{multipleChoice}[1][
1242   {% Environment Start Code
1243   \setkeys{multipleChoice}{#1}%
1244   \recordvariable{\mc@id}%
1245   \begin{trivlist}
1246     \item[\hskip \labelsep\small\bfseries Multiple Choice:]\hfil
1247     \begin{enumerate}
1248       }% Note this means that \item has to be the first line after \begin{multipleChoice}.
1249     {% Environment End Code
1250     \end{enumerate}
1251     \end{trivlist}
1252   }
1253
1254   %multipleChoice@ is for internal use only! (used in wordChoice)
1255   %this is simply a wrapper for the sole showing (other)choice.
1256   \newenvironment{multipleChoice@}[1][{}]{}%
1257 </classXimera>

```

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

```

1258 <*htXimera>
1259 \renewenvironment{multipleChoice}[1][
1260   {\setkeys{multipleChoice}{#1}%
1261   \stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div class="multiple-choice"
1262   \ifthenelse{\equal{\mc@id}{}}{\}{\HCode{data-id="\mc@id" }}}%
1263   \HCode{id="problem\arabic{identification}">}}%
1264   {\HCode{</div>}\IgnoreIndent}
1265   \ConfigureEnv{multipleChoice}{\}{\}{\}
1266 </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.

```

1267 <*classXimera>
1268 \newcommand{\wordChoice}[1]{%
1269   \let\choicetemp\choice% Assign a "choicetemp" command to duplicate choice.
1270   \ifwordchoicegiven% If wordchoice option is on, we need to juggle around some definitions.
1271   \let\choice\otherchoice%
1272   %\begin{multipleChoice@}% -unnecessary (REMOVE THIS LINE IF THE YEAR IS 2019 or Beyond)
1273   #1
1274   %\end{multipleChoice@}% -unnecessary (REMOVE THIS LINE IF THE YEAR IS 2019 or Beyond)
1275   \else% If it isn't the regular "choice" command should work.
1276   \let\choice\inlinechoice%
1277   \begin{multipleChoice@}%
1278   #1%
1279   \end{multipleChoice@}%
1280   \fi%
1281   \let\choice\choicetemp% Now that choicetmp has been manipulated to what we want, replace choi
1282 }%
1283
1284
1285 </classXimera>

```

This is actually just word choice

```

1286 <*htXimera>
1287 \renewenvironment{multipleChoice@}{\refstepcounter{problem}}{\}%
1288 \ConfigureEnv{multipleChoice@}{\stepcounter{identification}\IgnorePar\HCode{<span class="word
1289 </htXimera>

```

2.12 Select all

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

```

1290 \*classXimera)
1291 \newenvironment{selectAll}[1]{}
1292 {\begin{trivlist}\item[\hskip \labelsep\small\bfseries Select All Correct Answers:]\hfil\begin{
1293   {\end{enumerate}\end{trivlist}}
1294 \*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`.

```

1295 \htXimera>
1296 \renewenvironment{selectAll}{\refstepcounter{problem}}{}%
1297 \ConfigureEnv{selectAll}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div
1298 \htXimera>

```

2.12.1 Free response

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

```

1299 (*classXimera)
1300 \newboolean{given} %% required for freeResponse
1301 \setboolean{given}{true} %% could be replaced by a key=value pair later if needed
1302
1303 \ifhandout
1304 \newenvironment{freeResponse}[1][false]%
1305 {%
1306 \def\givenatend{\boolean{#1}}
1307 \ifthenelse{\boolean{#1}}
1308 {% Begin then result
1309 \begin{trivlist}
1310 \item
1311 }% End then result
1312 {% Begin else result
1313 \setbox0\vbox\bgroup
1314 }% End else result
1315 % {}% Don't think this is doing anything? -- Jason
1316 }
1317 {%
1318 \ifthenelse{\givenatend}
1319 {% Begin then result
1320 \end{trivlist}
1321 }% End then result
1322 {% Begin else result
1323 \egroup
1324 }% End else result
1325 % {}% Don't think this is doing anything? -- Jason
1326 }
1327 \else
1328 \newenvironment{freeResponse}[1][false]%
1329 {% Environment Beginning Code
1330 \ifthenelse{\boolean{#1}}%% Could probably change this with just putting the (given) in
1331 {% Begin then result
1332 \begin{trivlist}
1333 \item[\hskip \labelsep\bfseries Free Response (Given):\hspace{2ex}]
1334 }% End then result
1335 {% Begin else result
1336 \begin{trivlist}
1337 \item[\hskip \labelsep\bfseries Free Response:\hspace{2ex}]
1338 }% End else result

```

```

1339 }
1340 {% Environment Ending Code
1341 \end{trivlist}
1342 }
1343 \fi
1344
1345 \end{classXimera}

1346 \htXimera
1347
1348 \renewenvironment{freeResponse}{\refstepcounter{problem}}{}%
1349 \ConfigureEnv{freeResponse}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<
1350
1351 \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 provided 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.

```

1352 \classXimera
1353 \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.

1354 \newenvironment{validator}[1][]{
1355 \def\PH@Command{#1}% Use PH@Command to hold the content and be a target for "\expandafter"
1356 \mbox{\texttt{\detokenize\expandafter{\PH@Command}}}% Now expand PH@Command once and then d
1357 }{}

```

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

```

1358 \ifhandout%
1359 \newenvironment{feedback}
1360     {%
1361     \setbox0\vbox\bgroup
1362     }
1363     {%
1364     \egroup
1365     }

```

If this isn't a handout, then we want to display the Feedback by using a label, positioned and formatted 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.

```

1366 \else

1367 \newenvironment{feedback}[1][attempt]{
1368
1369 \def\PH@Command{#1}% Use PH@Command to hold the content and be a target for "\expandafter"
1370
1371 \begin{trivlist}% Begin the trivlist to use formatting of the "Feedback" label.
1372 \item[\hspace{1em}\labelsep\small\slshape\bfseries Feedback% Format the "Feedback" label. Don't
1373 (\texttt{\detokenize\expandafter{\PH@Command}}):% Format (and detokenize) the condition for
1374 \hspace{2ex}]\small\slshape% Insert some space before the actual feedback given.
1375 }{
1376 \end{trivlist}
1377 }
1378
1379 \fi
1380 \end{classXimera}

```

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

```

1381 <*htXimera>
1382 \def\feedback{\@ifnextchar[{\@feedbackcode}{\@feedbackattempt}}
1383 \def\@feedbackattempt{\@feedbackcode[attempt]}
1384 \def\@feedbackcode[#1]{\stepcounter{identification}%
1385 \ifvmode \IgnorePar\fi \EndP%
1386 \ifthenelse{equal{#1}{attempt}}{\HCode{<div class="feedback" data-feedback="attempt" id="fe
1387 {\ifthenelse{equal{#1}{correct}}{\HCode{<div class="feedback" data-feedback="correct" id="f
1388 {\HCode{<div class="feedback" data-feedback="script" id="feedback\arabic{identification}"><s
1389 \def\endfeedback{\HCode{</div>}}\IgnoreIndent}
1390 </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 L^AT_EX side, the **ungraded** environment does nothing.

```

1391 <*classXimera>
1392 \newenvironment{ungraded}{}{}
1393 </classXimera>

```

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

```

1394 <*htXimera>
1395 \renewenvironment{ungraded}{%
1396 \ifvmode \IgnorePar\fi \EndP\HCode{<div class="ungraded">}\IgnoreIndent%
1397 }{
1398 \ifvmode \IgnorePar\fi \EndP\HCode{</div>}\IgnoreIndent%
1399 }
1400 </htXimera>

```

2.13 Support for the web

2.13.1 MathJax support

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

First, create the .jax file. Redefine newcommand appropriately.

```

1401 <*classXimera>
1402 %% Pre-202412: .jax file written in non-\HCode, and in a next run inserted by ximera.cfg in
1403 %% Post-202412: .mjax file written in \HCode, and in luaxake post-processing inerted in .html
1404 %% For backward-compatibility, the pre-202412 code is kept around for some time
1405 %% (and the extension .mjax was used to make both versions coexist...)
1406 \newwrite\myfile
1407 \ifdefined\HCode
1408 \immediate\openout\myfile=\jobname.xmjax
1409 \else
1410 \immediate\openout\myfile=\jobname.jax
1411 \fi
1412
1413 %% From |only.dtx| we must also create |prompt| on the MathJax side.
1414 \immediate\write\myfile{\unexpanded{\newenvironment}{prompt}}{}{}
1415
1416 %% Write all newcommands to .xmjax file, that will be included in the .html via luaxake
1417 \let\oldargdef\argdef
1418 \long\def\argdef#1[#2]#3{%
1419 \immediate\write\myfile{\unexpanded{\newcommand}{\unexpanded{#1}}[\unexpanded{#2}]{\unexpanded{#3}}%
1420 \oldargdef#1[#2]{#3}%
1421 }
1422
1423 %% Same for \DeclareMathOperator
1424 \let\oldDeclareMathOperator\DeclareMathOperator

```

```

1425 \renewcommand{\DeclareMathOperator}[2]{\@oldDeclareMathOperator{#1}{#2}\immediate\write\myfi
1426
1427 \end{classXimera}
Include the jax'ed newcommands (pre-202412 versions ....)
1428 \begin{cfgXimera}
1429 % Remove commands that use @
1430 \immediate\write18{sed -i "/[:*@]/d" \jobname.jax}
1431 % Replace ##1 with #1 and so forth
1432 \immediate\write18{sed -i "s/\string#\string#\string\([0-9]\string\)/\string#\string\1/g"
1433
1434 \Configure{BVerbatimInput}{-}{-}{-}
1435
1436 \Configure{verbatiminput}{-}{-}{-}
1437
1438 % Instead of a nonbreaking space, use a standard space
1439 \makeatletter
1440 \def\FV@Space{\space}
1441 \makeatother
1442
1443 % Include the mathjax newcommands in a math/tex script right at the beginning of the body
1444 % (post 202412: this will hopefully (only) be done via luaxake post-processing!)
1445 \Configure{BODY}{%
1446 \HCode{<body>\Hnewline}%
1447 \Tg<div class="preamble">%
1448 %% If there is a .jax file, but no .xmjax file: include it
1449 %% (If there is only a .xmjax file, it will presumably be included by luaxake post-processing)
1450 %% Once post-202412 functionality is considered stable, this whole thing can be removed here
1451 \IfFileExists{\jobname.jax}{
1452 \IfFileExists{\jobname.xmjax}{
1453 %% DO NOTHING HERE, as the .xmjax file will presumably be added to the .html by luaxake
1454 }{
1455 \Tg<script type="math/tex">%
1456 \BVerbatimInput{\jobname.jax}%
1457 \Tg</script>%
1458 }}
1459 \Hnewline\HCode{<!--Mmm, no newcommands provided -->}\Hnewline}
1460
1461 %% Include the .ids file
1462 \IfFileExists{\jobname.ids}{\HCode{<script type="text/javascript">\Hnewline}%
1463 \BVerbatimInput{\jobname.ids}%
1464 \HCode{</script>\Hnewline}%
1465 }{
1466 \Tg</div>%
1467 }{
1468 \ifvmode\IgnorePar\fi\EndP\HCode{</body>\Hnewline}%
1469 }
1470
1471 % prevent spaces as in "\begin{align}" (it confuses Mathjax2)
1472 \renewcommand\VerbMathToks[2]{%
1473   \HCode{\string\begin{#2}}%
1474   \alreqtoks{#1}%
1475   \HCode{\string\end{#2}}%
1476 }
1477
1478 % This is a fix for the LAODE book, which uses matlabEquation as if it were an equation
1479 \ScriptEnv{matlabEquation}{\ifvmode \IgnorePar\fi \EndP\HCode{<script type="math/tex; mode=d
1480
1481 \end{cfgXimera}

```

2.13.2 Semantic HTML

`\textbf` Using `\textbf` emits a `` tag.

```

1482 \begin{cfgXimera}

```



```

1483 \Configure{textbf}{\ifvmode\ShowPar\fi\HCode{<strong>}}{\HCode{</strong>}}
1484 \</cfgXimera>

\textit Using \textit or similar emits an <em> tag.
1485 \<cfgXimera>
1486 \Configure{textit}{\ifvmode\ShowPar\fi\HCode{<em>}}{\HCode{</em>}}
1487 \Configure{emph}{\ifvmode\ShowPar\fi\HCode{<em>}}{\HCode{</em>}}
1488 \</cfgXimera>

\texttt Using \texttt emits a <code> tag.
1489 \<cfgXimera>
1490 \Configure{texttt}{\ifvmode\ShowPar\fi\HCode{<code>}}{\HCode{</code>}}
1491 \</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`.

```

1492 \<classXimera>
1493 \font\dummyft@=dummy \relax
1494 \def\suppress{%
1495   \begingroup\par
1496   \parskip\z@
1497   \offinterlineskip
1498   \baselineskip=\z@skip
1499   \lineskip=\z@skip
1500   \lineskiplimit=\maxdimen
1501   \dummyft@
1502   \count@\sixt@@n
1503   \loop\ifnum\count@ >\z@
1504     \advance\count@\m@ne
1505     \textfont\count@\dummyft@
1506     \scriptfont\count@\dummyft@
1507     \scriptscriptfont\count@\dummyft@
1508   \repeat
1509   \let\selectfont\relax
1510   \let\mathversion\@gobble
1511   \let\getanddefine@fonts\@gobbletwo
1512   \tracinglostchars\z@
1513   \frenchspacing
1514   \hbadness\@M}
1515 \def\endsuppress{\par\endgroup}
1516 \</classXimera>

```

2.14.2 The End

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

```

1517 \<htXimera>
1518 \Hinput{ximera}
1519 \</htXimera>

1520 \<htXourse>
1521 \Hinput{xourse}
1522 \</htXourse>

1523 \<cfgXimera>
1524 \begin{document}
1525 \EndPreamble
1526 \</cfgXimera>

```

3 xourse.cls

```

1527 < *classXourse>
notoc The default behavior of the class is to provide a table of contents listing all activities in
      the course. This option will suppress this table of contents.
1528 \newif\ifnotoc
1529 \notocfalse
1530 \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.
1531 \newif\ifnonewpage
1532 \nonewpagefalse
1533 \DeclareOption{nonewpage}{\nonewpagetrue}

1534 \DeclareOption*{\PassOptionsToClass{\CurrentOption}{ximera}}
1535 \ProcessOptions\relax
1536 \LoadClass{ximera}
1537 % \begin{macrocode}
1538 < /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`.

```

1539 < *classXourse>
1540 \newcommand{\skip@preamble}{%
1541     \let\document\relax\let\enddocument\relax%
1542     \newenvironment{document}{\let\input\otherinput}{}%
1543     \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:

```

1544 \let\otherinput\input
Store usual \maketitle as \othermaketitle
1545 \let\othermaketitle\maketitle

```

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

```

1546 \renewcommand{\maketitle}{ %
1547 \pagestyle{empty}
1548 \begin{center}
1549 ~\ \ %puts space at top of page to move title down.
1550 \vskip .25\textheight
1551 \hrulefill\
1552 \vskip 1em
1553 \bfseries\Huge \@title\ \
1554 \hrulefill\
1555 \vskip 3em
1556 {\Large \@author}
1557 \vskip 2em
1558 {\large \@date}
1559 \end{center}
1560 \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.

```

1561 \ifnotoc
1562 \else
1563     \tableofcontents\clearpage

```

```

1564 \clearpage
1565 \fi
Switch to main pagestyle, just like a document with documentclass ximera.
1566 \pagestyle{main}
Renew maketitle to usual definition.
1567 \let\maketitle\othermaketitle
And we finish with our redefinition of \maketitle.
1568 }
1569 \relax
1570 \end{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.

```

1571 \begin{classXourse}
1572 \ifnonewpage
1573 \newcommand{\activity}[2][]{\%
1574 \setkeys{activity}{#1}
1575 \renewcommand{\input}[1]{
1576 \begin{group}\skip@preamble\otherinput{#2}\end{group}\par\vspace{\topsep}
1577 \let\input\otherinput}
1578 \else
1579 \newcommand{\activity}[2][]{\%
1580 \setkeys{activity}{#1}
1581 \renewcommand{\input}[1]{
1582 \begin{group}\skip@preamble\otherinput{#2}\end{group}\clearpage
1583 \let\input\otherinput}
1584 \fi
1585 \relax
1586 \end{classXourse}

1587 \begin{htXourse}
1588 \renewcommand\activity[2][]{\%
1589 \ifvmode \IgnorePar\fi \EndP\HCode{<a class="activity card \activitystyle" href="#2" data-op
1590 }
1591 \end{htXourse}

```

When running xake, we can just ignore activities

```

1592 \begin{classXourse}
1593 \ifxake
1594 \renewcommand\activity[2][]{\%
1595 \fi
1596 \end{classXourse}

```

3.1.2 Practice activities

`\practice` Like `\activity` but not expecting a title.

```

1597 \begin{classXourse}
1598 \ifhandout
1599 \newcommand{\practice}[2][]{\%
1600 \setkeys{practice}{#1}%!!!!
1601 \renewcommand{\input}[1]{
1602 \begin{group}\skip@preamble\otherinput{#2}\end{group}
1603 \let\input\otherinput}
1604 \else

```

```

1605 \newcommand{\practice}[2][\texttt{\detokenize{#2}}}% gives file name for practice
1606 \setkeys{practice}{#1}%!!!!
1607 \renewcommand{\input}[1]{
1608 \begingroup\skip@preamble\otherinput{#2}\endgroup
1609 \let\input\otherinput}
1610 \fi
1611 \relax
1612 \end{classXourse}

```

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

```

1613 \begin{classXourse}
1614 \ifxake
1615 \renewcommand{\practice}[2][{}]{
1616 \fi
1617 \end{classXourse}

```

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

```

1618 \begin{htXourse}
1619 \renewcommand{\practice}[2][{}]{
1620 \ifvmode\IgnorePar\fi\EndP%
1621 \HCode{<a class="activity card practice" href="#2" data-options="#1">#2</a>}%
1622 \IgnoreIndent%
1623 }
1624 \end{htXourse}

```

3.2 Sectioning

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.

```

1625 \begin{classXourse}
1626 \renewcommand*\l@section{\@dottedtocline{1}{1.5em}{4.2em}}
1627 \end{classXourse}

```

\subsection The name of a subsection inside an activity.

```

1628 \begin{classXourse}
1629 \renewcommand*\l@subsection{\@dottedtocline{2}{3.8em}{4.2em}}
1630 \end{classXourse}

```

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

```

1631 \begin{htXourse}
1632 \newcounter{ximera@part}
1633 \setcounter{ximera@part}{0}
1634 \renewcommand\part[1]{%
1635 \stepcounter{ximera@part}%
1636 \ifvmode \IgnorePar\fi \EndP%
1637 %\HCode{<h1 id="part\arabic{ximera@part}" class="card part">#1\HCode{</h1>}}% makes cards dis
1638 \HCode{<h1 id="part\arabic{ximera@part}" class="card part">#1</h1>}}%
1639 \IgnoreIndent%
1640 }
1641 \end{htXourse}

```

\paragraph Paragraph commands emit spans. A small heading.

```

1642 \begin{cfgXimera}
1643 \renewcommand{\paragraph}[1]{%
1644 \HCode{<span class="paragraphHead">%
1645 #1%
1646 \HCode{</span>}\par\IgnorePar}
1647 \end{cfgXimera}

```

\subparagraph An even smaller heading.

```

1648 \begin{cfgXimera}
1649 \renewcommand{\subparagraph}[1]{%
1650 \HCode{<span class="subparagraphHead">}}%

```

```

1651 #1%
1652 \HCode{</span>}\par\IgnorePar}
1653 </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.

```

1654 <*classXourse>
1655 \newenvironment{graded}[1]{%{}
1656 </classXourse>

```

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

```

1657 <*htXourse>
1658 \renewenvironment{graded}[1]{%
1659 \ifvmode \IgnorePar\fi \EndP\HCode{<div class="graded" data-weight="#1">}\IgnoreIndent%
1660 }{
1661 \ifvmode \IgnorePar\fi \EndP\HCode{</div>}\IgnoreIndent%
1662 }
1663 </htXourse>

```

3.4 Logos

\logo A logo for the xourse.

```

1664 <*classXourse>
1665 \newcommand*{\logo}[1]{%
1666 \ifx\@onlypreamble\@notprerr
1667 \ClassError{xourse}{logo can only be used in the preamble}
1668 {Move your logo command to the preamble}
1669 \else %
1670 \IfFileExists{#1}%
1671 {\gdef\xourse@logo{#1}}%
1672 {\ClassError{xourse}{logo file does not exist}
1673 {To use logo, make sure that the referenced image file exists}}%
1674 \fi%
1675 }
1676
1677 </classXourse>

```

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

```

1678 <*htXourse>
1679 \Configure{@HEAD}{%
1680 \HCode{<meta name="og:image" content="}%
1681 \ifdefined\xourse@logo%
1682 \xourse@logo%
1683 \fi%
1684 \HCode{" />\Hnewline}}%
1685 </htXourse>

```