

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

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

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

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

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

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

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

```
19 \DeclareOption{noinstructornotes}{\instructornotestru}
```

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* p
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 \RequirePackage{tcolorbox}
90 \RequirePackage{currfile}
91 \RequirePackage{comment}
92
93 \</classXimera>
```

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

```
94 \<classXimera>
95 \RequirePackage{getttitlestring}
96 \RequirePackage{nameref}
97 \RequirePackage{epstopdf}
98
99 \</classXimera>
```

2.3 Page setup

We want non-indented spaced-out paragraphs.

```
100 \<classXimera>
101 \setlength{\parindent}{0pt}
102 \setlength{\parskip}{5pt}
103 \</classXimera>
```

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

```
104 \<classXimera>
105 \oddsidemargin 62pt
106 \evensidemargin 62pt
107 \textwidth 345pt
108 \headheight 14pt
109 \</classXimera>
```

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

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

```

130 \Configure{@HEAD}{\HCode{<script type="text/javascript" async src="https://ximera.osu.edu/pub
131
132 % OVERWRITE css in ximera-server (to be removed whenever this has been fixed in the server;
133 \catcode'\%=11
134 \Configure{@BODY}{\HCode{<style>
135 .activity-body pre {
136     white-space: pre;
137     background-color: lightgray;
138 }
139 .xmyoutube {
140     aspect-ratio: 16/9;
141     min-width: 75%;
142 }
143 .image-environment img {
144     width: unset;
145 }
146 </style>\Hnewline}}
147 \catcode'\%=14
148
149 </cfgXimera>

```

Disable certain ligatures in HTML.

```

150 <htXimera>
151 \usepackage{microtype}
152 \DisableLigatures[f]{encoding=*}
153 </htXimera>

```

I am not sure what this does.

```

154 <htXimera>
155 \NewEnviron{html}{\HCode{\BODY}}
156 </htXimera>

```

2.4 Structure

2.4.1 Macros

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

```

157 <classXimera>
158 \everymath{\displaystyle}
159 </classXimera>

```

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

```

160 <classXimera>
161 \let\prelim\lim
162 \renewcommand{\lim}{\displaystyle\prelim}
163 </classXimera>

```

2.4.2 Theorem and theorem-like environments

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

```

164 <htXimera>
165 \newcommand{\ConfigureTheoremEnv}[1]{%
166 \renewenvironment{#1}[1][]{\refstepcounter{problem}%
167 \ifthenelse{\equal{##1}{}}{\}%
168 \HCode{<span class="theorem-like-title">##1\HCode{</span>}}%
169 }}{\%
170 \ConfigureEnv{#1}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div class=
171 }
172 </htXimera>
173 <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

```

174 <classXimera>      \newtheorem{theorem}{\GetTranslation{Theorem}}

```

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

<code>observation (env.)</code>	Observation	
	210 <code><classXimera></code>	<code>\newtheorem{observation}{\GetTranslation{Observation}}</code>
	211 <code><htXimera></code>	<code>\ConfigureTheoremEnv{observation}</code>
<code>proposition (env.)</code>	Proposition	
	212 <code><classXimera></code>	<code>\newtheorem{proposition}{\GetTranslation{Proposition}}</code>
	213 <code><htXimera></code>	<code>\ConfigureTheoremEnv{proposition}</code>
<code>paradox (env.)</code>	Paradox	
	214 <code><classXimera></code>	<code>\newtheorem{paradox}{\GetTranslation{Paradox}}</code>
	215 <code><htXimera></code>	<code>\ConfigureTheoremEnv{paradox}</code>
<code>procedure (env.)</code>	Procedure	
	216 <code><classXimera></code>	<code>\newtheorem{procedure}{\GetTranslation{Procedure}}</code>
	217 <code><htXimera></code>	<code>\ConfigureTheoremEnv{procedure}</code>
<code>remark (env.)</code>	Remark	
	218 <code><classXimera></code>	<code>\newtheorem{remark}{\GetTranslation{Remark}}</code>
	219 <code><htXimera></code>	<code>\ConfigureTheoremEnv{remark}</code>
<code>summary (env.)</code>	Summary	
	220 <code><classXimera></code>	<code>\newtheorem{summary}{\GetTranslation{Summary}}</code>
	221 <code><htXimera></code>	<code>\ConfigureTheoremEnv{summary}</code>
<code>template (env.)</code>	Template	
	222 <code><classXimera></code>	<code>\newtheorem{template}{\GetTranslation{Template}}</code>
	223 <code><htXimera></code>	<code>\ConfigureTheoremEnv{template}</code>
<code>warning (env.)</code>	Warning	
	224 <code><classXimera></code>	<code>\newtheorem{warning}{\GetTranslation{Warning}}</code>
	225 <code><htXimera></code>	<code>\ConfigureTheoremEnv{warning}</code>

2.4.3 Enumerate fixes

Make enumerate use a letter

```

226 <*classXimera>
227 \renewcommand{\theenumi}{\textup{(\alph{enumi})}}
228 \renewcommand{\labelenumi}{\theenumi}
229 \renewcommand{\theenumii}{\textup{(\roman{enumii})}}
230 \renewcommand{\labelenumii}{\theenumii}
231 </classXimera>

```

2.4.4 Proofs

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

```

232 <*classXimera>
233 \renewcommand{\qedsymbol}{\blacktriangle}
234 \renewenvironment{proof}[1][\proofname]
235 {\begin{trivlist}\item[\hskip \labelsep \itshape \bfseries #1]{\hspace{2ex}}}
236 {\qed\end{trivlist}}
237 </classXimera>
238 <*htXimera>
239 % Mmm, (why) do we want/need this ...?
240 \ConfigureTheoremEnv{proof}
241 \ConfigureEnv{proof}{\ifvmode\IgnorePar\fi\EndP\HCode{<div class="proof">}}
242 \ConfigureList{trivlist}{\ifvmode\IgnorePar\fi\EndP}{\}{\}
243 {\ifvmode\IgnorePar\fi\EndP\HCode{</div>}}{\}{\}
244 </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>

```

245 <*classXimera>

```

```

246 \newcommand{\hang}{% top theorem decoration
247   \begingroup%
248   \setlength{\unitlength}{.005\linewidth}% \linewidth/200
249   \begin{picture}(0,0)(1.5,0)%
250     \linethickness{1pt} \color{black!50}%
251     \put(-3,2){\line(1,0){206}}% Top line
252     \multido{\iA=2+-1,\iB=50+-10}{5}{% Top hangs
253       \color{black!\iB}%
254       \put(-3,\iA){\line(0,-1){1}}% Top left hang
255       \put(203,\iA){\line(0,-1){1}}% Top right hang
256     }%
257   \end{picture}%
258   \endgroup%
259 }%
260 \newcommand{\hung}{% bottom theorem decoration
261   \nobreak
262   \begingroup%
263   \setlength{\unitlength}{.005\linewidth}% \linewidth/200
264   \begin{picture}(0,0)(1.5,0)%
265     \linethickness{1pt} \color{black!50}%
266     \put(60,0){\line(1,0){143}}% Bottom line
267     \multido{\iA=0+1,\iB=50+-10}{5}{% Bottom hangs
268       \color{black!\iB}%
269       \put(-3,\iA){\line(0,1){1}}% Bottom left hang
270       \put(203,\iA){\line(0,1){1}}% Bottom right hang
271       \put(\iB,0){\line(60,0){10}}% Left fade out
272     }%
273   \end{picture}%
274   \endgroup%
275 }%

```

Configure environment configuration commands

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

```

276 \MakeCounter{Iteration@probCnt}
277 \MakeCounter{problem}
278 \newif\ifnoNumberedProblems
279 \newcommand{\probNumber}{
280 % First we determine if we have a counter for this question depth level.
281 \ifcsname c@depth\Roman{problem@Depth}Count\endcsname% Check to see if counter exists
282 %If so, do nothing.
283 \else
284 %If not, create it.
285 \expandafter\newcounter{depth\Roman{problem@Depth}Count}
286 \expandafter\setcounter{depth\Roman{problem@Depth}Count}{0}
287 \fi
288
289 \expandafter\stepcounter{depth\Roman{problem@Depth}Count}
290 \arabic{depthICount}% The first problem depth, what use to be |\theproblem|.
291
292 \forloop{Iteration@probCnt}{2}{\arabic{Iteration@probCnt} < \numexpr \value{problem@Depth} +
293 .\expandafter\arabic{depth\Roman{Iteration@probCnt}Count}}% Get the problem number of the next
294 }
295 }
296 %%%% Configure various problem environment commands
297 \Make@Counter{problem@Depth}
298 %%%% Configure environments start content
299 \newcommand{\problemEnvironmentStart}[2]{%
300 \stepcounter{problem@Depth}% Started a problem, so we've sunk another problem layer.
301 \def\spaceatend{#1}%
302 %\edef\probNumDisp{\probNumber}
303 \begin{trivlist}%
304 \item[\hskip\labelsep\ssfamily\bfseries\GetTranslation{#2} \probNumber% Determine the correct
305 ]%

```



```

306 \slshape
307 }
308
309 %%%% Configure environments end content %%%%
310 \newcommand{\problemEnvironmentEnd}{% This configures all the end content for a problem.
311 \stepcounter{problem@Depth}
312 \ifcsname c@depth\Roman{problem@Depth}Count\endcsname
313 \expandafter\ifnum\expandafter\value{depth\Roman{problem@Depth}Count}>0
314 \expandafter\setcounter{depth\Roman{problem@Depth}Count}{0}
315 \fi
316 \fi
317 \addtocounter{problem@Depth}{-2}% Exited a problem so we've exited a problem layer. Need -2
318 \ifhandout
319 \ifnewpage
320 \newpage
321 \fi
322 \fi
323 \end{trivlist}
324 }
325 %% Add a simple command that handles all the problem creation aspects:
326 \newcommand{\createProblemEnv}[2]{% This is a nice command to define a new problem-like environment
327 \newenvironment{#1}[1][2in]%
328 {%Env start code
329 \problemEnvironmentStart{#1}{#2}
330 }
331 {%Env end code
332 \problemEnvironmentEnd
333 }
334 }
335
336 %%%% Now populate the old environment names
337 %
338 % Old environments were "problem", "exercise", "exploration", and "question".
339 % Note that you can add content to the start/end code on top of these base code pieces if you want
340 %
341 % These definitions will be overwritten in ximera.4ht !
342
343 \createProblemEnv{problem}{Problem}
344 \createProblemEnv{exercise}{Exercise}
345 \createProblemEnv{exploration}{Exploration}
346 \createProblemEnv{question}{Question}
347 \end{classXimera}
348 \end{*htXimera}
349 \newcounter{identification}
350 \setcounter{identification}{0}
351 \def\probNumDisp{}% Otherwise don't display a problem number.
352 \newcommand{\ConfigureQuestionEnv}[2]{%
353 \renewenvironment{#1}{}{}
354 \ConfigureEnv{#1}
355 {
356 \stepcounter{problem@Depth}% Started a problem, so we've sunk another problem layer.
357 \ifcsname c@depth\Roman{problem@Depth}Count\endcsname
358 \else
359 \expandafter\newcounter{depth\Roman{problem@Depth}Count}
360 \expandafter\setcounter{depth\Roman{problem@Depth}Count}{0}
361 \fi
362 \expandafter\stepcounter{depth\Roman{problem@Depth}Count}
363 \ifnumberedProblems% Because of how conditional syntax works, we need to make sure that
364 \noNumberedProblemsfalse
365 \else
366 \noNumberedProblemstrue
367 \fi
368 \ifnoNumberedProblems% The code below is all to generate online problem numbering if opt

```

```

369 \def\probNumDisp{}
370 \else
371 \def\probNumDisp{
372 \space\arabic{depthICount}% Top Level Problem Number: X.1.1.1.1 Number.
373 \ifcsname c@depthIICount\endcsname\ifnum\value{problem@Depth}>1 .\arabic{depthIICount}\fi
374 \ifcsname c@depthIIICount\endcsname\ifnum\value{problem@Depth}>2 .\arabic{depthIIICount}\fi
375 \ifcsname c@depthIVCount\endcsname\ifnum\value{problem@Depth}>3 .\arabic{depthIVCount}\fi
376 \ifcsname c@depthVCount\endcsname\ifnum\value{problem@Depth}>4 .\arabic{depthVCount}\fi
377 \fi\fi\fi\fi
378 }
379 \fi
380 \stepcounter{identification}
381 \ifvmode
382 \IgnorePar
383 \fi
384 \EndP
385 \HCode{<div role="article" class="problem-environment #1" id="problem\arabic{identification}"
386 }
387 {
388 \stepcounter{problem@Depth}
389 \ifcsname c@depth\Roman{problem@Depth}Count\endcsname
390 \expandafter\ifnum\expandafter\value{depth\Roman{problem@Depth}Count}>0
391 \expandafter\setcounter{depth\Roman{problem@Depth}Count}{0}
392 \fi
393 \fi
394 \addtocounter{problem@Depth}{-2}% Exited a problem so we've exited a problem layer. Need -2
395
396 \ifvmode
397 \IgnorePar
398 \fi
399 \EndP
400 \HCode{</div>}\IgnoreIndent
401 }{}{}%
402 }
403
404 \ConfigureQuestionEnv{problem}{Problem}
405 \ConfigureQuestionEnv{exercise}{Exercise}
406 \ConfigureQuestionEnv{question}{Question}
407 \ConfigureQuestionEnv{exploration}{Exploration}
408 %\ifdefined\xmNotHintAsExpandable
409 % \ConfigureQuestionEnv{hint}{hint} % 2024: hint is no longer a 'question-environment'.
410 %\fi
411 </htXimera>

```

2.4.6 Hints

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

```
412 <*classXimera>
```

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

```
413 \newcounter{hintLevel}
414 \setcounter{hintLevel}{0}
```

Create an empty shell to renew

```
415 \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.

```
416 \renewenvironment{hint}
417 {
418 \ifhandout
419 \setbox0\vbox\bgroup
420 \else

```

```

421 \begin{trivlist}\item[\hskip \labelsep\small\slshape\bfseries \GetTranslation{hint}:\hspace{1ex}]
422 \small\slshape
423 \fi
424 \stepcounter{hintLevel}
425 }
426 {
427 \ifhandout
428 \egroup\ignorespacesafterend
429 \else
430 \end{trivlist}
431 \fi
432 \addtocounter{hintLevel}{-1}
433 }
434
435 \ifhints
436 \renewenvironment{hint}{
437 \begin{trivlist}\item[\hskip \labelsep\small\slshape\bfseries \GetTranslation{hint}:\hspace{1ex}]
438 \small\slshape
439 }
440 {
441 \end{trivlist}
442 }
443 \fi
444
445 \end{classXimera}

```

2.4.7 Solution

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

```

446 \begin{classXimera}
447 %% solution environment
448 \ifhandout % what follows is handout behavior
449 \newenvironment{solution}%
450     {%
451     \setbox0\vbox\bgroup
452     }
453     {%
454     \egroup
455     }
456 \else
457 \newenvironment{solution}%
458     {%
459     \begin{trivlist}
460     \item[\hskip \labelsep\bfseries \GetTranslation{Solution}:\hspace{2ex}]
461     }
462     % %% line at the bottom}
463     {
464     \end{trivlist}
465     % (202410: no longer \par\addvspace{.5ex}\nobreak\noindent\hung
466     }
467 \fi
468
469
470
471 \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.

```

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

```

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

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

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

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

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

```
485 %\begin{classXimera}
486 %\ConfigureEnv{verbatim}{\ifvmode\IgnorePar\fi\EndP\HCode{<pre style="white-space: pre; background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;">}}{\ifvmode\IgnorePar\fi\EndP\HCode{</pre>}}
487 %\ConfigureEnv{lstlisting}{\ifvmode\IgnorePar\fi\EndP\HCode{<pre>}}{\ifvmode\IgnorePar\fi\EndP\HCode{</pre>}}
488 %\end{classXimera}
489 %
```

2.4.9 Dialogues

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

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

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

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

2.4.10 Instructor notes

```
511 \begin{classXimera}
512
513 %% instructor intro/instructor notes
514 %%
515 \ifhandout % what follows is handout behavior
516 \ifinstructornotes
517 \newenvironment{instructorIntro}{%
518   {%
519     \begin{trivlist}
520     \item[\hspace{\labelsep}\bfseries \GetTranslation{Instructor Introduction}:\hspace{2ex}]
521     }
522     % %% line at the bottom}
```

```

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

```

```

586 \ifinstructornotes
587 \newenvironment{instructorNotes}%
588     {%
589     \setbox0\vbox\bgroup
590     }
591     {%
592     \egroup
593     }
594     \else
595     \newenvironment{instructorNotes}%
596     {%
597     \begin{trivlist}
598     \item[\hskip \labelsep\bfseries \GetTranslation{Instructor Notes}:\hspace{2ex}]
599     }
600     % %% line at the bottom}
601     {
602     \end{trivlist}
603     \par\addvspace{.5ex}\nobreak\noindent\hung
604     }
605     \fi
606     \fi
607
608 </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.

```

609 %\iftikzexport\else\RequirePackage[framemethod=TikZ]{mdframed}\fi
foldable (env.) Does it fold?
610 <*classXimera>
611
612 \colorlet{textColor}{black} % since textColor is referenced below
613 \colorlet{background}{white} % since background is referenced below
614
615 % The core environments. Find results in 4ht file.
616 %% pretty-foldable
617 %\iftikzexport
618 \newenvironment{foldable}{%
619 }{%
620 }
621 %\else
622 %\renewmdenv[
623 % font=\upshape,
624 % outerlinewidth=3,
625 % topline=false,
626 % bottomline=false,
627 % leftline=true,
628 % rightline=false,
629 % leftmargin=0,
630 % innertopmargin=0pt,
631 % innerbottommargin=0pt,
632 % skipbelow=\baselineskip,
633 % linecolor=textColor!20!white,
634 % fontcolor=textColor,
635 % backgroundcolor=background
636 %]{foldable}%
637 %\fi
638
639 %% pretty-expandable
640 %\iftikzexport
641 %% Overwritten in .4ht, but probably also in accordion!

```

```

642 \ifdefined\xmNotExpandableAsAccordion
643 \newenvironment{expandable}{}{}
644 \else
645 \newenvironment{expandable}[2]{}{}
646 \fi
647 %\else
648 %\newmdenv[
649 %   font=\upshape,
650 %   outerlinewidth=3,
651 %   topline=false,
652 %   bottomline=false,
653 %   leftline=true,
654 %   rightline=false,
655 %   leftmargin=0,
656 %   innertopmargin=0pt,
657 %   innerbottommargin=0pt,
658 %   skipbelow=\baselineskip,
659 %   linecolor=black,
660 %]{expandable}%
661 %\fi
662
663 \newcommand{\unfoldable}[1]{#1}
664
665 \end{classXimera}

```

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

```

666 \begin{htXimera}
667 \renewenvironment{foldable}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<
668
669 \ifdefined\xmNotExpandableAsAccordion
670 \renewenvironment{expandable}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{
671 \fi
672
673 \renewcommand{\unfoldable}[1]{\HCode{<span class="unfoldable">#1\HCode{</span>}}
674 \end{htXimera}

```

2.4.12 Leashes

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

```

675 \begin{classXimera}
676
677 \newenvironment{leash}[1]{%
678 }{%
679 }
680
681
682 \end{classXimera}
683 \begin{htXimera}
684 \renewenvironment{leash}[1]{\ifvmode \IgnorePar\fi \EndP\HCode{<div style="overflow: auto; h
685 \end{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.

```

686 \begin{classXimera}
687 \newcommand{\license}{\excludecomment}
688 \end{classXimera}

```

`\acknowledgement` In the preamble, use `\acknowledgement` to credit others who contributed to the

intellectual content beside the author.

```
689 <*classXimera>
690 \newcommand{\acknowledgement}{\excludecomment}
691 </classXimera>
```

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

```
692 <*classXimera>
693 \renewcommand{\tag}{\excludecomment}
694 </classXimera>
```

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

```
695 <*htXourse>
696 % Mark this as a xourse file
697 \Configure{@HEAD}{\HCode{<meta name="description" content="xourse" />\Hnewline}}
698 </htXourse>
```

2.5.2 Abstract

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

```
699 <*classXimera>
700 \let\abstract\relax
701 \let\endabstract\relax
702 % Use of environ package, may want to find a better way.
703 % see the messing around with \theabstract in title.dtx ... Is this really needed/wanted?
704 \NewEnviron{abstract}{\protected@xdef\theabstract{\BODY}}
705 </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.

```
706 <*cfgXimera>
707 \ifvmode\IgnorePar\fi\EndP
708 \ConfigureEnv{abstract}{\ifvmode\IgnorePar\fi\EndP\HCode{\Hnewline<div class="abstract">}\par}
709 </cfgXimera>
710 <*htXimera>
711 \RenewEnviron{abstract}{\BODY}
712 <*htXimera>
```

2.5.3 Titles and authors

2.5.4 Authors

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

```
713 <*classXimera>
714 \let\emptyauthor\@author
715 \def\author#1{\gdef\author{#1}}
716 \def\@author{\@latex@warning@no@line{No \noexpand\author given}}
717 </classXimera>
```

Include author name in meta tags

```
718 <*htXimera>
719 \Configure{@HEAD}{\HCode{<meta name="author" content="\@author\HCode{" />\Hnewline}}
720 </htXimera>
```

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

```
721 <htXimera | classXimera>\def\and{and }
```

2.5.5 Title

`\title` Activities have titles.

```
722 <*classXimera>
723 \let\title\relax
724 \newcommand{\title}[1][ ]{\protected@xdef\pretitle{#1}}\protected@xdef\@title{
```



```

725
726 \title{}
727
728 \newcounter{titlenumber}
729 \renewcommand{\thetitlenumber}{\arabic{titlenumber}}
730 %\renewcommand{\thesection}{\arabic{titlenumber}} %% Makes section numbers work
731 \setcounter{titlenumber}{0}
732
733 \newpagestyle{main}{
734 \sethead[\textsl{\ifnumbers\thetitlenumber\hspace{1em}}\fi\@title]{}{} % even
735 {}{}{\textsl{\ifnumbers\thetitlenumber\hspace{1em}}\fi\@title} % odd
736 \setfoot[\thepage]{}{} % even
737 {}{}{\thepage} % odd
738 }
739 \pagestyle{main}

\maketitle In a ximera document, redefine \maketitle and put them in a table of contents. The
\phantomsection is to fix the hrefs.
740 \renewcommand\maketitle{%
741   \addtocounter{titlenumber}{1}%
742   {\flushleft\large\bfseries \@pretitle\par\vspace{-1em}}
743   {\flushleft\LARGE\bfseries {\ifnumbers\thetitlenumber\fi}{\ifnumbers\hspace{1em}}\else\hspa
744   \phantomsection%
745   \ifnumbers\addcontentsline{toc}{section}{\thetitlenumber~\@title}\else\addcontentsline{toc
746   \vskip .6em\noindent\textit{theabstract}\setcounter{problem}{0}\setcounter{section}{0}\setco
747   %\ifnooutcomes\else\let\thefootnote\relax\footnote{Learning outcomes: \theoutcomes}\fi% Dep
748   \ifnoauthor\else\let\thefootnote\relax\footnote{Author(s):~\@author}\fi
749   \aftergroup\@afterindentfalse
750   \aftergroup\@afterheading}
751
752 \ifnumbers
753 \setcounter{secnumdepth}{2}
754 \renewcommand{\thesection}{\arabic{titlenumber}.\arabic{section}}
755 \renewcommand{\thesubsection}{\arabic{titlenumber}.\arabic{section}.\arabic{subsection}}
756 \else
757 \setcounter{secnumdepth}{-2}
758 \fi
759
760 \def\activitystyle{}
761 \newcounter{sectiontitlenumber}
762 \setcounter{secnumdepth}{2}
763 \setcounter{tocdepth}{2}
764 \newcommand\chapterstyle{%
765   \def\activitystyle{activity-chapter}
766   \def\maketitle{%
767     \addtocounter{titlenumber}{1}%
768     {\flushleft\small\sffamily\bfseries\@pretitle\par\vspace{-1.5em}}%
769     {\flushleft\LARGE\sffamily\bfseries\thetitlenumber\hspace{1em}}\@title \pa
770     {\vskip .6em\noindent\textit{theabstract}\setcounter{problem}{0}\setcount
771     \par\vspace{2em}
772     \phantomsection\addcontentsline{toc}{section}{\textbf{\thetitlenumber\hsp
773   }}
774
775
776 \newcommand\sectionstyle{%
777   \def\activitystyle{activity-section}
778   \def\maketitle{%
779     \addtocounter{section}{1}
780     \setcounter{sectiontitlenumber}{\value{section}}
781     {\flushleft\small\sffamily\bfseries\@pretitle\par\vspace{-1.5em}}%
782     {\flushleft\Large\sffamily\bfseries\thetitlenumber.\thesectiontitlenumber\hspace{1em}}\@t
783     {\vskip .6em\noindent\textit{theabstract}\setcounter{subsection}{0}}%
784     \par\vspace{2em}
785     \phantomsection\addcontentsline{toc}{section}{\thetitlenumber.\thesectiontitlenumber\hsp

```

```

786 \renewcommand\section{\@startsection{subsection}{2}{\z@}%
787                               {-3.25ex\@plus -1ex \@minus -.2ex}%
788                               {1.5ex \@plus .2ex}%
789                               {\normalfont\large\bfseries}}
790
791 \renewcommand\subsection{\@startsection{subsubsection}{3}{\z@}%
792                               {-3.25ex\@plus -1ex \@minus -.2ex}%
793                               {1.5ex \@plus .2ex}%
794                               {\normalfont\normalsize\bfseries}}
795
796 }}
797
798
799 \iftikzexport%% allows xake to handle \chapterstyle and \sectionstyle
800 \renewcommand\chapterstyle{\def\activitystyle{chapter}}
801 \renewcommand\sectionstyle{\def\activitystyle{section}}
802 \else
803 \fi
804
805 \</classXimera>

```

Eliminate some formatting that we'll handle later with CSS

```

806 \<htXimera>
807 \renewcommand{\maketitle}{}
808 \</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 L^AT_EX 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

```

809 \<classXimera>
810 \ifxake
811     \newenvironment{prompt}{}{}
812 \else
813 \ifhandout
814     \NewEnviron{prompt}{}
815     % Breaks when put in mathmode ?
816     % \newenvironment{prompt}{\suppress}{\endsuppress}
817 \else
818     \newenvironment{prompt}{\bgroup\color{gray!50!black}}{\egroup}
819 \fi
820 \fi

```

`onlyHtml (env.)` Only display online

`onlyPdf (env.)` Only display in the PDF

`onlineOnly (env.)` Only display online (deprecated: use `onlyHtml` instead)

```

821 \ifdefined\HCode
822     \newenvironment{onlyPdf}{\setbox0\vbox\bgroup}{\egroup}
823     \newenvironment{onlyHtml}{\bgroup}{\egroup}

```

```

824 \newenvironment{onlineOnly}{\bgroup}{\egroup}
825 \else
826 \newenvironment{onlyPdf}{\bgroup}{\egroup}
827 \ifdefined\xmPrintHtmlOnlyAlsoInPdf
828 \newenvironment{onlyHtml}{\bgroup\color{red!50!black}}{\egroup}
829 \newenvironment{onlineOnly}{\bgroup\color{red!50!black}}{\egroup}
830 \else
831 \newenvironment{onlyHtml}{\setbox0\vbox\bgroup}{\egroup}
832 \newenvironment{onlineOnly}{\setbox0\vbox\bgroup}{\egroup}
833 \fi
834 \fi
835
\htmlOnly Only display online
\pdfOnly Only display in the PDF

836
837 \ifdefined\HCode
838 \newcommand{\pdfOnly}[1]{
839 \newcommand{\htmlOnly}[1]{#1}
840 \else
841 \ifdefined\xmPrintHtmlOnlyAlsoInPdf
842 \newcommand{\pdfOnly}[1]{#1}
843 \newcommand{\htmlOnly}[1]{\bgroup\color{red!50!black}#1\egroup}
844 \else
845 \newcommand{\pdfOnly}[1]{#1}
846 \newcommand{\htmlOnly}[1]{
847 \fi
848 \fi
849
\ifonline Only execute online (ie in HTML version)
\ifonlineTF Different output online vs PDF

850 % An alternative for \pdfOnly/\begin{htmlOnly} :
851 % Usage: Hello \ifonlineTF{online reader}{PDF reader}
852 \providecommand{\ifonlineTF}[2]{\htmlOnly{#1}\pdfOnly{#2}}
853 \newif{\ifonline}
854 \ifdefined\HCode
855 \onlinetrue
856 \else
857 \onlinefalse
858 \fi
859 \end{classXimera}

```

2.5.7 Learning Outcomes

```

860 \classXimera
861 \newcommand{\preOutcomeLine}{\item }
862 \newcommand{\postOutcomeLine}{}
863 \newcommand{\preOutcomeBlock}{After completing this content, students should be able to: \begin{itemize}
864 \newcommand{\postOutcomeBlock}{\end{itemize} So go forth and learn!}
865
866 \newcommand{\outcomeHeader}{Goals for this Section}
867 \htmlOnly{
868 \newcommand{\outcomeBlock}{\ifvmode\IgnorePar\fi\EndP\HCode{<div class="outcomeHead"> \outcomeHeader
869 }
870
871
872 \newwrite\outcomefile
873 \immediate\openout\outcomefile=\jobname.oc
874 \newcommand{\outcome}[1]{%
875 \immediate\write\outcomefile{\expandafter\unexpanded\expandafter{\preOutcomeLine #1} \outcomeBlock}
876 }
877
878 \newcommand{\displayOutcomes}[1][1]{%

```

```

879 \immediate\closeout\outcomefile
880 \IfFileExists{\currfiledir\currfilebase.oc}{
881   \htmlOnly{\outcomeBlock}
882   \expandafter\preOutcomeBlock
883   \input{\currfiledir\currfilebase.oc}
884   \postOutcomeBlock
885   \htmlOnly{\ifvmode\IgnorePar\fi\EndP\HCode{</div>}}
886 }
887 {
888   \IfFileExists{\currfilebase.oc}{
889     \htmlOnly{\outcomeBlock}
890     \expandafter\preOutcomeBlock
891     \input{\currfilebase.oc}
892     \postOutcomeBlock
893     \htmlOnly{\ifvmode\IgnorePar\fi\EndP\HCode{</div>}}
894   }
895   {
896     No outcome file found.
897   }
898 }
899 }
900 %
901 </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.

```

902 <*cfgXimera>
903 \renewcommand{\outcome}[1]{
904   \Configure{@HEAD}{\HCode{<meta name="learning-outcome" content="#1"/>\Hnewline}}
905 }
906 % Sometimes there are no outcomes at all
907 \IfFileExists{\jobname.oc}{\input{\jobname.oc}}{}
908
909 \renewcommand{\outcome}[1]{%
910   \HCode{<span class="learning-outcome">#1</span>}}
911 }
912 </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.

```

913 <*htXimera>
914 \let\oldlabel\label
915 \renewcommand{\label}[1]{\oldlabel{#1}\HCode{<a class="ximera-label" id="#1"></a>}}
916 </htXimera>

```

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

```

917 <*htXimera>
918 \renewcommand{\ref}[1]{\HCode{<a class="reference" href="#1">#1</a>}}
919 </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!

```

920 <*classXimera>
921 % Provide a default graphicspath
922 % (somewhat tricky: an activity can be included in a xourse in a wildly different path !)
923 % Suggested convention: put all images in i /pictures folder in the root of your project
924 \providecommand{\xmDefaultGraphicsPath}{/xmPictures}

```

```

925 \graphicspath{ %% When looking for images,
926 {./} %% look here first,
927 {\xmDefaultGraphicsPath/} %% then look for a pictures folder,
928 {..\xmDefaultGraphicsPath/} %% then look for a pictures folder,
929 {../../xmDefaultGraphicsPath/} %% then look for a pictures folder,
930 {../../..\xmDefaultGraphicsPath/} %% then look for a pictures folder,
931 }
932 \newenvironment{image}[1][\begin{center}]{\end{center}}
933 \NewEnviron{image}[1][3in]{%
934 \begin{center}\resizebox{#1}{!}{\BODY}\end{center}% resize and center
935 }
936 \end{classXimera}
\alt Inside an image environment, \alt provides alt-text for assistive technology like screen-
readers.

```

```

937 \end{classXimera}
938 \newcommand{\alt}[1]{}
939 \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.

```

940 \end{htXimera}
941 \newcounter{imagealt}
942 \setcounter{imagealt}{0}
943 \renewenvironment{image}[1][\stepcounter{imagealt}%
944 \ifvmode \IgnorePar\fi \EndP%
945 \HCode{<div class="image-environment" role="img" aria-labelledby="image-alt-\arabic{imagealt}>}
946 }{\HCode{</div>}}
947 \renewcommand{\alt}[1]{\HCode{<div style="display: none;" id="image-alt-\arabic{imagealt}>>}}
948 \end{htXimera}
949 \end{cfgXimera}
950 %% Although we accept many formats, SVG is preferred on the web.
951 %% Since we have a different mechanism for producing |alt| text, we
952 %% want to ignore tex4ht's own method for producing alt text.
953 %% 2024: is now in TeX4ht ...
954 \DeclareGraphicsExtensions{.jpg,.png,.gif,.svg}
955 \Configure{graphics*}{
956 % {svg}{
957 % {\Configure{Needs}{File: \Gin@base.svg}\Needs{}}
958 % \Picture[]{\csname Gin@base\endcsname.svg \csname a:Gin-dim\endcsname}%
959 % }
960 \end{cfgXimera}

```

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

```

961 \end{cfgXimera}
962 \ifcsname ifstandalone\endcsname
963 \ifstandalone
964 \renewcommand\includegraphics[2][\fi
965 \fi
966 \end{cfgXimera}

```

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

```

967 \end{htXimera}
968 \providecommand{\pgfsyspdfmark}[3]{}
969 \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`.

```

970 <*classXimera>
971 % everything skipped, assume TeX4ht does the job now
972 \ifdefined\reallyneverever
973
974 \ifdefined\HCode
975   \tikzexporttrue
976 \fi
977
978 \iftikzexport
979   \usetikzlibrary{external}
980
981   \ifdefined\HCode
982     % in htlatex, just include the svg files
983     \def\pgfsys@imagesuffixlist{.svg}
984
985     \tikzexternalize[prefix=./,mode=graphics if exists]
986   \else
987     % in pdflatex, actually generate the svg files
988     \tikzset{
989       /tikz/external/system call={
990         pdflatex \tikzexternalcheckshellescape
991         -halt-on-error -interaction=batchmode
992         -jobname "\image" "\PassOptionsToClass{tikzexport}{ximera}\texsource";
993         mutool draw -F svg \image.pdf > \image.svg ;      % mutool adds "1" to filename ???
994         mutool draw -o \image.svg \image.pdf ;
995         mutool draw -r 150 -c rgbalpha -o \image.png \image.pdf ;
996         ebb -x \image.png
997       }
998     }
999     \tikzexternalize[optimize=false,prefix=./]
1000 \fi
1001
1002 \fi
1003 \fi
1004 </classXimera>

```

2.6.3 XKCD

`\xkcd` Reference an XKCD cartoon.

```

1005 <*classXimera>
1006 \newcommand{\xkcd}[1]{#1}
1007 </classXimera>

```

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

```

1008 <*htXimera>
1009 \renewcommand{\xkcd}[1]{\ifvmode \IgnorePar\fi \EndP\HCode{#2 $\rangle$  $\}$ 
1082  $\langle$ /htXimera $\rangle$ 
```

2.8.6 Video

\backslash youtube Youtube command. Requires id.

```
1083  $\langle$ *classXimera $\rangle$ 
1084  $\backslash$ newcommand $\{\backslash$ youtube $\}[1]\{YouTube\ link: \url{https://www.youtube.com/watch?v=#1}\}$ 
1085  $\langle$ /classXimera $\rangle$ 

1086  $\langle$ *htXimera $\rangle$ 
1087  $\%$   $\backslash$ renewcommand $\{\backslash$ youtube $\}[1]\{\text{ifvmode } \backslash$ IgnorePar $\backslash$ fi  $\backslash$ EndP $\}$ 
1088  $\%$  Fixes no-youtube-when-no-cookies-accepted. Class xmyoutube allows for css customization.
1089  $\backslash$ renewcommand $\{\backslash$ youtube $\}[1]\{\text{ifvmode } \backslash$ IgnorePar $\backslash$ fi  $\backslash$ EndP $\}$ 
1090
1091  $\langle$ /htXimera $\rangle$ 
```

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

```
1092  $\langle$ *htXourse $\rangle$ 
1093  $\backslash$ renewcommand $\backslash$ youtube $[1]\%$ 
1094  $\text{ifvmode } \backslash$ IgnorePar $\backslash$ fi  $\backslash$ EndP $\}$ 
1095  $\}$ 
1096  $\langle$ /htXourse $\rangle$ 
```

2.8.7 JavaScript

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

```
1097  $\langle$ *classXimera $\rangle$ 
1098  $\backslash$ DefineVerbatimEnvironment $\{javascript\}\{Verbatim\}\{numbers=left,frame=lines,label=JavaScript,language=JavaScript\}$ 
1099  $\langle$ /classXimera $\rangle$ 

1100  $\langle$ *htXimera $\rangle$ 
1101  $\%$  for programming javascript
1102  $\backslash$ renewenvironment $\{javascript\}\{NoFonts\}\{EndNoFonts\}$ 
1103  $\backslash$ ScriptEnv $\{javascript\}\{\text{stepcounter}\{identification\}\text{ifvmode } \backslash$ IgnorePar $\backslash$ fi  $\backslash$ EndP $\}$ 
1104  $\langle$ /htXimera $\rangle$ 
```


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

```

1105 \classXimera
1106 \def\js#1{\mbox{\texttt{\detokenize{#1}}}}
1107 \endclassXimera

1108 \htXimera
1109 \def\js#1{\stepcounter{identification}\HCode{<span class="inline-javascript" id="javascript\
1110 \endhtXimera

```

2.9 SageMath support

Load SageTeX if it exists.

```

1111 \classXimera
1112 \IfFileExists{sagetex.sty}{\RequirePackage{sagetex}}{}
1113 \endclassXimera

```

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

```

1114 \classXimera
1115 \DefineVerbatimEnvironment{sageCell}{Verbatim}{numbers=left,frame=lines,label=SAGE,labelposition=
1116 \endclassXimera

1117 \htXimera
1118 \renewenvironment{sageCell}{\NoFonts}{\EndNoFonts}
1119 \ScriptEnv{sageCell}{\ifvmode \IgnorePar\fi \EndP\HCode{<div class="sage"><script type="text/
1120 \endhtXimera

```

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

```

1121 \classXimera
1122 \DefineVerbatimEnvironment{sageOutput}{Verbatim}{numbers=left,frame=lines,label=SAGE-Output,
1123 \endclassXimera

1124 \htXimera
1125 \renewenvironment{sageOutput}{\NoFonts}{\EndNoFonts}
1126 \ScriptEnv{sageOutput}{\ifvmode \IgnorePar\fi \EndP\HCode{<div class="sageOutput"><script typ
1127 \endhtXimera

```

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

```

1128 \htXimera
1129 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1130 \ifdefined\sagesilent
1131 \renewenvironment{sagesilent}{\NoFonts}{\EndNoFonts}
1132 \fi
1133 \ScriptEnv{sagesilent}{\ifvmode \IgnorePar\fi \EndP\HCode{<script type="text/sagemath">}\Htm
1134 \endhtXimera

```

2.10 Answerables

2.10.1 Answers

`\answer` A math answer

```

1135 \classXimera
1136
1137 \ifdefined\HCode
1138 \newcommand{\recordvariable}[1]{}
1139 \else
1140 \newwrite\idfile
1141 \immediate\openout\idfile=\jobname.ids
1142 \newcommand{\recordvariable}[1]{\ifthenelse{equal{#1}}{}{\immediate\write\idfile{var #1};}
1143 \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

```

1144 \define@key{answer}{given}[true]{\def\ans@given{#1}}

```

Used for setting numeric answer tolerance for online student input.

```
1145 \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.

```
1146 \define@key{answer}{validator}{}
```

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

```
1147 \define@key{answer}{id}{\def\ans@id{#1}}
```

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

```
1148 \define@key{answer}{format}{}
```

Used to hide the answer input box on the web.

```
1149 \define@key{answer}{onlinenoinput}[false]{}
```

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

```
1150 \define@key{answer}{onlineshowanswerbutton}[false]{}
```

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

```
1151 \setkeys{answer}{id=,given=false,onlinenoinput=false,onlineshowanswerbutton=false}
```

Basic code for \answer.

```
1152
```

```
1153 % Options for handout
```

```
1154 \newcommand{\answerFormatLength}{2cm}
```

```
1155
```

```
1156 \newcommand{\answerFormatDots}[1]{\ldots\ldots}
```

```
1157 \newcommand{\answerFormatLine}[1]{\protect\rule{\answerFormatLength}{0.4pt}}
```

```
1158 \newcommand{\answerFormatFlexibleLine}[1]{\protect\rule{\widthof{${#1$}*2}{0.4pt}}
```

```
1159 \newcommand{\answerFormatFlexibleBox}[1]{\fbox{\scalebox{2}{\phantom{${#1$}}}}
```

```
1160
```

```
1161 % options for default (i.e with answers filled in)
```

```
1162 \newcommand{\answerFormatPlain}[1]{\ensuremath{#1}}
```

```
1163 \newcommand{\answerFormatBlue}[1]{\color{blue}\ensuremath{#1}}
```

```
1164 \newcommand{\answerFormatBoxed}[1]{\fbox{\ensuremath{#1}}}
```

```
1165 \newcommand{\answerFormatBoxedGiven}[1]{\underset{\scriptstyle\mathrm{given}}{\fbox{\ensuremath{#1}}}}
```

```
1166
```

```
1167 % defaults for handout and default mode, and for \answer[given]
```

```
1168 \let\handoutAnswerFormat\answerFormatDots
```

```
1169 \let\defaultAnswerFormat\answerFormatBlue
```

```
1170 \let\givenAnswerFormat\answerFormatBoxedGiven
```

```
1171
```

```
1172 \newcommand{\answer}[2][{}]{%
```

```
1173 \ifmode%
```

```
1174 \setkeys{answer}{#1}%
```

```
1175 \recordvariable{\ans@id}
```

```
1176 \ifthenelse{\boolean{\ans@given}}{
```

```
1177 {% Start then statement
```

```
1178 \ifhandout
```

```
1179 #2
```

```
1180 \else
```

```
1181 \givenAnswerFormat{#2} %% in case the argument helps formatting
```

```
1182 \fi
```

```
1183 }% End then statement
```

```
1184 {% Start else statement
```

```
1185 \ifhandout
```

```
1186 \handoutAnswerFormat{#2} %% in case the argument helps formatting
```

```
1187 \else% show answer in box outside handout mode
```

```
1188 \defaultAnswerFormat{#2} %% in case the argument helps formatting
```

```
1189 \fi
```

```
1190 }% End else statement
```

```
1191 \else%
```

```
1192 \GenericError{\space\space\space\space}% Throw an error based on... something? -- Jason
```

```
1193 {Attempt to use \@backslashchar answer outside of math mode}
```

```
1194 {See https://github.com/ximeraProject/ximeraLatex for explanation.}
```

```

1195 {Need to use either inline or display math.}%
1196 \fi
1197 }
1198 \end{classXimera}

```

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

```

1199 \begin{classXimera}
1200 \renewcommand{\answer}[2][false]{\HCode{<span class="answer_respondable">}#2\HCode{</span>}}
1201
1202 \def\validator[#1]{\stepcounter{identification}\HCode{<div class="validator" id="validator\an
1203 \def\endvalidator{\HCode{</div>}}
1204
1205 \end{classXimera}

```

2.10.2 Multiple choice and the like

`multipleChoice` (*env.*) Multiple choice

```

1206 \begin{classXimera}
1207 % Jim: Originally this was \renewcommand{\theenumi}{$(\mathrm{\alph{enumi}})$}
1208 % but that breaks tex4ht because mathmode can only be processed by mathjax.
1209 % so now I made this just italicized.

```

2.10.3 Options

```

1210 \define@key{choice}{value}[]{\def\choice@value{#1}}

```

This flags the answer as the correct answer

```

1211 \define@boolkey{choice}{correct}[true]{\def\choice@correct{#1}}

```

Use an ID to refer to the choice.

```

1212 \define@key{multipleChoice}{id}{\def\mc@id{#1}}

```

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

```

1213 \define@key{otherchoice}{value}[]{\def\otherchoice@value{#1}}

```

```

1214 \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".

```

1215 \setkeys{choice}{correct=false,value=}

```

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

```

1216 \setkeys{multipleChoice}{id=}

```

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

```

1217 \setkeys{otherchoice}{correct=false,value=}

```

```

1218 \end{classXimera}

```

2.10.4 Choices

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

```

1219 \begin{classXimera}
1220 \newcommand{\choice}[2][]{%
1221 \setkeys{choice}{#1}%
1222 \item{#2}
1223 \ifthenelse{\boolean{\choice@correct}}{
1224   {% Begin then result
1225     \ifhandout% if it's a handout do nothing.
1226     \else% otherwise place a checkmark when you select the "correct choice"... maybe? -- Jas
1227       \, \checkmark \, \setkeys{choice}{correct=false}
1228     \fi
1229   }% End then result
1230   }{% Begin/End else result.
1231 }
1232
1233 %Define an expandable version of choice Not really meant to be used outside this package (use

```

```

1234 % Is there a reason we can't just always use this as default? -- Jason
1235 \newcommand{\choiceEXP}[2][]{%
1236   \expandafter\setkeys\expandafter{\choice}{#1}%
1237   \item{#2}
1238   \ifthenelse{\boolean{\choice@correct}}{
1239     {% Begin then result
1240       \ifhandout
1241         \else
1242         \, \checkmark \, \setkeys{\choice}{correct=false}
1243       \fi
1244     }% End then result
1245   }{% Begin/End else result.
1246 } %% note all the {} are needed in case the choice has [] in it.
1247
1248 % \otherchoice is the \choice used in wordChoice command.
1249 \newcommand{\otherchoice}[2][]{%
1250   \ignorespaces%
1251   \setkeys{\otherchoice}{#1}%
1252   \ifthenelse{\boolean{\otherchoice@correct}}{
1253     {% Start then result
1254       #2\ignorespaces\setkeys{\otherchoice}{correct=false}\ignorespaces%
1255     }% End then result
1256   }{% Start/End else result
1257   \ignorespaces%
1258 }%
1259 \newcommand{\inlinechoice}[2][]{%
1260   \setkeys{\choice}{#1}%
1261   \iffirstinlinechoice
1262     (\hspace{-.25em}
1263     \firstinlinechoicefalse
1264   \else
1265   /
1266   \fi
1267   #2
1268   \ifthenelse{\boolean{\choice@correct}}{
1269     {% Start then result
1270       \ifhandout\else\checkmark\ignorespaces\setkeys{\choice}{correct=false}\ignorespaces\fi
1271     }% End then result
1272   }{% Start/End else result
1273     \hspace{-.25em}\ignorespaces%
1274   }
1275
1276 \</classXimera>

```

On the HTML side, \choice emits s.

```

1277 \*htXimera>
1278 \newcounter{choiceId}
1279 \renewcommand{\choice}[2][]{%
1280   \setkeys{\choice}{correct=false}%
1281   \setkeys{\choice}{#1}%
1282   \stepcounter{choiceId}\IgnorePar%
1283   \HCode{<span class="choice }%
1284   \ifthenelse{\boolean{\choice@correct}}{\HCode{correct}}{}
1285   \HCode{" }
1286   \ifthenelse{\equal{\choice@value}{}}{}{\HCode{data-value="\choice@value" }}
1287   \HCode{id="choice\arabic{choiceId}">}%
1288   #2\HCode{</span>}}
1289 \let\inlinechoice\choice
1290 \</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.

```

1291 <*classXimera>
1292 \newenvironment{multipleChoice}[1]{}
1293 {% Environment Start Code
1294 \setkeys{multipleChoice}{#1}%
1295 \recordvariable{\mc@id}%
1296 \begin{trivlist}
1297 \item[\hspace\labelsep\small\bfseries \GetTranslation{Multiple Choice}:]\hfil
1298 \begin{enumerate}
1299 }% Note this means that \item has to be the first line after \begin{multipleChoice}.
1300 {% Environment End Code
1301 \end{enumerate}
1302 \end{trivlist}
1303 }
1304
1305 %multipleChoice@ is for internal use only! (used in wordChoice)
1306 %this is simply a wrapper for the sole showing (other)choice.
1307 \newenvironment{multipleChoice@}[1]{}{}{}
1308 </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.

```

1309 <*htXimera>
1310 \renewenvironment{multipleChoice}[1]{}
1311 {%\setkeys{multipleChoice}{#1}%
1312 \stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div class="multiple-choice"
1313 \ifthenelse{\equal{\mc@id}{}}{\}\{\HCode{data-id="\mc@id" }}}%
1314 \HCode{id="problem\arabic{identification}" titletext=" \GetTranslation{Multiple Choice}">}%
1315 \}\{\HCode{</div>}\IgnoreIndent}
1316 \ConfigureEnv{multipleChoice}{\}\{\}\{\}
1317 </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.

```

1318 <*classXimera>
1319 \newcommand{\wordChoice}[1]{%
1320 \let\choicetemp\choice% Assign a "choicetemp" command to duplicate choice.
1321 \ifwordchoicegiven% If wordchoice option is on, we need to juggle around some definitions.
1322 \let\choice\otherchoice%
1323 %\begin{multipleChoice@}% -unnecessary (REMOVE THIS LINE IF THE YEAR IS 2019 or Beyond)
1324 #1
1325 %\end{multipleChoice@}% -unnecessary (REMOVE THIS LINE IF THE YEAR IS 2019 or Beyond)
1326 \else% If it isn't the regular "choice" command should work.
1327 \let\choice\inlinechoice%
1328 \begin{multipleChoice@}%
1329 #1%
1330 \end{multipleChoice@}%
1331 \fi%
1332 \let\choice\choicetemp% Now that choicetmp has been manipulated to what we want, replace choi
1333 }%
1334
1335
1336 </classXimera>

```

This is actually just word choice

```

1337 <*htXimera>
1338 \renewenvironment{multipleChoice@}{\refstepcounter{problem}}{}%
1339 \ConfigureEnv{multipleChoice@}{\stepcounter{identification}\IgnorePar\HCode{<span class="word
1340 </htXimera>

```

2.12 Select all

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

```
1341 \classXimera
1342 \newenvironment{selectAll}[1]{}
1343 {\begin{trivlist}\item[\hskip \labelsep\small\bfseries \GetTranslation{Select All Correct Answer}]
1344   {\end{enumerate}\end{trivlist}}
1345 \endclassXimera
```

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`.

```
1346 \htXimera
1347 \renewenvironment{selectAll}{\refstepcounter{problem}}{}
1348 \ConfigureEnv{selectAll}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div
1349 \endhtXimera
```

2.12.1 Free response

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

```
1350 \classXimera
1351 \newboolean{given} %% required for freeResponse
1352 \setboolean{given}{true} %% could be replaced by a key=value pair later if needed
1353
1354 \ifhandout
1355   \newenvironment{freeResponse}[1][false]{}
1356   {}
1357   \def\givenatend{\boolean{#1}}
1358   \ifthenelse{\boolean{#1}}{}
1359   {% Begin then result
1360     \begin{trivlist}
1361       \item
1362     }% End then result
1363     {% Begin else result
1364       \setbox0\vbox\bgroup
1365     }% End else result
1366 %   {}% Don't think this is doing anything? -- Jason
1367   }
1368   {}
1369   \ifthenelse{\givenatend}{}
1370   {% Begin then result
1371     \end{trivlist}
1372   }% End then result
1373   {% Begin else result
1374     \egroup
1375   }% End else result
1376 %   {}% Don't think this is doing anything? -- Jason
1377   }
1378 \else
1379   \newenvironment{freeResponse}[1][false]{}
1380   {} Environment Beginning Code
1381   \ifthenelse{\boolean{#1}}{}% Could probably change this with just putting the (given) in t
1382   {% Begin then result
1383     \begin{trivlist}
1384       \item[\hskip \labelsep\bfseries \GetTranslation{Free Response (Given)}:\hspace{2ex}]
1385     }% End then result
1386     {% Begin else result
1387       \begin{trivlist}
1388         \item[\hskip \labelsep\bfseries \GetTranslation{Free Response}:\hspace{2ex}]
1389       }% End else result
```

```

1390 }
1391 {% Environment Ending Code
1392 \end{trivlist}
1393 }
1394 \fi
1395
1396 \end{classXimera}
1397 \end{*htXimera}
1398
1399 \renewenvironment{freeResponse}{\refstepcounter{problem}}{}%
1400 \ConfigureEnv{freeResponse}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<
1401
1402 \end{*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.

```

1403 \begin{classXimera}
1404 \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.

```

1405 \newenvironment{validator}[1][]{
1406   \def\PH@Command{#1}% Use PH@Command to hold the content and be a target for "\expandafter"
1407   \mbox{\texttt{\detokenize\expandafter{\PH@Command}}}% Now expand PH@Command once and then d
1408 }{}

```

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

```

1409 \ifhandout%
1410 \newenvironment{feedback}
1411   {%
1412   \setbox0\vbox\bgroup
1413   }
1414   {%
1415   \egroup
1416   }

```

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.

```

1417 \else
1418 \newenvironment{feedback}[1][attempt]{
1419
1420 \edef\PH@Command{\GetTranslation{#1}}% Use PH@Command to hold the content and be a target for
1421
1422 \begin{trivlist}% Begin the trivlist to use formatting of the "Feedback" label.
1423 \item[\hskip \labelsep\small\slshape\bfseries \GetTranslation{Feedback}]% Format the "Feedback
1424 \ifonlineTF{% If the feedback is on a pdf, we don't need to detokenize - which messes with t
1425 (\texttt{\expandafter\detokenize\expandafter{\PH@Command}})% Keep the online version the sa
1426 {(\expandafter\texttt{\PH@Command}}):% No need for detokenize in the pdf version
1427 \hspace{2ex}}\small\slshape% Insert some space before the actual feedback given.
1428 }{
1429 \end{trivlist}
1430 }
1431

```

```

1432 \fi
1433 \endclassXimera

```

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

```

1434 \beginXimera
1435 \def\feedback{\ifnextchar[{\@feedbackcode}{\@feedbackattempt}}
1436 \def\@feedbackattempt{\@feedbackcode[attempt]}
1437 \def\@feedbackcode[#1]{\stepcounter{identification}%
1438 \ifvmode \IgnorePar\fi \EndP%
1439 \ifthenelse{\equal{#1}{attempt}}{\HCode{<div class="feedback" data-feedback="attempt" id="fe
1440 {\ifthenelse{\equal{#1}{correct}}{\HCode{<div class="feedback" data-feedback="correct" id="f
1441 {\HCode{<div class="feedback" data-feedback="script" id="feedback\arabic{identification}" ti
1442 \def\endfeedback{\HCode{</div>}\IgnoreIndent}
1443 \endXimera

```

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.

```

1444 \beginclassXimera
1445 \newenvironment{ungraded}{}{}
1446 \endclassXimera

```

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

```

1447 \beginXimera
1448 \renewenvironment{ungraded}{}%
1449 \ifvmode \IgnorePar\fi \EndP\HCode{<div class="ungraded">}\IgnoreIndent%
1450 }{
1451 \ifvmode \IgnorePar\fi \EndP\HCode{</div>}\IgnoreIndent%
1452 }
1453 \endXimera

```

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.

```

1454 \beginclassXimera
1455 %% Pre-202412: .jax file written in non-\HCode, and in a next run inserted by ximera.cfg in
1456 %% Post-202501: .mjax file written only in \HCode, and in luaxake post-processing inserted in
1457 %% ( used luaxake rather than sed ...)
1458 \newwrite\myfile
1459 \ifdefined\HCode
1460 \immediate\openout\myfile=\jobname.xmjax
1461
1462 %% From |only.dtx| we must also create |prompt| on the MathJax side.
1463 \immediate\write\myfile{\unexpanded{\newenvironment}{prompt}}{}{}
1464
1465 %% Write all newcommands to .xmjax file, that will be included in the .html via luaxake
1466 \let\oldargdef\argdef
1467 \long\def\argdef#1[#2]#3{%
1468 \immediate\write\myfile{\unexpanded{\newcommand}{\unexpanded{#1}}[\unexpanded{#2}]{\unexpanded{#3}}%
1469 \oldargdef#1[#2]{#3}%
1470 }
1471
1472 %% Same for \DeclareMathOperator
1473 \let\oldDeclareMathOperator\DeclareMathOperator
1474 \renewcommand{\DeclareMathOperator}[2]{\oldDeclareMathOperator{#1}{#2}\immediate\write\myfile{

```



```

1475
1476 \fi
1477
1478
1479 </classXimera>
Include the jax'ed newcommands (pre-202412 versions ....)
1480 <*cfgXimera>
1481
1482 % 202501: removed sed-manipulation of .jax file; see luaxake now
1483
1484 \Configure{BVerbatimInput}{\}\}\}\}
1485
1486 \Configure{verbatiminput}{\}\}\}\}
1487
1488 % Instead of a nonbreaking space, use a standard space
1489 \makeatletter
1490 \def\FV@Space{\space}
1491 \makeatother
1492
1493 % Include the (problem-?) .ids in a text/javascript script right at the beginning of the body
1494 \Configure{BODY}{%
1495 \HCode{<body>\Hnewline}%
1496 \Tg<div class="preamble">%
1497 %% 202501: removed .jax inclusion (see luaxake)
1498
1499 %% Include the .ids file
1500 \IfFileExists{\jobname.ids}{\HCode{<script type="text/javascript">\Hnewline}%
1501 \BVerbatimInput{\jobname.ids}%
1502 \HCode{</script>\Hnewline}%
1503 }\}
1504 \Tg</div>%
1505 }{%
1506 \ifvmode\IgnorePar\fi\EndP\HCode{</body>\Hnewline}%
1507 }
1508
1509 % 202501: removed 'prevent spaces as in "\begin{align}": this is done in luaxake now
1510
1511 % This is a fix for the LAODE book, which uses matlabEquation as if it were an equation
1512 \ScriptEnv{matlabEquation}{\ifvmode \IgnorePar\fi \EndP\HCode{<script type="math/tex; mode=d
1513
1514 </cfgXimera>

```

2.13.2 Semantic HTML

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

```

1515 <*cfgXimera>
1516 \Configure{textbf}{\ifvmode\ShowPar\fi\HCode{<strong>}}{\HCode{</strong>}}
1517 </cfgXimera>

```

`\textit` Using `\textit` or similar emits an `` tag.

```

1518 <*cfgXimera>
1519 \Configure{textit}{\ifvmode\ShowPar\fi\HCode{<em>}}{\HCode{</em>}}
1520 \Configure{emph}{\ifvmode\ShowPar\fi\HCode{<em>}}{\HCode{</em>}}
1521 </cfgXimera>

```

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

```

1522 <*cfgXimera>
1523 \Configure{texttt}{\ifvmode\ShowPar\fi\HCode{<code>}}{\HCode{</code>}}
1524 </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`.

```
1525 <*classXimera>
1526 \font\dummyft@=dummy \relax
1527 \def\suppress{%
1528   \begingroup\par
1529   \parskip\z@
1530   \offinterlineskip
1531   \baselineskip=\z@skip
1532   \lineskip=\z@skip
1533   \lineskiplimit=\maxdimen
1534   \dummyft@
1535   \count@\sixt@@n
1536   \loop\ifnum\count@ >\z@
1537     \advance\count@\m@ne
1538     \textfont\count@\dummyft@
1539     \scriptfont\count@\dummyft@
1540     \scriptscriptfont\count@\dummyft@
1541   \repeat
1542   \let\selectfont\relax
1543   \let\mathversion\@gobble
1544   \let\getanddefine@fonts\@gobbletwo
1545   \tracinglostchars\z@
1546   \frenchspacing
1547   \hbadness\@M}
1548 \def\endsuppress{\par\endgroup}
1549 </classXimera>
```

2.14.2 The End

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

```
1550 <*htXimera>
1551 \Hinput{ximera}
1552 </htXimera>

1553 <*htXourse>
1554 \Hinput{xourse}
1555 </htXourse>

1556 <*cfgXimera>
1557 \begin{document}
1558 \EndPreamble
1559 </cfgXimera>
```

3 xourse.cls

```
1560 <*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.

```
1561 \newif\ifnotoc
1562 \notocfalse
1563 \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.

```
1564 \newif\ifnonewpage
1565 \nonewpagefalse
1566 \DeclareOption{nonewpage}{\nonewpagetrue}
```

```

1567 \DeclareOption*{\PassOptionsToClass{\CurrentOption}{ximera}}
1568 \ProcessOptions\relax
1569 \LoadClass{ximera}
1570 % \begin{macrocode}
1571 \end{macrocode}

```

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`.

```

1572 (*classXourse)
1573 \newcommand{\skip@preamble}{%
1574     \let\document\relax\let\enddocument\relax%
1575     \newenvironment{document}{\let\input\otherinput}{}%
1576     \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:

```
1577 \let\otherinput\input
```

Store usual `\maketitle` as `\othermaketitle`

```
1578 \let\othermaketitle\maketitle
```

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

```

1579 \renewcommand{\maketitle}{%
1580 \pagestyle{empty}
1581 \begin{center}
1582 ~\ \ %puts space at top of page to move title down.
1583 \vskip .25\textheight
1584 \hrulefill\
1585 \vskip 1em
1586 \bfseries{\Huge \@title} \
1587 \hrulefill\
1588 \vskip 3em
1589 {\Large \@author}
1590 \vskip 2em
1591 {\large \@date}
1592 \end{center}
1593 \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.

```

1594 \ifnotoc
1595 \else
1596     \tableofcontents\clearpage
1597     \clearpage
1598 \fi

```

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

```
1599 \pagestyle{main}
```

Renew `maketitle` to usual definition.

```
1600 \let\maketitle\othermaketitle
```

And we finish with our redefinition of `\maketitle`.

```

1601 }
1602 \relax
1603 \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.

```

1604 \classXourse
1605 \ifnonewpage
1606 \newcommand{\activity}[2][]{%
1607   \setkeys{activity}{#1}
1608   \renewcommand{\input}[1]{
1609     \begingroup\skip@preamble\otherinput{#2}\endgroup\par\vspace{\topsep}
1610     \let\input\otherinput}
1611 \else
1612 \newcommand{\activity}[2][]{%
1613   \setkeys{activity}{#1}
1614   \renewcommand{\input}[1]{
1615     \begingroup\skip@preamble\otherinput{#2}\endgroup\clearpage
1616     \let\input\otherinput}
1617 \fi
1618 \relax
1619 \endclassXourse
1620 \htXourse
1621 \renewcommand\activity[2][]{%
1622 \ifvmode \IgnorePar\fi \EndP\HCode{<a class="activity card \activitystyle" href="#2" data-op
1623 }
1624 \endhtXourse

```

When running xake, we can just ignore activities

```

1625 \classXourse
1626 \ifxake
1627 \renewcommand\activity[2][]{%
1628 \fi
1629 \endclassXourse

```

3.1.2 Practice activities

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

```

1630 \classXourse
1631 \ifhandout
1632 \newcommand{\practice}[2][]{%
1633   \setkeys{practice}{#1}%!!!!
1634   \renewcommand{\input}[1]{
1635     \begingroup\skip@preamble\otherinput{#2}\endgroup
1636     \let\input\otherinput}
1637 \else
1638 \newcommand{\practice}[2][]{\texttt{\detokenize{#2}}}% gives file name for practice
1639 \setkeys{practice}{#1}%!!!!
1640 \renewcommand{\input}[1]{
1641   \begingroup\skip@preamble\otherinput{#2}\endgroup
1642   \let\input\otherinput}
1643 \fi
1644 \relax
1645 \endclassXourse

```

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

```

1646 \classXourse
1647 \ifxake
1648 \renewcommand\practice[2][]{%

```

```

1649 \fi
1650 \end{classXourse}

```

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

```

1651 \begin{htXourse}
1652 \renewcommand\practice[2][]{\%
1653   \ifvmode\IgnorePar\fi\EndP\%
1654   \HCode{\<a class="activity card practice" href="#2" data-options="#1">#2</a>}}\%
1655   \IgnoreIndent\%
1656 }
1657 \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.

```

1658 \begin{classXourse}
1659 \renewcommand*\l@section{\@dottedtocline{1}{1.5em}{4.2em}}
1660 \end{classXourse}

```

`\subsection` The name of a subsection inside an activity.

```

1661 \begin{classXourse}
1662 \renewcommand*\l@subsection{\@dottedtocline{2}{3.8em}{4.2em}}
1663 \end{classXourse}

```

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

```

1664 \begin{htXourse}
1665 \newcounter{ximera@part}
1666 \setcounter{ximera@part}{0}
1667 \renewcommand\part[1]{\%
1668   \stepcounter{ximera@part}\%
1669   \ifvmode \IgnorePar\fi \EndP\%
1670   \% \HCode{\<h1 id="part\arabic{ximera@part}" class="card part">#1\HCode{\</h1>}}\% makes cards dis
1671   \HCode{\<h1 id="part\arabic{ximera@part}" class="card part">#1\</h1>}}\%
1672   \IgnoreIndent\%
1673 }
1674 \end{htXourse}

```

`\paragraph` Paragraph commands emit spans. A small heading.

```

1675 \begin{cfgXimera}
1676 \renewcommand{\paragraph}[1]{\%
1677   \HCode{\<span class="paragraphHead">}}\%
1678   #1\%
1679   \HCode{\</span>}\par\IgnorePar}
1680 \end{cfgXimera}

```

`\subparagraph` An even smaller heading.

```

1681 \begin{cfgXimera}
1682 \renewcommand{\subparagraph}[1]{\%
1683   \HCode{\<span class="subparagraphHead">}}\%
1684   #1\%
1685   \HCode{\</span>}\par\IgnorePar}
1686 \end{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.

```

1687 \begin{classXourse}
1688 \newenvironment{graded}[1]{}{}
1689 \end{classXourse}

```

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

```

1690 <*htXourse>
1691 \renewenvironment{graded}[1]{%
1692 \ifvmode \IgnorePar\fi \EndP\HCode{<div class="graded" data-weight="#1">}\IgnoreIndent%
1693 }{
1694 \ifvmode \IgnorePar\fi \EndP\HCode{</div>}\IgnoreIndent%
1695 }
1696 </htXourse>

```

3.4 Logos

`\logo` A logo for the xourse.

```

1697 <*classXourse>
1698 \newcommand*{\logo}[1]{%
1699 \ifx\@onlypreamble\@notprerr
1700 \ClassError{xourse}{logo can only be used in the preamble}
1701 {Move your logo command to the preamble}
1702 \else %
1703 \IfFileExists{#1}%
1704 {\gdef\xourse@logo{#1}}%
1705 {\ClassError{xourse}{logo file does not exist}
1706 {To use logo, make sure that the referenced image file exists}}%
1707 \fi%
1708 }
1709
1710 </classXourse>

```

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

```

1711 <*htXourse>
1712 \Configure{@HEAD}{%
1713 \HCode{<meta name="og:image" content="}%
1714 \ifdefined\xourse@logo%
1715 \xourse@logo%
1716 \fi%
1717 \HCode{" />\Hnewline}}%
1718 </htXourse>

```