

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 \RequirePackage{xkeyval}
88 \RequirePackage{expl3}% for tables to use LaTeX3 programming
89 \RequirePackage{currfile}
90 \RequirePackage{comment}
91 \RequirePackage{tabulararray}% For accessibility features in tables
92 \classXimera
```

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

```
93 \classXimera
94 \RequirePackage{getttitlestring}
95 \RequirePackage{nameref}
96 \RequirePackage{epstopdf}
97 \classXimera
```

2.3 Page setup

We want non-indented spaced-out paragraphs.

```
98 \classXimera
99 \setlength{\parindent}{0pt}
100 \setlength{\parskip}{5pt}
101 \classXimera
```

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

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

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

```
108 \cfgXimera
109 \Preamble{xhtml,mathjax,minipage-width}
110
111 % We don't want to translate font suggestions with ugly wrappers like
112 % <span class="cmti-10"> for italic text
113 \NoFonts
114
115 % Don't output xml version tag
116 % \Configure{VERSION}{\HCode{<?xml version="1.0" />}}
117
118 % Output HTML5 doctype instead of the default for HTML4
119 % \Configure{DOCTYPE}{\HCode{<!doctype html>}}
120
121 % Custom page opening
122 % \Configure{HTML}{\HCode{<html lang="en">}}
123
124 % Reset <head>, aka delete all default boilerplate; alternatively set up new content
125 % \Configure{@HEAD}{\HCode{<meta name="generator" content="TeX4ht (http://www.cse.ohio-state.edu/~dpc/TeX4ht/)>}}
126 \Configure{@HEAD}{\HCode{<meta name="ximera" content="version 2.5.1" />}}
127 \Configure{@HEAD}{\HCode{<link href="https://ximera.osu.edu/public/stylesheets/standalone.css" />}}
128 \Configure{@HEAD}{\HCode{<script type="text/javascript" async src="https://ximera.osu.edu/public/js/ximera.js">}}
129
```

```

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

```

Disable certain ligatures in HTML.

```

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

```

I am not sure what this does.

```

152 <*htXimera>
153 \NewEnviron{html}{\HCode{\BODY}}
154 </htXimera>

```

2.4 Structure

2.4.1 Macros

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

```

155 <*classXimera>
156 \everymath{\displaystyle}
157 </classXimera>

```

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

```

158 <*classXimera>
159 \let\prelim\lim
160 \renewcommand{\lim}{\displaystyle\prelim}
161 </classXimera>

```

2.4.2 Theorem and theorem-like environments

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

```

162 <*htXimera>
163 \newcommand{\ConfigureTheoremEnv}[1]{%
164 \renewenvironment{#1}[1][\refstepcounter{problem}%
165 \ifthenelse{\equal{##1}{}}{}{}%
166 \HCode{<span class="theorem-like-title">##1\HCode{</span>}}%
167 }{}%
168 \ConfigureEnv{#1}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div class=
169 }
170 </htXimera>
171 <classXimera>\theoremstyle{definition} % No italic (because this makes also text in TikZ italic

```

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

```

theorem Theorem
172 <classXimera> \newtheorem{theorem}{\GetTranslation{Theorem}}
173 <htXimera> \ConfigureTheoremEnv{theorem}

```

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

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

2.4.3 Enumerate fixes

Make enumerate use a letter

```

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

230 \*cfgXimera
231 \catcode'\:=11
232 % Insert <section> around thebibliography
233 \ConfigureEnv{thebibliography}{\ifvmode\IgnorePar\fi \EndP \HCode{<section role="doc-bibliography">}}
234 % now configure thebibliography to produce a description list
235 % \en:bib insertes delimiters for particular bibitems. at the beginning, it is empty, as then
236 % it is then defined to insert the delimiter after the first bibitem
237 \ConfigureList{thebibliography}%
238 {\ifvmode\IgnorePar\fi\EndP\HCode{<dl><dt>}\let\en:bib=\empty}% opening tags
239 {\ifvmode\IgnorePar\fi\EndP\HCode{</dd></dl>}} % closing tags
240 {\en:bib\def\en:bib{\ifvmode\IgnorePar\fi\HCode{</dd><dt>}}}% at the bibitem
241 {\HCode{</dt><dd>}}}% after biblabel
242 \catcode'\:=12
243 \Css{.thebibliography dl {
244 display: grid;
245 grid-auto-columns: min-content 1fr;
246 grid-auto-flow: column;
247 }}
248 \Css{.thebibliography dt {
249 grid-column: 1;
250 margin-bottom: 0.5em;
251 }}
252 \catcode'\:=11
253 \ConfigureList{enumerate}%
254 {\EndP\HCode{<ol \a:enumerate:\space
255 class="enumerate\expandafter\the\csname @enumdepth\endcsname"
256 \a:LRdir
257 >}\PushMacro\end:itm

```

```

258 \global\let\end:itm=\empty
259 }
260         {\PopMacro\end:itm \global\let\end:itm \end:itm
261 %
262 \EndP\HCode{</li></ol>}\ShowPar
263 }
264         {\end:itm \gdef\end:itm{\EndP\Tg</li>}\DeleteMark
265 }
266         {{\Configure{Link}{li}{}}{ class="enumerate" id=}{}}%
267 \let\EndLink=\empty\par\ShowPar
268 \AnchorLabel }%
269 }
270 \catcode'\:=12
271 \</cfgXimera>

```

2.4.4 Proofs

proof A mathematical proof environment.

```

272 \*classXimera>
273 \renewcommand{\qedsymbol}{\blacksquare}
274 \renewenvironment{proof}[1][\proofname]
275   {\begin{trivlist}\item[\hskip \labelsep \itshape \bfseries #1]{\hspace{2ex}}}{
276   {\qed\end{trivlist}}}
277 \</classXimera>
278 \*htXimera>
279         % Mmm, (why) do we want/need this ...?
280         \ConfigureTheoremEnv{proof}
281 \ConfigureEnv{proof}{\ifvmode\IgnorePar\fi\EndP\HCode{<div class="proof">}
282 \ConfigureList{trivlist}{\ifvmode\IgnorePar\fi\EndP}{\}{\}}
283 {\ifvmode\IgnorePar\fi\EndP\HCode{</div>}}{\}{\}}
284 \</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>

```

285 \*classXimera>
286 \newcommand{\hang}{% top theorem decoration
287   \begingroup%
288   \setlength{\unitlength}{.005\linewidth}% \linewidth/200
289 \begin{picture}(0,0)(1.5,0)%
290   \linethickness{1pt} \color{black!50}%
291   \put(-3,2){\line(1,0){206}}% Top line
292   \multido{\iA=2+-1,\iB=50+-10}{5}{% Top hangs
293     \color{black!\iB}%
294     \put(-3,\iA){\line(0,-1){1}}% Top left hang
295     %\put(203,\iA){\line(0,-1){1}}% Top right hang
296   }%
297 \end{picture}%
298   \endgroup%
299 }%
300 \newcommand{\hung}{% bottom theorem decoration
301   \nobreak
302   \begingroup%
303   \setlength{\unitlength}{.005\linewidth}% \linewidth/200
304 \begin{picture}(0,0)(1.5,0)%
305   \linethickness{1pt} \color{black!50}%
306   \put(60,0){\line(1,0){143}}% Bottom line
307   \multido{\iA=0+1,\iB=50+-10}{5}{% Bottom hangs
308     \color{black!\iB}%
309     %\put(-3,\iA){\line(0,1){1}}% Bottom left hang

```

```

310 \put(203,\iA){\line(0,1){1}}% Bottom right hang
311 \put(\iB,0){\line(60,0){10}}% Left fade out
312 }%
313 \end{picture}%
314 \endgroup%
315 }%

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.

316 \MakeCounter{Iteration@probCnt}
317 \MakeCounter{problem}
318 \newcommand{\problemNumber}{
319 % First we determine if we have a counter for this question depth level.
320 \ifcsname c@depth\Roman{problem@Depth}Count\endcsname% Check to see if counter exists
321 %If so, do nothing.
322 \else
323 %If not, create it.
324 \expandafter\newcounter{depth\Roman{problem@Depth}Count}
325 \expandafter\setcounter{depth\Roman{problem@Depth}Count}{0}
326 \fi
327
328 \expandafter\stepcounter{depth\Roman{problem@Depth}Count}
329 \arabic{depthICount}% The first problem depth, what use to be |\theproblem|.
330
331 \forloop{Iteration@probCnt}{2}{\arabic{Iteration@probCnt} < \numexpr \value{problem@Depth} +
332 .\expandafter\arabic{depth\Roman{Iteration@probCnt}Count}}% Get the problem number of the next
333 }
334 }
335 %%%% Configure various problem environment commands
336 \Make@Counter{problem@Depth}
337 %%%% Configure environments start content
338 \newcommand{\problemEnvironmentStart}[2]{%
339 \stepcounter{problem@Depth}% Started a problem, so we've sunk another problem layer.
340 \def\spaceatend{#1}%
341 \begin{trivlist}%
342 \item[\hspace\labelsep\sffamily\bfseries\GetTranslation{#2} \problemNumber% Determine the cor
343 ]%
344 \slshape
345 }
346 %%%% Configure environments end content %%%%
347 \newcommand{\problemEnvironmentEnd}{%This configures all the end content for a problem.
348 \stepcounter{problem@Depth}
349 \ifcsname c@depth\Roman{problem@Depth}Count\endcsname
350 \expandafter\ifnum\expandafter\value{depth\Roman{problem@Depth}Count}>0
351 \expandafter\setcounter{depth\Roman{problem@Depth}Count}{0}
352 \fi
353 \fi
354 \addtocounter{problem@Depth}{-2}% Exited a problem so we've exited a problem layer. Need -2 b
355 \ifhandout
356 \ifnewpage
357 \newpage
358 \fi
359 \fi
360 \end{trivlist}
361 }
362 %% Add a simple command that handles all the problem creation aspects:
363 \newcommand{\createProblemEnv}[2]{% This is a nice command to define a new problem-like envi
364 \newenvironment{#1}[1][2in]%
365 {%Env start code
366 \problemEnvironmentStart{#1}{#2}
367 }
368 {%Env end code
369 \problemEnvironmentEnd

```

```

370 }
371 }
372
373 %%% Now populate the old environment names
374 %
375 % Old environments were "problem", "exercise", "exploration", and "question".
376 % Note that you can add content to the start/end code on top of these base code pieces if you
377 %
378 % These definitions will be overwritten in ximera.4ht !
379
380 \createProblemEnv{problem}{Problem}
381 \createProblemEnv{exercise}{Exercise}
382 \createProblemEnv{exploration}{Exploration}
383 \createProblemEnv{question}{Question}
384 \end{classXimera}
385 \begin{Ximera}
386 \newcounter{identification}
387 \setcounter{identification}{0}
388 \newcommand{\ConfigureQuestionEnv}[2]{%
389 \renewenvironment{#1}{
390   }
391   {
392   }%
393   \ConfigureEnv{#1}
394   {
395 %     \ifnumberedProblems% The code below is all to generate online problem numbering if optio
396 %     \stepcounter{problem@Depth}% Started a problem, so we've sunk another problem layer.
397 %     \ifcsname c@depth\Roman{problem@Depth}Count\endcsname
398 %     \else
399 %       \expandafter\newcounter{depth\Roman{problem@Depth}Count}
400 %       \expandafter\setcounter{depth\Roman{problem@Depth}Count}{0}
401 %     \fi
402 %     \expandafter\stepcounter{depth\Roman{problem@Depth}Count}
403 %     \def\problemNumDisp{
404 %       \arabic{depthICount}% Top Level Problem Number: X.1.1.1.1 Number.
405 %       \ifcsname c@depthIICount\endcsname\ifnum\value{problem@Depth}>1 .\arabic{depthIICount}\fi
406 %       \ifcsname c@depthIIICount\endcsname\ifnum\value{problem@Depth}>2 .\arabic{depthIIICount}\fi
407 %       \ifcsname c@depthIVCount\endcsname\ifnum\value{problem@Depth}>3 .\arabic{depthIVCount}\fi
408 %       \ifcsname c@depthVCount\endcsname\ifnum\value{problem@Depth}>4 .\arabic{depthVCount}\fi
409 %     \fi\fi\fi\fi
410 %   }
411 %   \else
412 %     \def\problemNumDisp{}% Otherwise don't display a problem number.
413 %   \fi
414 %   \stepcounter{identification}
415 %   \ifvmode
416 %     \IgnorePar
417 %   \fi
418 \EndP
419 \HCode{<div role="article" class="problem-environment #1" id="problem\arabic{identification}"
420 }
421 {
422 \stepcounter{problem@Depth}
423 \ifcsname c@depth\Roman{problem@Depth}Count\endcsname
424 \expandafter\ifnum\expandafter\value{depth\Roman{problem@Depth}Count}>0
425 \expandafter\setcounter{depth\Roman{problem@Depth}Count}{0}
426 \fi
427 \fi
428 \addtocounter{problem@Depth}{-2}% Exited a problem so we've exited a problem layer. Need -2
429 \ifvmode
430 \IgnorePar
431 \fi
432 \EndP

```

```

433 \HCode{</div>}\IgnoreIndent
434 }-{}-%
435 }
436
437 \ConfigureQuestionEnv{problem}{Problem}
438 \ConfigureQuestionEnv{exercise}{Exercise}
439 \ConfigureQuestionEnv{question}{Question}
440 \ConfigureQuestionEnv{exploration}{Exploration}
441
442 \ifdefined\xmNotHintAsExpandable
443   \ConfigureQuestionEnv{hint}{hint} % 2024: hint is no longer a 'question-environment'.
444 \fi
445 </htXimera>

```

2.4.6 Hints

hint Hint environments can be embedded inside problems.

```
446 <*classXimera>
```

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

```

447 \newcounter{hintLevel}
448 \setcounter{hintLevel}{0}

```

Create an empty shell to renew

```
449 \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.

```

450 \renewenvironment{hint}
451 {
452   \ifhandout
453     \setbox0\vbox\bgroup
454   \else
455     \begin{trivlist}\item[\hskip \labelsep\small\slshape\bfseries \GetTranslation{Hint}:\hspace{2em}]
456     \small\slshape
457   \fi
458   \stepcounter{hintLevel}
459 }
460 {
461   \ifhandout
462     \egroup\ignorespacesafterend
463   \else
464     \end{trivlist}
465   \fi
466   \addtocounter{hintLevel}{-1}
467 }
468
469 \ifhints
470 \renewenvironment{hint}{
471 \begin{trivlist}\item[\hskip \labelsep\small\slshape\bfseries \GetTranslation{Hint}:\hspace{2em}]
472 \small\slshape
473 }
474 {
475 \end{trivlist}
476 }
477 \fi
478
479 </classXimera>

```

2.4.7 Solution

solution The solution to a problem.

```
480 <*classXimera>
```

```

481 %% solution environment
482 \ifhandout % what follows is handout behavior
483 \newenvironment{solution}%
484     {%
485     \setbox0\vbox\bgroup
486     }
487     {%
488     \egroup
489     }
490 \else
491 \newenvironment{solution}%
492     {%
493     \begin{trivlist}
494     \item[\hskip \labelsep\bfseries \GetTranslation{Solution}:\hspace{2ex}]
495     }
496     % %% line at the bottom}
497     {
498     \end{trivlist}
499     % (202410: no longer \par\addvspace{.5ex}\nobreak\noindent\hung
500     }
501 \fi
502
503
504
505 \</classXimera>

```

2.4.8 Code listing environments

code A code answer environment You cannot use Environ with the fancyvrb/listings package if you want nested environments.

```

506 \<classXimera>
507 \DefineVerbatimEnvironment{code}{Verbatim}{numbers=left,frame=lines,label=Code,labelposition=
508 \</classXimera>

```

python A python answer environment You cannot use Environ with the fancyvrb/listings package if you want nested environments

```

509 \<classXimera>
510 \DefineVerbatimEnvironment{python}{Verbatim}{numbers=left,frame=lines,label=Python,labelpositi
511 \</classXimera>

```

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

```

512 \<classXimera>
513 \DefineVerbatimEnvironment{javascriptCode}{Verbatim}{numbers=left,frame=lines,label=JavaScrip
514 \</classXimera>
515 \<cfgXimera>
516 \renewenvironment{javascriptCode}{\NoFonts}{\EndNoFonts}
517 \ScriptEnv{javascriptCode}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<d
518 \</cfgXimera>

```

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

```

519 %%%<cfgXimera>
520 %%\ConfigureEnv{verbatim}{\ifvmode\IgnorePar\fi\EndP\HCode{<pre style="white-space: pre; back
521 %%\ConfigureEnv{lstlisting}{\ifvmode\IgnorePar\fi\EndP\HCode{<pre>}}{\ifvmode\IgnorePar\fi\En
522 %%%</cfgXimera>
523 %%%

```

2.4.9 Dialogues

dialogue A dialogue between people.

```

524 \<classXimera>
525 \newenvironment{dialogue}{%
526     \renewcommand\descriptionlabel[1]{\hspace{\labelsep}\textbf{##1:}}
527     \begin{description}%

```

```

528 }{%
529   \end{description}%
530 }
531 \end{classXimera}

```

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

```

532 (*htXimera)
533 \renewenvironment{dialogue}{\begin{description}}{\end{description}}
534
535 \ConfigureList{dialogue}%
536   {\EndP\HCode{<dl \a:LRdir class="dialogue">}}%
537   \PushMacro\end:itm
538 \global\let\end:itm=\empty
539   {\PopMacro\end:itm \global\let\end:itm \end:itm}
540 \EndP\HCode{</dd></dl>}\ShowPar}
541   {\end:itm \global\def\end:itm{\EndP\Tg</dd>}\HCode{<dt
542     class="actor">}\bgroup \bf}
543   {\egroup\EndP\HCode{</dt><dd\Hnewline class="speech">}}
544 \end{htXimera}

```

2.4.10 Instructor notes

```

545 (*classXimera)
546
547 %% instructor intro/instructor notes
548 %%
549 \ifhandout % what follows is handout behavior
550 \ifinstructornotes
551 \newenvironment{instructorIntro}%
552   {%
553   \begin{trivlist}
554   \item[\hspace{1.5cm} \labelsep\bfseries \GetTranslation{Instructor Introduction}:\hspace{2ex}]
555   }
556   % %% line at the bottom}
557   {
558   \end{trivlist}
559   \par\addvspace{.5ex}\nobreak\noindent\hung
560   }
561 \else
562 \newenvironment{instructorIntro}%
563   {%
564   \setbox0\vbox\bgroup
565   }
566   {%If this mysteriously starts breaking
567   % remove \ignorespacesafterend
568   \egroup\ignorespacesafterend
569   }
570   \fi
571 \else% for handout, so what follows is default
572 \ifinstructornotes
573 \newenvironment{instructorIntro}%
574   {%
575   \setbox0\vbox\bgroup
576   }
577 {%
578   \egroup
579 }
580   \else
581   \newenvironment{instructorIntro}%
582 {%
583   \begin{trivlist}
584   \item[\hspace{1.5cm} \labelsep\bfseries \GetTranslation{Instructor Introduction}:\hspace{2ex}]
585   }
586   % %% line at the bottom}

```

```

587 {
588   \end{trivlist}
589   \par\addvspace{.5ex}\nobreak\noindent\hung
590 }
591           \fi
592 \fi
593
594
595
596
597 %% instructorNotes environment
598 \ifhandout % what follows is handout behavior
599 \ifinstructornotes
600 \newenvironment{instructorNotes}%
601   {%
602   \begin{trivlist}
603   \item[\hskip \labelsep\bfseries \GetTranslation{Instructor Notes}:\hspace{2ex}]
604     }
605     % %% line at the bottom}
606     {
607 \end{trivlist}
608 \par\addvspace{.5ex}\nobreak\noindent\hung
609   }
610   \else
611 \newenvironment{instructorNotes}%
612   {%
613     \setbox0\vbox\bgroup
614   }
615   {%
616     \egroup
617   }
618   \fi
619 \else% for handout, so what follows is default
620 \ifinstructornotes
621 \newenvironment{instructorNotes}%
622   {%
623     \setbox0\vbox\bgroup
624   }
625   {%
626     \egroup
627   }
628   \else
629     \newenvironment{instructorNotes}%
630       {%
631         \begin{trivlist}
632         \item[\hskip \labelsep\bfseries \GetTranslation{Instructor Notes}:\hspace{2ex}]
633           }
634           % %% line at the bottom}
635           {
636         \end{trivlist}
637         \par\addvspace{.5ex}\nobreak\noindent\hung
638       }
639       \fi
640       \fi
641
642 \end{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.

```

643 %\iftikzexport\else\RequirePackage[framemethod=TikZ]{mdframed}\fi

```

`foldable` Does it fold?

```

644 <*classXimera>
645
646 \colorlet{textColor}{black} % since textColor is referenced below
647 \colorlet{background}{white} % since background is referenced below
648
649 % The core environments. Find results in 4ht file.
650 %% pretty-foldable
651 %\iftikzexport
652 \newenvironment{foldable}{%
653 }{%
654 }
655 %\else
656 %\renewmdenv[
657 % font=\upshape,
658 % outerlinewidth=3,
659 % topline=false,
660 % bottomline=false,
661 % leftline=true,
662 % rightline=false,
663 % leftmargin=0,
664 % innertopmargin=0pt,
665 % innerbottommargin=0pt,
666 % skipbelow=\baselineskip,
667 % linecolor=textColor!20!white,
668 % fontcolor=textColor,
669 % backgroundcolor=background
670 %]{foldable}%
671 %\fi
672
673 %% pretty-expandable
674 %\iftikzexport
675 %% Overwritten in .4ht, but probably also in accordion!
676 \ifdefined\xmNotExpandableAsAccordion
677 \newenvironment{expandable}{}{}
678 \else
679 \newenvironment{expandable}[2]{}{}
680 \fi
681 %\else
682 %\newmdenv[
683 % font=\upshape,
684 % outerlinewidth=3,
685 % topline=false,
686 % bottomline=false,
687 % leftline=true,
688 % rightline=false,
689 % leftmargin=0,
690 % innertopmargin=0pt,
691 % innerbottommargin=0pt,
692 % skipbelow=\baselineskip,
693 % linecolor=black,
694 %]{expandable}%
695 %\fi
696
697 \newcommand{\unfoldable}[1]{#1}
698
699 </classXimera>

```

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

```

700 <*htXimera>
701 \renewenvironment{foldable}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<
702
703 \ifdefined\xmNotExpandableAsAccordion
704 \renewenvironment{expandable}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{
705 \fi

```

```

706
707 \renewcommand{\unfoldable}[1]{\HCode{<span class="unfoldable">}#1\HCode{</span>}}
708 \</htXimera>

```

2.4.12 Leashes

leash Put content inside a scrollable box.

```

709 \<classXimera>
710
711 \newenvironment{leash}[1]{%
712 }{%
713 }
714
715
716 \</classXimera>

717 \<htXimera>
718 \renewenvironment{leash}[1]{\ifvmode \IgnorePar\fi \EndP\HCode{<div style="overflow: auto; height: 100px; border: 1px solid black; padding: 5px;">}#1\HCode{</div>}}
719 \</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.

```

720 \<classXimera>
721 \newcommand{\license}{\excludecomment}
722 \</classXimera>

```

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

```

723 \<classXimera>
724 \newcommand{\acknowledgement}{\excludecomment}
725 \</classXimera>

```

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

```

726 \<classXimera>
727 \renewcommand{\tag}{\excludecomment}
728 \</classXimera>

```

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

```

729 \<htXourse>
730 % Mark this as a xourse file
731 \Configure{@HEAD}{\HCode{<meta name="description" content="xourse" />\Hnewline}}
732 \</htXourse>

```

2.5.2 Abstract

abstract Every activity should include a short abstract.

```

733 \<classXimera>
734 \let\abstract\relax
735 \let\endabstract\relax
736 % Use of environ package, may want to find a better way.
737 % see the messing around with \theabstract in title.dtx ... Is this really needed/wanted?
738 \NewEnviron{abstract}{\protected@xdef\theabstract{\BODY}}
739 \</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.

```

740 \<cfgXimera>
741 \ifvmode\IgnorePar\fi\EndP

```

```

742 \ConfigureEnv{abstract}{\ifvmode\IgnorePar\fi\EndP\HCode{\Hnewline<div class="abstract">}\par}
743 \</cfgXimera>
744 \*htXimera>
745 \RenewEnviron{abstract}{\BODY}
746 \*htXimera>

```

2.5.3 Titles and authors

2.5.4 Authors

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

```

747 \*classXimera>
748 \let\@emptyauthor\@author
749 \def\@authorfootnote{\gdef\@thefnmark{}\@footnotetext}
750 \def\author#1{\gdef\@author{#1}}
751 \def\@author{\@latex@warning@no@line{No \noexpand\author given}}
752 \</classXimera>

```

Include author name in meta tags

```

753 \*htXimera>
754 \Configure{@HEAD}{\HCode{<meta name="author" content="\@author\HCode{" />\Hnewline}}
755 \</htXimera>

```

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

```

756 \htXimera | classXimera>\def\and{and }

```

2.5.5 Title

`\title` Activities have titles.

```

757 \*classXimera>
758 \let\title\relax
759 \newcommand{\title}[1][\protected@xdef\@prettitle{#1}]{\protected@xdef\@title{#1}}
760
761 \title{}
762
763 \newcounter{titlenumber}
764 \renewcommand{\thetitlenumber}{\arabic{titlenumber}}
765 %\renewcommand{\thesection}{\arabic{titlenumber}} %% Makes section numbers work
766 \setcounter{titlenumber}{0}
767
768 \newpagestyle{main}{
769 \sethead[\textsl{\ifnumbers\thetitlenumber\hspace{1em}\fi\@title}][\ifnumbers\thetitlenumber\hspace{1em}\fi\@title]} % even
770 {}{\textsl{\ifnumbers\thetitlenumber\hspace{1em}\fi\@title}} % odd
771 \setfoot[\thepage][\ifnumbers\thetitlenumber\hspace{1em}\fi\@title]} % even
772 {}{\thepage} % odd
773 }
774 \pagestyle{main}

```

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

```

775 \renewcommand\maketitle{%
776 \addtocounter{titlenumber}{1}%
777 {\flushleft\large\bfseries \@prettitle\par\vspace{-1em}}
778 {\flushleft\LARGE\bfseries {\ifnumbers\thetitlenumber\fi}{\ifnumbers\hspace{1em}\else\hspace{1em}}\par}
779 \phantomsection%
780 \ifnumbers\addcontentsline{toc}{section}{\thetitlenumber~\@title}\else\addcontentsline{toc}{section}{\thetitlenumber~\@title}
781 \vskip .6em\noindent\textit{\theabstract}\setcounter{problem}{0}\setcounter{section}{0}\setcounter{subsection}{0}
782 %\ifnooutcomes\else\let\thefootnote\relax\footnote{Learning outcomes: \theoutcomes}\fi% Dependent on \ifnooutcomes
783 \ifnoauthor\else\@authorfootnote{Author(s):~\@author}\fi
784 \aftergroup\@afterindentfalse
785 \aftergroup\@afterheading}
786
787 \ifnumbers

```

```

788 \setcounter{secnumdepth}{2}
789 \renewcommand{\thesection}{\arabic{titlenumber}.\arabic{section}}
790 \renewcommand{\thesubsection}{\arabic{titlenumber}.\arabic{section}.\arabic{subsection}}
791 \else
792 \setcounter{secnumdepth}{-2}
793 \fi
794
795 \def\activitystyle{}
796 \newcounter{sectiontitlenumber}
797 \setcounter{secnumdepth}{2}
798 \setcounter{tocdepth}{2}
799 \newcommand\chapterstyle{%
800   \def\activitystyle{activity-chapter}
801   \def\maketitle{%
802     \addtocounter{titlenumber}{1}%
803     {\flushleft\small\sffamily\bfseries\@pretitle\par\vspace{-1.5em}}%
804     {\flushleft\LARGE\sffamily\bfseries\thetitle\hspace{1em}\@title \par}
805     {\vskip .6em\noindent\textit{theabstract}\setcounter{problem}{0}\setcount
806     \par\vspace{2em}
807     \phantomsection\addcontentsline{toc}{section}{\textbf{\thetitle\hsp
808   }}
809
810
811 \newcommand\sectionstyle{%
812   \def\activitystyle{activity-section}
813   \def\maketitle{%
814     \addtocounter{section}{1}
815     \setcounter{sectiontitlenumber}{\value{section}}
816     {\flushleft\small\sffamily\bfseries\@pretitle\par\vspace{-1.5em}}%
817     {\flushleft\Large\sffamily\bfseries\thetitle\hspace{1em}\@title
818     {\vskip .6em\noindent\textit{theabstract}\setcounter{subsection}{0}}%
819     \par\vspace{2em}
820     \phantomsection\addcontentsline{toc}{section}{\thetitle.\thesectiontitlenumber\hsp
821   \renewcommand\section{\@startsection{subsection}{2}{\z@}%
822     {-3.25ex\@plus -1ex \@minus -.2ex}%
823     {1.5ex \@plus .2ex}%
824     {\normalfont\large\bfseries}}
825
826   \renewcommand\subsection{\@startsection{subsubsection}{3}{\z@}%
827     {-3.25ex\@plus -1ex \@minus -.2ex}%
828     {1.5ex \@plus .2ex}%
829     {\normalfont\normalsize\bfseries}}
830
831 }}
832
833
834 \iftikzexport% allows xake to handle \chapterstyle and \sectionstyle
835 \renewcommand\chapterstyle{\def\activitystyle{chapter}}
836 \renewcommand\sectionstyle{\def\activitystyle{section}}
837 \else
838 \fi
839
840 \end{classXimera}

```

Eliminate some formatting that we'll handle later with CSS

```

841 \end{htXimera}
842 \renewcommand{\maketitle}{}
843 \end{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      The prompt part for mathmode
844 <*classXimera>
845 \ifxake
846     \newenvironment{prompt}{}{}
847 \else
848 \ifhandout
849 \NewEnviron{prompt}{}
850     % Breaks when put in mathmode ?
851     % \newenvironment{prompt}{\suppress}{\endsuppress}
852 \else
853 \newenvironment{prompt}{\bgroup\color{gray!50!black}}{\egroup}
854 \fi
855 \fi

onlyHtml    Only display online
onlyPdf     Only display in the PDF
onlineOnly  Only display online (deprecated: use onlyHtml instead)
856 \ifdefined\HCode
857 \newenvironment{onlyPdf}{\setbox0\vbox\bgroup}{\egroup}
858 \newenvironment{onlyHtml}{\bgroup}{\egroup}
859 \newenvironment{onlineOnly}{\bgroup}{\egroup}
860 \else
861 \newenvironment{onlyPdf}{\bgroup}{\egroup}
862 \ifdefined\xmPrintHtmlOnlyAlsoInPdf
863 \newenvironment{onlyHtml}{\bgroup\color{red!50!black}}{\egroup}
864 \newenvironment{onlineOnly}{\bgroup\color{red!50!black}}{\egroup}
865 \else
866 \newenvironment{onlyHtml}{\setbox0\vbox\bgroup}{\egroup}
867 \newenvironment{onlineOnly}{\setbox0\vbox\bgroup}{\egroup}
868 \fi
869 \fi
870

\htmlOnly  Only display online
\pdfOnly   Only display in the PDF
871
872 \ifdefined\HCode
873 \newcommand{\pdfOnly}[1]{}
874 \newcommand{\htmlOnly}[1]{#1}
875 \else
876 \ifdefined\xmPrintHtmlOnlyAlsoInPdf
877 \newcommand{\pdfOnly}[1]{#1}
878 \newcommand{\htmlOnly}[1]{\bgroup\color{red!50!black}#1\egroup}
879 \else
880 \newcommand{\pdfOnly}[1]{#1}
881 \newcommand{\htmlOnly}[1]{}
882 \fi
883 \fi
884

\ifonline  Only execute online (ie in HTML version)
```

```

\ifonlineTF Different output online vs PDF
885 % An alternative for \pdfOnly/\begin{htmlOnly} :
886 % Usage: Hello \ifonlineTF{online reader}{PDF reader}
887 \providecommand{\ifonlineTF}[2]{\htmlOnly{#1}\pdfOnly{#2}}
888 \newif{\ifonline}
889 \ifdefined\HCode
890 \onlinetrue
891 \else
892 \onlinefalse
893 \fi
894 \end{classXimera}

```

2.5.7 Learning Outcomes

```

895 \begin{classXimera}
896 \newcommand{\preOutcomeLine}{\item }
897 \newcommand{\postOutcomeLine}{}
898 \newcommand{\preOutcomeBlock}{After completing this content, students should be able to: \begin{itemize}
899 \newcommand{\postOutcomeBlock}{\end{itemize} So go forth and learn!}
900
901 \newcommand{\outcomeHeader}{Goals for this Section}
902 \htmlOnly{
903   \newcommand{\outcomeBlock}{\ifvmode\IgnorePar\fi\EndP\HCode{<div class="outcomeHead">} \outcomeHeader}
904 }
905
906
907 \newwrite\outcomefile
908 \immediate\openout\outcomefile=\jobname.oc
909 \newcommand{\outcome}[1]{%
910   \immediate\write\outcomefile{\expandafter\unexpanded\expandafter{\preOutcomeLine #1} \preOutcomeBlock}
911 }
912
913 \newcommand{\displayOutcomes}[1][1]{%
914   \immediate\closeout\outcomefile
915   \IfFileExists{\currfiledir\currfilebase.oc}{
916     \htmlOnly{\outcomeBlock}
917     \expandafter\preOutcomeBlock
918     \input{\currfiledir\currfilebase.oc}
919     \postOutcomeBlock
920     \htmlOnly{\ifvmode\IgnorePar\fi\EndP\HCode{</div>}}
921   }
922   {
923     \IfFileExists{\currfilebase.oc}{
924       \htmlOnly{\outcomeBlock}
925       \expandafter\preOutcomeBlock
926       \input{\currfilebase.oc}
927       \postOutcomeBlock
928       \htmlOnly{\ifvmode\IgnorePar\fi\EndP\HCode{</div>}}
929     }
930     {
931       No outcome file found.
932     }
933   }
934 }
935 %
936 \end{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.

```

937 \begin{cfgXimera}
938 \renewcommand{\outcome}[1]{
939   \Configure{@HEAD}{\HCode{<meta name="learning-outcome" content="#1"/>\Hnewline}}
940 }

```

```

941 % Sometimes there are no outcomes at all
942 \IfFileExists{\jobname.oc}{\input{\jobname.oc}}{}
943
944 \renewcommand{\outcome}[1]{%
945   \HCode{<span class="learning-outcome">#1</span>}
946 }
947 \</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.

```

948 \<htXimera>
949 \let\oldlabel\label
950 \renewcommand{\label}[1]{\oldlabel{#1}\HCode{<a class="ximera-label" id="#1"></a>}}
951 \</htXimera>

```

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

```

952 \<htXimera>
953 \renewcommand{\ref}[1]{\HCode{<a class="reference" href="#1">#1</a>}}
954 \</htXimera>

```

2.6 Images

2.6.1 Images

`image` 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`!

`\xmDefaultGraphicsPath`

```

955 \<classXimera>
956 % Provide a default graphicspath
957 % (somewhat tricky: an activity can be included in a xourse in a wildly different path !)
958 % Suggested convention: put all images in i /pictures folder in the root of your project
959 \providecommand{\xmDefaultGraphicsPath}{/xmPictures}
960 \graphicspath{ %% When looking for images,
961 {./} %% look here first,
962 {.\xmDefaultGraphicsPath/} %% then look for a pictures folder,
963 {..\xmDefaultGraphicsPath/} %% then look for a pictures folder,
964 {../../xmDefaultGraphicsPath/} %% then look for a pictures folder,
965 {../../..\xmDefaultGraphicsPath/} %% then look for a pictures folder,
966 }
967 %\newenvironment{image}[1][\begin{center}]{\end{center}}
968 \NewEnviron{image}[1][3in]{%
969   \begin{center}\resizebox{#1}{!}{\BODY}\end{center}% resize and center
970 }
971 \</classXimera>

```

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

```

972 \<classXimera>
973 \newcommand{\alt}[1]{}
974 \</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.

```

975 \<htXimera>
976 \newcounter{imagealt}
977 \setcounter{imagealt}{0}
978 \renewenvironment{image}[1][\stepcounter{imagealt}%
979   \ifvmode \IgnorePar\fi \EndP%
980   \HCode{<div class="image-environment" role="img" aria-labelledby="image-alt-\arabic{imagealt}>}
981 }{\HCode{</div>}}
982 \renewcommand{\alt}[1]{\HCode{<div style="display: none;" id="image-alt-\arabic{imagealt}">}}
983 \</htXimera>

```

```

984 <*cfgXimera>
985 %% Although we accept many formats, SVG is preferred on the web.
986 %% Since we have a different mechanism for producing |alt| text, we
987 %% want to ignore tex4ht's own method fo producing alt text.
988 %% 2024: is now in TeX4ht ...
989 % \DeclareGraphicsExtensions{.jpg,.png,.gif,.svg}
990 % \Configure{graphics*}
991 % {svg}-{
992 %   {\Configure{Needs}{File: \Gin@base.svg}\Needs{}}
993 %   \Picture[]{\csname Gin@base\endcsname.svg \csname a:Gin-dim\endcsname}%
994 % }
995 </cfgXimera>

```

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

```

996 <*cfgXimera>
997 \ifcsname ifstandalone\endcsname
998   \ifstandalone
999     \renewcommand\includegraphics[2][]{ }
1000   \fi
1001 </cfgXimera>

```

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

```

1002 <*htXimera>
1003 \providecommand{\pgfsyspdfmark}[3]{}
1004 </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.

```

1005 <*classXimera>
1006 % everything skipped, assume TeX4ht does the jbb now
1007 \ifdefined\reallyneverever
1008
1009 \ifdefined\HCode
1010   \tikzexporttrue
1011 \fi
1012
1013 \iftikzexport
1014   \usetikzlibrary{external}
1015
1016   \ifdefined\HCode
1017     % in htlatex, just include the svg files
1018     \def\pgfsys@imagesuffixlist{.svg}
1019
1020     \tikzexternalize[prefix=./,mode=graphics if exists]
1021   \else
1022     % in pdflatex, actually generate the svg files
1023     \tikzset{
1024       /tikz/external/system call={
1025         pdflatex \tikzexternalcheckshellescape
1026         -halt-on-error -interaction=batchmode
1027         -jobname "\image" "\PassOptionsToClass{tikzexport}{ximera}\texsource";
1028         mutool draw -F svg \image.pdf > \image.svg ;      % mutool adds "1" to filename ???
1029         mutool draw -o \image.svg \image.pdf ;
1030         mutool draw -r 150 -c rgbalpha -o \image.png \image.pdf ;
1031         ebb -x \image.png
1032       }
1033     }
1034     \tikzexternalize[optimize=false,prefix=./]

```

```

1035 \fi
1036
1037 \fi
1038 \fi
1039 \endclassXimera

```

2.6.3 XKCD

`\xkcd` Reference an XKCD cartoon.

```

1040 \classXimera
1041 \newcommand{\xkcd}[1]{#1}
1042 \endclassXimera

```

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

```

1043 \htXimera
1044 \renewcommand{\xkcd}[1]{\ifvmode \IgnorePar\fi \EndP\HCode{

```

2.8.3 Geogebra

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

```

1083 <classXimera>
1084 %Geogebra link
1085 \newcommand{\geogebra}[3]{GeoGebra link: \url{https://www.geogebra.org/m/#1}}
1086 </classXimera>

Define keys for answer geogebra key=value pairs.

1087 <htXimera>
1088 \define@key{geogebra}{rc}[true]{\def\geo@rc{#1}}
1089 \define@key{geogebra}{sdz}[true]{\def\geo@sdz{#1}}
1090 \define@key{geogebra}{smb}[true]{\def\geo@smb{#1}}
1091 \define@key{geogebra}{stb}[true]{\def\geo@stb{#1}}
1092 \define@key{geogebra}{stbh}[true]{\def\geo@stbh{#1}}
1093 \define@key{geogebra}{ld}[true]{\def\geo@ld{#1}}
1094 \define@key{geogebra}{sri}[true]{\def\geo@sri{#1}}
1095 %set default key values
1096 \setkeys{geogebra}{rc=false,sdz=false,smb=false,stb=false,stbh=false,ld=false,sri=false}
1097 %command definition
1098 \renewcommand{\geogebra}[4][ ]{%
1099   \setkeys{geogebra}{#1}% Set new keys
1100   \HCode{<iframe scrolling="no" src="https://www.geogebra.org/material/iframe/id/#2/width/#3
1101 </htXimera>

```

2.8.4 Desmos

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

```

1102 <classXimera>
1103 \newcommand{\desmos}[3]{Desmos link: \url{https://www.desmos.com/calculator/#1}}
1104 \newcommand{\desmosThreeD}[3]{Desmos3D link: \url{https://www.desmos.com/3d/#1}}
1105 </classXimera>

1106 <htXimera>
1107 \catcode'\%=11
1108 \renewcommand{\desmos}[3]{\HCode{<iframe src="https://www.desmos.com/calculator/#1" width="1
1109 \catcode'\%=14
1110 \renewcommand{\desmosThreeD}[3]{\HCode{<iframe src="https://www.desmos.com/3d/#1" width="#2p
1111 </htXimera>

```

2.8.5 Graphs

`\graph` An embedded graph (in math mode).

```

1112 <classXimera>
1113 \newcommand{\graph}[2][ ]{\text{Graph of $#2$}}
1114 </classXimera>

1115 <htXimera>
1116 \renewcommand{\graph}[2][ ]{\HCode{<div class="graph" data-options="#1">#2\HCode{</div>}}
1117 </htXimera>

```

2.8.6 Video

`\youtube` Youtube command. Requires id.

```

1118 <*classXimera>
1119 \newcommand{\youtube}[1]{YouTube link: \url{https://www.youtube.com/watch?v=#1}}
1120 </classXimera>

1121 <*htXimera>
1122 %% \renewcommand{\youtube}[1]{\ifvmode \IgnorePar\fi \EndP\HCode{<div class="video youtube-p
1123 % Fixes no-youtube-when-no-cookies-accepted. Class xmyoutube allows for css customization.
1124 \renewcommand{\youtube}[1]{\ifvmode \IgnorePar\fi \EndP\HCode{<iframe class="xmyoutube" src=
1125
1126 </htXimera>

```

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

```

1127 <*htXourse>
1128 \renewcommand{\youtube}[1]{%
1129 \ifvmode \IgnorePar\fi \EndP\HCode{<a class="youtube" href="https://www.youtube.com/watch?v=
1130 }
1131 </htXourse>

```

2.8.7 JavaScript

`javascript` Code inside a javascript environment is printed on paper, but executed on the web.

```

1132 <*classXimera>
1133 \DefineVerbatimEnvironment{javascript}{Verbatim}{numbers=left,frame=lines,label=JavaScript,l
1134 </classXimera>

1135 <*htXimera>
1136 % for programming javascript
1137 \renewenvironment{javascript}{\NoFonts}{\EndNoFonts}
1138 \ScriptEnv{javascript}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div c
1139 </htXimera>

```

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

```

1140 <*classXimera>
1141 \def\js#1{\mbox{\texttt{\detokenize{#1}}}}
1142 </classXimera>

1143 <*htXimera>
1144 \def\js#1{\stepcounter{identification}\HCode{<span class="inline-javascript" id="javascript\
1145 </htXimera>

```

2.9 SageMath support

Load Sage \TeX if it exists.

```

1146 <*classXimera>
1147 \IfFileExists{sagetex.sty}{\RequirePackage{sagetex}}{}
1148 </classXimera>

```

`sageCell` Create an interactive SageMath widget.

```

1149 <*classXimera>
1150 \DefineVerbatimEnvironment{sageCell}{Verbatim}{numbers=left,frame=lines,label=SAGE,labelposi
1151 </classXimera>

1152 <*htXimera>
1153 \renewenvironment{sageCell}{\NoFonts}{\EndNoFonts}
1154 \ScriptEnv{sageCell}{\ifvmode \IgnorePar\fi \EndP\HCode{<div class="sage"><script type="text
1155 </htXimera>

```

`sageOutput` Execute SageMath code and output the result.

```

1156 <*classXimera>
1157 \DefineVerbatimEnvironment{sageOutput}{Verbatim}{numbers=left,frame=lines,label=SAGE-Output,
1158 </classXimera>

```

```

1159 <*htXimera>
1160 \renewenvironment{sageOutput}{\NoFonts}{\EndNoFonts}
1161 \ScriptEnv{sageOutput}{\ifvmode \IgnorePar\fi \EndP\HCode{<div class="sageOutput"><script ty
1162 </htXimera>

```

sageSilent Execute SageMath code without outputting the result.

```

1163 <*htXimera>
1164 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1165 \ifdefined\sagesilent
1166 \renewenvironment{sagesilent}{\NoFonts}{\EndNoFonts}
1167 \fi
1168 \ScriptEnv{sagesilent}{\ifvmode \IgnorePar\fi \EndP\HCode{<script type="text/sagemath">\Htm
1169 </htXimera>

```

2.10 Answerables

2.10.1 Answers

\answer A math answer

```

1170 <*classXimera>
1171
1172 \ifdefined\HCode
1173 \newcommand{\recordvariable}[1]{}
1174 \else
1175 \newwrite\idfile
1176 \immediate\openout\idfile=jobname.ids
1177 \newcommand{\recordvariable}[1]{\ifthenelse{\equal{#1}{}}{\}\{\immediate\write\idfile{var #1};}
1178 \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

```
1179 \define@key{answer}{given}[true]{\def\ans@given{#1}}
```

Used for setting numeric answer tolerance for online student input.

```
1180 \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.

```
1181 \define@key{answer}{validator}{}

```

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

```
1182 \define@key{answer}{id}{\def\ans@id{#1}}
```

Used to set anticipated input format; eg “string”.

```
1183 \define@key{answer}{format}{}

```

Used to hide the answer input box on the web.

```
1184 \define@key{answer}{onlinenoinput}[false]{}

```

Used to add a ‘show answer’ button to the answer blank.

```
1185 \define@key{answer}{onlineshowanswerbutton}[false]{}

```

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

```
1186 \setkeys{answer}{id=,given=false,onlinenoinput=false,onlineshowanswerbutton=false}
```

Basic code for `\answer`.

```

1187
1188 % Options for handout
1189 \newcommand{\answerFormatLength}{2cm}
1190
1191 \newcommand{\answerFormatDots}[1]{\ldots\ldots}
1192 \newcommand{\answerFormatLine}[1]{\protect\rule{\answerFormatLength}{0.4pt}}
1193 \newcommand{\answerFormatFlexibleLine}[1]{\protect\rule{\widthof{#1$}*2}{0.4pt}}
1194 \newcommand{\answerFormatFlexibleBox}[1]{\fbox{\scalebox{2}{\phantom{#1$}}}}
1195
1196 % options for default (i.e with answers filled in)

```

```

1197 \newcommand{\answerFormatPlain}[1]{\ensuremath{#1}}
1198 \newcommand{\answerFormatBlue}[1]{\color{blue}\ensuremath{#1}}
1199 \newcommand{\answerFormatBoxed}[1]{\fbox{\ensuremath{#1}}}
1200 \newcommand{\answerFormatBoxedGiven}[1]{\underset{\scriptstyle\mathrm{given}}{\fbox{\ensuremath{#1}}}}
1201
1202 % defaults for handout and default mode, and for \answer[given]
1203 \let\handoutAnswerFormat\answerFormatDots
1204 \let\defaultAnswerFormat\answerFormatBlue
1205 \let\givenAnswerFormat\answerFormatBoxedGiven
1206
1207 \newcommand{\answer}[2][{}]{%
1208 \ifmmode%
1209 \setkeys{answer}{#1}%
1210 \recordvariable{\ans@id}
1211 \ifthenelse{\boolean{\ans@given}}{
1212 {% Start then statement
1213 \ifhandout
1214 #2
1215 \else
1216 \givenAnswerFormat{#2} %% in case the argument helps formatting
1217 \fi
1218 }% End then statement
1219 {% Start else statement
1220 \ifhandout
1221 \handoutAnswerFormat{#2} %% in case the argument helps formatting
1222 \else% show answer in box outside handout mode
1223 \defaultAnswerFormat{#2} %% in case the argument helps formatting
1224 \fi
1225 }% End else statement
1226 \else%
1227 \GenericError{\space\space\space\space}% Throw an error based on... something? -- Jason
1228 {Attempt to use \@backslashchar answer outside of math mode}
1229 {See https://github.com/ximeraProject/ximeraLatex for explanation.}
1230 {Need to use either inline or display math.}%
1231 \fi
1232 }
1233 \endclassXimera

```

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

```

1234 \beginXimera
1235 \renewcommand{\answer}[2][false]{\HCode{<span class="answer_respondable">}#2\HCode{</span>}}
1236
1237 \def\validator[#1]{\stepcounter{identification}\HCode{<div class="validator" id="validator">#1}}
1238 \def\endvalidator{\HCode{</div>}}
1239
1240 \endXimera

```

2.10.2 Multiple choice and the like

`multipleChoice` Multiple choice

```

1241 \beginXimera
1242 % Jim: Originally this was \renewcommand{\theenumi}{$(\mathrm{\alph{enumi}})$}
1243 % but that breaks tex4ht because mathmode can only be processed by mathjax.
1244 % so now I made this just italicized.

```

2.10.3 Options

```

1245 \define@key{choice}{value}[]{\def\choice@value{#1}}

```

This flags the answer as the correct answer

```

1246 \define@boolkey{choice}{correct}[true]{\def\choice@correct{#1}}

```

Use an ID to refer to the choice.

```

1247 \define@key{multipleChoice}{id}{\def\mc@id{#1}}

```

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

```
1248 \define@key{otherchoice}{value}[]{\def\otherchoice@value{#1}}
1249 \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".

```
1250 \setkeys{choice}{correct=false,value=}
```

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

```
1251 \setkeys{multipleChoice}{id=}
```

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

```
1252 \setkeys{otherchoice}{correct=false,value=}
```

```
1253 \endclassXimera
```

2.10.4 Choices

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

```
1254 \classXimera
1255 \newcommand{\choice}[2][]{%
1256 \setkeys{choice}{#1}%
1257 \item{#2}
1258 \ifthenelse{\boolean{\choice@correct}}{
1259   {% Begin then result
1260     \ifhandout% if it's a handout do nothing.
1261     \else% otherwise place a checkmark when you select the "correct choice"... maybe? -- Jason
1262       \, \checkmark \, \setkeys{choice}{correct=false}
1263     \fi
1264   }% End then result
1265   {% Begin/End else result.
1266 }
1267
1268 %Define an expandable version of choice Not really meant to be used outside this package (use
1269 % Is there a reason we can't just always use this as default? -- Jason
1270 \newcommand{\choiceEXP}[2][]{%
1271 \expandafter\setkeys\expandafter{choice}{#1}%
1272 \item{#2}
1273 \ifthenelse{\boolean{\choice@correct}}{
1274 {% Begin then result
1275 \ifhandout
1276 \else
1277 \, \checkmark \, \setkeys{choice}{correct=false}
1278 \fi
1279 }% End then result
1280 }{% Begin/End else result.
1281 } %% note all the {} are needed in case the choice has [] in it.
1282
1283 % \otherchoice is the \choice used in wordChoice command.
1284 \newcommand{\otherchoice}[2][]{%
1285 \ignorespaces%
1286 \setkeys{otherchoice}{#1}%
1287 \ifthenelse{\boolean{\otherchoice@correct}}{%
1288 {% Start then result
1289 #2\ignorespaces\setkeys{otherchoice}{correct=false}\ignorespaces%
1290 }% End then result
1291 }{% Start/End else result
1292 \ignorespaces%
1293 }%
1294 \newcommand{\inlinechoice}[2][]{%
1295 \setkeys{choice}{#1}%
1296 \iffirstinlinechoice
1297 (\hspace{-.25em}
1298 \firstinlinechoicefalse
1299 \else
```

```

1300 /
1301 \fi
1302 #2
1303 \ifthenelse{\boolean{\choice@correct}}{%
1304 {% Start then result
1305 \ifhandout\else\checkmark\ignorespaces\setkeys{choice}{correct=false}\ignorespaces\fi%
1306 }% End then result
1307 }% Start/End else result
1308 \hspace{-.25em}\ignorespaces%
1309 }
1310
1311 \end{classXimera}

```

On the HTML side, `\choice` emits `s`.

```

1312 \begin{htXimera}
1313 \newcounter{choiceId}
1314 \renewcommand{\choice}[2][]{%
1315 \setkeys{choice}{correct=false}%
1316 \setkeys{choice}{#1}%
1317 \stepcounter{choiceId}\IgnorePar%
1318 \HCode{<span class="choice }%
1319 \ifthenelse{\boolean{\choice@correct}}{\HCode{correct}}{\HCode{}}
1320 \HCode{" }
1321 \ifthenelse{\equal{\choice@value}{}}{\HCode{data-value="\choice@value" }}
1322 \HCode{id="choice\arabic{choiceId}">}}%
1323 #2\HCode{</span>}}
1324 \let\inlinechoice\choice
1325 \end{htXimera}

```

2.10.5 Environment(s)

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

```

1326 \begin{classXimera}
1327 \newenvironment{multipleChoice}[1]{}
1328 {% Environment Start Code
1329 \setkeys{multipleChoice}{#1}%
1330 \recordvariable{mc{id}}%
1331 \begin{trivlist}
1332 \item[\hspace{\labelsep}\small\bfseries \GetTranslation{Multiple Choice:}]\hfil
1333 \begin{enumerate}
1334 }% Note this means that \item has to be the first line after \begin{multipleChoice}.
1335 {% Environment End Code
1336 \end{enumerate}
1337 \end{trivlist}
1338 }
1339
1340 %multipleChoice@ is for internal use only! (used in wordChoice)
1341 %this is simply a wrapper for the sole showing (other)choice.
1342 \newenvironment{multipleChoice@}[1]{}{}
1343 \end{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.

```

1344 \begin{htXimera}
1345 \renewenvironment{multipleChoice}[1]{}
1346 {\setkeys{multipleChoice}{#1}%
1347 \stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div class="multiple-choice" }
1348 \ifthenelse{\equal{\mc{id}}{}}{\HCode{data-id="\mc{id" }}%
1349 \HCode{id="problem\arabic{identification}" titletext=" \GetTranslation{Multiple Choice}">}}%
1350 }{\HCode{</div>}\IgnoreIndent}
1351 \ConfigureEnv{multipleChoice}{}{}{}
1352 \end{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.

```

1353 <*classXimera>
1354 \newcommand{\wordChoice}[1]{%
1355 \let\choicetemp\choice% Assign a "choicetemp" command to duplicate choice.
1356 \ifwordchoicegiven% If wordchoice option is on, we need to juggle around some definitions.
1357 \let\choice\otherchoice%
1358 %\begin{multipleChoice@}% -unnecessary (REMOVE THIS LINE IF THE YEAR IS 2019 or Beyond)
1359 #1
1360 %\end{multipleChoice@}% -unnecessary (REMOVE THIS LINE IF THE YEAR IS 2019 or Beyond)
1361 \else% If it isn't the regular "choice" command should work.
1362 \let\choice\inlinechoice%
1363 \begin{multipleChoice@}%
1364 #1%
1365 \end{multipleChoice@}%
1366 \fi%
1367 \let\choice\choicetemp% Now that choicetmp has been manipulated to what we want, replace choi
1368 }%
1369
1370
1371 </classXimera>

```

This is actually just word choice

```

1372 <*htXimera>
1373 \renewenvironment{multipleChoice@}{\refstepcounter{problem}}{}%
1374 \ConfigureEnv{multipleChoice@}{\stepcounter{identification}\IgnorePar\HCode{<span class="word
1375 </htXimera>

```

2.12 Select all

`selectAll` A multiple-multiple choice question

```

1376 <*classXimera>
1377 \newenvironment{selectAll}[1][
1378 {\begin{trivlist}\item[\hskip \labelsep\small\bfseries \GetTranslation{Select All Correct Ans
1379 {\end{enumerate}\end{trivlist}}
1380 </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`.

```

1381 <*htXimera>
1382 \renewenvironment{selectAll}{\refstepcounter{problem}}{}%
1383 \ConfigureEnv{selectAll}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div
1384 </htXimera>

```

2.12.1 Free response

`freeResponse` A freeform input box.

```

1385 <*classXimera>
1386 \newboolean{given} %% required for freeResponse
1387 \setboolean{given}{true} %% could be replaced by a key=value pair later if needed
1388
1389 \ifhandout
1390 \newenvironment{freeResponse}[1][false]%
1391 {%
1392 \def\givenatend{\boolean{#1}}
1393 \ifthenelse{\boolean{#1}}
1394 {% Begin then result

```

```

1395 \begin{trivlist}
1396 \item
1397 }% End then result
1398 {% Begin else result
1399 \setbox0\vbox\bgroup
1400 }% End else result
1401 % {}% Don't think this is doing anything? -- Jason
1402 }
1403 {%
1404 \ifthenelse{\givenatend}
1405 {% Begin then result
1406 \end{trivlist}
1407 }% End then result
1408 {% Begin else result
1409 \egroup
1410 }% End else result
1411 % {}% Don't think this is doing anything? -- Jason
1412 }
1413 \else
1414 \newenvironment{freeResponse}[1][false]%
1415 {% Environment Beginning Code
1416 \ifthenelse{\boolean{#1}}% Could probably change this with just putting the (given) in the
1417 {% Begin then result
1418 \begin{trivlist}
1419 \item[\hskip \labelsep\bfseries \GetTranslation{Free Response (Given)}:\hspace{2ex}]
1420 }% End then result
1421 {% Begin else result
1422 \begin{trivlist}
1423 \item[\hskip \labelsep\bfseries \GetTranslation{Free Response}:\hspace{2ex}]
1424 }% End else result
1425 }
1426 {% Environment Ending Code
1427 \end{trivlist}
1428 }
1429 \fi
1430
1431 \</classXimera>
1432 \<htXimera>
1433
1434 \renewenvironment{freeResponse}{\refstepcounter{problem}}{}%
1435 \ConfigureEnv{freeResponse}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<
1436
1437 \</htXimera>

```

2.12.2 Feedback

feedback 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.

```

1438 \<classXimera>
1439 \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.

1440 \newenvironment{validator}[1][]{
1441 \def\PH@Command{#1}% Use PH@Command to hold the content and be a target for "\expandafter" to
1442 \mbox{\texttt{\detokenize\expandafter{\PH@Command}}}% Now expand PH@Command once and then det
1443 }{}

```

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

```

1444 \ifhandout%
1445 \newenvironment{feedback}
1446         {%
1447   \setbox0\vbox\bgroup
1448       }
1449         {%
1450   \egroup
1451       }

```

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.

```

1452 \else
1453 \newenvironment{feedback}[1][attempt]{
1454
1455   \edef\PH@Command{\GetTranslation{#1}}% Use PH@Command to hold the content and be a target for
1456
1457   \begin{trivlist}% Begin the trivlist to use formatting of the "Feedback" label.
1458     \item[\hskip \labelsep\small\slshape\bfseries \GetTranslation{Feedback}]% Format the "Feedback"
1459     \ifonlineTF{% If the feedback is on a pdf, we don't need to detokenize - which messes with the
1460       (\texttt{\expandafter\detokenize\expandafter{\PH@Command}})}% Keep the online version the same
1461       {(\expandafter\texttt{\PH@Command}}}% No need for detokenize in the pdf version
1462       \hspace{2ex}}\small\slshape% Insert some space before the actual feedback given.
1463   }{
1464   \end{trivlist}
1465 }
1466
1467 \fi
1468 \end{classXimera}

```

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

```

1469 \begin{classXimera}
1470 \def\feedback{\@ifnextchar[{\@feedbackcode}{\@feedbackattempt}}
1471 \def\@feedbackattempt{\@feedbackcode[attempt]}
1472 \def\@feedbackcode[#1]{\stepcounter{identification}%
1473 \ifvmode \IgnorePar\fi \EndP%
1474 \ifthenelse{\equal{#1}{attempt}}{\HCode{<div class="feedback" data-feedback="attempt" id="feedback-@feedbackcode[#1]>}}{\HCode{<div class="feedback" data-feedback="correct" id="feedback-@feedbackcode[#1]>}}
1475 \ifthenelse{\equal{#1}{correct}}{\HCode{<div class="feedback" data-feedback="correct" id="feedback-@feedbackcode[#1]>}}{\HCode{<div class="feedback" data-feedback="script" id="feedback\arabic{identification}>}}
1476 \ifthenelse{\equal{#1}{script}}{\HCode{<div class="feedback" data-feedback="script" id="feedback\arabic{identification}>}}{\HCode{<div class="feedback" data-feedback="other" id="feedback-@feedbackcode[#1]>}}
1477 \def\endfeedback{\HCode{</div>}\IgnoreIndent}
1478 \end{classXimera}

```

2.12.3 Ungraded activities

ungraded 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.

```

1479 \begin{classXimera}
1480 \newenvironment{ungraded}{}{}
1481 \end{classXimera}

```

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

```

1482 \begin{classXimera}
1483 \renewenvironment{ungraded}{%
1484   \ifvmode \IgnorePar\fi \EndP\HCode{<div class="ungraded">}\IgnoreIndent%
1485 }{
1486   \ifvmode \IgnorePar\fi \EndP\HCode{</div>}\IgnoreIndent%

```

```

1487 }
1488 </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.

```

1489 (*classXimera)
1490 %% Pre-202412: .jax file written in non-\HCode, and in a next run inserted by ximera.cfg in
1491 %% Post-202501: .mjax file written only in \HCode, and in luaxake post-processing inserted in
1492 %% ( used luaxake rather than sed ...)
1493 \newwrite\myfile
1494 \ifdefined\HCode
1495 \immediate\openout\myfile=\jobname.xmjax
1496
1497 %% From |only.dtx| we must also create |prompt| on the MathJax side.
1498 \immediate\write\myfile{\unexpanded{\newenvironment}{\prompt}{\}}
1499
1500 %% Write all newcommands to .xmjax file, that will be included in the .html via luaxake
1501 \let\oldargdef\argdef
1502 \long\def\argdef#1[#2]#3{%
1503 \immediate\write\myfile{\unexpanded{\newcommand}{\unexpanded{#1}}[\unexpanded{#2}]{\unexpanded{#3}}}%
1504 \oldargdef#1[#2]{#3}%
1505 }
1506
1507 %% Same for \DeclareMathOperator
1508 \let\oldDeclareMathOperator\DeclareMathOperator
1509 \renewcommand{\DeclareMathOperator}[2]{\oldDeclareMathOperator{#1}{#2}\immediate\write\myfile{\unexpanded{\newcommand}{\unexpanded{#1}}[\unexpanded{#2}]{\unexpanded{#3}}}%
1510
1511 \fi
1512
1513
1514 </classXimera>

```

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

```

1515 (*cfgXimera)
1516
1517 % 202501: removed sed-manipulation of .jax file; see luaxake now
1518
1519 \Configure{BVerbatimInput}{\}\}\}\}
1520
1521 \Configure{verbatiminput}{\}\}\}\}
1522
1523 % Instead of a nonbreaking space, use a standard space
1524 \makeatletter
1525 \def\FV@Space{\space}
1526 \makeatother
1527
1528 % Include the (problem-?) .ids in a text/javascript script right at the beginning of the body
1529 \Configure{BODY}{%
1530 \HCode{<body>\Hnewline}%
1531 \Tg<div class="preamble">%
1532 %% 202501: removed .jax inclusion (see luaxake)
1533
1534 %% Include the .ids file
1535 \IfFileExists{\jobname.ids}{\HCode{<script type="text/javascript">\Hnewline}%
1536 \BVerbatimInput{\jobname.ids}%
1537 \HCode{</script>\Hnewline}%
1538 }\}
1539 \Tg</div>%
1540 }\}
1541 \ifvmode\IgnorePar\fi\EndP\HCode{</body>\Hnewline}%

```

```

1542 }
1543
1544 % 202501: removed 'prevent spaces as in "\begin {align}": this is done in luaxake now
1545
1546 % This is a fix for the LAODE book, which uses matlabEquation as if it were an equation
1547 \ScriptEnv{matlabEquation}{\ifvmode \IgnorePar\fi \EndP\HCode{<script type="math/tex; mode=d
1548
1549 \</cfgXimera>

```

2.13.2 Semantic HTML

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

```

1550 \<cfgXimera>
1551 \Configure{textbf}{\ifvmode\ShowPar\fi\HCode{<strong>}}{\HCode{</strong>}}
1552 \</cfgXimera>

```

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

```

1553 \<cfgXimera>
1554 \Configure{textit}{\ifvmode\ShowPar\fi\HCode{<em>}}{\HCode{</em>}}
1555 \Configure{emph}{\ifvmode\ShowPar\fi\HCode{<em>}}{\HCode{</em>}}
1556 \</cfgXimera>

```

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

```

1557 \<cfgXimera>
1558 \Configure{texttt}{\ifvmode\ShowPar\fi\HCode{<code>}}{\HCode{</code>}}
1559 \</cfgXimera>

```

2.13.3 Enumerate fixes

Make enumerate use a letter

```

1560 \<classXimera>
1561 \renewcommand{\theenumi}{\textup{(\alph{enumi})}}
1562 \renewcommand{\labelenumi}{\theenumi}
1563 \renewcommand{\theenumii}{\textup{(\roman{enumii})}}
1564 \renewcommand{\labelenumii}{\theenumii}
1565 \</classXimera>

1566 \<cfgXimera>
1567 \catcode'\:=11
1568 % Insert <section> around thebibliography
1569 \ConfigureEnv{thebibliography}{\ifvmode\IgnorePar\fi \EndP \HCode{<section role="doc-bibliog
1570 % now configure thebibliography to produce a description list
1571 % \en:bib insertes delimiters for particular bibitems. at the beginning, it is empty, as the
1572 % it is then defined to insert the delimiter after the first bibitem
1573 \ConfigureList{thebibliography}%
1574   {\ifvmode\IgnorePar\fi\EndP\HCode{<dl><dt>}\let\en:bib=\empty}% opening tags
1575   {\ifvmode\IgnorePar\fi\EndP\HCode{</dd></dl>}} % closing tags
1576   {\en:bib\def\en:bib{\ifvmode\IgnorePar\fi\HCode{</dd><dt>}}}% at the bibitem
1577   {\HCode{</dt><dd>}}}% after biblabel
1578 \catcode'\:=12
1579 \Css{.thebibliography dl {
1580     display: grid;
1581     grid-auto-columns: min-content 1fr;
1582     grid-auto-flow: column;
1583 }}
1584 \Css{.thebibliography dt {
1585     grid-column: 1;
1586     margin-bottom: 0.5em;
1587 }}
1588 \catcode'\:=11
1589 \ConfigureList{enumerate}%
1590   {\EndP\HCode{<ol \a:enumerate:\space
1591     class="enumerate\expandafter\the\csname @enumdepth\endcsname"
1592     \a:LRdir

```

```

1593     >}\PushMacro\end:itm
1594 \global\let\end:itm=\empty
1595 }
1596     {\PopMacro\end:itm \global\let\end:itm \end:itm
1597 %
1598 \EndP\HCode{</li></ol>}\ShowPar
1599 }
1600     {\end:itm \gdef\end:itm{\EndP\Tg</li>}\DeleteMark
1601 }
1602     {{\Configure{Link}{li}{}}{ class="enumerate" id=}{}}%
1603 \let\EndLink=\empty\par\ShowPar
1604 \AnchorLabel }%
1605 }
1606 \catcode'\:=12
1607 </cfgXimera>

```

2.13.4 Making tables accessible

TailorSwiftBots developed the following code to make tables more accessible.

```

1608 (*classXimera)
1609 \ExplSyntaxOn
1610 \__tblr_keys_define:nn { table/inner }
1611 {
1612     xmcollhead .code:n = \__tblr_keys_gput:nn { xmcollhead } {#1},
1613 }
1614
1615 \int_new:N \lTblrxmColHeadInt
1616
1617 \int_new:N \lTblrxmColDiffInt
1618
1619 \int_new:N \lTblrxmRowDiffInt
1620
1621 \int_new:N \lTblrxmCellRowSpanInt
1622
1623 \int_new:N \lTblrxmColGroupCheck
1624
1625 \int_new:N \myInteger
1626
1627 \int_new:N \lTblrTestingInt
1628
1629 \int_new:N \l_my_test_int
1630
1631
1632 \tl_new:N \ColSpanEx_tl
1633
1634 \int_new:N \ColSpanEx
1635
1636 \str_new:N \myString
1637
1638 \tl_new:N \l_my_tl
1639
1640
1641 \prop_new:N \g__tblr_testing_prop
1642
1643 \prop_new:N \l__tblr_testing_prop
1644
1645 \str_new:N \l__tblr_testing_prop_str
1646
1647 \cs_new:Npn \add_integer_entry:nn #1 #2
1648 {
1649     \prop_put:Nnn \l__tblr_testing_prop { #1 } { #2 }
1650 }
1651

```

```

1652 \NewDocumentCommand{\AddIntegerEntry}{mm}
1653 {
1654     \add_integer_entry:nn { #1 } { #2 }
1655 }
1656
1657 %\AddIntegerEntry{first_entry}{44}
1658
1659 \cs_new:Npn \string_to_int_var:nN #1 #2
1660 {
1661     \int_set:Nn #2 { #1 }
1662 }
1663
1664 \cs_new:Npn \prop_get_to_int:NnN #1 #2 #3
1665 {
1666     \int_set:Nn #3 { \prop_item:Nn #1 { #2 } }
1667 }
1668
1669 \cs_new_protected:Npn \__tblr_build_col_head_foot:
1670 {
1671     %% \lTblrColHeadInt can not be empty, so we append '+ 0'.
1672     \int_set:Nn \lTblrxmColHeadInt
1673     { \__tblr_prop_item:ne { inner } { xmcolhead } + 0 }
1674     \int_compare:nNnTF { \lTblrxmColHeadInt } > { 0 }
1675     {
1676         \__tblr_build_one_table:nnNN {1} { \lTblrxmColHeadInt }
1677         \c_true_bool \c_true_bool
1678     }
1679 }
1680
1681 \prop_new:N \l__tblr_colgrouptest_prop
1682
1683 \cs_new:Npn \add_integer_entry_v:nn #1 #2
1684 {
1685     \prop_put:Nnn \l__tblr_colgrouptest_prop { #1 } { #2 }
1686 }
1687
1688 \NewDocumentCommand{\AddIntegerEntryV}{mm}
1689 {
1690     \add_integer_entry_v:nn { #1 } { #2 }
1691 }
1692
1693 \tl_new:N \l__tblr_z_tl
1694
1695
1696 \tl_new:N \l__tblr_A_tl
1697
1698
1699 \tl_new:N \l__tblr_B_tl
1700
1701 %\prop_put:Nnn \l__tblr_testing_prop {first_entry} { 47 }
1702
1703 %\AddIntegerEntryV{firsttest}{ 42 }
1704 %\AddIntegerEntry{first_entry}{43}
1705
1706
1707
1708
1709
1710 %% #1: data name; #2: data index 1; #3: data index 2; #4: key
1711 \cs_new:Npn \__tblr_data_test_item:nnnn #1 #2 #3 #4
1712 {
1713     \intarray_item:cn { g__tblr_#1_ \int_use:N \gTblrLevelInt _intarray }
1714     { \__tblr_data_key_to_int:nnnn {#1} {#2} {#3} {#4} }

```

```

1715
1716 }
1717
1718 \str_new:N \l_test_string
1719 \__tblr_keys_define:nn { table/inner }
1720 {
1721     caption .code:n = \__tblr_keys_gput:nn { caption } {#1},
1722     summary .code:n = \__tblr_keys_gput:nn { summary } {#1}
1723 }
1724
1725
1726 \tl_new:N \l__tblr_caption_tl
1727 \str_new:N \l__tblr_caption_str
1728
1729 \tl_new:N \l__tblr_summary_tl
1730 \str_new:N \l__tblr_summary_str
1731
1732 \cs_new:Npn \l__tblr_write_caption: #1
1733 {
1734     \tl_set:Nn \l__tblr_caption_tl
1735     { \__tblr_prop_item:ne { inner } { caption } }
1736     \str_set:Nc \l__tblr_caption_str
1737     { \tl_to_str:e \l__tblr_caption_tl }
1738     \tl_set:Nn \l__tblr_summary_tl
1739     { \__tblr_prop_item:ne { inner } { summary } }
1740     \str_set:Nc \l__tblr_summary_str
1741     { \tl_to_str:e \l__tblr_summary_tl }
1742 }
1743
1744 \int_new:N \l__tblr_number_cols_int
1745
1746 \tl_new:N \l__tblr_colgroup_tl
1747 \ExplSyntaxOff
1748 \</classXimera>
1749 \*cfgXimera>
1750
1751 \ExplSyntaxOn
1752 % Get the value of rowhead key and save it in the \lTblrRowHeadInt for later use
1753 \def\setrowhead{
1754     \int_set:Nn \lTblrRowHeadInt
1755     { \__tblr_prop_item:ne { inner } { rowhead } + 0 }
1756 }
1757
1758
1759
1760
1761 % Get the value of xmclohead key and save it in the \lTblrxmColHeadInt for later use and \lTbl
1762 % between the column header and the actual column
1763 \def\setcellinfo{
1764     \int_set:Nn \lTblrxmColHeadInt
1765     { \__tblr_prop_item:ne { inner } { xmclohead } + 0 }
1766     \int_set:Nn \lTblrxmColDiffInt
1767     { \HCol - \lTblrxmColHeadInt }
1768     \int_set:Nn \lTblrxmRowDiffInt
1769     { \HRow - \lTblrRowHeadInt }
1770     %\prop_get_to_int:NnN \l__tblr_testing_prop { Y} \lTblrxmColGroupCheck
1771     %\prop_get_to_int:NnN \l__tblr_colgrouptest_prop { first } \myInteger
1772     %\xdef\output{\@arabic\lTblrxmColGroupCheck}
1773     %\string_to_int_var:nN \lTblrxmColGroupCheck \myInteger
1774     %\int_set:Nn \lTblrxmColGroupCheck {\prop_item:Ne \l__tblr_testing_prop {first} }
1775     %\int_set:Nn \output_test{\@arabic\lTblrxmColGroupCheck}
1776 }
1777

```

```

1778
1779 % print either th with scope or td ...
1780 \def\testcell{
1781     \ifnum\lTblrxmColDiffInt<1
1782         \ifnum\lTblrxmRowDiffInt<1
1783             td
1784         \else
1785             \ifnum\HRowspan>1
1786                 th \space scope="rowgroup"
1787             \else
1788                 th \space scope="row"
1789             \fi
1790         \fi
1791     \else
1792         \ifnum\lTblrxmRowDiffInt<1
1793             \ifnum\HMultispan>1
1794                 th \space scope="colgroup"
1795             \else
1796                 th \space scope="col"
1797             \fi
1798         \else
1799             td
1800         \fi
1801     \fi}
1802
1803
1804
1805
1806
1807 \ExplSyntaxOff
1808
1809
1810
1811
1812 % this is the configuration for the tabularray package with the support for <th>
1813 \catcode'\:=11
1814 \makeatletter
1815 \Configure{tabularray}{%
1816     \ifvmode\IgnorePar\fi\EndP%
1817     \gHAdvance\Next:TableNo by 1 \global\let\TableNo=\Next:TableNo%
1818     \HCode{<table class="tabularray \@currenvir" id="tbl-\TableNo">}
1819 }{\ifvmode\IgnorePar\fi\EndP\HCode{</table>}}
1820 {\setrowhead\HCode{<tr id="row-\TableNo-\HRow-">}}{\HCode{</tr>}}
1821 {\setcellinfo\HCode{<\:testcell\space\:colgroupcheck\:testcheck colspan="HMultispan" rowsp}}
1822 %{\Configure{tabularrayattributes}{rowspan="lTblrCellRowSpanInt"}}
1823 %the line below cannot be deleted
1824 {\Configure{tabularrayattributes}{}}{}
1825 \catcode'\:=12
1826 \makeatother
1827 </cfgXimera>
1828 <*htXimera>
1829 \makeatletter
1830 \ExplSyntaxOn
1831 \cs_new:Npn \l__tblr_testing_func_three:n #1
1832 {
1833     \tl_set:Nn \l__tblr_colgroup_tl { }
1834     \int_set:Nn \l__tblr_number_cols_int { \c@colcount }
1835     \int_step_variable:nNn { 1 } \l__tblr_A_tl %%this checks the first row for colhead, I don
1836     {
1837         \int_step_variable:nNn { \l__tblr_number_cols_int } \l__tblr_z_tl
1838         {
1839             \tl_set:Nn \ColSpanEx_tl
1840             {

```

```

1841         \tblr_data_item:neen { cell } { \tblr_A_tl } { \tblr_z_tl } { colspan } {
1842     }
1843     %\int_set:Nn \tblr_testing_int {8000 + \tblr_colgroup_check + #1}
1844     \int_set:Nn \tblr_testing_int {\ColSpanEx_tl} %currently this is an integer, if a later up
1845     \ifnum\int_eval:n{\tblr_z_tl}>0 %%this if statement is always true, it literally does
1846         \prop_put:Nnn \tblr_testing_prop {\tblr_z_tl} {\tblr_testing_int}
1847         \prop_get_to_int:Nnn \tblr_testing_prop { \tblr_z_tl } \tblr_colgroup_check
1848         \def\outputV{\@arabic\tblr_colgroup_check }
1849         \tblr_put_right:Ne \tblr_colgroup_tl { <colgroup \space span="\outputV"></colgroup> }
1850     \fi
1851 }
1852 }
1853 }
1854
1855
1856 %\prop_get_to_int:Nnn \tblr_testing_prop {Y} \tblr_colgroup_check %% needs to be in Arab
1857
1858
1859
1860 \long\def\tempa#1#2#3#4{%
1861     % insert <table>...</table>
1862     \a:tabularray \o:tblr_environs_code:nnnn:{#1}{#2}{#3}{#4}\b:tabularray
1863 }
1864
1865
1866 \HLet\tblr_environs_code:nnnn:\tempa
1867
1868
1869 \long\def\tempa#1{
1870     \int_set:Nn \c@rownum {#1}\xdef\HRow{\@arabic\c@rownum}
1871     \int_step_function:nnnN { 13 } { 1 } {14} \tblr_testing_func_three:n
1872     %\int_step_function:nnnN { 13 } { 1 } {14} \tblr_testing_func_four:n
1873     \int_step_function:nnnN { 13 } { 1 } {14} \tblr_write_caption:
1874     \def\caption_string{\tblr_caption_str}
1875     \def\colgroup_string{\tblr_colgroup_tl}
1876     \def\outputV{\@arabic\tblr_colgroup_check }
1877     %% <col> <colgroup \space span="\outputV"></colgroup>
1878     \ifnum\HRow=1
1879         \HCode{ <caption>\tblr_caption_str <span>\tblr_summary_str </span> </caption> \col
1880     \fi
1881     \c:tabularray\o:tblr_build_row:N:{#1}\d:tabularray
1882 }
1883 \HLet\tblr_build_row:N:\tempa
1884
1885 % disable rules
1886 \ht:special{t4ht@_}
1887
1888 % #1 row number, #2 column, #3 hline number (there can be multiple), #4 css property to be s
1889 \def\tblr:hlinestyle#1#2#3#4{
1890     % get line height
1891     \tblr_set:Ne \tblr_h_tl{ \tblr_spec_item:ne { hline } { [#1](#3) / @hline-height } }
1892     % get dash style
1893     \def\tblr@dash{} % remove "dash" word from the variable for correct CSS string
1894     \tblr_set:Ne \tblr_f_tl{\tblr_spec_item:ne { hline } { [#1][#2](#3) / @dash }}
1895     % create CSS only when a dash style is set
1896     \tblr_if_empty:NF\tblr_f_tl{
1897         % get hline color
1898         \tblr_set:Ne \tblr_g_tl{ \tblr_spec_item:ne { hline } { [#1][#2](#3) / fg } }
1899         \def:hlinecolor{#000000}
1900         % convert color to CSS value if color is set
1901         \tblr_if_empty:NF\tblr_g_tl{\get:xcolorcss{\tblr_g_tl}\hlinecolor}
1902         % \Configure{tabularraystyles} doesn't expand attributes, so we need to expand it here
1903         % otherwise, we would get wrong color and hline style in the last row, because this macro

```

```

1904 \cs_set:ce{#4:}{#4:\dim_to_decimal_in_unit:nn{\l__tblr_h_tl*2}{1px}px~\l__tblr_f_tl\space
1905 \Configure{tabularraystyles}{\csname#4:\endcsname}
1906 }
1907 }
1908
1909
1910
1911 \def\tblr:vlinestyle#1#2#3#4{
1912 \tl_set:Ne \l__tblr_t_tl{ \__tblr_spec_item:ne { vline } { [#2](#3) / @vline-width } }
1913 \def\tblr@dash{ % remove "dash" word from the variable for correct CSS string
1914 \tl_set:Ne \l__tblr_f_tl{\__tblr_spec_item:ne { vline } { [#1][#2](#3) / @dash }}
1915 \tl_if_empty:NF\l__tblr_f_tl{
1916 \tl_set:Ne \l__tblr_g_tl { \__tblr_spec_item:ne { vline } { [#1][#2](#3) / fg } }
1917 \def\hlinecolor{#000000}
1918 % convert color to CSS value if color is set
1919 \tl_if_empty:NF\l__tblr_g_tl{\get:xcolorcss{\l__tblr_g_tl}\hlinecolor}
1920 % \Configure{tabularraystyles} doesn't expand attributes, so we need to expand it here
1921 % otherwise, we would get wrong color and hline style in the last row, because this macro
1922 \cs_set:ce{#4:}{#4:\dim_to_decimal_in_unit:nn{\l__tblr_t_tl*2}{1px}px~\l__tblr_f_tl\space
1923 \Configure{tabularraystyles}{\csname#4:\endcsname}
1924 }
1925 }
1926
1927 \long\def\tempa#1#2{%
1928 % find columns that are covered by rowspan and colspan
1929 \xdef\HCol{\@arabic\c@colnum}
1930 \xdef\HRow{\@arabic\c@rownum}
1931 %I added the arabic below -- Jeff
1932 \xdef\HMultispan{\@arabic\lTblrCellColSpanInt}
1933 \xdef\HRowspan{\@arabic\lTblrCellRowSpanInt}
1934
1935
1936 %\xdef\XMCOLGROUPCHECK{\@arabic\lTblrxmColGroupCheck}
1937 %\xdef\output_test{\@arabic\lTblrxmColGroupCheck}
1938 \let\CellAttributes\empty
1939 \let\CellStyle\empty
1940 % calculate ignored cells, if the current cell uses colspan or rowspan
1941 \int_step_inline:nnn{\c@rownum }{\c@rownum - 1 + \lTblrCellRowSpanInt}{
1942 \int_step_inline:nnn{\c@colnum }{\c@colnum - 1 + \lTblrCellColSpanInt}{
1943 % the loop always matches the current cell, we must ignore it
1944 \str_if_eq:eeF{\HCol.\HRow}{####1.##1}{%
1945 \cs_gset:cpn{ignoredcell-####1-##1}{}
1946 }
1947 }
1948 }
1949 \cs_if_exist_use:c{tabularray_halign:\g__tblr_cell_halign_tl}
1950 \cs_if_exist_use:c{tabularray_valign:\g__tblr_cell_valign_tl}
1951 % the vertical alignment can be set also in \g__tblr_cell_middle_tl, so we should try it as
1952 \cs_if_exist_use:c{tabularray_valign:\g__tblr_cell_middle_tl}
1953 % calculate column width
1954 \dim_compare:nNnT {\__tblr_data_item:nen{column}{\HCol}{@col-width}} > {0pt}{
1955 \__tblr_get_table_width:% initialize \tablewidth
1956 \edef\HColWidth{\fp_eval:n{\__tblr_data_item:nen{column}{\HCol}{@col-width}/\tablewidth*}
1957 % save table width, preferably in CSS
1958 \a:tabularraycolumnwidth%
1959 }
1960 % there can be multiple hlines for each cell, but we only suport the first one, because of
1961 \tblr:hlinestyle{#1}{#2}{1}{border-top}
1962 \int_compare:nNnT{\HRow + \HRowspan - 1} = {\c@rowcount}{%
1963 % draw hline below the last row
1964 \tblr:hlinestyle{\int_eval:n{\c@rownum + 1}}{#2}{1}{border-bottom}
1965 }
1966 % the same is true for vlines

```

```

1967 \tblr:vlinestyle{#1}{#2}{1}{border-left}
1968 \int_compare:nNnT{\HCol + \HMultispan - 1} = {\c@colcount}{%
1969 % draw hline below the last row
1970 \tblr:vlinestyle{#1}{\int_eval:n{\c@colnum + 1}}{1}{border-right}
1971 }
1972 % support for the background color
1973 \tl_set:Nx \l__tblr_b_tl
1974 { \__tblr_data_item:neen { cell } {#1} {#2} { background } }
1975 % save background color to the list of CSS, if it is set
1976 \tl_if_empty:NF \l__tblr_b_tl{
1977 \get:xcolorcss{\l__tblr_b_tl}{:bgcolor}
1978 \Configure{tabulararraystyles}{background-color: \:bgcolor;}
1979 }
1980 % We can use something like \Configure{tabulararrayattributes}{rowspan="\HRowspan"} in \Conf
1981 % to declare correct attributes for joined cells
1982 \int_compare:nNnT {\HRowspan} > {1}{\g:tabulararray}
1983 \int_compare:nNnT {\HMultispan} > {1}{\h:tabulararray}
1984 \cs_if_exist:cTF{ignoredcell-\the\c@colnum-\the\c@rownum}{%
1985 \a:tabulararrayignoredcell\the\c@colnum-\the\c@rownum}{%
1986 }{%
1987 \e:tabulararray\o:__tblr_build_cell_content:NN:{#1}{#2}\f:tabulararray
1988 }
1989 % the ignored cell is global, so we must undefine it after the thes
1990 \cs_undefine:c{ignoredcell-\the\c@colnum-\the\c@rownum}%
1991 }
1992 \HLet\__tblr_build_cell_content:NN:\tempa
1993
1994 \HLet\__tblr_get_vcenter_box:N:\gobble
1995
1996 %Jeff moved this here
1997 \def\output{\@arabic\lTblrxmColGroupCheck}
1998 \def\colgroupcheck{
1999 \ifnum\lTblrxmColGroupCheck>0
2000 test="\output" \space
2001 \else
2002 \space
2003 \fi
2004 }
2005
2006 \def\testcheck{
2007 \ifnum\myInteger>0
2008 test="taylor swift" \space
2009 \else
2010 \space
2011 \fi
2012 }
2013 \ExplSyntaxOff
2014 \makeatother
2015 \</htXimera>

```

2.14 Tools

2.14.1 Suppress

suppress 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`.

```

2016 \<classXimera>
2017 \font\dummyft@=dummy \relax
2018 \def\suppress{%
2019 \begingroup\par
2020 \parskip\z@
2021 \offinterlineskip

```

```

2022 \baselineskip=\z@skip
2023 \lineskip=\z@skip
2024 \lineskiplimit=\maxdimen
2025 \dummyft@
2026 \count@\sist@@n
2027 \loop\ifnum\count@ >\z@
2028 \advance\count@m@one
2029 \textfont\count@\dummyft@
2030 \scriptfont\count@\dummyft@
2031 \scriptscriptfont\count@\dummyft@
2032 \repeat
2033 \let\selectfont\relax
2034 \let\mathversion\@gobble
2035 \let\getanddefine@fonts\@gobbletwo
2036 \tracinglostchars\z@
2037 \frenchspacing
2038 \hbadness\@M}
2039 \def\endsuppress{\par\endgroup}
2040 \end{classXimera}

```

2.14.2 The End

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

```

2041 \end{htXimera}
2042 \Hinput{ximera}
2043 \end{htXimera}

2044 \end{htXourse}
2045 \Hinput{xourse}
2046 \end{htXourse}

2047 \end{cfgXimera}
2048 \begin{document}
2049 \EndPreamble
2050 \end{cfgXimera}

```

3 xourse.cls

```

2051 \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.

```

2052 \newif\ifnotoc
2053 \notocfalse
2054 \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.

```

2055 \newif\ifnonewpage
2056 \nonewpagefalse
2057 \DeclareOption{nonewpage}{\nonewpagetrue}

2058 \DeclareOption*{\PassOptionsToClass{\CurrentOption}{ximera}}
2059 \ProcessOptions\relax
2060 \LoadClass{ximera}
2061 % \begin{macrocode}
2062 \end{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`.

```
2063 <*classXourse>
2064 \newcommand{\skip@preamble}{%
2065     \let\document\relax\let\enddocument\relax%
2066     \newenvironment{document}{\let\input\otherinput}{}%
2067     \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:

```
2068 \let\otherinput\input
Store usual \maketitle as \othermaketitle
2069 \let\othermaketitle\maketitle
```

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

```
2070 \renewcommand{\maketitle}{ %
2071 \pagestyle{empty}
2072 \begin{center}
2073 ~\ \ %puts space at top of page to move title down.
2074 \vskip .25\textheight
2075 \hrulefill\
2076 \vskip 1em
2077 \bfseries\Huge \@title\ \
2078 \hrulefill\
2079 \vskip 3em
2080 {\Large \@author}
2081 \vskip 2em
2082 {\large \@date}
2083 \end{center}
2084 \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.

```
2085 \ifnotoc
2086 \else
2087     \tableofcontents\clearpage
2088     \clearpage
2089 \fi
```

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

```
2090 \pagestyle{main}
Renew maketitle to usual definition.
2091 \let\maketitle\othermaketitle
```

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

```
2092 }
2093 \relax
2094 </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.

```
2095 <*classXourse>
2096 \ifnonewpage
2097 \newcommand{\activity}[2][ ]{%
2098 \setkeys{activity}{#1}}
```

```

2099 \renewcommand{\input}[1]{
2100 \begingroup\skip@preamble\otherinput{#2}\endgroup\par\vspace{\topsep}
2101 \let\input\otherinput}
2102 \else
2103 \newcommand{\activity}[2][]{%
2104 \setkeys{activity}{#1}
2105 \renewcommand{\input}[1]{
2106 \begingroup\skip@preamble\otherinput{#2}\endgroup\clearpage
2107 \let\input\otherinput}
2108 \fi
2109 \relax
2110 \endclassXourse}

2111 \htXourse}
2112 \renewcommand\activity[2][]{%
2113 \ifvmode \IgnorePar\fi \EndP\HCode{<a class="activity card \activitystyle" href="#2" data-op
2114 }
2115 \endhtXourse}

```

When running xake, we can just ignore activities

```

2116 \classXourse}
2117 \ifxake
2118 \renewcommand\activity[2][]{
2119 \fi
2120 \endclassXourse}

```

3.1.2 Practice activities

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

```

2121 \classXourse}
2122 \ifhandout
2123 \newcommand{\practice}[2][]{
2124 \setkeys{practice}{#1}%!!!!
2125 \renewcommand{\input}[1]{
2126 \begingroup\skip@preamble\otherinput{#2}\endgroup
2127 \let\input\otherinput}
2128 \else
2129 \newcommand{\practice}[2][]{\texttt{\detokenize{#2}}}% gives file name for practice
2130 \setkeys{practice}{#1}%!!!!
2131 \renewcommand{\input}[1]{
2132 \begingroup\skip@preamble\otherinput{#2}\endgroup
2133 \let\input\otherinput}
2134 \fi
2135 \relax
2136 \endclassXourse}

```

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

```

2137 \classXourse}
2138 \ifxake
2139 \renewcommand\practice[2][]{
2140 \fi
2141 \endclassXourse}

```

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

```

2142 \htXourse}
2143 \renewcommand\practice[2][]{%
2144 \ifvmode\IgnorePar\fi\EndP%
2145 \HCode{<a class="activity card practice" href="#2" data-options="#1">#2</a>}%
2146 \IgnoreIndent%
2147 }
2148 \endhtXourse}

```

3.2 Sectioning

<code>\section</code>	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.
	<pre> 2149 <*classXourse> 2150 \renewcommand*{l@section}{\@dottedtocline{1}{1.5em}{4.2em}} 2151 </classXourse> </pre>
<code>\subsection</code>	The name of a subsection inside an activity.
	<pre> 2152 <*classXourse> 2153 \renewcommand*{l@subsection}{\@dottedtocline{2}{3.8em}{4.2em}} 2154 </classXourse> </pre>
<code>\part</code>	Xourse files can have parts. The name of a large part of a xourse.
	<pre> 2155 <*htXourse> 2156 \newcounter{ximera@part} 2157 \setcounter{ximera@part}{0} 2158 \renewcommand\part[1]{% 2159 \stepcounter{ximera@part}% 2160 \ifvmode \IgnorePar\fi \EndP% 2161 %\HCode{<h1 id="part\arabic{ximera@part}" class="card part">}#1\HCode{</h1>}}% makes cards di 2162 \HCode{<h1 id="part\arabic{ximera@part}" class="card part">}#1</h1>}}% 2163 \IgnoreIndent% 2164 } 2165 </htXourse> </pre>
<code>\paragraph</code>	Paragraph commands emit spans. A small heading.
	<pre> 2166 <*cfgXimera> 2167 \renewcommand{\paragraph}[1]{% 2168 \HCode{}}% 2169 #1% 2170 \HCode{}\par\IgnorePar} 2171 </cfgXimera> </pre>
<code>\subparagraph</code>	An even smaller heading.
	<pre> 2172 <*cfgXimera> 2173 \renewcommand{\subparagraph}[1]{% 2174 \HCode{}}% 2175 #1% 2176 \HCode{}\par\IgnorePar} 2177 </cfgXimera> </pre>

3.3 Grading by points

`graded` 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.

```

2178 <*classXourse>
2179 \newenvironment{graded}[1]{}{}
2180 </classXourse>

```

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

```

2181 <*htXourse>
2182 \renewenvironment{graded}[1]{%
2183 \ifvmode \IgnorePar\fi \EndP\HCode{<div class="graded" data-weight="#1">}\IgnoreIndent%
2184 }{
2185 \ifvmode \IgnorePar\fi \EndP\HCode{</div>}\IgnoreIndent%
2186 }
2187 </htXourse>

```

3.4 Logos

`\logo` A logo for the xourse.

```

2188 <*classXourse>

```

```

2189 \newcommand*{\logo}[1]{%
2190   \ifx\@onlypreamble\@notprerr
2191     \ClassError{xourse}{logo can only be used in the preamble}
2192     {Move your logo command to the preamble}
2193   \else %
2194     \IfFileExists{#1}%
2195     {\gdef\xourse@logo{#1}}%
2196     {\ClassError{xourse}{logo file does not exist}
2197      {To use logo, make sure that the referenced image file exists}}%
2198   \fi%
2199 }
2200
2201 \end{classXourse}

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

2202 \begin{htXourse}
2203 \Configure{@HEAD}{%
2204   \HCode{<meta name="og:image" content="}%
2205   \ifdefined\xourse@logo%
2206     \xourse@logo%
2207   \fi%
2208   \HCode{" />\Hnewline}}%
2209 \end{htXourse}

```