

# 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  $\text{\TeX}$  ends.” The `ximera` class aids in the creation of hand-outs, worksheets, exercises, and sections of textbooks to be used either individually or “glued” together via a `xourse` file. All `ximera` documents can be deployed in an online interactive form via `xake`. See: [Ximera Project](#) and the source code on [GitHub](#).

## 1 Introduction

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

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

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

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

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

Online examples can be found at

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

---

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

## 2 ximera.cls

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

### 2.1 Options for the class

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

```
6 \classXimera
```

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

```
26 \newif\ifnumbers
27 \numbersfalse
28 \DeclareOption{numbers}{\numberstrue}
```

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

```

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

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

65 </classXimera>
66 <*classXimera>

```

## 2.2 Loading packages

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

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

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

```

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

```

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

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

Now we load even more packages.

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

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

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

## 2.3 Page setup

We want non-indented spaced-out paragraphs.

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

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

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

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

```
108 \begincfgXimera
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{}}
117
118 % Output HTML5 doctype instead of the default for HTML4
119 % \Configure{DOCTYPE}{\HCode{<!doctype html>\Hnewline}}
120
121 % Custom page opening
122 % \Configure{HTML}{\HCode{<html lang="en">\Hnewline}}{\HCode{\Hnewline</html>}}
123
124 % Reset <head>, aka delete all default boilerplate; alternatively set up new content
125 % \Configure{@HEAD}{\HCode{<meta name="generator" content="TeX4ht (http://www.cse.ohio-state.edu/~dpc/TeX4ht/)>\Hnewline}}
126 \Configure{@HEAD}{\HCode{<meta name="ximera" content="version 2.5.1" /\Hnewline}}
127 \Configure{@HEAD}{\HCode{<link href="https://ximera.osu.edu/public/stylesheets/standalone.css" type="text/css" /\Hnewline}}
128 \Configure{@HEAD}{\HCode{<script type="text/javascript" async src="https://ximera.osu.edu/public/js/ximera.js" /\Hnewline}}
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 (env.)    Theorem
172 <classXimera>    \newtheorem{theorem}{\GetTranslation{Theorem}}
173 <htXimera>       \ConfigureTheoremEnv{theorem}

```

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

<b>proposition</b> ( <i>env.</i> )	Proposition	
	210 <code>&lt;classXimera&gt;</code>	<code>\newtheorem{proposition}{\GetTranslation{Proposition}}</code>
	211 <code>&lt;htXimera&gt;</code>	<code>\ConfigureTheoremEnv{proposition}</code>
<b>paradox</b> ( <i>env.</i> )	Paradox	
	212 <code>&lt;classXimera&gt;</code>	<code>\newtheorem{paradox}{\GetTranslation{Paradox}}</code>
	213 <code>&lt;htXimera&gt;</code>	<code>\ConfigureTheoremEnv{paradox}</code>
<b>procedure</b> ( <i>env.</i> )	Procedure	
	214 <code>&lt;classXimera&gt;</code>	<code>\newtheorem{procedure}{\GetTranslation{Procedure}}</code>
	215 <code>&lt;htXimera&gt;</code>	<code>\ConfigureTheoremEnv{procedure}</code>
<b>remark</b> ( <i>env.</i> )	Remark	
	216 <code>&lt;classXimera&gt;</code>	<code>\newtheorem{remark}{\GetTranslation{Remark}}</code>
	217 <code>&lt;htXimera&gt;</code>	<code>\ConfigureTheoremEnv{remark}</code>
<b>summary</b> ( <i>env.</i> )	Summary	
	218 <code>&lt;classXimera&gt;</code>	<code>\newtheorem{summary}{\GetTranslation{Summary}}</code>
	219 <code>&lt;htXimera&gt;</code>	<code>\ConfigureTheoremEnv{summary}</code>
<b>template</b> ( <i>env.</i> )	Template	
	220 <code>&lt;classXimera&gt;</code>	<code>\newtheorem{template}{\GetTranslation{Template}}</code>
	221 <code>&lt;htXimera&gt;</code>	<code>\ConfigureTheoremEnv{template}</code>
<b>warning</b> ( <i>env.</i> )	Warning	
	222 <code>&lt;classXimera&gt;</code>	<code>\newtheorem{warning}{\GetTranslation{Warning}}</code>
	223 <code>&lt;htXimera&gt;</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** (*env.*) 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[\hskip\labelsep\sffamily\bfseries\GetTranslation{#2} \problemNumber% Determine the cor
343 ]%
344 \slshape
345 }
346 %%%% Configure environments end content %%%%
347 \newcommand{\problemEnvironmentEnd}[2]{%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
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 opti
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** (*env.*) 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{1cm}}
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{1cm}}
472   \small\slshape
473 }
474 {
475   \end{trivlist}
476 }
477 \fi
478
479 </classXimera>

```

## 2.4.7 Solution

**solution** (*env.*) 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 \end{classXimera}

```

### 2.4.8 Code listing environments

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

```

506 \begin{classXimera}
507 \DefineVerbatimEnvironment{code}{Verbatim}{numbers=left,frame=lines,label=Code,labelposition=left}
508 \end{classXimera}

```

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

```

509 \begin{classXimera}
510 \DefineVerbatimEnvironment{python}{Verbatim}{numbers=left,frame=lines,label=Python,labelposition=left}
511 \end{classXimera}

```

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

```

512 \begin{classXimera}
513 \DefineVerbatimEnvironment{javascriptCode}{Verbatim}{numbers=left,frame=lines,label=JavaScript,labelposition=left}
514 \end{classXimera}
515 \begin{cfgXimera}
516 \renewenvironment{javascriptCode}{\NoFonts}{\EndNoFonts}
517 \ScriptEnv{javascriptCode}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<d
518 \end{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; backgroun
521 %%\ConfigureEnv{lstlisting}{\ifvmode\IgnorePar\fi\EndP\HCode{<pre>}}{\ifvmode\IgnorePar\fi\EndP
522 %%%</cfgXimera>
523 %%%

```

### 2.4.9 Dialogues

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

```

524 \begin{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 \begin{classXimera}
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{classXimera}

```

#### 2.4.10 Instructor notes

```

545 \begin{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{1.5ex}\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{1.5ex}\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 (env.)` 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` (*env.*) 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` (*env.*) 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}\@t
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<sup>A</sup>T<sub>E</sub>X to have a command and an environment with the same name. We opted for the above (confusing...) names.

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

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

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

```
prompt (env.)    The prompt part for mathmode
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 (env.) Only display online
onlyPdf (env.)  Only display in the PDF
onlineOnly (env.) 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<sub>l</sub>atex, 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<sub>E</sub>X 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 (env.)` Place images inside an `image` environment. On paper, this centers the image. On the web, this provides additional benefits. Base `graphicspath`, default `'/xmPictures'`. Can only be changed BEFORE loading `ximera.cls`!

```

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. Generic command to provide alt-text for assistive technology like screenreaders, via the HTML `details` tag.

```

972 \<classXimera>
973 \newcommand{\alt}[1]{} %% Deprecated
974
975 \providecommand\xmaltprint{}
976 %% use \undef\xmaltprint to NOT have \xmalt output in the PDF
977 \newcommand{\xmalt}[1]{% %% see infra for HTML implementation; 2026, J. Kuan
978   \ifdefined\xmaltprint%
979     {\smaller\color{gray}Alt text: #1}%
980   \fi%
981 }
982 \</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.

```

983 <*htXimera>
984 \newcounter{imagealt}
985 \setcounter{imagealt}{0}
986 \renewenvironment{image}[1][\stepcounter{imagealt}%
987   \ifvmode \IgnorePar\fi \EndP%
988   \HCode{<div class="image-environment" role="img" aria-labelledby="image-alt-\arabic{imagea
989 }{\HCode{</div>}}
990 \renewcommand{\alt}[1]{\HCode{<div style="display: none;" id="image-alt-\arabic{imagealt}">}}
991 \renewcommand{\xmalt}[1]{% %% implementation 2026, J. Kuan
992   \ifvmode \IgnorePar\fi \EndP%
993   \HCode{<details><summary>Show Alt Text</summary>} #1 \HCode{</details>}}%
994 }
995 </htXimera>
996 <*cfgXimera>
997 %% Although we accept many formats, SVG is preferred on the web.
998 %% Since we have a different mechanism for producing |alt| text, we
999 %% want to ignore tex4ht's own method fo producing alt text.
1000 %% 2024: is now in TeX4ht ...
1001 % \DeclareGraphicsExtensions{.jpg,.png,.gif,.svg}
1002 % \Configure{graphics*}
1003 % {svg}{
1004 %   {\Configure{Needs}{File: \Gin@base.svg}\Needs{}}
1005 %   \Picture[]{\csname Gin@base\endcsname.svg \csname a:Gin-dim\endcsname}%
1006 % }
1007 </cfgXimera>

```

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

```

1008 <*cfgXimera>
1009 \ifcsname ifstandalone\endcsname
1010   \ifstandalone
1011     \renewcommand\includegraphics[2][\fi
1012   \fi
1013 </cfgXimera>

```

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

```

1014 <*htXimera>
1015 \providecommand{\pgfsyspdfmark}[3]{\fi
1016 </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.

```

1017 <*classXimera>
1018 % everything skipped, assume TeX4ht does the job now
1019 \ifdefined\reallyneverever
1020
1021 \ifdefined\HCode
1022   \tikzexporttrue
1023 \fi
1024
1025 \iftikzexport
1026   \usetikzlibrary{external}
1027
1028   \ifdefined\HCode
1029     % in htlatex, just include the svg files
1030     \def\pgfsys@imagesuffixlist{.svg}
1031
1032     \tikzexternalize[prefix=./,mode=graphics if exists]
1033   \else

```

```

1034 % in pdflatex, actually generate the svg files
1035 \tikzset{
1036   /tikz/external/system call={
1037     pdflatex \tikzexternalcheckshellescape
1038     -halt-on-error -interaction=batchmode
1039     -jobname "\image" "\PassOptionsToClass{tikzexport}{ximera}\texsource";
1040     mutool draw -F svg \image.pdf > \image.svg ; % mutool adds "1" to filename ???
1041     mutool draw -o \image.svg \image.pdf ;
1042     mutool draw -r 150 -c rgbalpha -o \image.png \image.pdf ;
1043     ebb -x \image.png
1044   }
1045 }
1046 \tikzexternalize[optimize=false,prefix=./]
1047 \fi
1048
1049 \fi
1050 \fi
1051 \end{classXimera}

```

### 2.6.3 XKCD

`\xkcd` Reference an XKCD cartoon.

```

1052 \begin{classXimera}
1053 \newcommand{\xkcd}[1]{#1}
1054 \end{classXimera}

```

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

```

1055 \begin{htXimera}
1056 \renewcommand{\xkcd}[1]{\ifvmode \IgnorePar\fi \EndP\HCode{<img src="https://imgs.xkcd.com/c
1057 \end{htXimera}

```

## 2.7 Links

We put `hyperref` after all other packages because that is better.

```

1058 \begin{classXimera}
1059 % Don't use hyperref when using Tex4ht
1060 \ifdefined\HCode
1061 \RequirePackage{hyperref}
1062 \else
1063 \RequirePackage[pdfpagelabels,colorlinks=true,allcolors=blue!30!black]{hyperref}
1064 \pdfstringdefDisableCommands{\def\hskip{}}%% quiets warning
1065 \fi
1066 \end{classXimera}

```

## 2.8 Interactives

### 2.8.1 Including widgets

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

```

1067 \begin{classXimera}
1068 \define@key{interactive}{id}{\def\interactive{id}{#1}}
1069 \setkeys{interactive}{id=}
1070 \newcommand{\includeinteractive}[2][]{
1071 \setkeys*{interactive}{#1}%
1072 \ifthenelse{\equal{\interactive{id}}{}}{\recordvariable{\interactive{id}}}
1073 Interactive
1074 }
1075 \end{classXimera}

1076 \begin{htXimera}
1077 \renewcommand{\includeinteractive}[2][]{\stepcounter{identification}\ifvmode \IgnorePar\fi \
1078 \end{htXimera}

```

### 2.8.2 Google Sheet

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

```
1079 (*classXimera)
1080 % Google Spreadsheet link (read only)
1081 \newcommand{\googleSheet}[5]{%
1082   Google Spreadsheet link: \expandafter\url{\detokenize{https://docs.google.com/spreadsheets/
1083 }}
1084 }/classXimera)

1085 (*htXimera)
1086 \renewcommand{\googleSheet}[5]{%
1087   \ifthenelse{\equal{#4}{}}{%
1088     {\HCode{<iframe width="#2px" height="#3px" src="https://docs.google.com/spreadsheets/d/#
1089     {\ifthenelse{\equal{#5}{}}%
1090       {\HCode{<iframe width="#2px" height="#3px" src="https://docs.google.com/spreadsheets/
1091       {\HCode{<iframe width="#2px" height="#3px" src="https://docs.google.com/spreadsheets/
1092     }}
1093   }}
1094 }/htXimera)
```

### 2.8.3 Geogebra

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

```
1095 (*classXimera)
1096 %Geogebra link
1097 \newcommand{\geogebra}[3]{GeoGebra link: \url{https://www.geogebra.org/m/#1}}
1098 }/classXimera)

Define keys for answer geogebra key=value pairs.

1099 (*htXimera)
1100 \define@key{geogebra}{rc}[true]{\def\geo@rc{#1}}
1101 \define@key{geogebra}{sdz}[true]{\def\geo@sdz{#1}}
1102 \define@key{geogebra}{smb}[true]{\def\geo@smb{#1}}
1103 \define@key{geogebra}{stb}[true]{\def\geo@stb{#1}}
1104 \define@key{geogebra}{stbh}[true]{\def\geo@stbh{#1}}
1105 \define@key{geogebra}{ld}[true]{\def\geo@ld{#1}}
1106 \define@key{geogebra}{sri}[true]{\def\geo@sri{#1}}
1107 %set default key values
1108 \setkeys{geogebra}{rc=false,sdz=false,smb=false,stb=false,stbh=false,ld=false,sri=false}
1109 %command definition
1110 \renewcommand{\geogebra}[4][ ]{%
1111   \setkeys{geogebra}{#1}% Set new keys
1112   \HCode{<iframe scrolling="no" src="https://www.geogebra.org/material/iframe/id/#2/width/#3
1113 }/htXimera)
```

### 2.8.4 Desmos

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

```
1114 (*classXimera)
1115 \newcommand{\desmos}[3]{Desmos link: \url{https://www.desmos.com/calculator/#1}}
1116 \newcommand{\desmosThreeD}[3]{Desmos3D link: \url{https://www.desmos.com/3d/#1}}
1117 }/classXimera)

1118 (*htXimera)
1119 \catcode'\%=11
1120 \renewcommand{\desmos}[3]{\HCode{<iframe src="https://www.desmos.com/calculator/#1" width="#1
1121 \catcode'\%=14
1122 \renewcommand{\desmosThreeD}[3]{\HCode{<iframe src="https://www.desmos.com/3d/#1" width="#2p
1123 }/htXimera)
```



## 2.8.5 Graphs

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

```
1124 \classXimera>
1125 \newcommand{\graph}[2][\text{Graph of } \mathbb{R}^2}
1126 \endclassXimera>

1127 \htXimera>
1128 \renewcommand{\graph}[2][\HCode{<div class="graph" data-options="#1">#2\HCode{</div>}}
1129 \htXimera>
```

## 2.8.6 Video

`\youtube` Youtube command. Requires id.

```
1130 \classXimera>
1131 \newcommand{\youtube}[1]{YouTube link: \url{https://www.youtube.com/watch?v=#1}}
1132 \endclassXimera>

1133 \htXimera>
1134 %% \renewcommand{\youtube}[1]{\ifvmode \IgnorePar\fi \EndP\HCode{<div class="video youtube-p
1135 % Fixes no-youtube-when-no-cookies-accepted. Class xmyoutube allows for css customization.
1136 \renewcommand{\youtube}[1]{\ifvmode \IgnorePar\fi \EndP\HCode{<iframe class="xmyoutube" src=
1137
1138 \htXimera>
```

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

```
1139 \htXourse>
1140 \renewcommand\youtube[1]{%
1141 \ifvmode \IgnorePar\fi \EndP\HCode{<a class="youtube" href="https://www.youtube.com/watch?v=
1142 }
1143 \htXourse>
```

## 2.8.7 JavaScript

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

```
1144 \classXimera>
1145 \DefineVerbatimEnvironment{javascript}{Verbatim}{numbers=left,frame=lines,label=JavaScript,la
1146 \endclassXimera>

1147 \htXimera>
1148 % for programming javascript
1149 \renewenvironment{javascript}{\NoFonts}{\EndNoFonts}
1150 \ScriptEnv{javascript}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div c
1151 \htXimera>
```

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

```
1152 \classXimera>
1153 \def\js#1{\mbox{\texttt{\detokenize{#1}}}}
1154 \endclassXimera>

1155 \htXimera>
1156 \def\js#1{\stepcounter{identification}\HCode{<span class="inline-javascript" id="javascript\
1157 \htXimera>
```

## 2.9 SageMath support

Load SageTeX if it exists.

```
1158 \classXimera>
1159 \IfFileExists{sagetex.sty}{\RequirePackage{sagetex}}{}
1160 \endclassXimera>
```

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

```
1161 \classXimera>
1162 \DefineVerbatimEnvironment{sageCell}{Verbatim}{numbers=left,frame=lines,label=SAGE,labelposi
1163 \endclassXimera>
```

```

1164 <*htXimera>
1165 \renewenvironment{sageCell}{\NoFonts}{\EndNoFonts}
1166 \ScriptEnv{sageCell}{\ifvmode \IgnorePar\fi \EndP\HCode{<div class="sage"><script type="text/
1167 </htXimera>

sageOutput (env.)    Execute SageMath code and output the result.
1168 <*classXimera>
1169 \DefineVerbatimEnvironment{sageOutput}{Verbatim}{numbers=left,frame=lines,label=SAGE-Output,
1170 </classXimera>

1171 <*htXimera>
1172 \renewenvironment{sageOutput}{\NoFonts}{\EndNoFonts}
1173 \ScriptEnv{sageOutput}{\ifvmode \IgnorePar\fi \EndP\HCode{<div class="sageOutput"><script ty
1174 </htXimera>

sageSilent (env.)    Execute SageMath code without outputting the result.
1175 <*htXimera>
1176 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1177 \ifdefined\sagesilent
1178 \renewenvironment{sagesilent}{\NoFonts}{\EndNoFonts}
1179 \fi
1180 \ScriptEnv{sagesilent}{\ifvmode \IgnorePar\fi \EndP\HCode{<script type="text/sagemath">}\Htm
1181 </htXimera>

```

## 2.10 Answerables

### 2.10.1 Answers

```

\answer A math answer
1182 <*classXimera>
1183
1184 \ifdefined\HCode
1185 \newcommand{\recordvariable}[1]{
1186 \else
1187 \newwrite\idfile
1188 \immediate\openout\idfile=\jobname.ids
1189 \newcommand{\recordvariable}[1]{\ifthenelse{\equal{#1}{}}{\}\{\immediate\write\idfile{var #1};}
1190 \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
1191 \define@key{answer}{given}[true]{\def\ans@given{#1}}

Used for setting numeric answer tolerance for online student input.
1192 \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.
1193 \define@key{answer}{validator}{

Used for assigning a js ID to answer for dynamic code (eg validators).
1194 \define@key{answer}{id}{\def\ans@id{#1}}

Used to set anticipated input format; eg “string”.
1195 \define@key{answer}{format}{

Used to hide the answer input box on the web.
1196 \define@key{answer}{onlinenoinput}[false]{

Used to add a ‘show answer’ button to the answer blank.
1197 \define@key{answer}{onlineshowanswerbutton}[false]{

Set default values for \answer command key=value pairs. Default values are given = false.
1198 \setkeys{answer}{id=,given=false,onlinenoinput=false,onlineshowanswerbutton=false}

```

Basic code for `\answer`.

```

1199
1200 % Options for handout
1201 \newcommand{\answerFormatLength}{2cm}
1202
1203 \newcommand{\answerFormatDots}[1]{\ldots\ldots}
1204 \newcommand{\answerFormatLine}[1]{\protect\rule{\answerFormatLength}{0.4pt}}
1205 \newcommand{\answerFormatFlexibleLine}[1]{\protect\rule{\widthof{$#1$}*2}{0.4pt}}
1206 \newcommand{\answerFormatFlexibleBox}[1]{\fbox{\scalebox{2}{\phantom{$#1$}}}}
1207
1208 % options for default (i.e with answers filled in)
1209 \newcommand{\answerFormatPlain}[1]{\ensuremath{#1}}
1210 \newcommand{\answerFormatBlue}[1]{\color{blue}\ensuremath{#1}}
1211 \newcommand{\answerFormatBoxed}[1]{\fbox{\ensuremath{#1}}}
1212 \newcommand{\answerFormatBoxedGiven}[1]{\underset{\scriptstyle\mathrm{given}}{\fbox{\ensuremath{#1}}}}
1213
1214 % defaults for handout and default mode, and for \answer[given]
1215 \let\handoutAnswerFormat\answerFormatDots
1216 \let\defaultAnswerFormat\answerFormatBlue
1217 \let\givenAnswerFormat\answerFormatBoxedGiven
1218
1219 \newcommand{\answer}[2][{}]{%
1220   \ifmmode%
1221     \setkeys{answer}{#1}%
1222     \recordvariable{\ans@id}
1223     \ifthenelse{\boolean{\ans@given}}{
1224       {% Start then statement
1225         \ifhandout
1226           #2
1227         \else
1228           \givenAnswerFormat{#2} %% in case the argument helps formatting
1229         \fi
1230       }% End then statement
1231     }{% Start else statement
1232       \ifhandout
1233         \handoutAnswerFormat{#2} %% in case the argument helps formatting
1234       \else% show answer in box outside handout mode
1235         \defaultAnswerFormat{#2} %% in case the argument helps formatting
1236       \fi
1237     }% End else statement
1238   \else%
1239     \GenericError{\space\space\space\space}% Throw an error based on... something? -- Jason
1240     {Attempt to use \@backslashchar answer outside of math mode}
1241     {See https://github.com/ximeraProject/ximeraLatex for explanation.}
1242     {Need to use either inline or display math.}%
1243   \fi
1244 }
1245 \end{classXimera}

```

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

```

1246 \begin{classXimera}
1247 \renewcommand{\answer}[2][false]{\HCode{<span class="answer_respondable">}#2\HCode{</span>}}
1248
1249 \def\validator[#1]{\stepcounter{identification}\HCode{<div class="validator" id="validator">}#1\HCode{</div>}}
1250 \def\endvalidator{\HCode{</div>}}
1251
1252 \end{classXimera}

```

## 2.10.2 Multiple choice and the like

`multipleChoice (env.)` Multiple choice

```

1253 \begin{classXimera}

```

```

1254 % Jim: Originally this was \renewcommand{\theenumi}{$(\mathrm{\alph{enumi}})$}
1255 % but that breaks tex4ht because mathmode can only be processed by mathjax.
1256 % so now I made this just italicized.

```

### 2.10.3 Options

```

1257 \define@key{choice}{value}[]{\def\choice@value{#1}}

```

This flags the answer as the correct answer

```

1258 \define@boolkey{choice}{correct}[true]{\def\choice@correct{#1}}

```

Use an ID to refer to the choice.

```

1259 \define@key{multipleChoice}{id}{\def\mc@id{#1}}

```

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

```

1260 \define@key{otherchoice}{value}[]{\def\otherchoice@value{#1}}

```

```

1261 \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".

```

1262 \setkeys{choice}{correct=false,value=}

```

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

```

1263 \setkeys{multipleChoice}{id=}

```

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

```

1264 \setkeys{otherchoice}{correct=false,value=}

```

```

1265 \endclassXimera

```

### 2.10.4 Choices

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

```

1266 \beginXimera

```

```

1267 \newcommand{\choice}[2][]{%

```

```

1268 \setkeys{choice}{#1}%

```

```

1269 \item{#2}

```

```

1270 \ifthenelse{\boolean{\choice@correct}}{

```

```

1271   {% Begin then result

```

```

1272   \ifhandout% if it's a handout do nothing.

```

```

1273   \else% otherwise place a checkmark when you select the "correct choice"... maybe? -- Jason

```

```

1274     \,\checkmark\,\setkeys{choice}{correct=false}

```

```

1275   \fi

```

```

1276   }% End then result

```

```

1277   }% Begin/End else result.

```

```

1278 }

```

```

1279

```

```

1280 %Define an expandable version of choice Not really meant to be used outside this package (use

```

```

1281 % Is there a reason we can't just always use this as default? -- Jason

```

```

1282 \newcommand{\choiceEXP}[2][]{%

```

```

1283 \expandafter\setkeys\expandafter{choice}{#1}%

```

```

1284 \item{#2}

```

```

1285 \ifthenelse{\boolean{\choice@correct}}{

```

```

1286   {% Begin then result

```

```

1287   \ifhandout

```

```

1288   \else

```

```

1289     \,\checkmark\,\setkeys{choice}{correct=false}

```

```

1290   \fi

```

```

1291   }% End then result

```

```

1292   }% Begin/End else result.

```

```

1293 } %% note all the {} are needed in case the choice has [] in it.

```

```

1294

```

```

1295 % \otherchoice is the \choice used in wordChoice command.

```

```

1296 \newcommand{\otherchoice}[2][]{%

```

```

1297 \ignorespaces%

```

```

1298 \setkeys{otherchoice}{#1}%

```

```

1299 \ifthenelse{\boolean{\otherchoice@correct}}{

```

```

1300 {% Start then result
1301 #2\ignorespaces\setkeys{otherchoice}{correct=false}\ignorespaces%
1302 }% End then result
1303 {}% Start/End else result
1304 \ignorespaces%
1305 }%
1306 \newcommand{\inlinechoice}[2][{}]{%
1307 \setkeys{choice}{#1}%
1308 \iffirstinlinechoice
1309 (\hspace{-.25em}
1310 \firstinlinechoicetrue
1311 \else
1312 /
1313 \fi
1314 #2
1315 \ifthenelse{\boolean{choice@correct}}{%
1316 {% Start then result
1317 \ifhandout\else\checkmark\ignorespaces\setkeys{choice}{correct=false}\ignorespaces\fi%
1318 }% End then result
1319 {}% Start/End else result
1320 \hspace{-.25em}\ignorespaces%
1321 }
1322
1323 \end{classXimera}

```

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

```

1324 (*htXimera)
1325 \newcounter{choiceId}
1326 \renewcommand{\choice}[2][{}]{%
1327 \setkeys{choice}{correct=false}%
1328 \setkeys{choice}{#1}%
1329 \stepcounter{choiceId}\IgnorePar%
1330 \HCode{<span class="choice }%
1331 \ifthenelse{\boolean{choice@correct}}{\HCode{correct}}{}%
1332 \HCode{" }
1333 \ifthenelse{\equal{\choice@value}{}}{\HCode{data-value="\choice@value" }}%
1334 \HCode{id="choice\arabic{choiceId}">}%
1335 #2\HCode{</span>}}
1336 \let\inlinechoice\choice
1337 \end{htXimera}

```

### 2.10.5 Environment(s)

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

```

1338 (*classXimera)
1339 \newenvironment{multipleChoice}[1][{}]{%
1340 {% Environment Start Code
1341 \setkeys{multipleChoice}{#1}%
1342 \recordvariable{mc@id}%
1343 \begin{trivlist}
1344 \item[\hspace{\labelsep}\small\bfseries \GetTranslation{Multiple Choice}:]\hfil
1345 \begin{enumerate}
1346 }% Note this means that \item has to be the first line after \begin{multipleChoice}.
1347 {% Environment End Code
1348 \end{enumerate}
1349 \end{trivlist}
1350 }
1351
1352 %multipleChoice@ is for internal use only! (used in wordChoice)
1353 %this is simply a wrapper for the sole showing (other)choice.
1354 \newenvironment{multipleChoice@}[1][{}]{%
1355 \end{classXimera}

```

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

```

1356 (*htXimera)
1357 \renewenvironment{multipleChoice}[1] []
1358 {\setkeys{multipleChoice}{#1}%
1359 \stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div class="multiple-choice"
1360 \ifthenelse{\equal{\mc@id}{}}{\}\HCode{data-id="\mc@id" }}%
1361 \HCode{id="problem\arabic{identification}" titletext=" \GetTranslation{Multiple Choice}">}%
1362 }\HCode{</div>}\IgnoreIndent}
1363 \ConfigureEnv{multipleChoice}{\}\{\}\{\}
1364 </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.

```

1365 (*classXimera)
1366 \newcommand{\wordChoice}[1]{%
1367 \let\choicetemp\choice% Assign a "choicetemp" command to duplicate choice.
1368 \ifwordchoicegiven% If wordchoice option is on, we need to juggle around some definitions.
1369 \let\choice\otherchoice%
1370 %\begin{multipleChoice@}% -unnecessary (REMOVE THIS LINE IF THE YEAR IS 2019 or Beyond)
1371 #1
1372 %\end{multipleChoice@}% -unnecessary (REMOVE THIS LINE IF THE YEAR IS 2019 or Beyond)
1373 \else% If it isn't the regular "choice" command should work.
1374 \let\choice\inlinechoice%
1375 \begin{multipleChoice@}%
1376 #1%
1377 \end{multipleChoice@}%
1378 \fi%
1379 \let\choice\choicetemp% Now that choicetmp has been manipulated to what we want, replace choi
1380 }%
1381
1382
1383 </classXimera>

```

This is actually just word choice

```

1384 (*htXimera)
1385 \renewenvironment{multipleChoice@}{\refstepcounter{problem}}{\}%
1386 \ConfigureEnv{multipleChoice@}{\stepcounter{identification}\IgnorePar\HCode{<span class="word
1387 </htXimera>

```

## 2.12 Select all

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

```

1388 (*classXimera)
1389 \newenvironment{selectAll}[1] []
1390 {\begin{trivlist}\item[\hspace \labelsep\small\bfseries \GetTranslation{Select All Correct Ans
1391 {\end{enumerate}\end{trivlist}}
1392 </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`.

```

1393 (*htXimera)
1394 \renewenvironment{selectAll}{\refstepcounter{problem}}{\}%
1395 \ConfigureEnv{selectAll}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<div
1396 </htXimera>

```

### 2.12.1 Free response

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

```

1397 <*classXimera>
1398 \newboolean{given} %% required for freeResponse
1399 \setboolean{given}{true} %% could be replaced by a key=value pair later if needed
1400
1401 \ifhandout
1402 \newenvironment{freeResponse}[1][false]%
1403 {%
1404 \def\givenatend{\boolean{#1}}
1405 \ifthenelse{\boolean{#1}}
1406 {% Begin then result
1407 \begin{trivlist}
1408 \item
1409 }% End then result
1410 {% Begin else result
1411 \setbox0\vbox\bgroup
1412 }% End else result
1413 % {}% Don't think this is doing anything? -- Jason
1414 }
1415 {%
1416 \ifthenelse{\givenatend}
1417 {% Begin then result
1418 \end{trivlist}
1419 }% End then result
1420 {% Begin else result
1421 \egroup
1422 }% End else result
1423 % {}% Don't think this is doing anything? -- Jason
1424 }
1425 \else
1426 \newenvironment{freeResponse}[1][false]%
1427 {% Environment Beginning Code
1428 \ifthenelse{\boolean{#1}}%% Could probably change this with just putting the (given) in t
1429 {% Begin then result
1430 \begin{trivlist}
1431 \item[\hspace{1cm}\labelsep\bfseries \GetTranslation{Free Response (Given)}:\hspace{2ex}]
1432 }% End then result
1433 {% Begin else result
1434 \begin{trivlist}
1435 \item[\hspace{1cm}\labelsep\bfseries \GetTranslation{Free Response}:\hspace{2ex}]
1436 }% End else result
1437 }
1438 {% Environment Ending Code
1439 \end{trivlist}
1440 }
1441 \fi
1442
1443 </classXimera>
1444 <*htXimera>
1445
1446 \renewenvironment{freeResponse}{\refstepcounter{problem}}{}%
1447 \ConfigureEnv{freeResponse}{\stepcounter{identification}\ifvmode \IgnorePar\fi \EndP\HCode{<
1448
1449 </htXimera>

```

### 2.12.2 Feedback

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

Define a placeholder command for validator and feedback.

```
1450 <classXimera>
1451 \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.

```
1452 \newenvironment{validator}[1][]{
1453   \def\PH@Command{#1}% Use PH@Command to hold the content and be a target for "\expandafter"
1454   \mbox{\texttt{\detokenize\expandafter{\PH@Command}}}% Now expand PH@Command once and then d
1455 }{}

```

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

```
1456 \ifhandout%
1457 \newenvironment{feedback}
1458   {%
1459   \setbox0\vbox\bgroup
1460   }
1461   {%
1462   \egroup
1463   }

```

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.

```
1464 \else
1465 \newenvironment{feedback}[1][attempt]{
1466
1467 \edef\PH@Command{\GetTranslation{#1}}% Use PH@Command to hold the content and be a target for
1468
1469 \begin{trivlist}% Begin the trivlist to use formatting of the "Feedback" label.
1470 \item[\hskip \labelsep\small\slshape\bfseries \GetTranslation{Feedback}]% Format the "Feedback
1471 \ifonlineTF{% If the feedback is on a pdf, we don't need to detokenize - which messes with t
1472 (\texttt{\expandafter\detokenize\expandafter{\PH@Command}})}% Keep the online version the sa
1473 {(\expandafter\texttt{\PH@Command}}):% No need for detokenize in the pdf version
1474 \hspace{2ex}}\small\slshape% Insert some space before the actual feedback given.
1475 }{
1476 \end{trivlist}
1477 }
1478
1479 \fi
1480 </classXimera>

```

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

```
1481 <htXimera>
1482 \def\feedback{\@ifnextchar{\@feedbackcode}{\@feedbackattempt}}
1483 \def\@feedbackattempt{\@feedbackcode[attempt]}
1484 \def\@feedbackcode[#1]{\stepcounter{identification}%
1485 \ifvmode \IgnorePar\fi \EndP%
1486 \ifthenelse{equal{#1}{attempt}}{\HCode{<div class="feedback" data-feedback="attempt" id="fe
1487 \ifthenelse{equal{#1}{correct}}{\HCode{<div class="feedback" data-feedback="correct" id="f
1488 {\HCode{<div class="feedback" data-feedback="script" id="feedback\arabic{identification}" ti
1489 \def\endfeedback{\HCode{</div>}\IgnoreIndent}
1490 </htXimera>

```

### 2.12.3 Ungraded activities

`ungraded (env.)` The `ungraded` environment is used to record that certain parts of activities should not be worth points. For example, if you want to use a `multipleChoice` as a survey question,



you can place it inside an `ungraded` environment. On the L<sup>A</sup>T<sub>E</sub>X side, the `ungraded` environment does nothing.

```
1491 \classXimera
1492 \newenvironment{ungraded}{}{}
1493 \endclassXimera
```

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

```
1494 \htXimera
1495 \renewenvironment{ungraded}{%
1496 \ifmode \IgnorePar\fi \EndP\HCode{<div class="ungraded">}\IgnoreIndent%
1497 }{
1498 \ifmode \IgnorePar\fi \EndP\HCode{</div>}\IgnoreIndent%
1499 }
1500 \endhtXimera
```

## 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.

```
1501 \classXimera
1502 %% Pre-202412: .jax file written in non-\HCode, and in a next run inserted by ximera.cfg in
1503 %% Post-202501: .mjax file written only in \HCode, and in luaxake post-processing inserted in
1504 %% ( used luaxake rather than sed ...)
1505 \newwrite\myfile
1506 \ifdefined\HCode
1507 \immediate\openout\myfile=\jobname.xmjax
1508
1509 %% From |only.dtx| we must also create |prompt| on the MathJax side.
1510 \immediate\write\myfile{\unexpanded{\newenvironment}{prompt}}{}{}
1511
1512 %% Write all newcommands to .xmjax file, that will be included in the .html via luaxake
1513 \let\@oldargdef\@argdef
1514 \long\def\@argdef#1[#2]#3{%
1515 \immediate\write\myfile{\unexpanded{\newcommand}{#1}}[\unexpanded{#2}]{\unexpanded{#3}}%
1516 \@oldargdef#1[#2]#3}%
1517 }
1518
1519 %% Same for \DeclareMathOperator
1520 \let\@oldDeclareMathOperator\DeclareMathOperator
1521 \renewcommand{\DeclareMathOperator}[2]{\@oldDeclareMathOperator{#1}{#2}\immediate\write\myfile{\unexpanded{\newcommand}{#1}}[\unexpanded{#2}]{\unexpanded{#3}}%
1522 }
1523 \fi
1524
1525
1526 \endclassXimera
```

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

```
1527 \cfgXimera
1528
1529 % 202501: removed sed-manipulation of .jax file; see luaxake now
1530
1531 \Configure{BVerbatimInput}{}{}{}
1532
1533 \Configure{verbatiminput}{}{}{}
1534
1535 % Instead of a nonbreaking space, use a standard space
1536 \makeatletter
1537 \def\FV@Space{\space}
1538 \makeatother
1539
1540 % Include the (problem-?) .ids in a text/javascript script right at the beginning of the body
```

```

1541 \Configure{BODY}{%
1542 \HCode{<body>\Hnewline}%
1543 \Tg<div class="preamble">%
1544 %% 202501: removed .jax inclusion (see luaxake)
1545
1546 %% Include the .ids file
1547 \IfFileExists{\jobname.ids}{\HCode{<script type="text/javascript">\Hnewline}%
1548 \BVerbatimInput{\jobname.ids}%
1549 \HCode{</script>\Hnewline}%
1550 }{}
1551 \Tg</div>%
1552 }{%
1553 \ifvmode\IgnorePar\fi\EndP\HCode{</body>\Hnewline}%
1554 }
1555
1556 % 202501: removed 'prevent spaces as in "\begin {align}": this is done in luaxake now
1557
1558 % This is a fix for the LAODE book, which uses matlabEquation as if it were an equation
1559 \ScriptEnv{matlabEquation}{\ifvmode \IgnorePar\fi \EndP\HCode{<script type="math/tex; mode=d
1560
1561 \</cfgXimera>

```

### 2.13.2 Semantic HTML

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

```

1562 \<cfgXimera>
1563 \Configure{textbf}{\ifvmode\ShowPar\fi\HCode{<strong>}}{\HCode{</strong>}}
1564 \</cfgXimera>

```

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

```

1565 \<cfgXimera>
1566 \Configure{textit}{\ifvmode\ShowPar\fi\HCode{<em>}}{\HCode{</em>}}
1567 \Configure{emph}{\ifvmode\ShowPar\fi\HCode{<em>}}{\HCode{</em>}}
1568 \</cfgXimera>

```

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

```

1569 \<cfgXimera>
1570 \Configure{texttt}{\ifvmode\ShowPar\fi\HCode{<code>}}{\HCode{</code>}}
1571 \</cfgXimera>

```

## 2.14 Tools

### 2.14.1 Suppress

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

```

1572 \<classXimera>
1573 \font\dummyft@=dummy \relax
1574 \def\suppress{%
1575   \begingroup\par
1576   \parskip\z@
1577   \offinterlineskip
1578   \baselineskip=\z@skip
1579   \lineskip=\z@skip
1580   \lineskiplimit=\maxdimen
1581   \dummyft@
1582   \count@\sixt@@n
1583   \loop\ifnum\count@ >\z@
1584     \advance\count@\m@ne
1585     \textfont\count@\dummyft@
1586     \scriptfont\count@\dummyft@
1587     \scriptscriptfont\count@\dummyft@
1588   \repeat

```

```

1589 \let\selectfont\relax
1590 \let\mathversion\@gobble
1591 \let\getanddefine@fonts\@gobbletwo
1592 \tracinglostchars\z@
1593 \frenchspacing
1594 \hbadness\@M}
1595 \def\endsuppress{\par\endgroup}
1596 \endclassXimera

```

### 2.14.2 The End

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

```

1597 \endhtXimera}
1598 \Hinput{ximera}
1599 \endhtXimera}

1600 \endhtXourse}
1601 \Hinput{xourse}
1602 \endhtXourse}

1603 \endcfgXimera}
1604 \begin{document}
1605 \EndPreamble
1606 \endcfgXimera}

```

## 3 xourse.cls

```

1607 \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.

```

1608 \newif\ifnotoc
1609 \notocfalse
1610 \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.

```

1611 \newif\ifnonewpage
1612 \nonewpagefalse
1613 \DeclareOption{nonewpage}{\nonewpagetrue}

1614 \DeclareOption*{\PassOptionsToClass{\CurrentOption}{ximera}}
1615 \ProcessOptions\relax
1616 \LoadClass{ximera}
1617 % \begin{macrocode}
1618 \endclassXourse}

```

### 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**.

```

1619 \classXourse}
1620 \newcommand{\skip@preamble}{%
1621   \let\document\relax\let\enddocument\relax%
1622   \newenvironment{document}{\let\input\otherinput}{}%
1623   \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:

```

1624 \let\otherinput\input

```

```

Store usual \maketitle as \othermaketitle
1625 \let\othermaketitle\maketitle
\maketitle In a xourse file, \maketitle is redefined to give course packet title page and toc.
1626 \renewcommand{\maketitle}{%
1627 \pagestyle{empty}
1628 \begin{center}
1629 ~\ \ %puts space at top of page to move title down.
1630 \vskip .25\textheight
1631 \hrulefill\
1632 \vskip 1em
1633 \bfseries{\Huge \@title} \
1634 \hrulefill\
1635 \vskip 3em
1636 {\Large \@author}
1637 \vskip 2em
1638 {\large \@date}
1639 \end{center}
1640 \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.
1641 \ifnotoc
1642 \else
1643 \tableofcontents\clearpage
1644 \clearpage
1645 \fi

Switch to main pagestyle, just like a document with documentclass ximera.
1646 \pagestyle{main}

Renew maketitle to usual definition.
1647 \let\maketitle\othermaketitle

And we finish with our redefinition of \maketitle.
1648 }
1649 \relax
1650 \end{classXourse}

```

### 3.1.1 Regular activities

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

```

1651 \begin{classXourse}
1652 \ifnonepage
1653 \newcommand{\activity}[2][{}]{%
1654 \setkeys{activity}{#1}
1655 \renewcommand{\input}[1]{%
1656 \begin{group}\skip@preamble\otherinput{#2}\end{group}\par\vspace{\topsep}
1657 \let\input\otherinput}
1658 \else
1659 \newcommand{\activity}[2][{}]{%
1660 \setkeys{activity}{#1}
1661 \renewcommand{\input}[1]{%
1662 \begin{group}\skip@preamble\otherinput{#2}\end{group}\clearpage
1663 \let\input\otherinput}
1664 \fi
1665 \relax
1666 \end{classXourse}

```

```

1667 <*htXourse>
1668 \renewcommand\activity[2][]{%
1669 \ifvmode \IgnorePar\fi \EndP\HCode{<a class="activity card \activitystyle" href="#2" data-op
1670 }
1671 </htXourse>

```

When running xake, we can just ignore activities

```

1672 <*classXourse>
1673 \ifxake
1674 \renewcommand\activity[2][]{%
1675 \fi
1676 </classXourse>

```

### 3.1.2 Practice activities

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

```

1677 <*classXourse>
1678 \ifhandout
1679 \newcommand{\practice}[2][]{%
1680 \setkeys{practice}{#1}%!!!!
1681 \renewcommand{\input}[1]{%
1682 \begingroup\skip@preamble\otherinput{#2}\endgroup
1683 \let\input\otherinput}%
1684 \else
1685 \newcommand{\practice}[2][]{\texttt{\detokenize{#2}}}% gives file name for practice
1686 \setkeys{practice}{#1}%!!!!
1687 \renewcommand{\input}[1]{%
1688 \begingroup\skip@preamble\otherinput{#2}\endgroup
1689 \let\input\otherinput}%
1690 \fi
1691 \relax
1692 </classXourse>

```

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

```

1693 <*classXourse>
1694 \ifxake
1695 \renewcommand\practice[2][]{%
1696 \fi
1697 </classXourse>

```

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

```

1698 <*htXourse>
1699 \renewcommand\practice[2][]{%
1700 \ifvmode\IgnorePar\fi\EndP%
1701 \HCode{<a class="activity card practice" href="#2" data-options="#1">#2</a>}%
1702 \IgnoreIndent%
1703 }
1704 </htXourse>

```

## 3.2 Sectioning

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

```

1705 <*classXourse>
1706 \renewcommand*\l@section{\@dottedtocline{1}{1.5em}{4.2em}}
1707 </classXourse>

```

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

```

1708 <*classXourse>
1709 \renewcommand*\l@subsection{\@dottedtocline{2}{3.8em}{4.2em}}
1710 </classXourse>

```

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

```

1711 <*htXourse>
1712 \newcounter{ximera@part}
1713 \setcounter{ximera@part}{0}
1714 \renewcommand\part[1]{%
1715 \stepcounter{ximera@part}%
1716 \ifvmode \IgnorePar\fi \EndP%
1717 %\HCode{<h1 id="part\arabic{ximera@part}" class="card part">}#1\HCode{</h1>}}% makes cards di
1718 \HCode{<h1 id="part\arabic{ximera@part}" class="card part">}#1</h1>}}%
1719 \IgnoreIndent%
1720 }
1721 </htXourse>

```

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

```

1722 <*cfgXimera>
1723 \renewcommand{\paragraph}[1]{%
1724 \HCode{<span class="paragraphHead">}}%
1725 #1%
1726 \HCode{</span>}\par\IgnorePar}
1727 </cfgXimera>

```

`\subparagraph` An even smaller heading.

```

1728 <*cfgXimera>
1729 \renewcommand{\subparagraph}[1]{%
1730 \HCode{<span class="subparagraphHead">}}%
1731 #1%
1732 \HCode{</span>}\par\IgnorePar}
1733 </cfgXimera>

```

### 3.3 Grading by points

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

```

1734 <*classXourse>
1735 \newenvironment{graded}[1]{\{}
1736 </classXourse>

```

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

```

1737 <*htXourse>
1738 \renewenvironment{graded}[1]{%
1739 \ifvmode \IgnorePar\fi \EndP\HCode{<div class="graded" data-weight="#1">}\IgnoreIndent%
1740 }{
1741 \ifvmode \IgnorePar\fi \EndP\HCode{</div>}\IgnoreIndent%
1742 }
1743 </htXourse>

```

### 3.4 Logos

`\logo` A logo for the xourse.

```

1744 <*classXourse>
1745 \newcommand*{\logo}[1]{%
1746 \ifx\@onlypreamble\@notprerr
1747 \ClassError{xourse}{logo can only be used in the preamble}
1748 {Move your logo command to the preamble}
1749 \else %
1750 \IfFileExists{#1}%
1751 {\gdef\xourse@logo{#1}}%
1752 {\ClassError{xourse}{logo file does not exist}
1753 {To use logo, make sure that the referenced image file exists}}%
1754 \fi%
1755 }
1756
1757 </classXourse>

```

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

```
1758 <*htXourse>
1759 \Configure{@HEAD}{%
1760   \HCode{<meta name="og:image" content="}%
1761   \ifdefined\xourse@logo%
1762     \xourse@logo%
1763   \fi%
1764   \HCode{" />\Hnewline}}%
1765 </htXourse>
```