



# The **lwarp** package

v0.20 — 2017/02/09

L<sup>A</sup>T<sub>E</sub>X to HTML5

© 2016 Brian Dunn  
bd@BDTechConcepts.com

## Abstract

The **lwarp** package causes L<sup>A</sup>T<sub>E</sub>X to directly produce HTML5 output, using external utility programs only for the final conversion of text and images. Math may be represented by SVG files or MathJax.

Documents may be produced by L<sup>A</sup>T<sub>E</sub>X, LuaL<sup>A</sup>T<sub>E</sub>X, or XeL<sup>A</sup>T<sub>E</sub>X. A TexLua script removes the need for system utilities such as **make** and **gawk**, and also supports **xindy** and **latexmk**. Configuration is automatic at the first manual compile.

Print and HTML versions of each document may coexist, each with its own set of auxiliary files. Support files are self-generated on request.

A modular package-loading system uses the **lwarp** version of a package for HTML when available. Several dozen L<sup>A</sup>T<sub>E</sub>X packages are supported with these high-level source compatibility replacements.

A tutorial is provided to quickly introduce the user to the major components of the package.

To update existing projects, see [section 1, Updates](#).

**Note that this is still a “beta” version of **lwarp**, and some things may change in response to user feedback and further project development.**

# Contents

<b>1</b>	<b>lwarp.sty</b>	<b>14</b>
<b>1</b>	<b>Updates</b>	<b>14</b>
<b>2</b>	<b>Introduction</b>	<b>16</b>
2.1	Supported packages and features . . . . .	17
<b>3</b>	<b>Alternatives</b>	<b>22</b>
3.1	TeX4ht . . . . .	22
3.2	Translators . . . . .	22
3.3	AsciiDoc . . . . .	22
3.4	Pandoc . . . . .	23
3.5	Word Processors . . . . .	23
3.6	Commercial Systems . . . . .	23
3.7	Comparisons . . . . .	23
<b>4</b>	<b>Installation</b>	<b>25</b>
4.1	Installing the lwarp package . . . . .	26
4.2	Installing the lwarpmk utility . . . . .	27
4.2.1	Using a local copy of lwarpmk . . . . .	28
4.3	Installing additional utilities . . . . .	29
<b>5</b>	<b>Tutorial</b>	<b>30</b>
5.1	Starting a new project . . . . .	30
5.2	Compiling the print version with lwarpmk . . . . .	34
5.3	Compiling the HTML version with lwarpmk . . . . .	35
5.4	Generating the SVG images . . . . .	36
5.5	Using MathJax for math . . . . .	37
5.6	Changing the CSS style . . . . .	38
5.7	Customizing the HTML output . . . . .	39
5.8	Using latexmk . . . . .	43
5.9	Using XeLaTeX or LuaLaTeX . . . . .	44
5.10	Cleaning auxiliary files . . . . .	45
5.11	Cleaning auxiliary and output files . . . . .	45
5.12	Processing multiple projects in the same directory . . . . .	45
5.13	Using the make utility . . . . .	45
<b>6</b>	<b>Additional details</b>	<b>46</b>
6.1	Font and UTF-8 support . . . . .	46
6.2	lwarp package loading and options . . . . .	47
6.3	Selecting the operating system . . . . .	47
6.4	Selecting actions for print or HTML output . . . . .	47
6.5	Commands to be placed into the warpprint environment . . . . .	49
6.6	Commands for a successful HTML conversion . . . . .	49

6.7	Title page . . . . .	50
6.8	HTML page meta descriptions . . . . .	51
6.9	CSS . . . . .	51
6.10	Special cases and limitations . . . . .	52
6.10.1	Text formatting . . . . .	52
6.10.2	Verse and Memoir packages . . . . .	52
6.10.3	Math . . . . .	53
6.10.4	Graphics . . . . .	55
6.10.5	Xcolor . . . . .	55
6.10.6	Tikz . . . . .	55
6.10.7	Tabular . . . . .	56
6.10.8	Minipages . . . . .	57
6.10.9	Mdframed . . . . .	57
6.10.10	float, trivfloat, and/or algorithmicx together . . . . .	58
6.10.11	caption and subcaption packages . . . . .	58
6.10.12	subfig package . . . . .	58
6.10.13	floatrow package . . . . .	58
6.10.14	siunitx package . . . . .	59
6.10.15	newclude package . . . . .	59
6.10.16	newtxmath package . . . . .	59
6.10.17	babel package . . . . .	59
6.10.18	enumitem package . . . . .	60
<b>7</b>	<b>Expanding the code base</b>	<b>60</b>
7.1	Creating an lwarp version of a package . . . . .	60
<b>8</b>	<b>Troubleshooting</b>	<b>61</b>
8.1	Using the <code>lwarp.sty</code> package . . . . .	61
8.1.1	Debug tracing output . . . . .	63
8.2	Compiling the <code>lwarp.dtx</code> file . . . . .	63
<b>9</b>	<b>Implementation</b>	<b>63</b>
<b>10</b>	<b>Stack depths</b>	<b>64</b>
<b>11</b>	<b>Source Code</b>	<b>66</b>
<b>12</b>	<b>Detecting the T<sub>E</sub>X Engine — pdf<sub>l</sub>atex, lua<sub>l</sub>atex, xel<sub>l</sub>atex</b>	<b>67</b>
<b>13</b>	<b>Unicode Input Characters</b>	<b>67</b>
<b>14</b>	<b>Handling package options</b>	<b>67</b>
<b>15</b>	<b>Required packages</b>	<b>70</b>
<b>16</b>	<b>Loading Packages</b>	<b>75</b>

---

<b>17</b>	<b>Copying a file</b>	<b>77</b>
<b>18</b>	<b>Debugging messages</b>	<b>78</b>
<b>19</b>	<b>Remembering original formatting macros</b>	<b>79</b>
<b>20</b>	<b>Operating-System portability</b>	<b>80</b>
20.1	Common portability code . . . . .	80
20.2	Unix, Linux, and Mac OS . . . . .	80
20.3	MS-Windows . . . . .	81
<b>21</b>	<b>Stacks</b>	<b>81</b>
21.1	Assigning depths . . . . .	82
21.2	Closing actions . . . . .	82
21.3	Closing depths . . . . .	83
21.4	Pushing and popping the stack . . . . .	83
<b>22</b>	<b>Data arrays</b>	<b>85</b>
<b>23</b>	<b>HTML entities</b>	<b>85</b>
<b>24</b>	<b>HTML filename generation</b>	<b>86</b>
<b>25</b>	<b>Homepage link</b>	<b>88</b>
<b>26</b>	<b>\PrintStack diagnostic tool</b>	<b>89</b>
<b>27</b>	<b>Closing stack levels</b>	<b>89</b>
<b>28</b>	<b>HTML tags, spans, divs, elements</b>	<b>90</b>
28.1	Mapping L <sup>A</sup> T <sub>E</sub> X Sections to HTML Sections . . . . .	90
28.2	HTML tags . . . . .	91
28.3	Block tags and comments . . . . .	92
28.4	Div class and element class . . . . .	93
28.5	Single-line elements . . . . .	94
28.6	HTML5 semantic elements . . . . .	95
28.7	High-level block and inline classes . . . . .	95
28.8	Closing HTML tags . . . . .	96
<b>29</b>	<b>Paragraph handling</b>	<b>97</b>
<b>30</b>	<b>Paragraph start/stop handling</b>	<b>100</b>
<b>31</b>	<b>Page headers and footers</b>	<b>103</b>
<b>32</b>	<b>CSS</b>	<b>104</b>
<b>33</b>	<b>HTML meta description</b>	<b>104</b>

<b>34</b>	<b>Footnotes</b>	<b>105</b>
34.1	Regular page footnotes . . . . .	105
34.2	Minipage footnotes . . . . .	105
34.3	Titlepage thanks . . . . .	105
34.4	Regular page footnote implementation . . . . .	106
<b>35</b>	<b>Marginpars</b>	<b>107</b>
<b>36</b>	<b>Splitting HTML files</b>	<b>108</b>
<b>37</b>	<b>Sectioning</b>	<b>112</b>
37.1	Book class commands . . . . .	113
37.2	Sectioning support macros . . . . .	113
37.3	\section and friends . . . . .	118
<b>38</b>	<b>Starting a new file</b>	<b>120</b>
<b>39</b>	<b>Starting HTML output</b>	<b>122</b>
<b>40</b>	<b>Ending HTML output</b>	<b>125</b>
<b>41</b>	<b>Titles and the titling package</b>	<b>127</b>
41.1	Setting the title, etc. . . . .	128
41.2	Changes to HTML titlepage and titlingpage . . . . .	130
41.3	Printing the title, etc. in HTML . . . . .	131
41.4	Printing the title, etc. in print form . . . . .	134
41.5	\maketitle for print output . . . . .	135
41.6	\maketitle for HTML output . . . . .	137
<b>42</b>	<b>Abstract</b>	<b>140</b>
<b>43</b>	<b>Quote and verse</b>	<b>141</b>
43.1	Citations and attributions . . . . .	141
43.2	Quotes, quotations . . . . .	141
43.3	Verse . . . . .	142
<b>44</b>	<b>Verbatim</b>	<b>144</b>
<b>45</b>	<b>Fancyvrb</b>	<b>144</b>
<b>46</b>	<b>Theorems</b>	<b>150</b>
<b>47</b>	<b>Lists</b>	<b>151</b>
47.1	Itemize . . . . .	151
47.2	Enumerate . . . . .	152
47.3	Description . . . . .	153
<b>48</b>	<b>Tabular</b>	<b>154</b>

48.1	Token lookahead	155
48.2	Booleans	156
48.3	Handling &	157
48.4	Handling \\	159
48.5	Variables	159
48.6	Parsing @, >, <, ! columns	160
48.7	Parsing 'l', 'c', or 'r' columns	162
48.8	Parsing 'p', 'm', or 'b' columns	163
48.9	Parsing the column specifications	163
48.10	Starting a new row	166
48.11	Data opening tag	167
48.12	Midrules	168
48.13	Multicolumns	169
48.13.1	Parsing multicolumns	169
48.13.2	High-level multicolumn interface	171
48.13.3	Longtable captions	172
48.13.4	\tabledatamulticolumntag	174
48.14	Multirow	175
48.15	Utility macros inside a table	175
48.16	Checking for a new table cell	175
48.17	\mrowcell	177
48.18	New \tabular definition	177
48.19	Array	179
<b>49</b>	<b>Cross-references</b>	<b>180</b>
49.1	Setup	180
49.2	Zref setup	182
49.3	Labels	183
49.4	References	185
49.5	Hyper-references	187
<b>50</b>	<b>Floats</b>	<b>189</b>
50.1	Float captions	190
50.1.1	Caption inside a float environment	191
50.1.2	Caption and LOF linking and tracking	191
<b>51</b>	<b>Table of Contents, LOF, LOT</b>	<b>194</b>
51.1	Reading and printing the TOC	195
51.2	High-level TOC commands	197
51.3	Side TOC	197
51.4	Low-level TOC line formatting	199
<b>52</b>	<b>Index</b>	<b>200</b>
<b>53</b>	<b>Math</b>	<b>202</b>
53.1	Inline and display math	205

53.2	MathJax support . . . . .	206
53.3	Equation environment . . . . .	209
53.4	AMS Math environments . . . . .	211
53.4.1	Support macros . . . . .	211
53.4.2	Environment patches . . . . .	212
<b>54</b>	<b>Lateximages</b>	<b>216</b>
<b>55</b>	<b>center, flushleft, flushright</b>	<b>222</b>
<b>56</b>	<b>Siunitx</b>	<b>223</b>
<b>57</b>	<b>Graphics</b>	<b>223</b>
57.1	\graphicspath . . . . .	224
57.2	Length conversions and graphics options . . . . .	225
57.3	\includegraphics . . . . .	228
57.4	\rotatebox, \scalebox, \reflectbox . . . . .	231
57.5	Null functions . . . . .	234
<b>58</b>	<b>Cleverref</b>	<b>234</b>
<b>59</b>	<b>Picture</b>	<b>236</b>
<b>60</b>	<b>Tikzpicture</b>	<b>236</b>
<b>61</b>	<b>Boxes and Minipages</b>	<b>237</b>
61.1	Counters and lengths . . . . .	238
61.2	Footnote handling . . . . .	238
61.3	Minipage handling . . . . .	239
61.4	Parbox, makebox, framebox, fbox, raisebox . . . . .	243
<b>62</b>	<b>Direct formatting</b>	<b>244</b>
<b>63</b>	<b>Skips, spaces, font sizes</b>	<b>247</b>
<b>64</b>	<b>\phantomsection</b>	<b>253</b>
<b>65</b>	<b>\LaTeX and other logos</b>	<b>254</b>
65.1	HTML logos . . . . .	254
65.2	Print logos . . . . .	256
<b>66</b>	<b>\AtBeginDocument, \AtEndDocument</b>	<b>256</b>
<b>67</b>	<b>lwarpmk.conf</b>	<b>257</b>
<b>68</b>	<b>Trademarks</b>	<b>258</b>

<b>2</b>	<b>lwarp-newproject.sty</b>	<b>259</b>
<b>69</b>	<b>lwarp-newproject</b>	<b>259</b>
69.1	project_html.tex . . . . .	261
69.2	project.lwarpmkconf . . . . .	261
69.3	lwarp.css . . . . .	262
69.4	lwarp_sagebrush.css . . . . .	282
69.5	lwarp_formal.css . . . . .	287
69.6	sample_project.css . . . . .	291
69.7	lwarp_html.xdy . . . . .	292
69.8	lwarp_mathjax.txt . . . . .	292
69.9	Lwarpmk option . . . . .	294
<b>3</b>	<b>lwarp-afterpage.sty</b>	<b>302</b>
<b>70</b>	<b>Afterpage</b>	<b>302</b>
<b>4</b>	<b>lwarp-algorithmicx.sty</b>	<b>303</b>
<b>71</b>	<b>Algorithmicx</b>	<b>303</b>
<b>5</b>	<b>lwarp-alltt.sty</b>	<b>303</b>
<b>72</b>	<b>Alltt</b>	<b>304</b>
<b>6</b>	<b>lwarp-bookmark.sty</b>	<b>305</b>
<b>73</b>	<b>Bookmark</b>	<b>305</b>
<b>7</b>	<b>lwarp-booktabs.sty</b>	<b>306</b>
<b>74</b>	<b>Booktabs</b>	<b>306</b>
<b>8</b>	<b>lwarp-draftwatermark.sty</b>	<b>306</b>
<b>75</b>	<b>Draftwatermark</b>	<b>306</b>
<b>9</b>	<b>lwarp-epigraph.sty</b>	<b>306</b>
<b>76</b>	<b>Epigraph</b>	<b>307</b>



---

<b>10</b>	<b>lwarp-eso-pic.sty</b>	<b>308</b>
77	Eso-pic	308
<b>11</b>	<b>lwarp-everypage.sty</b>	<b>308</b>
78	Everypage	309
<b>12</b>	<b>lwarp-extramarks.sty</b>	<b>309</b>
79	Extramarks	309
<b>13</b>	<b>lwarp-fancyhdr.sty</b>	<b>309</b>
80	Fancyhdr	310
<b>14</b>	<b>lwarp-float.sty</b>	<b>311</b>
81	Float and \newfloat	311
<b>15</b>	<b>lwarp-floatrow.sty</b>	<b>313</b>
82	Floatrow	313
<b>16</b>	<b>lwarp-geometry.sty</b>	<b>317</b>
83	Geometry	317
<b>17</b>	<b>lwarp-graphics.sty</b>	<b>317</b>
84	Graphics	317
<b>18</b>	<b>lwarp-graphicx.sty</b>	<b>319</b>
85	Graphicx	319
<b>19</b>	<b>lwarp-hyperref.sty</b>	<b>320</b>
86	Hyperref	320

---

<b>20</b>	<b>lwarp-keyfloat.sty</b>	<b>323</b>
87	Keyfloat	323
<b>21</b>	<b>lwarp-letterspace.sty</b>	<b>324</b>
88	Letterspace	324
<b>22</b>	<b>lwarp-listings.sty</b>	<b>325</b>
89	Listings	325
<b>23</b>	<b>lwarp-longtable.sty</b>	<b>329</b>
90	Longtable	329
<b>24</b>	<b>lwarp-ltcaption.sty</b>	<b>331</b>
91	Ltcaption	331
<b>25</b>	<b>lwarp-mdframed.sty</b>	<b>332</b>
92	Mdframed	332
<b>26</b>	<b>lwarp-microtype.sty</b>	<b>339</b>
93	Microtype	339
<b>27</b>	<b>lwarp-multicol.sty</b>	<b>340</b>
94	Multicol	340
<b>28</b>	<b>lwarp-multirow.sty</b>	<b>342</b>
95	Multirow	342
<b>29</b>	<b>lwarp-needspace.sty</b>	<b>342</b>
96	Needspace	342

---

<b>30</b>	<b>lwarp-nowidow.sty</b>	<b>342</b>
97	Nowidow	342
<b>31</b>	<b>lwarp-pagenote.sty</b>	<b>343</b>
98	Pagenote	343
<b>32</b>	<b>lwarp-placeins.sty</b>	<b>344</b>
99	Placeins	344
<b>33</b>	<b>lwarp-ragged2e.sty</b>	<b>345</b>
100	Ragged2e	345
<b>34</b>	<b>lwarp-rotating.sty</b>	<b>346</b>
101	Rotating	346
<b>35</b>	<b>lwarp-setspace.sty</b>	<b>346</b>
102	Setspace	346
<b>36</b>	<b>lwarp-textpos.sty</b>	<b>347</b>
103	Textpos	347
<b>37</b>	<b>lwarp-threeparttable.sty</b>	<b>349</b>
104	Threeparttable	349
<b>38</b>	<b>lwarp-titleps.sty</b>	<b>350</b>
105	Titleps	350
<b>39</b>	<b>lwarp-titlesec.sty</b>	<b>352</b>
106	Titlesec	352

<b>40</b>	<b>lwarp-titletoc.sty</b>	<b>353</b>
107	Titletoc	353
<b>41</b>	<b>lwarp-titling.sty</b>	<b>354</b>
108	Titling	354
<b>42</b>	<b>lwarp-tocloft.sty</b>	<b>355</b>
109	Tocloft	355
<b>43</b>	<b>lwarp-trivfloat.sty</b>	<b>360</b>
110	Trivfloat	360
110.1	Combining <code>\newfloat</code> , <code>\trivfloat</code> , and <code>algorithmicx</code> . . . . .	360
<b>44</b>	<b>lwarp-wallpaper.sty</b>	<b>361</b>
111	Wallpaper	361
<b>45</b>	<b>lwarp-wrapfig.sty</b>	<b>362</b>
112	Wrapfig	362
<b>46</b>	<b>lwarp-xcolor.sty</b>	<b>364</b>
113	Xcolor	364
<b>47</b>	<b>lwarp-xfrac.sty</b>	<b>367</b>
114	Xfrac	367
	<b>Change History and Index</b>	<b>370</b>

## List of Figures

1	tutorial.tex listing . . . . .	31
---	--------------------------------	----

## List of Tables

1	L <sup>A</sup> T <sub>E</sub> X–HTML generation — lwarp package — Supported functions	18
2	Required software programs . . . . .	25
3	Files created in a new project . . . . .	33
4	Section depths and HTML headings . . . . .	65
5	Tabular baseline . . . . .	163
6	Tabular HTML column conversions . . . . .	164
7	Cross-referencing data structures . . . . .	181
8	Float data structures . . . . .	189

## Package 1

# lwarp.sty

## 1 Updates

For a detailed list of changes, see the Change History on page 370.

### v0.20:

- The `makefile` and related infrastructure has been replaced by the `lwarpmk` utility. This provides increased portability, reduced dependencies, and much simpler installation and setup.
- The `lwarp-newproject` package is now used to locally create support files.
- The print and HTML versions of a document may co-exist with their own sets of auxiliary files.
- Package handling is now controlled by a modular system which looks for and loads an `lwarp-<package>` version if available.
- High-level source compatibility is provided for all supported packages, almost totally eliminating the need for `warpprint` and `warpHTML` environments.
- A large number of additional packages are supported.
- A new tutorial is included in the documentation, and many obsolete sections have been removed.
- `\NewHTMLdescription` sets the HTML meta description tag for each file. See section 6.8.
- `\HTMLFileName` may now be empty, allowing filenames without a prefix. Lwarp no longer automatically appends a `-` character. For existing projects, add a `-` to the end of `\HTMLFileName`.
- `\HomeHTMLFileName` and `\HTMLFileName` no longer use escaped underscore characters. Underscores may be used in filenames as-is.
- `lwarp` now tries to auto-detect the operating system, and `\warpOSwindows` is only needed if the auto-detection fails to detect Windows.
- Tabular column types `@`, `>`, and `<` are now supported.
- `BlockClass` and `\InlineClass` add an optional style.
- The `sidebar` and `example` environments have been moved to the test suite, and are no longer included in `lwarp`.



**v0.19:**

- MATHJAX now may be used to display math via the `mathjax` option. See sections 6.2 and 6.10.3. To use MathJax with a pre-existing project, copy or link the file `lwarp_mathjax.txt` to the project's directory.
- `\rule` added, supporting width, height, raise, `\textcolor`.
- `\LateximageFontSizeName` provides user-adjustable font size for math and `lateximages`.
- `\minipagefullwidth` requests that the next minipage be full-width in HTML, but still the assigned width in print.
- `minipage` improved side-by-side rendering.
- CSS class `tablenotes` is provided for table note items.
- `\warpprintonly` replaces `\rowprintedonly`, and `\warpHTMLonly` is added. These behave like the `warpprint` and `warpHTML` environments, and are generally useful, so they replace the previously table-specific syntax.
- `cleveref` is loaded `\AtEndPreamble` for improved reliability. See section 58.
- `\xfracHTMLfontsize` controls `xfrac` font size in HTML.
- `Tikz` improved catcode handling.

**v0.18:**

- The `verse` package and the verse-related commands from the `memoir` package are now supported.
- Responsive web design has been improved for the sideTOC.
- `\includegraphics` now maintains relative sized for `em`, `ex`, and `%`.

**v0.17:**

- `mdframed` package is supported.

**v0.16:**

- Font and input encoding are now controlled by the user, and `lwarp` is loaded after fonts have been selected.
- Support for  $\text{\LaTeX}$  and  $\text{\LuaTeX}$ . See section 6.1.

## 2 Introduction

The `lwarp` project aims to allow a rich  $\text{\LaTeX}$  document to be converted to a reasonable HTML interpretation. No attempt has been made to force  $\text{\LaTeX}$  to provide for every HTML-related possibility, and HTML cannot exactly render every possible  $\text{\LaTeX}$  concept. Where compromise is necessary, it is desirable to allow the print output to remain typographically rich, and compromise only in the HTML conversion.

Several “modern” features of HTML5, CSS3, and SVG are employed to allow a fairly feature-rich document without relying on the use of Javascript. Limited testing on older browsers show that these new features degrade gracefully, although the SVG format for math may not be available on small cell phones.

`pdflatex`, `xelatex`, or `lualatex` is used, allowing `lwarp` to process the usual image formats. While generating HTML output, SVG files are used in placed of PDF. Other formats such as JPG are used as-is.

SVG images may be used for math, and are also used for `picture`, and `Tikz` environments, as this format has better browser and e-book support than MathML (as of this writing), while still allowing for the high-quality display and printing of images (again, subject to potentially bug-ridden<sup>1</sup> browser support).<sup>2</sup> Further, SVG images allow math to be presented with the same precise formatting as in the print version. Math is accompanied by ALT tags holding the  $\text{\LaTeX}$  source for the expression, allowing it to be copy/pasted into other documents.<sup>3</sup> Custom  $\text{\LaTeX}$  macros may be used as-is in math expressions, since the math is evaluated entirely inside  $\text{\LaTeX}$ .

The MATHJAX JavaScript display engine may be selected for math display instead of using SVG images. Subject to browser support and Internet access, MathJax allows an HTML page to display math without relying on a large number of external image files, one per math expression. `lwarp` maintains  $\text{\LaTeX}$  control for cross-referencing and equation numbering/formatting.

---

The `lwarp` package allows  $\text{\LaTeX}$  to directly generate HTML5 tags from a  $\text{\LaTeX}$  source document, with only minor intervention on the user’s part. A `texlua`

---

<sup>1</sup>Firefox has had an on-again/off-again bug for quite some time regarding printing SVGs at high resolution.

<sup>2</sup>Aside — The author, being a programmer, believes in the movement towards “logical punctuation”. (Look it up.)

Aside to the aside — The author also believes that space should be on either side of an em-dash, employing the symbol as a separator rather than something looking like a chemical **bond—as** shown here, but is willing to abide by Chicago style when necessary.

<sup>3</sup>There seems to be some debate as to whether MathML is actually an improvement over  $\text{\LaTeX}$  for sharing math. The author has no particular opinion on the matter, except to say that in this case  $\text{\LaTeX}$  is much easier to implement!



program called `lwarpmk` is used to process either the print or HTML version of the document. A few external utility programs are used to finish the conversion from a  $\text{\LaTeX}$ -generated PDF file which happens to have HTML5 tags, to a number of HTML5 plain-text files and accompanying images.

A special `lwarp-newproject` package is used to set up the auxiliary files necessary for the HTML conversion. Included is a parallel version of the user's source document, `<sourcename>-html.tex`, which selects HTML output and then inputs the user's own source. This process allows both the printed and HTML versions to co-exist side-by-side, each with their own auxiliary files.

When requesting packages during HTML conversion, `lwarp` first looks to see if it has its own modified version to use instead of the usual  $\text{\LaTeX}$  version. These `lwarp-packagename.sty` files contain code used to emulate or replace functions for HTML output.

Enough functionality is provided to convert a typical article containing technical content. Not every package has been tested, but many of the most useful ones are known to work, either as-is or through emulation with replacement code. (See table 1 on page 18.)

## 2.1 Supported packages and features

Table 1 lists the various  $\text{\LaTeX}$  packages and macros which may be used. *Supported* means that the package or macro may be used as-is, perhaps with minor limitations. *Emulated* means that the original package or macro is not used, but similar functionality is provided in a way which is intended to be compatible with the user's  $\text{\LaTeX}$  code. *Ignored* means that the package is not used, but nullified user-level macros and related counters, lengths, etc. are provided for source-level compatibility.

Any package listed in table 1 probably works with little or no change to the user's source code. Special environments are provided to mark blocks of code which are for print only, HTML only, or both, should it be necessary.

Table 1: L<sup>A</sup>T<sub>E</sub>X–HTML generation — lwarp package — Supported functions

Package or Macro	Status
Engines:	pdfL <sup>A</sup> T <sub>E</sub> X, X <sub>Y</sub> L <sup>A</sup> T <sub>E</sub> X, LuaL <sup>A</sup> T <sub>E</sub> X
Classes:	book, report, or article. memoir is planned.
Sectioning:	Supported, with hyperlinks. Honors <code>tocdepth</code> and <code>secnumdepth</code> . Adds <code>filedepth</code> for splitting the HTML output. Files may be numbered sequentially or named according to section name. Common short words and punctuation is removed from the filenames.
<code>titlesec</code>	Ignored.
Table of Contents, Figures, Tables:	Supported, with hyperlinks.
<code>\maketitle</code> , <code>titlepage</code> , titling:	Supported, with optional titling-based commands for published and subtitle.
<code>titletoc</code>	Ignored.
<code>tocloft</code>	Ignored, with support for <code>\newlistof</code> and <code>\ctfchapterprecis</code> .
<code>abstract</code> :	Supported
Cross-references:	Emulated, with hyperlinks.
<code>cleverref</code> :	Supported, including ranges and lists, with hyperlinks.
<code>nameref</code> :	Emulated
<code>hyperref</code> :	Emulated. HTML hyperlinks are generated for TOC, LOF, LOT, <code>\nameref</code> , <code>\ref</code> , the <code>cleverref</code> commands, and index entries.
<code>bookmark</code> :	Emulated.
Footnotes:	Supported, emulated as pagenotes per HTML page.
<code>pagenote</code> :	Supported.
Indexing:	<code>texindy</code> is used, with hyperlinks.
Bibliography:	Supported, without hyperlinks so far.

## lwarp Supported Functions — continued

Package or Macro	Status
babel:	Supported.
Math:	Supported. Converted to SVG images with HTML ALT tags containing the $\text{\LaTeX}$ source for the math expression. MathJax supported as an alternative. $\text{\LaTeX}$ environments are supported. User-defined macros are available during conversion, due to native $\text{\LaTeX}$ processing.
Floats:	Supported, appear where declared. float is emulated, newfloat is supported, caption and subcaption are supported, capt-of is supported, placeins is ignored, trivfloat is supported, floatrow is emulated, keyfloat is supported, wrapfig is emulated.
tabular:	Emulated. <code>\multirow</code> and <code>\multicolumn</code> are available, but cannot be used at the same time. Nested tables are not supported.
multirow:	Emulated.
longtable:	Emulated. Converted to a tabular. Captions supported. Extra headings and <code>\kill</code> lines must be enclosed in <code>\warpprintonly{}</code>
booktabs:	Emulated. <code>\toprule</code> and <code>\bottomrule</code> form black rules, <code>\midrule</code> forms silver rules, as demonstrated on this table. <code>\cmidrule</code> , demonstrated at this line, does not use width or trim options.
threeparttable:	Emulated
geometry:	Ignored.
titlesp:	Used by lwarp, but emulated for the user.
fancyhdr:	Ignored.
extramarks:	Ignored.
afterpage, draftwatermark, eso-pic, everypage, wallpaper:	Ignored.

## lwarp Supported Functions — continued

Package or Macro	Status
graphics, graphicx:	Emulated. <code>\includegraphics</code> supports <code>width</code> , <code>height</code> , <code>origin</code> , <code>angle</code> , and <code>scale</code> tags, and adds <code>class</code> . References to PDF files are changed to SVG, other image types are accepted as well. <code>\rotatebox</code> and <code>\scalebox</code> are supported as well as HTML can handle.
rotating:	Emulated. All objects are displayed unrotated.
Lists:	Supported
enumitem:	Supported, although spacing is still controlled by CSS.
Environments:	Standard L <sup>A</sup> T <sub>E</sub> X environments are supported.
theorem:	Supported, with added CSS tags.
epigraph:	Emulated
csquotes:	Supported
verse and memoir's verse-related macros	Supported
picture and tikz:	Converted to an SVG image.
minipage:	Supported with some HTML5-imposed limitations. Nested minipages are supported. Footnotes appear at the bottom of the HTML page.
fancyvrb:	Supported except for verbatim footnotes.
alltt, listings:	Supported
algorithmicx:	Supported
mdframed:	Supported
multicol:	Emulated, with CSS3. Converted to up to three columns with an optional heading, per browser support. Single-column if unsupported.
siunitx:	Supported except for <code>per-mode=fraction</code> .
xfrac:	Supported

## lwarp Supported Functions — continued

Package or Macro	Status
Direct formatting:	<code>\emph</code> , <code>\textsuperscript</code> , <code>\textbf</code> , etc are supported. <code>\bfseries</code> , etc. are not yet supported.
Ordinals:	<code>nth</code> , <code>fntcount</code> , and <code>engord</code> are supported.
Text ligatures:	Ligatures for symbols are supported. Ligatures for f, q, t are intentionally turned off because many simpler browsers do not display them correctly. Modern full-featured browsers re-create these ligatures on-the-fly.
Horizontal space:	HTML output for thin-unbreakable, unbreakable, <code>\enskip</code> , <code>\quad</code> , <code>\qquad</code> , <code>\hspace</code> .
Rules:	<code>\rule</code> with width, height, raise, text color.
<code>placeins</code> , <code>setspace</code> , <code>needspace</code> :	Ignored.
<code>textpos</code> :	Emulated.
<code>nowidow</code> :	Ignored.
<code>microtype</code> , <code>letterspace</code> :	Ignored. <code>microtype</code> is used to disable ligatures.
HTML reserved characters:	<code>\&amp;</code> , <code>\textless</code> , and <code>\textgreater</code> are converted to HTML entities.
<code>xcolor</code> :	<b>Supported.</b> Full package color names, any color models, and <code>mixing</code> is converted to hex web colors via <code>\convertcolorspec</code> . Patched commands are <code>\textcolor</code> , <code>\colorbox</code> , and <code>\fcolorbox</code> . <code>\pagecolor</code> is not supported.
CJKutf8:	Supported
Where:	
<b>Supported:</b>	The existing $\text{\LaTeX}$ package is used.
<b>Emulated:</b>	The $\text{\LaTeX}$ package is not used, but some/all of its functions are emulated. Null functions, lengths, and counters are provided for source compatibility.
<b>Ignored:</b>	The $\text{\LaTeX}$ package is not used, but null functions are provided for source compatibility.

### 3 Alternatives

Summarized below are several other ways to convert a  $\text{\LaTeX}$  or other document to HTML. Where an existing  $\text{\LaTeX}$  document is to be converted to HTML, **lwarp** may be a good choice. For new projects with a large number of documents, it may be worth investigating the alternatives before decided which path to take.

#### 3.1 TeX4ht

Prog **TeX4ht** <http://tug.org/tex4ht/>

This system uses native  $\text{\LaTeX}$  processing to produce a DVI file containing special commands, and then uses additional post-processing for the HTML conversion by way of numerous configuration files. In some cases, **lwarp** provides a better HTML conversion, and it supports a different set of packages. TeX4ht produces several other forms of output beyond HTML.

#### 3.2 Translators

These systems use external programs to translate a subset of  $\text{\LaTeX}$  syntax into HTML:

Prog **Hevea** **H<sup>E</sup>v<sup>E</sup>a**: <http://hevea.inria.fr/>

Prog **TtH** **T<sub>T</sub>H**: <http://hutchinson.belmont.ma.us/tth/>

Prog **GELLMU** **GELLMU**: <http://www.albany.edu/~hammond/gellmu/>

Prog **LaTeXML** **LaTeXML**: <http://dlmf.nist.gov/LaTeXML/>

Prog **Plastex** **Plastex**: <https://github.com/tiarno/plastex>

Prog **LaTeX2html** **LaTeX2html**: <http://www.latex2html.org/>  
and <http://ctan.org/pkg/latex2html>.

#### 3.3 AsciiDoc

AsciiDoc is one of the most capable markup languages, providing enough features to produce the typical technical-writing document with cross-references, and it writes  $\text{\LaTeX}$  and HTML.

Prog **AsciiDoc** **Asciidoctor**: <http://asciidoctor.org/> (More active.)

Prog   **AsciiDoctor**   **AsciiDoc:** <http://asciidoc.org/> (The original version.)

The AsciiDoctor-LaTeX project is adding additional L<sup>A</sup>T<sub>E</sub>X-related features.

**AsciiDoctor-LaTeX:**

Prog   **AsciiDoctor-LaTeX**   <http://www.noteshare.io/book/asciidoc-lateX-manual>  
<https://github.com/asciidoc/asciidoc-lateX>

### 3.4 Pandoc

Prog   **Pandoc**

A markup system which also reads and writes L<sup>A</sup>T<sub>E</sub>X and HTML.

**Pandoc:** <http://pandoc.org/>

(Watch for improvements in cross-references to figures and tables.)

### 3.5 Word Processors

Prog   **Word**   It should be noted that the popular word processors have advanced through the  
 Prog   **LibreOffice**   years in their abilities to represent math with a L<sup>A</sup>T<sub>E</sub>X-ish input syntax, unicode  
 Prog   **OpenOffice**   math fonts, and high-quality output, and also generate HTML with varying success.  
 See recent developments in Microsoft® Word® and LibreOffice™ Writer.

### 3.6 Commercial Systems

Prog   **Adobe**   Likewise, several professional systems exist whose abilities have been advancing in  
 Prog   **FrameMaker**   the areas of typesetting, cross-referencing, and HTML generation. See Adobe®  
 Prog   **InDesign**   FrameMaker®, Adobe® InDesign®, and Madcap Flare™.  
 Prog   **Flare**  
 Prog   **Madcap**

### 3.7 Comparisons

AsciiDoc, Pandoc, and various other markup languages typically have a syntax which tries to be natural and human-readable, but the use of advanced features tends to require many combinations of special characters, resulting in a complicated mess of syntax. By contrast, L<sup>A</sup>T<sub>E</sub>X spells things out in readable words but takes longer to type, although integrated editors exist which can provide faster entry and a graphic user interface. For those functions which are covered by the typical

markup language it is arguable that  $\text{\LaTeX}$  is comparably easy to learn, while  $\text{\LaTeX}$  provides many more advanced features where needed, along with a large number of pre-existing packages which provide solutions to numerous common tasks.

Text-based document-markup systems share some of the advantages of  $\text{\LaTeX}$  vs. a typical word processor. Documents formats are stable. The documents themselves are portable, work well with revision control, do not crash or become corrupted, and are easily generated under program control. Formatting commands are visible, cross-referencing is automatic, and editing is responsive. Search/replace with regular expressions provides a powerful tool for the manipulation of both document contents and structure. Markup systems and some commercial systems allow printed output through a  $\text{\LaTeX}$  back end, yielding high-quality results especially when the  $\text{\LaTeX}$  template is adjusted, but they lose the ability to use  $\text{\LaTeX}$  macros and other  $\text{\LaTeX}$  source-document features.

The effort required to customize the output of each markup system varies. For print output,  $\text{\LaTeX}$  configuration files are usually used. For HTML output, a CSS file will be available, but additional configuration may require editing some form of control file with a different syntax, such as XML. In the case of `lwarp`, CSS is used, and much HTML output is adjusted through the usual  $\text{\LaTeX}$  optional macro parameters, but further customization may require patching  $\text{\LaTeX}$  code.

The popular word processors and professional document systems each has a large base of after-market support including pre-designed styles and templates, and often include content-management systems for topic reuse.



## 4 Installation

Table 2 shows the tools which are used for the L<sup>A</sup>T<sub>E</sub>X to HTML conversion. In most cases, these will be available via the standard package-installation tools.

Table 2: Required software programs

---

**Provided by your L<sup>A</sup>T<sub>E</sub>X distribution:**

From T<sub>E</sub>XLive: <http://tug.org/texlive/>.

**L<sup>A</sup>T<sub>E</sub>X:** pdf<sub>l</sub>atex, xelatex, or lualatex.

**The lwarp package:** This package.

**The lwarp-newproject package:** Accompanies lwarp, and used to create configuration files.

**The lwarpmk utility:** Provided along with this package. This should be an operating-system executable in the same way that pdf<sub>l</sub>atex or latexmk is. It is possible to have the lwarp-newproject package generate a local copy of lwarpmk called lwarpmk.lua. See table 3.

**luatex:** Used by the lwarpmk program to simplify and automate document generation.

**xindy:** The xindy package is used by lwarp to create indexes. On a MiK<sub>T</sub>E<sub>X</sub> system this may have to be acquired separately, but it is part of the regular installer as of mid 2015.

**latexmk:** Optionally used by lwarpmk to compile L<sup>A</sup>T<sub>E</sub>X code. On a MiK<sub>T</sub>E<sub>X</sub> system, Perl may need to be installed first.

**pdfcrop:** Used to pull images out of the L<sup>A</sup>T<sub>E</sub>X PDF.

**Extra packages provided with your operating sytem:**

From Poppler: [poppler.freedesktop.org](http://poppler.freedesktop.org).

**pdftotext:** Used to convert PDF to text.

**pdfseparate:** Used to pull images out of the L<sup>A</sup>T<sub>E</sub>X PDF.

**pdftocairo:** Used to convert images to SVG.

**Automatically downloaded from the internet as required:**

**MathJax:** Optionally used to display math. Automatically loaded from the MathJax website when needed.

From: [mathjax.org](http://mathjax.org)

---

## 4.1 Installing the lwarp package

There are several ways to install `lwarp`. These are listed here with the preferred methods listed first:

**Pre-installed:** Try entering into a command line:

```
Enter ⇒ lwarpmk
```

If the `lwarpmk` help message appears, then `lwarp` is already installed.

**T<sub>E</sub>X Live:** If using a T<sub>E</sub>X Live distribution, try installing via `tlmgr`:

```
Enter ⇒ tlmgr install lwarp
```

**MiK<sub>T</sub>E<sub>X</sub>:** If using MiK<sub>T</sub>E<sub>X</sub>, try using the package installer.

**Operating-system package:** The operating-system package manager may already have `lwarp`, perhaps as part of a set of T<sub>E</sub>X-related packages.

**CTAN TDS archive:** `lwarp` may be downloaded from the Comprehensive T<sub>E</sub>X Archive:

1. See <http://ctan.org/pkg/lwarp> for the `lwarp` package.
2. Download the TDS archive: `lwarp.tds.zip`
3. Find the T<sub>E</sub>X local directory:

**T<sub>E</sub>X Live:**

```
Enter ⇒ kpsewhich -var-value TEXMFLOCAL
```

**MiK<sub>T</sub>E<sub>X</sub>:**

In the “Settings” window, “Roots” tab, look for a local TDS root.

This should be something like:

```
/usr/local/texlive/texmf-local/
```

4. Unpack the archive in the TDS local directory.
5. Renew the cache:

```
Enter ⇒ mktexlsr — or —
```

```
Enter ⇒ texhash
```

**CTAN .dtx and .ins files:** Another form of T<sub>E</sub>X package is the `.dtx` and `.ins` source files. These files are used to create the documentation and `.sty` files.

1. See <http://ctan.org/pkg/lwarp> for the `lwarp` package.
2. Download the zip archive `lwarp.zip` into your own `lwarp` directory.
3. Unpack `lwarp.zip`.

4. Locate the contents `lwarp.dtx` and `lwarp.ins`
5. Create the documentation:
 

```
Enter ⇒ pdflatex lwarp.dtx (several times)
```
6. Create the `.sty` files:
 

```
Enter ⇒ pdflatex lwarp.ins
```
7. Copy the `.sty` files somewhere such as the T<sub>E</sub>X Live local tree found in the previous CTAN TDS section, under the subdirectory:
 

```
<texlocal>/tex/latex/local/lwarp
```
8. Copy the documentation `lwarp.pdf` to a source directory in the local tree, such as:
 

```
<texlocal>/doc/local/lwarp
```
9. Renew the cache:
 

```
Enter ⇒ mktexlsr — or —
```

```
Enter ⇒ texhash
```
10. See section 4.2.1 to generate your local copy of `lwarpmk`.
11. Once the local version of `lwarpmk.lua` is installed, it may be made available system-wide as per section 4.2.

**Project-local CTAN `.dtx` and `.ins` files:** The `.dtx` and `.ins` files may be downloaded to a project directory, then compiled right there, alongside the document source files. The resultant `*.sty` and `lwarpmk.lua` files may be used as-is, so long as they are in the same directory as the document source. This approach is especially useful if you would like to temporarily test `lwarp` before deciding whether to permanently install it.

Just testing!

## 4.2 Installing the `lwarpmk` utility

(Note: It is possible to use a local copy of `lwarpmk` instead of installing it system-wide. See section 4.2.1.)

After the `lwarp` package is installed, you may need to setup the `lwarpmk` utility:

1. At a command line, try executing `lwarpmk`. If the help message appears, then `lwarpmk` is already set up.
2. Locate the file `lwarpmk.lua`, which should be in the `scripts` directory of the TDS tree. On a T<sub>E</sub>X Live system you may use

```
Enter ⇒ kpsewhich lwarpmk.lua
```

### 3. Create lwarpmk:

**The easy way:** Copy `lwarpmk.lua` to wherever `pdflatex` is located. For Unix, rename `lwarpmk.lua` to `lwarpmk` and make it executable.

**Unix:** Create a symbolic link and make it executable:

- (a) Locate the T<sub>E</sub>X Live binaries:

Enter  $\Rightarrow$  `kpsewhich -var-value TEXMFROOT`

This will be something like:

`/usr/local/texlive/<year>`

The binaries are then located in the `bin/<arch>` directory under the root:

`/usr/local/texlive/<year>/bin/<architecture>/`

In this directory you will find programs such as `pdflatex` and `makeindex`.

- (b) In the binaries directory, create a new symbolic link from the binaries directory to `lwarpmk.lua`:

Enter  $\Rightarrow$  `ln -s <pathtolwarpmk.lua> lwarpmk`

- (c) Make the link executable:

Enter  $\Rightarrow$  `chmod 0755 lwarpmk`

**Windows T<sub>E</sub>X Live:** Create a new `lwarpmk.exe` file:

- (a) Locate the T<sub>E</sub>X Live binaries as shown above for Unix.
- (b) In the binaries directory, make a *copy* of `runscript.exe` and call it `lwarpmk.exe`

**Windows MiK<sub>T</sub>E<sub>X</sub>:** Copy `lwarpmk.lua` to a folder in the executable PATH:

- (a) Locate the binaries. These will be in a directory such as:

`C:\Program Files\MiKTeX 2.9\miktex\bin`

In this directory you will find programs such as `pdflatex` and `makeindex`.

- (b) Copy `lwarpmk.lua` to this directory.

#### 4.2.1 Using a local copy of lwarpmk

It is also possible to use a local version of `lwarpmk`:

1. When compiling the tutorial in section 5, use the `lwarpmk` option for the `lwarp-newproject` package:  

$$\backslash\text{usepackage}[lwarpmk]\{lwarp-newproject\}$$
2. When the tutorial is compiled with `pdflatex`, the file `lwarpmk.lua` will be generated along with the other configuration files.
3. `lwarpmk.lua` may be used for this project:

**Unix:**

- (a) Make `lwarpmk.lua` executable:

Enter  $\Rightarrow$  `chmod 0755 lwarpmk.lua`

- (b) Compile documents with

`./lwarpmk.lua html`

`./lwarpmk.lua print` etc.

- (c) It may be useful to rename or link to a version without the `.lua` suffix.

**Windows:**

Compile documents with

`lwarpmk html`

`lwarpmk print` etc.

### 4.3 Installing additional utilities

The T<sub>E</sub>X utilities `latexmk` and `pdfcrop` may require the installation of additional T<sub>E</sub>X-relatex operating-system packages.

The tools from the POPPLER project should be provided by your operating system's package-installation tools.

To test for the existence of the additional utilities, try:

Enter  $\Rightarrow$  `luatex -version`

Enter  $\Rightarrow$  `xindy -version`

Enter  $\Rightarrow$  `latexmk -version`

Enter  $\Rightarrow$  `pdfcrop -version`

Enter  $\Rightarrow$  `pdftotext -v`

Enter  $\Rightarrow$  `pdfseparate -version`

Enter  $\Rightarrow$  `pdftocairo -v`

## 5 Tutorial

This section shows an example of how to create an `lwarp` document.

### 5.1 Starting a new project

1. Create a new project directory called `tutorial`.
2. Inside the `tutorial` directory, create a new file called `tutorial.tex`, as seen in fig. 1. This may be copied/pasted from this documentation directly into your own editor.
3. Compile the project:

Enter  $\Rightarrow$  `pdflatex tutorial.tex` (several times)

(`xelatex` or `lualatex` may be used as well.)

A number of new files are created when `tutorial.tex` is compiled, as shown in table 3. These files are created by the `lwarp-newproject` package.

(Two of the new files are configuration files for the helper program `lwarpmk`. Whenever a print version of the document is created while the `lwarp-newproject` package is used, the configuration files for `lwarpmk` are updated to record the operating system, L<sup>A</sup>T<sub>E</sub>X program (`pdflatex`, `xelatex`, or `lualatex`), the filenames of the source code and HTML output, and whether the additional helper program `latexmk` will be used to compile the document.)

Figure 1: tutorial.tex listing

Note: There are two pages!

```

\documentclass{book}

\usepackage{iftex}

% --- LOAD FONT SELECTION AND ENCODING BEFORE LOADING LWARP ---
\ifPDFTeX
\usepackage{lmodern}           % pdflatex
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\else
\usepackage{fontspec}         % XeLaTeX or LuaLaTeX
\fi

% --- HTML FILENAME AND LATEXMK SETTINGS ---
% \newcommand{\HomeHTMLFilename}{index} % Filename of the homepage.
% \newcommand{\HTMLFileName}{node-} % Filename prefix of other pages.
% \newcommand{\UseLatexmk}{true}% Uncomment to use latexmk

% --- LWARP IS LOADED NEXT ---
\usepackage{lwarp-newproject} % Possibly with the [lwarpmk] option.
\usepackage{lwarp}
% \boolfalse{FileSectionNames} % If false, numbers the files.

% --- OTHER PACKAGES ARE LOADED AFTER LWARP ---
\usepackage{makeidx} \makeindex
\usepackage{xcolor}

\usepackage{hyperref,cleveref} % LOAD THESE LAST!

% --- LATEX AND HTML CUSTOMIZATION ---
\title{The Lwarp Tutorial}
\author{Some Author}
\setcounter{tocdepth}{2} % Include subsections in the TOC.
\setcounter{secnumdepth}{2} % Number down to subsections.
\setcounter{FileDepth}{1} % Split HTML files at sections
\booltrue{CombineHigherDepths} % Combine parts/chapters/sections
\setcounter{SideTOCDepth}{1} % Include subsections in the sideTOC
\SetFirstPageTop{Name and \fbox{HOMEPAGE LOGO}}
\SetPageTop{\fbox{LOGO}}
\SetPageBottom{Contact Information and Copyright}
\NewCSS{lwarp_sagebrush.css}

\begin{document}
\maketitle % (or a titlepage environment)
% --- An abstract may be placed here. ---

```

```
\tableofcontents \listoffigures % --- MUST BE BEFORE THE FIRST SECTION.

\chapter{First chapter}
\section{A section}
This is some text which is indexed.\index{Some text.}

\subsection{A subsection}
See \cref{fig:withtext}.
\begin{figure}\begin{center}
\fbbox{\textcolor{blue!50!green}{Text in a figure.}}
\caption{A figure with text\label{fig:withtext}}
\end{center}\end{figure}

\section{Some math}
Inline math:  $r = r_0 + vt - \frac{1}{2}at^2$ 
\begin{equation}
a^2 + b^2 = c^2
\end{equation}

\printindex
\end{document}
```



Table 3: Files created in a new project

**tutorial.pdf:** The PDF output from L<sup>A</sup>T<sub>E</sub>X.

**tutorial\_html.tex:** A small .tex file used to create a parallel HTML version of the document, which co-exists with usual the PDF version, and which will have its own auxiliary files. In this way, both PDF and HTML documents may co-exist side-by-side.

**Auxiliary files:** The usual L<sup>A</sup>T<sub>E</sub>X files `tutorial.aux`, `tutorial.log`. When an HTML version of the document is created, `_html` versions of the auxiliary files will also be generated.

**lwarpmk.conf:** A configuration file for `lwarpmk`, which is used to automate the compilation of PDF or HTML versions of the document.

**tutorial.lwarpmkconf:** Another configuration file used by `lwarpmk`, which is only useful if you wish to have several projects residing in the same directory.

**.css files:** `lwarp.css`, `lwarp_formal.css`, `lwarp_sagebrush.css` These files are standard for `lwarp`, and are not meant to be modified by the user.

**sample\_project.css:** An example of a user-customized CSS file, which may be used for project-specific changes to the `lwarp` defaults.

**lwarp\_html.xdy:** Used by `lwarp` while creating an index. This file should not need to be modified by the user.

**lwarp\_mathjax.txt:** Inserted into the HTML files when `MathJax` is used to display math. This file should not need to be modified by the user.

**comment.cut:** A temporary file used by `lwarp` to conditionally process blocks of text. This file may be ignored.

---

When the `lwarpmk` option is given to the `lwarp-newproject` package:

**lwarpmk.lua:** A local copy of the `lwarpmk` utility.

On Unix-related operating systems this file must be made executable:  
`chmod u+x lwarpmk.lua`

This may be useful to have to archive with a project for future use.

## 5.2 Compiling the print version with `lwarpmk`

The `lwarpmk` utility program is used to compile either the printed or the HTML version of the document.

`lwarpmk print` is used to recompile a printed version of the document.

1. Re-compile the print version:

Enter  $\Rightarrow$  `lwarpmk print`

`lwarpmk` prints an introduction then checks to see if the document must be recompiled. If it seems that the files are up-to-date, then `lwarpmk` informs you of that fact and then exits.

2. Make a small change in the original document, such as adding a space character.
3. Recompile again.

Enter  $\Rightarrow$  `lwarpmk print`

The document is recompiled when a change is seen in the source. Several compilations may be necessary to resolve cross-references.

4. Force a recompile to occur.

Enter  $\Rightarrow$  `lwarpmk again`

Enter  $\Rightarrow$  `lwarpmk print`

`lwarpmk again` updates the date code for the file, triggering a recompile the next time the document is made.<sup>4</sup>

5. Process the index.

Enter  $\Rightarrow$  `lwarpmk printindex`

6. Recompile again to include the index.

Enter  $\Rightarrow$  `lwarpmk print`

Note that the HTML customization commands are ignored while making the print version.

---

<sup>4</sup> Although, when using the utility `latexmk` (introduced later), the changed date is ignored and an actual change in contents must occur to cause a recompile.

### 5.3 Compiling the HTML version with `lwarpmk`

`lwarpmk html` is used to recompile an HTML version of the document.

1. Compile the HTML version:

```
Enter ⇒ lwarpmk html
```

- (a) `lwarpmk` uses  $\text{\LaTeX}$  to process `tutorial_html.tex` to create `tutorial_html.pdf`.
- (b) `pdftotext` is then used to convert into the file `tutorial_html.html`. This file is a plain-text file containing HTML tags and content for the entire document.
- (c) `lwarpmk` manually splits `tutorial_html.html` into individual HTML files according to the HTML settings. For this tutorial, the result is `tutorial.html` (the home page), along with `First-chapter.html`<sup>5</sup>, `Some-math.html`, and the document's index in `_Index.html`.<sup>6</sup>

2. View the homepage in a web browser.

Open the file `tutorial.html` in a web browser.

math

Note that `math` is still displayed as its plain-text  $\text{\LaTeX}$  source until the images of the math expressions have not yet been generated. Math may be displayed as SVG images or by a MathJax script, as seen in sections 5.4 and 5.5.

3. Force a recompile:

```
Enter ⇒ lwarpmk again
```

```
Enter ⇒ lwarpmk html
```

```
Enter ⇒ lwarpmk print
```

4. Process the HTML index and recompile:

```
Enter ⇒ lwarpmk htmlindex
```

```
Enter ⇒ lwarpmk html
```

`_Index.html`, is updated for the new  $\text{\LaTeX}$  index.

5. Reload the web page to see the added index.

<sup>5</sup> `First-chapter.html` also contains the first section, even though the second section is its own HTML page. This behavior is controlled by the boolean `CombineHigherDepths`.

<sup>6</sup> `index.html` is commonly used as a homepage, so the document index is in `_Index.html`.

## 5.4 Generating the SVG images

**math as SVG images** By default `lwarp` represents math as SVG images with the  $\text{\LaTeX}$  source included in `alt` tags. In this way, the math displays as it was drawn by  $\text{\LaTeX}$ , and the  $\text{\LaTeX}$  source may be copied and pasted into some other document.


**picture and Tikz** `lwarp` uses the same mechanism for `picture` and `Tikz` environments.


1. Create the SVG images:

```
Enter ⇒  lwarpmk limages
```

```
Enter ⇒  lwarpmk html
```

2. Move to the tutorial's math page and reload.
3. The math images are displayed using the same font and formatting as the printed version.
4. Copy/paste a math expression into a text editor to see the  $\text{\LaTeX}$  source.


 **Adding/removing** When a math expression, `picture`, or `Tikz` environment is added or removed, the SVG images must be re-created with `lwarpmk limages` to maintain the proper image file sequence numbers.

 **Lots of files!** Expressing math as SVG images has the advantage of representing the math exactly as  $\text{\LaTeX}$  would, but has the disadvantage of requiring an individual file for each math expression. There is no attempt at reusing the same file each time the same expression occurs, so each time  $\$x\$$  is used, for example, yet another file is created. For a document with a large amount of math, see section 5.5 to use MathJax instead.

## 5.5 Using MathJax for math

**math with MathJax** Math may also be represented using the MathJax Javascript project.

1. In the tutorial's source code, change  
`\usepackage{lwarp}`  
to  
`\usepackage[mathjax]{lwarp}`
2. Recompile  
  
Enter  $\Rightarrow$  `lwarpmk html`
3. Reload the math page.

 **MathJax requirements** MathJax requires web access unless a local copy of MathJax is available, and it also requires that Javascript is enabled for the web page. The math is rendered by MathJax. Right-click on math to see several options for rendering, and for copying the  $\text{\LaTeX}$  source.

While using MathJax has many advantages, it may not be able to represent complex expressions or spacing adjustments as well as  $\text{\LaTeX}$ .

## 5.6 Changing the CSS style

**\NewCSS** **\NewCSS** may be used to choose which `.css` file is used to display each section of the web page. Use **\NewCSS** before `\begin{document}` to assign the style of the home page. If different parts of the website should have different styles, call **\NewCSS** again before each section heading which creates a new file.

The styles provided by `lwarp` include:


**lwarp.css:** A default style if **\NewCSS** is not used. This style is comparable to a plain L<sup>A</sup>T<sub>E</sub>X document. To set this style, you may use **\NewCSS{lwarp.css}**, or no **\NewCSS** call at all.

**lwarp\_formal.css:** A formal style with a serif fonts and a traditional look.

**lwarp\_sagebrush.css:** A style with muted colors, gradient backgrounds, additional borders, and rounded corners.

To see each style in use, change the **\NewCSS** entry in the tutorial `lwarpmk.html`, and then reload the webpage.

**Custom CSS** A customized style may also be created. For each new project a file called `sample_project.css` is generated. This may be renamed to `<project>.css` then used by assigning **\NewCSS{<project>.css}**.

 **Rename it!** Note that `sample_project.css` is overwritten whenever `lwarp-newproject` is loaded in print mode. It is therefore important to rename the file to something like `<project>.css` before using it, so that your own changes are not overwritten.

`<project>.css` has an entry which loads `lwarp.css`, and this entry may be changed to load `lwarp_formal.css` or `lwarp_sagebrush.css` if desired. Additional changes to the CSS may be made by making entries later in the `<project>.css` file.

## 5.7 Customizing the HTML output

### Placement!

Several settings may be used to customize the HTML output. Watch for the correct placement of each!

### Changes!

Note that if changes are made, it is best to first use `lwarpmk clearall` to clear all the HTML, PDF, and auxiliary files. Also, if `\HomeHTMLFilename`, `\HTMLFileName`, or `\UseLatexmk` are changed, it is necessary to reenale the `lwarp-newproject` package and then recompile the print version in order to recreate the configuration files for `lwarpmk`.

### Placed in the preamble just before `lwarp` is loaded:

<code>\HomeHTMLFilename</code>	<b>\HomeHTMLFilename:</b> Filename of the homepage, without the “.html” suffix. Defaults to the <code>\BaseJobname</code> . A common setting is: <code>\newcommand{HomeHTMLFilename}{index}</code> causing the homepage to be the file <code>index.html</code> . Underscores are allowed in filenames.
<code>\HTMLFileName</code>	<b>\HTMLFileName:</b> A filename prefix for the rest of the HTML web pages. Useful for numbered web pages with a common prefix. May be empty.
<code>\UseLatexmk</code>	<b>\UseLatexmk:</b> A macro which controls whether <code>lwarp</code> uses <code>latexmk</code> to compile the document. This setting is written to <code>lwarpmk</code> ’s configuration files. Defaults to false. Set to true with: <code>\newcommand*{\UseLatexmk}{true}</code>

### Placed in the preamble before `\begin{document}`:

Ctrl <code>tocdepth</code>	<b>tocdepth:</b> Sectioning depth of the table of contents. See section 10 for a list of L <sup>A</sup> T <sub>E</sub> X stack depths.
Ctrl <code>SideTOCDepth</code>	<b>SideTOCDepth:</b> Sectioning depth of the sideTOC. Defaults to 1, causing the sidetoc to show sections but not subsections.
<code>sideTOC</code>	Each subpage of the website has its own small table of contents on the side (the “sideTOC”). Its depth is set by <code>SideTOCDepth</code> . This sideTOC is only shown if the web page is wide enough. When using a narrow web browser window, “responsive web design” is used to show the sideTOC at the top of the page and a link back to “Home” at the bottom. It is recommended to set: <code>SideTOCDepth = FileDepth</code> , or <code>SideTOCDepth = FileDepth + 1</code> If <code>SideTOCDepth &lt; FileDepth</code> , web pages will be inaccessible via the sideTOC.
Ctrl <code>FileDepth</code>	<b>FileDepth:</b> Sectioning depth of file splits. Defaults to -5, causing the entire HTML website to be one single file.

- To place the entire file into one HTML page, use:  
`\setcounter{FileDepth}{-5}`
- To split the HTML file at `\section` depth, use:  
`\setcounter{FileDepth}{1}`
- To ensure that the HTML pages/files are accessible:  
Place a `\tableofcontents` somewhere before the first section break (therefore in the “home page”), and set `tocdepth >= FileDepth`.



Bool `CombineHigherDepths`

**CombineHigherDepths:** Combine a higher section with its first lower subsections, down to the `FileDepth`. Defaults to true. Set to false to simulate the concept of a chapter opening on its own page, for example.

The file splits are controlled by the counter `FileDepth` and the boolean `CombineHigherDepths`. Setting `FileDepth` to 0 splits the file at chapters, 1 at sections, etc. `CombineHigherDepths` controls whether to combine pages at levels higher than the chosen `FileDepth`, such as in this tutorial where the page which opens the chapter also contains the first section. Be careful to set `tocdepth` and `SideTOCDepth` to allow access to each page of the website. Set `tocdepth` and `SideTOCDepth` to be greater than or equal to `FileDepth`.

⚠ Inaccessible pages!

⚠ Lost in an old page!

When making changes to the file structure, it is possible to end up with the web browser pointing to an old file which is no longer in use. When this occurs, changes to the web site will not appear in the browser, even if reloading the page, because that page is no longer in use. It is best to return to the home page, clean the files (`lwarpmk cleanall`), change `FileDepth` and/or `CombineHigherDepths`, then finally recompile and renavigate to the desired page using the new file structure.

Bool `FileSectionNames`

**FileSectionNames:** If true, web page filenames are derived from a sanitized version of the section names. If false, web pages are numbered. Either way, the `\HTMLFileName` is prefixed.

HTML filenames

Example HTML filenames:

**Numbered HTML nodes:**

Example: Homepage `index.html`, and `node-1`, `node-2`. (See `\SetHTMLFileNumber` to number grouped by chapter, for example.)

---

```
\newcommand{\HomeHTMLFileName}{index}
\newcommand{\HTMLFileName}{node-}
\usepackage{lwarp}
\boolfalse{FileSectionNames}
```

---

**Named HTML sections, no prefix:**

Example: `index.html`, and `About.html`, `Products.html`

---

```
\newcommand{\HomeHTMLFileName}{index}
\newcommand{\HTMLFileName}{}
```

---



---

```
\usepackage{lwrap}
\booltrue{FileSectionNames}
```

---

### Named HTML sections, with prefix:

Example: Homepage `mywebsite.html`, and additional pages such as `mywebsite-About.html`, etc.

---

```
\newcommand{\HomeHTMLFileName}{mywebsite}
\newcommand{\HTMLFileName}{mywebsite-}
\usepackage{lwrap}
\booltrue{FileSectionNames}
```

---

<code>\abstractname</code>	<b>\abstractname:</b> The name of the abstract. This may also be over-written by the babel package. Defaults to “Abstract”.
<code>\MetaLanguage</code>	<b>\MetaLanguage:</b> The HTML language meta header. Defaults to en-US.
<code>\SetFirstPageTop</code>	<b>\SetFirstPageTop:</b> <code>{\langle contents \rangle}</code> A user-definable custom action applied to the top of the home page. Useful for logos, etc. Defaults empty. Ignored in print output.
<code>\SetPageTop</code>	<b>\SetPageTop:</b> <code>{\langle contents \rangle}</code> A user-definable custom action applied to the top of pages other than the home page. Useful for logos, etc. Defaults empty. <code>\LinkHome</code> may be used to place a link back to the homepage. Ignored in print output.
<code>\SetPageBottom</code>	<b>\SetPageBottom:</b> <code>{\langle contents \rangle}</code> A user-definable custom action applied to the bottom of each web page. Useful for authors, copyright notices, contact information, etc. Defaults empty. <code>\LinkHome</code> may be used to place a link back to the homepage. Ignored in print output.

### Placed in the home page before the first section break:

<code>\tableofcontents</code>	<b>\tableofcontents:</b> Used to place a table of contents on the home page. This command must be used before the first file split, so that a way is available to navigate to other files from the homepage. Links to each chapter/section are provided, as selected by <code>tocdepth</code> .
-------------------------------	---

### Placed before any sectioning command which causes a file break:

<code>\NewCSS</code>	<b>\NewCSS:</b> <code>{\langle filename.css \rangle}</code> Sets the CSS file to use for the following files. May be changed before each each sectioning command which would cause a file split.  The CSS styles of the web pages are set by the <code>\NewCSS</code> command. If <code>\NewCSS</code> is not used, a default plain style is used to mimic printed L <sup>A</sup> T <sub>E</sub> X output. <code>lwrap_sagebrush.css</code> is a semi-fancy colored style as shown in this tutorial. Change it to <code>lwrap_formal.css</code> for a more formal look, or comment out the <code>\NewCSS</code> command to see the default. <code>\NewCSS</code> may be used before each file break to set the CSS for individual pages of the website.
----------------------	---

---

**\NewHTMLdescription**      **\NewHTMLdescription:**  $\{\langle description \rangle\}$     Sets the HTML description tag for the following files. May be changed before each sectioning command which would cause a file split.

**Placed in the document wherever necessary:**

Env    **warpprint**      **warpprint:** An environment which is only used while generating print output. Place here anything which does not apply to HTML and which may cause problems with **lwarp**. If **lwarp** knows about and emulates or supports a package then its related macros, lengths, counters, etc. probably won't have to be placed inside a **warpprint** environment, but unknown packages may cause problems which may be isolated from **lwarp** using this environment.

Env    **warpHTML**      **warpHTML:** An environment which is only used while generating HTML output. This is useful for website logos and other items which have no purpose in printed output.

**\warpprintonly**      **\warpprintonly:**  $\{\langle contents \rangle\}$     A macro version of the **warpprint** environment.

**\warpHTMLonly**      **\warpHTMLonly:**  $\{\langle contents \rangle\}$     A macro version of the **warpHTML** environment.

## 5.8 Using latexmk

`latexmk` is a  $\text{\LaTeX}$  utility used to monitor changes in source files and recompile as needed.

1. In the tutorial’s source code, at “HTML FILENAME AND LATEXMK SETTINGS”, insert the line:

```
\newcommand{\UseLatexmk}{true}
```

2. Directly recompile the printed version of the document.

⚠ NOT `lwarpmk`!

```
Enter ⇒ pdflatex tutorial.tex
```

```
(Or xelatex or lualatex)
```

`lwarp` updates its own configuration files (`lwarpmk.conf` and `tutorial.lwarpmkconf`) whenever the printed version of the document is compiled. These configuration files remember that `lwarpmk` should use `latexmk` to compile the document.

3. Recompile the document.

```
Enter ⇒ lwarpmk print and/or
```

```
Enter ⇒ lwarpmk html
```

Changes are detected by comparing checksums rather than modification times, so `lwarpmk` again will not trigger a recompile, but `latexmk` has a much better awareness of changes than the `lwarpmk` utility does and it is likely to correctly know when to recompile. A recompile may be forced by making a small change to the source.

## 5.9 Using XeLaTeX or LuaLaTeX

X<sub>Y</sub>LaTeX or LuaLaTeX may be used instead of LaTeX.

1. Remove the auxiliary files for the project:

```
Enter ⇒  lwarpmk cleanall
```

2. Be sure that

```
    \usepackage{lwarp-newproject}  
is still enabled in tutorial.tex.
```

3. Use **xelatex** or **lualatex** to recompile the printed version.

```
Enter ⇒  xelatex tutorial.tex  -or-
```

```
Enter ⇒  lualatex tutorial.tex
```

When the recompile occurs and **lwarp-newproject** is loaded, the configuration files for **lwarpmk** are modified to remember which TeX engine was used. X<sub>Y</sub>LaTeX or LuaLaTeX will be used for future runs of **lwarpmk**.

4. To recompile the document:

```
Enter ⇒  lwarpmk print  -and-
```

```
Enter ⇒  lwarpmk html
```

5. Also rememeber to update the indexes and recompile again.

## 5.10 Cleaning auxiliary files

To remove the auxiliary files `.aux`, `.toc`, `.lof`, `.lot`, `.idx`, `.ind`, and `.log`:

```
Enter ⇒ lwarpmk clean
```

## 5.11 Cleaning auxiliary and output files

To remove the auxiliary files, and also remove the `.pdf` and `.html` files:

```
Enter ⇒ lwarpmk cleanall
```

## 5.12 Processing multiple projects in the same directory

It is possible to have several projects in the same directory. `lwarpmk` has an optional parameter which is the document to compile.

To create each project:

```
Enter ⇒ pdflatex project_a
```

```
Enter ⇒ pdflatex project_b
```

Each project is given its own configuration file:

```
project_a.lwarpmkconf, project_b.lwarpmkconf
```

To compile each project with `lwarkmk`:

```
Enter ⇒ lwarpmk print project_a
```

```
Enter ⇒ lwarpmk html project_b
```

## 5.13 Using the make utility

`lwarpmk` has an action which may be useful for integration with the common `make` utility:

```
lwarpmk pdftohtml [project]
```

`make` may be used to compile the code to PDF with HTML tags (`project_html.pdf`), then `lwarpmk` may be used to convert each target to HTML files.

## 6 Additional details

### 6.1 Font and UTF-8 support

lwarp uses `pdftotext` to convert PDF output into UTF-8-encoded text. This process requires that UTF-8 information be embedded in the PDF file, which usually prevents the use of bit-mapped fonts.

vector fonts  
Computer Modern



While using `pdflatex`, if no font-related package is specified, the default bit-mapped Computer Modern font is used, so simply add

```
usepackage{lmodern}
```

to the preamble to enable the related vector font instead, or use

```
\usepackage{dejavu}
```

or other other font packages, which may provide an increased coverage of Unicode mappings. Avoid bit-mapped fonts.



X<sub>Y</sub>LaTeX and LuaLaTeX users must use the `fontspec` package. Do NOT use `fontenc`!

Place `fontspec` or `fontenc` and other font and UTF-8 related commands after the `\documentclass` command and before `\usepackage{lwarp}`:

1. `documentclass{article/book/report}` goes here, followed by any of:
2. Font and UTF-8 related commands:

- For X<sub>Y</sub>LaTeX or LuaLaTeX:

Pkg `fontspec`

- `fontspec` and font choices

`ligatures`

lwarp sets the following to turn off TeX ligatures during the generation of HTML tags, and turn off common ligatures in regular text, since older browsers may not display them correctly and newer browsers can automatically re-create them.

---

```
\defaultfontfeatures[\rmfamily]{Ligatures={NoCommon,TeX}}
\defaultfontfeatures[\sffamily]{Ligatures={NoCommon,TeX}}
\defaultfontfeatures[\ttfamily]{Ligatures=NoCommon}
```

---

- For `pdflatex`:

Pkg `lmodern`

- `lmodern` or other font-related packages

Pkg `fontenc`

- `fontenc`

Pkg `inputenc`

- `inputenc`

Pkg `newunicodechar`

- `newunicodechar`

File `glyphtounicode`

- `\input glyphtounicode.tex`
- `\input glyphtounicode-cmr.tex`% from the `pdfx` package
- `\pdfgentounicode=1`

Pkg	<code>cmap</code>	– <code>cmap</code>
Pkg	<code>textcomp</code>	– <code>textcomp</code>
Pkg	<code>microtype</code> <code>ligatures</code>	– <code>microtype</code> is automatically used by <code>lwarp</code> to turn off f,q,t,T,Q ligatures for the same browser-related reasons shown above. Also, the monospaced font is used during HTML tag generation to turn off $\text{\TeX}$ ligatures.

3. `\usepackage{lwarp}` (section 6.2) goes after any of the above, followed by:

4. ... the rest of the preamble and the main document.

## 6.2 lwarp package loading and options

`lwarp` supports `book`, `report`, and `article` classes.

Pkg	<code>lwarp</code>	Load the <code>lwarp</code> package immediately after the font and UTF-8 setup commands.
Opt lwarp	<code>warpprint</code>	Select the <code>warpprint</code> option to generate print output (default), or the <code>warpHTML</code> option to generate HTML5 output. The default is print output, so the print version may be compiled with the usual <code>pdflatex</code> , etc. When <code>lwarp-newproject</code> is loaded in print mode, it creates <code>&lt;project&gt;_html.tex</code> , which sets the <code>warpHTML</code> option before calling the user's source code <code>&lt;project&gt;.tex</code> . In this way, <code>&lt;project&gt;.tex</code> can <code>\usepackage{lwarp}</code> without any options to create a printed version, while <code>&lt;project&gt;_html.tex</code> will create an HTML version.
Opt lwarp	<code>warpHTML</code>	
Opt lwarp	<code>mathsvg</code>	For math display, select <code>mathsvg</code> (default), or <code>mathjax</code> . For more information about the math options, see section 6.10.3.
Opt lwarp	<code>mathjax</code>	

## 6.3 Selecting the operating system

`lwarp` tries to detect which operating system is being used.

Prog	<code>MS-Windows</code>	If MS-WINDOWS is not correctly detected, use the command <code>\warpOSwindows</code> in the document preamble after <code>lwarp</code> is loaded. This modifies the creation of the <code>lateximages.sh</code> or <code>lateximages.cmd</code> batch file, which is an operating-system-specific shell script used to take individual pages of the $\text{\LaTeX}$ PDF output and convert and store them in individual SVG files.
Prog	<code>Windows</code>	
	<code>\warpOSwindows</code>	

## 6.4 Selecting actions for print or HTML output

The following environments and macros are used to select actions which only apply to either traditional  $\text{\LaTeX}$  print-formatted PDF generation, or to HTML generation.

For most of built-in L<sup>A</sup>T<sub>E</sub>X and many additional packages there is user-level source code support or emulation, so no special handling will be required. For those cases which **lwarp** does not handle by itself, the following environments and macros may be used to isolate sections of code for print-only or HTML-only.

These environments are also useful for creating a special version of the titlepage for print and another for HTML.

Env **warpHTML** Anything which is to be done only for HTML5 output is surrounded by a **warpHTML** environment:

---

```
\begin{warpHTML}
... something to be done only during HTML generation
\end{warpHTML}
```

---

Env **warpprint** Anything which is to be done only for print output is surrounded by a **warpprint** environment:

---

```
\begin{warpprint}
... something to be done only during traditional PDF generation
\end{warpprint}
```

---

Env **warpall** Anything which is to be done for any output may be surrounded by a **warpall** environment. Doing so is optional.

---

```
\begin{warpall}
... something to be done during print PDF or HTML output
\end{warpall}
```

---

Macros are also provided for print-only or HTML-only code:

**\warpprintonly**  $\{\langle actions \rangle\}$

Performs the given actions only when print output is being generated.

**\warpHTMLonly**  $\{\langle actions \rangle\}$

Performs the given actions only when HTML output is being generated.



## 6.5 Commands to be placed into the `warpptest` environment

Certain print-related commands should always be placed inside a `warpptest` environment, or may need other special handling. These are unrelated to HTML output, but are hard to isolate automatically. For example:

- Paragraph formatting: `\parindent` `\parskip`
- Variable spaces such as `\vspace`. `\hfill` is turned into a `\quad`. Fixed spaces such as `\quad` are emulated correctly.
- Manual page positions such as the `textpos` package, which is emulated but only in a limited way.

Some packages require additional setup commands. Where these packages are emulated for HTML, setup commands may work for the emulated HTML output as well as for print output. See the details for each package in this document for more information.

Also see section 8: [Troubleshooting](#).

## 6.6 Commands for a successful HTML conversion

Some commonly-used L<sup>A</sup>T<sub>E</sub>X expressions should be modified to allow for a smooth conversion to both HTML and print-formatted outputs:

**`\bfseries`, etc:** Use `\textbf` instead.

**`\centering`, `\raggedright`, `\raggedleft`:**

Use the environments `center`, `flushright`, `flushleft` instead.

**Superscripts and other non-math uses of math mode:**

Use `\textsuperscript{x}` instead of  $\text{\textsuperscript{x}}$

**Empty `\item` followed by a new line of text or a nested list:**

Use a trailing backslash: `\item[label] \`

**`\fbox` around a minipage:**

`\fbox` can only be used around inline items during HTML output.

For an `\fbox` around a minipage, you may:

- Place the `\fbox` command and its closing brace inside `warpptest` environments.

- Use `\mdframed` instead.
- Use a custom environment to create a sidebar, containing a `BlockClass` environment with custom CSS formatting, and `\warpprintonly{\hrule}` command:

---

```
\begin{BlockClass}{frameminipage}% ignored in print output
% use CSS to format div class ``framedminipage''
\warpprintonly{\hrule} % only appears in print output
Contents
\warpprintonly{\hrule} % only appears in print output
\end{BlockClass}
```

---

Also see section 8: [Troubleshooting](#).

## 6.7 Title page

In the preamble, place an additional block of code to set the following:

---

```
\title{Document Title} % One line only
\subtitle{Optional Document Subtitle \\ with optional multiple lines}
\author{Author One\affiliation{Affiliation One} \and
        Author Two\affiliation{Affiliation Two} }
\date{Optional date}
\published{Optional Journal Name \\ Optional multiple lines}
```

---

The title is used in the meta tags in the HTML files, and the rest are used in `\maketitle`.

- |                              |   |
|------------------------------|---|
| <code>\maketitle</code>      | Use <code>\maketitle</code> just after the <code>\begin{document}</code> , as this will establish the title of the homepage. Optionally, use a <code>titlepage</code> environment instead.  |
| Env <code>titlepage</code>   | The <code>titlepage</code> environment may be used to hold a custom title page. The <code>titlepage</code> will be set in a div class <code>titlepage</code> , and <code>\printtitle</code> , etc. may be used inside this environment. |
| Env <code>titlingpage</code> | Another form of custom title page, where <code>\maketitle</code> is allowed, and additional information may be included as well.  |

`\title` `{\langle title \rangle}`



Avoid newlines in the `\title`; these will interfere with the file break and CSS detection. Use the `\subtitle` command instead. In HTML, the title will appear in a heading “h1”.

`\author` `{\langle author \rangle}`

⚠ In `\author`, use `\protect` before formatting commands such as `\textsc`. In HTML, the author will appear in a div class `author`. `\affiliation` is a new addition to lwarp.

`\date` `{\langle date \rangle}`

`\date` works as expected. In HTML, this will appear in a div class “`titledate`”.

`\subtitle` `{\langle subtitle \rangle}`

A new command which sets a subtitle. Newlines are allowed. The default is empty. In HTML, this will appear in a div class “`subtitle`”.

`\published` `{\langle published \rangle}`

A new command which sets a publisher. The default is empty. In HTML, this will appear in a div class “`published`”.

`\thanks` `{\langle text \rangle}`

`\thanks` are allowed in the titlepage fields, and will be rendered as HTML notes at the bottom of the title page.

## 6.8 HTML page meta descriptions

`\NewHTMLdescription` `{\langle A description of the web page. \rangle}` The default is no description.

**limitations** Each page of HTML output should have its own HTML meta description, which usually shows up in web search results, is limited to around 150 characters in length, and should not include the ASCII double quote character (“”).

**placement** Use `\NewHTMLdescription` just before `\begin{document}` to set the description of the home page, and also just before each sectioning command such as `\chapter` or `\section` where a new file will be generated, depending on `FileDepth`. For example, if `FileDepth` is 1, use `\NewHTMLdescription` just before each `\section` command, and that description will be placed inside the HTML page for that `\section`. The same description will be used for all following HTML files as well, until reset by a new `\NewHTMLdescription`. It is best to use a unique description for each HTML file.

**disabling** To disable the generation of HTML description meta tags, use:  
`\NewCSSdescription{}`

## 6.9 CSS

File `lwarp.css` It is best to make a local project-specific CSS file such as `project.css`, containing

File `project.css`

File `sample_project.css`

only things which are different from `lwarp.css`. `project.css` should refer to `lwarp.css` as follows:

---

```
/* ( --- Start of project.css --- ) */
/* A sample project-specific CSS file for lwarp --- ) */

/* Load default lwarp settings: */
@import url("lwarp.css") ;
/* or lwarp_formal.css, lwarp_sagebrush.css */

/* Project-specific CSS setting follow here. */
/* . . . */

/* ( --- End of project.css --- ) */
```

---

An example file called `sample_project.css` is provided, and may be renamed `project.css`.

`\NewCSS` For each section at which HTML files are split, `\NewCSS` may be used before the sectioning command to select a CSS file for that and all following sections. This may be changed numerous times throughout the file, resulting in different HTML pages having different CSS files assigned.

## 6.10 Special cases and limitations

### 6.10.1 Text formatting

`\textbf`, etc. are supported, but `\bfseries`, etc. are not yet supported.

### 6.10.2 Verse and Memoir packages

`\attrib` The documentation for the `verse` and `memoir` packages suggest defining an `\attrib` command, which may already exist in current documents, but it will only work for print output. `lwarp` provides `\attribution`, which works for both print and HTML output. To combine the two so that `\attrib` is used for print and `\attribution` is used for HTML:

---

```
\begin{warpHTML}
\let\attrib\attribution
\end{warpHTML}
```

---

`\leftskip` These lengths are used by `verse` and `memoir` to control the left margin, and they  
`\leftmargini`

`\TMLvleftskip` may already be set by the user for print output. New lengths `\HTMLvleftskip` and `\HTMLleftmargini` are provided to control the margins in HTML output. These new lengths may be set by the user before any `verse` environment, and persist until they are manually changed again. One reason to change `\HTMLleftmargini` is if there is a wide `\flagverse` in use, such as the word “Chorus”, in which case the value of `\HTMLleftmargini` should be set to a wide enough length to contain “Chorus”. The default is wide enough for a stanza number.

Horizontal spacing relies on `pdftotext`’s ability to discern the `-layout` of the text in the HTML-tagged PDF output. For some settings of `\HTMLleftmargini` or `\HTMLleftskip` the horizontal alignment may not work out exactly, in which case a label may be shifted by one space.

### 6.10.3 Math

Math may be rendered as SVG graphics or using the MATHJAX JavaScript display engine.

**SVG math option** For SVG math, math is rendered as usual by  $\text{\LaTeX}$  into the initial PDF file using the current font<sup>7</sup>, then is captured from the PDF and converted to SVG graphics via a number of utility programs. The SVG format is a scalable-vector web format, so math may be typeset by  $\text{\LaTeX}$  with its fine control and precision, then displayed or printed at any size, depending on (sometimes broken) browser support. An HTML ALT tag carries the  $\text{\LaTeX}$  code which generated the math, allowing copy/paste of the  $\text{\LaTeX}$  math expression into other documents.

**SVG image font size** The size of the math and text used in the SVG image may be adjusted by setting `\LateximageFontSizeName` to a font size name — *without the backslash*, for ex:

```
\renewcommand{\LateximageFontSizeName}{large}
```

**SVG files** As currently implemented, each instance of math creates a new SVG file. In text with many references to math variables, this can result in a large number of files with duplicate content. In the future, some method of content-based naming and checksumming may be used to remove the need for duplicate files.

**SVG inline** Another approach would be to in-line the SVG files directly into the HTML. This avoids having a large number of files and potentially speeds loading the images, but dis-allows the possibility of sharing one file among many instances without user intervention.

**PNG files** Others have used PNG files, sometimes pre-scaled for print resolution but displayed on-screen at a scaled down size. This allows high-quality print output at the expense of larger files, but SVG files are also larger as well.

**MathML** Conversion to MathML might be a better approach, among other things allowing a

<sup>7</sup>See section 114 regarding fonts and fractions.

more compact representation of math than SVG drawings. Problems with MathML include limited browser support and some issues with the fine control of the appearance of the result.

### MathJax math option

The popular MathJax alternative ([mathjax.org](https://mathjax.org)) may be used to display math.

Prog MathJax

When MathJax is enabled, math is rendered twice:

1. As regular L<sup>A</sup>T<sub>E</sub>X PDF output placed inside an HTML comment, allowing equation numbering and cross referencing to be almost entirely under the control of L<sup>A</sup>T<sub>E</sub>X, and
2. As detokenized printed L<sup>A</sup>T<sub>E</sub>X commands placed directly into the HTML output for interpretation by the MathJax display scripts. An additional script is used to pre-set the equation number format and value according to the current L<sup>A</sup>T<sub>E</sub>X values, and the MathJax cross-referencing system is ignored in favor of the L<sup>A</sup>T<sub>E</sub>X internal system, seamlessly integrating with the rest of the L<sup>A</sup>T<sub>E</sub>X code.

### MathJax limitations

Limitations when using MathJax include:

Prog MathJax

#### chapter numbers

- In document classes which have chapters, `\tagged` equations have the chapter number prepended in HTML output, unlike L<sup>A</sup>T<sub>E</sub>X. `\tag*` equations (correctly) do not. This may be improved with future versions of the MathJax support script.

<https://groups.google.com/forum/#!topic/mathjax-users/jUtewUcE2bY>

#### subequations

- MathJax itself does not support subequations. This may be improved by parsing the L<sup>A</sup>T<sub>E</sub>X math expression to manually insert tags, but this has not yet been done.

#### footnotes in math

- Footnotes inside equations are not yet supported while using MathJax.

#### lateximage

- Math appearing inside a `lateximage`, and therefore also inside a `Tikz` or `picture` environment, is rendered as SVG math even if MathJax is used in the rest of the document.

#### siunitx

- Usage of `siunitx` inside a math equation is supported via a third-party MathJax extension. While inside a math expression, do not use `\SI` or `\si` inside `\text`, where it will be rendered as normal text.

<https://github.com/mathjax/MathJax-third-party-extensions/tree/master/siunitx>

△ siunitx inside an equation

#### L<sup>A</sup>T<sub>E</sub>X macros

- MathJax does not automatically support custom L<sup>A</sup>T<sub>E</sub>X macros, but they may be set up by the user.

### custom MathJax macros

For an example of using custom L<sup>A</sup>T<sub>E</sub>X macros with MathJax, see page 204.


### 6.10.4 Graphics

**\graphicspath** `\graphicspath` only works for a single directory; all graphics must be in this directory.


**units** For `\includegraphics`, avoid `px` and `%` units for width and height, or enclose them inside `warpHTML` environments. For font-proportional image sizes, use `ex` or `em`. For fixed-sized images, use `cm`, `mm`, `in`, `pt`, or `pc`. Using the keys `width=.5\linewidth`, or similar for `\textwidth` or `\textheight` to give fixed-sized images proportional to a 6 by 9 inch text area.

**options** `\includegraphics` accepts `width` and `height`, `origin`, `rotate` and `scale`, plus a new `class` key.

**HTML class** With HTML output, `\includegraphics` accepts an optional `class=xyz` keyval combination, and if this is given then the HTML output will include that class for the image. The class is ignored for print output.

 **image file types** For `\includegraphics` the user should provide both `.pdf` and `.svg` images, but always refer to `.pdf` images in the document source. All `\includegraphics` references to `.pdf` will automatically be changed to `.svg` for HTML output, and will be left as `.pdf` for print output. Images may also be `.jpg` and `.png`, and will be used as-is for either output.

**\rotatebox** `\rotatebox` accepts the optional `origin` key.

 **browser support** `\rotatebox`, `\scalebox`, and `\reflectbox` depend on modern browser support. The CSS3 standard declares that when an object is transformed the whitespace which they occupied is preserved, unlike  $\text{\LaTeX}$ , so expect some ugly results for scaling and rotating.

### 6.10.5 Xcolor


**support** Color definitions, models, and mixing are fully supported without any changes required.

**tables** Colored tables are ignored. Use CSS to style tables.

**colored text and boxes** `\textcolor`, `\colorbox`, and `\fcolorbox` are supported.

**\color and \pagecolor** `\color` and `\pagecolor` are ignored. Use CSS or `\textcolor` where possible.

### 6.10.6 Tikz






 **preamble** For recent versions of `Tikz` (v3.0.0 — 2013/12/20 and later), include in the document preamble:

`\usetikzlibrary{babel}` % v3.0.0 and up

For older versions of Tikz without the Tikz babel library, lwarp automatically handles the catcode change for the \$ math shift.

See section 60.

### 6.10.7 Tabular

column types	<ul style="list-style-type: none"> <li>• Vertical rules are not yet supported.</li> <li>• * in a column specification is not used (so far). Repeat the column type the correct number of times.</li> </ul>
 <code>\multirow &amp; \multicolumn</code>	<ul style="list-style-type: none"> <li>• Multirow and multicolumn cannot be used at the same time. (No rectangular holes wider than one column or taller than one row.)</li> </ul>
 <code>\multirow</code>	<ul style="list-style-type: none"> <li>• For multirow, insert <code>\mrowcell</code> into any empty multi-row cells. This will be a null function for the print output, and is a placeholder for parsing the table for HTML output.</li> </ul>
<code>\multirow with rules</code>	<ul style="list-style-type: none"> <li>• If a multirow reaches to the bottom of a table, and <code>\bottomrule</code> does not go over to that edge, try adding a line of empty cells below the <code>\bottomrule</code>. This may be a browser bug.</li> </ul>
rule at last row	<ul style="list-style-type: none"> <li>• If a <code>\midrule</code> is desired after the last row, an additional row of blank cells must be used.</li> </ul>
 paragraphs	<ul style="list-style-type: none"> <li>• Multiple paragraphs in one cell of a p, b, m column must have <code>\newline</code> between paragraphs.</li> </ul>
<code>\cmidrule width, trim</code>	<ul style="list-style-type: none"> <li>• <code>\cmidrule</code> does not support width or trim options due to CSS limitations.</li> </ul>
<code>longtable headings</code>	<ul style="list-style-type: none"> <li>• For longtable, place headings and footings which do not apply to HTML inside <code>\warpprintonly{}</code>.</li> </ul>
 <code>\warpprintonly</code>	<ul style="list-style-type: none"> <li>• For <code>\toprule</code> and <code>\bottomrule</code>, when combined with a <code>warpprint</code> or <code>warpHTML</code> environment, if a “misplaced <code>\noalign</code>” error occurs, change  <code>This &amp; That \endhead</code>  to  <code>\warpprintonly{This &amp; That \endhead}</code>  and likewise with the other <code>\end</code> headings. Keep the <code>\endfirsthead</code> row unchanged, as it is still relevant to HTML output.</li> </ul>
 S columns	<ul style="list-style-type: none"> <li>• For S columns (from the siunitx package), while producing print output, anything non-numeric must be placed inside { } braces, including commands such as <code>\multirow</code>. While producing HTML output, though, anything placed inside braces is not seen by lwarp’s tabular handling algorithm. To resolve this problem, make a copy of the row, with one version for print output,</li> </ul>



containing the extra braces, and another version for HTML output, without the extra braces, such as:

```
\warpprintonly{1 & 2 & {\multirow{2}{2cm}{Text}} & 3 \\}
\warpHTMLonly{1 & 2 & \multirow{2}{2cm}{Text} & 3 \\}
```

### 6.10.8 Minipages


**placement** Minipages and parboxes will be placed side-by-side in HTML unless you place a `\newline` between them.

**inline** A line of text with an inline minipage or parbox will have the minipage or parbox placed onto its own line, because a paragraph is a block element and cannot be made `inline-block`.

**in a span** There is limited support for minipages inside an HTML `|span|`. An HTML `|div|` cannot appear inside a `|span|`. While in a `|span|`, minipages and parboxes are ignored. Use `\newline` or `\par` for an HTML break.


**size** When using `\linewidth`, `\textwidth`, and `\textheight`, widths and heights are scaled proportionally to a 6×9 inch text area.

**full-width minipages** A new macro `\minipagefullwidth` requests that the next minipage be generated without an HTML `width` tag, allowing it to be the full width of the display rather than the fixed width given.

 **text alignment** Nested minipages adopt their parent's text alignment in HTML, whereas in regular L<sup>A</sup>T<sub>E</sub>X PDF output they do not. Use a `flushleft` or similar environment in the child minipage to force a text alignment.

### 6.10.9 Mdframed

**support** Most basic functionality is supported, including frame background colors and single-border colors and thickness, title and subtitle background colors and borders and thickness, border radius, and shadow. CSS classes are created for `mdframed` environments and frame titles.

 **loading** Only load `mdframed` in a `warpprint` environment. `lwarp` pre-loads `mdframed` in HTML with `framemethod=none`.

For title font, use

```
frametitlefont=\textbf,
```

**font** instead of

```
frametitlefont=\bfseries,
```

where `\textbf` must appear just before the comma and will receive the following text as its argument (since the text happens to be between braces in the `mdframed`


source). Since `lwrap` does not support `\bfseries` and friends, only one font selection may be made at a time.

`theoremtitlefont` `theoremtitlefont` is not supported, since the following text is not in braces in the `mdframed` source.

`footnotes` Footnotes are currently placed at the bottom of the HTML page.

`ignored options` `userdefinedwidth` and `align` are currently ignored.

#### 6.10.10 `float`, `trivfloat`, and/or `algorithmicx` together

 `package conflicts` If using `\newfloat`, `trivfloat`, and/or `algorithmicx` together, see section 110.1.

#### 6.10.11 `caption` and `subcaption` packages

To ensure proper float numbering, set caption positions such as:

```
\captionsetup[table]{position=top}
\captionsetup[figure]{position=bottom}
```


Similarly for `subtable`, `subfigure`, and `longtable`.

#### 6.10.12 `subfig` package

`LOF/LOT` At present, subfigures before the enclosing figure/table's `\caption` would appear in the List of Figures/Tables before the enclosing figure/table, therefore `lwrap` does not place subfigures in the LOF/LOT.

`horizontal spacing` In the document source, use `\hfill` and `\hspace*` between subfigures to spread them apart horizontally. The use of other forms of whitespace may cause paragraph tags to be generated, resulting in subfigures appearing on the following lines instead of all on a single line.

#### 6.10.13 `floatrow` package

 `\FBwidth`, `\FBheight` The emulation of `floatrow` does not support `\FBwidth` or `\FBheight`. These values are pre-set to `.3\linewidth` and `2in`. Possible solutions include:

- Use fixed lengths. `lwrap` will scale the HTML lengths appropriately.
- Use `warpprint` and `warpHTML` environments to select appropriate values for each case.

- Inside a `warpHTML` environment, manually change `\FBwidth` or `\FBheight` before the `\ffigbox` or `\ttabbox`. Use `\FBwidth` or `\FBheight` normally afterwards; it will be used as expected in print output, and will use your custom-selected value in HTML output. This custom value will be used repeatedly, until it is manually changed to a new value.

#### 6.10.14 siunitx package

Pkg `siunitx`



`per-mode`

Do not use `per-mode=fraction`, which cannot be seen by the final `pdftotext` conversion.

#### 6.10.15 newclude package

Pkg `newclude`



`loading`

`newclude` modifies `\label` in a non-adaptive way, so `newclude` must be loaded before `lwarp` is loaded.

Ex:

---

```
\documentclass{article}
...<font setup>
\usepackage{newclude}
\usepackage[lwarpHTML]{lwarp}
...
```

---

#### 6.10.16 newtxmath package

Pkg `newtxmath`



`loading`

`newtxmath` must be loaded after `lwarp`.

#### 6.10.17 babel package

Pkg `babel`

If using `babel` with French, use

```
\frenchbsetup{StandardLists=true}
```



`French`

to preserve the special HTML and `enumitem` list handling.

`\CaptionSeparator`

Also, when French is used, the caption separator is changed to a dash. The following

may be used to restore it to a colon:

```
\renewcommand*{\CaptionSeparator}{:~}
```

### 6.10.18 enumitem package

Pkg `enumitem`

`enumitem` is pre-loaded during HTML output. Many of the spacing options are rendered irrelevant by `pdftotext` and HTML. Numbering, labels, and `\newlist` function correctly.

## 7 Expanding the code base

Purely text-based packages probably will work as-is when generating HTML.

Look to existing code for ideas on how to expand into new code.

An environment may be converted to a `lateximage` then displayed with an image of the resulting L<sup>A</sup>T<sub>E</sub>X output. See section 59 for an example of the `picture` environment.

To create a custom HTML block or inline CSS class, see section 28.7.

### 7.1 Creating an lwarp version of a package

When creating HTML, `lwarp` redefines the `\usepackage` and `\RequirePackage` macros such that it first looks to see if a `lwarp-⟨packagename⟩.sty` version exists. If so, the `lwarp` version is used instead. This modular system allows the users to create their own versions of packages for `lwarp` to use for HTML, simply by creating a new package with a `lwarp-` prefix. If placed in the local directory along with the source code, it will be seen by that project alone. If placed alongside the other `lwarp-` packages where T<sub>E</sub>X can see it, then the user's new package will be seen by any documents using `lwarp`.

An `lwarp-⟨packagename⟩.sty` package is only used during HTML generation. Its purpose is to pretend to be the original package, while modify anything necessary to create a successful HTML conversion. For many packages it is sufficient to simply provide nullified macros, lengths, counters, etc. for anything which the original package does, while passing the raw text on to be typeset. See the pre-existing `lwarp-` packages for examples.

Note that anything the user might expect of the original package must be replaced or emulated by the new `lwarp-` package, including package options, user-adjustable counters, lengths, and booleans, and conditional behaviors.

Each `lwarp-` package should first call either `\LWR@ProvidesPackageDrop` or `\LWR@ProvidesPackagePass`. If dropped, the original print-version package is ignored, and only the `lwarp-` version is used. Use this where the original print version is useless for HTML. If passed, the original package is loaded first, with the supplied options, then the `lwarp-` version continues loading as well. Use this when HTML output only requires some modifications of the original package. For a case where the original package is usable as-is, there is no need to create a `lwarp-` version.

## 8 Troubleshooting

### 8.1 Using the `lwarp.sty` package

Also see:

Section 6.6: [Commands for a successful HTML conversion](#)

Section 6.5: [Commands to be placed into the `warpprint` environment](#)

Section 6.10: [Special cases and limitations](#)

#### Text is not converting:

- Font-related UTF-8 information must be embedded in the PDF file. See section 6.1 regarding vector fonts.

#### Undefined HTML settings:

See the warning regarding the placement of the HTML settings at section 5.7.

#### Obscure error messages:

- Be sure that a print version of the document compiles and that your document's  $\text{\LaTeX}$  code is correct, before attempting to generate an HTML version.

#### Missing sections:

- See section 5.7 regarding the `FileDepth` and `SideTOCDepth` counters, and the use of `\tableofcontents` in the home page.

#### Missing HTML files:

See the warning regarding changes to the HTML settings at section 5.7.

#### Missing / incorrect cross-references:

- Use `lwarpmk` again followed by `lwarpmk html` or `lwarpmk print` to compile the document one more time.

**Em-dashes or En-dashes in listing captions and titles:**

Use  $\text{Xe}\text{L}\text{A}\text{T}\text{E}\text{X}$  or  $\text{L}\text{u}\text{a}\text{L}\text{A}\text{T}\text{E}\text{X}$ .

**Floats out of sequence:**

**Mixed “Here” and floating:** Floats [H]ere and regular floats may become out of order. `\clearpage` if necessary.

**Caption setup:** With `\captionsetup` set the positions for the captions above or below to match their use in the source code.

**Print document contains HTML tags:**

- Be sure that the document selects `\usepackage[warpprint]{lwarp}` instead of `[warpHTML]`.

**HTML document contains a single unformatted print document:**

- Be sure that the document selects `\usepackage[warpHTML]{lwarp}` instead of `[warpprint]`.

**Images are appearing in strange places:**

- `lwarpmk limages` to refresh the `lateximage` images.

**“Leaders not followed by proper glue”:** This can be caused by a missing `l@<floattype>` or `l@<sectiontype>` definition. See `lwarp`’s definitions for examples.

**Plain-looking document:**

- The document’s CSS stylesheet may not be available, or may be linked incorrectly. Verify any `\NewCSS` statements point to a valid CSS file.

**Broken fragments of HTML:**

- Check the PDF file used to create HTML to see if the tags overflowed the margin. (This is why such large page size and margins are used.)

**Changes do not seem to be taking effect:**

- Be sure to `lwarpmk clean`, recompile, then start by reloading the home page. You may have been looking at an older version of the document. If you changed a section name, you may have been looking at the file for the old name.
- See the warning regarding changes to the HTML settings at section 5.7.
- Verify that the proper CSS is actually being used.

- The browser may compensate for some subtle changes, such as automatically generating ligatures, reflowing text, etc.

#### Un-matched conditional compiles:

- Verify the proper `begin/end` of `warpprint`, `warpHTML`, and `warpall` environments.

#### 8.1.1 Debug tracing output

`\tracinglwarp` When `\tracinglwarp` is used, `lwarp` will add extra tracing messages to the `.log` file. The last several messages may help track down errors.

Place `\tracinglwarp` just after `\usepackage{lwarp}` to activate tracing.

## 8.2 Compiling the `lwarp.dtx` file

#### Illogical error messages caused by an out-of-sync `lwarp.sty` file:

1. Delete the `lwarp.sty` file.
2. `pdflatex lwarp.ins` to generate a new `lwarp.sty` file.
3. `pdflatex lwarp.dtx` to recompile the `lwarp.pdf` documentation.

#### Un-nested environments:

Be sure to properly nest:

- `\begin{macrocode}` and `\end{macrocode}`
- `\begin{macro}` and `\end{macro}`
- `\begin{environment}` and `\end{environment}`

## 9 Implementation

This package is perhaps best described as a large collection of smaller individual technical challenges, in many cases solved through a number of ~~erude hacks~~ clever tricks. Reference sources are given for many of the solutions, and a quick internet search will provide additional possibilities.

Judgement calls were made, and are often commented. Improvements are possible. The author is open to ideas and suggestions.

Packages were patched for re-use where they provided significant functionality. Examples include `xcolor` with its color models and conversion to HTML color output,

and `siunitx` which provides many number and unit-formatting options, almost all of which are available in pure-text form, and thus easily used by `pdftotext`.

Packages were emulated where their primary purpose was visual formatting which is not relevant to HTML output. For example, packages related to sectioning are already patched by numerous other packages, creating a difficult number of combinations to try to support, and yet in HTML output all of the formatting is thrown away, so these packages are merely emulated.

Packages with graphical output are allowed as-is, but must be nested inside a `lateximage` environment to preserve the graphics.

There is still room to improve the factoring of the code, and doing so will become important if support for other output formats is added. Rather than wait until the code is pristine, the author felt it best to publish early and accept input before pushing on towards a perhaps less-than-ideal solution.

Testing has primarily been done with the Iceweasel/Firefox browser.

## 10 Stack depths

Stacks are created to track depth inside the  $\text{\LaTeX}$  document structure. This depth is translated to HTML headings as shown in table 4. “Depth” here is not depth in the traditional computer-science stack-usage sense, but rather a representation of the nesting depth inside the  $\text{\LaTeX}$  document structure.

When starting a new section, the program first must close out any existing sections and lists of a deeper level to keep the HTML tags nested correctly.

Support for the `memoir` package will require the addition of a `book` level, which will push the HTML headings down a step, and also cause `subsubsection` to become a `div` due to a limit of six HTML headings.

It is possible to use HTML5 `section` and `H1` for all levels, but this may not be well-recognized by older browsers.



Table 4: Section depths and HTML headings

Section	L <sup>A</sup> T <sub>E</sub> X depth	HTML headings
title of the entire website		h1
none	-5	new for this package
book	-2	<b>not yet used</b>
part	-1	h2
chapter	0	h3
section	1	h4
subsection	2	h5
subsubsection	3	h6
paragraph	4	span class = "paragraph"
subparagraph	5	span class = "subparagraph"
listitem	7	new for this package, used for list items

## 11 Source Code


This is where the documented source code for **lwarp** begins, continuing through the following sections all the way to the change log and index at the end of this document.

The following sections document the actual implementation of the **lwarp** package.

**line numbers** The small numbers at the left end of a line refer to line numbers in the **lwarp.sty** file.

**subjects** Blue-colored tags in the left margin aid in quickly identifying the subject of each paragraph.

**objects** Black-colored tags in the left margin are used to identify programming objects such as files, packages, environments, booleans, and counters. Items without a tag are command macros. Each of these also appears in the index as individual entries, and are also listed together under “files”, “packages”, “environments”, “booleans”, and “counters”.

 **warnings** Special warnings are marked with a warning icon.

**for HTML output:**  
**for PRINT output:**  
**for HTML & PRINT:** Green-colored tags in the left margin show which sections of source code apply to the generation of HTML, print, or both forms of output.

lwarp source code begins on the following page.

## 12 Detecting the T<sub>E</sub>X Engine — pdf<sub>l</sub>at<sub>e</sub>x, lua<sub>l</sub>at<sub>e</sub>x, x<sub>e</sub>lat<sub>e</sub>x

```

1 \RequirePackage{iftex}
2
3 \ifLuaTeX
4 \RequirePackage{luatex85}% until the geometry package is updated
5 \fi

```

## 13 Unicode Input Characters

**for HTML & PRINT:** If using pdf<sub>l</sub>at<sub>e</sub>x, convert a minimal set of Unicode characters. Additional characters may be defined by the user, as needed.

If using and of pdf<sub>l</sub>at<sub>e</sub>x, x<sub>e</sub>lat<sub>e</sub>x, or lua<sub>l</sub>at<sub>e</sub>x, \texttimes is declared for a common multiply symbol.

```

6
7 \RequirePackage{newunicodechar}
8
9 \newunicodechar{×}{\texttimes}
10
11 \ifPDFTeX
12 \newunicodechar{ff}{ff}
13 \newunicodechar{fi}{fi}
14 \newunicodechar{fl}{fl}
15 \newunicodechar{ffi}{ffi}
16 \newunicodechar{ffl}{ffl}
17 \newunicodechar{--}{--}
18 \newunicodechar{-}{-}

```

In verbatim text in PDF<sub>T</sub>ex, preserve upright quotes.

```

19 \RequirePackage{upquote}
20 \else
21 \fi

```

## 14 Handling package options

Pkg **etoolbox** Provides \ifbool and other functions.

```

22 \RequirePackage{etoolbox}[2011/01/03]
23 % requires v2.6 for \BeforeBeginEnvironment, etc.

```

Pkg **ifplatform** Provides \ifwindows to try to automatically detect Windows OS.

---

```

24 \RequirePackage{ifplatform}% sense op-system platform

Pkg  comment  Provides conditional code blocks.

25 \RequirePackage{comment}
26 \excludecomment{testing}

Pkg  kvoptions  Allows key/value package options.

27 \RequirePackage{kvoptions}
28 \SetupKeyvalOptions{family=LWR,prefix=LWR@}

Bool  warpingprint  Set to true/false depending on the package option selections for print/HTML output
Bool  warpingHTML   and mathsvg/mathjax:
    Bool  mathjax
29 \newbool{warpingprint}
30 \newbool{warpingHTML}
31 \newbool{mathjax}

\warpprintonly  {\contents}

Only process the contents if producing printed output.

32 \newcommand{\warpprintonly}[1]{\ifbool{warpingprint}{#1}{}}

\warpHTMLonly  {\contents}

Only process the contents if producing HTML output.

33 \newcommand{\warpHTMLonly}[1]{\ifbool{warpingHTML}{#1}{}}

Env  warpall  Anything in the warpall environment will be generated for print or HTML outputs.

34 \includecomment{warpall}

Env  warpprint  Anything in the warpprint environment will be generated for print output only.

Opt lwarp  warpprint  If the warpprint option is given, boolean warpingprint is true and boolean
warpingHTML is false, and may be used for \ifbool tests.

35 \DeclareVoidOption{warpprint}{%
36 \PackageInfo{lwarp}{Using option 'warpprint'}
37 \includecomment{warpprint}%
38 \excludecomment{warpHTML}%
39 \booltrue{warpingprint}%
40 \boolfalse{warpingHTML}%
41 }

```

Env	<b>warpHTML</b>	Anything in the warpHTML environment will be generated for HTML output only.
Opt lwarp	<b>warpHTML</b>	<p>If the warpHTML option is given, boolean warpingHTML is true and boolean warpingprint is false, and may be used for \ifbool tests.</p> <pre> 42 \DeclareVoidOption{warpHTML}{% 43 \PackageInfo{lwarp}{Using option 'warpHTML'} 44 \excludecomment{warpprint}% 45 \includecomment{warpHTML}% 46 \booltrue{warpingHTML}% 47 \boolfalse{warpingprint}% 48 }</pre>
Opt lwarp	<b>mathsvg</b>	<p>Option mathsvg selects SVG math display: If the mathsvg option is given, boolean mathjax is false, and may be used for \ifbool tests.</p> <pre> 49 \DeclareVoidOption{mathsvg}{% 50 \PackageInfo{lwarp}{Using option 'mathsvg'} 51 \boolfalse{mathjax}% 52 }</pre>
Opt lwarp	<b>mathjax</b>	<p>Option mathjax selects MathJax math display: If the mathjax option is given, boolean mathjax is true, may be used for \ifbool tests.</p> <pre> 53 \DeclareVoidOption{mathjax}{% 54 \PackageInfo{lwarp}{Using option 'mathjax'} 55 \booltrue{mathjax}% 56 }</pre>
Opt lwarp	<b>BaseJobname</b>	<p>Option BaseJobname sets the \BaseJobname for this document.</p> <p>This is the \jobname of the printed version, even if currently compiling the HTML version. I.e. this is the \jobname without _html appended. This is used to set \HomeHTMLFileName if the user did not provide one.</p> <pre> 57 \DeclareStringOption[\jobname]{BaseJobname}</pre> <p>The default is print output, and SVG math if the user chose HTML output.</p> <pre> 58 \includecomment{warpprint}% 59 \excludecomment{warpHTML}% 60 \booltrue{warpingprint}% 61 \boolfalse{warpingHTML}% 62 \boolfalse{mathjax}% 63 64 65 % \ExecuteOptions{warpprint,mathsvg} 66 67 \ProcessKeyvalOptions*\relax</pre>

Assign the `\BaseJobname` if the user hasn't provided one:

```
68 \providecommand*{\BaseJobname}{\LWR@BaseJobname}
```

```
\@nameedef {<name>} {<defn>}
```

Expands `<defn>` before assigning to `<name>`

```
69 \def\@nameedef#1{\expandafter\edef\csname #1\endcsname}
```

## 15 Required packages

These packages are automatically loaded by `lwarp` when generating HTML output. Some of them are also automatically loaded when generating print output, but some are not.

In the document preamble, create a `warpprint` environment, and place inside it any of the following packages which are required and which are labeled as “Print: OK to Load in a `warpprint` environment”. Those packages which are labeled as “Print: Pre-Loaded” need not be placed into the document preamble.

**for HTML & PRINT:** 70 `\begin{warppall}`

See: <http://tex.stackexchange.com/a/47579>.

Detects  $\text{X}\text{\TeX}$  and  $\text{Lua}\text{\TeX}$ :

```
71 \RequirePackage{iftex}
72 \newif\ifxetexorluatex
73 \ifXeTeX
74   \xetexorluatextrue
75 \else
76   \ifLuaTeX
77     \xetexorluatextrue
78   \else
79     \xetexorluatexfalse
80   \fi
81 \fi
```

```
82 \end{warppall}
```

**for HTML output:** 83 `\begin{warppHTML}`

```
84 \ifxetexorluatex
85 % ^^A \usepackage[no-math]{fontspec}
```

The monospaced font is used for HTML tags, so turn off its TeX ligatures and common ligatures:

```
86 \defaultfontfeatures[\rmfamily]{Ligatures={NoCommon,TeX}}
87 \defaultfontfeatures[\sffamily]{Ligatures={NoCommon,TeX}}
88 \defaultfontfeatures[\ttfamily]{Ligatures=NoCommon}
89 \else
```

Pkg **microtype**

**pdflatex only:** Only pre-loaded if pdflatex is being used.

**ligatures** Older browsers don't display ligatures. Turn off letter ligatures, keeping L<sup>A</sup>T<sub>E</sub>X dash and quote ligatures, which may fail on older browsers but at least won't corrupt written words.

```
90 \RequirePackage {microtype}
91
92 \microtypesetup{
93 protrusion=false,
94 expansion=false,
95 tracking=false,
96 kerning=false,
97 spacing=false}
98
99 \DisableLigatures[f,q,t,T,Q]{encoding = *,family = *}

100 \fi

101 \end{warpHTML}
```

Pkg **geometry** Tactics to avoid unwanted page breaks and margin overflow:

- Uses a very long and wide page to minimize page breaks and margin overflow.
- Will also use a scriptsize font.
- Also uses extra space at the margin to avoid HTML tag overflow off the page.
- Will also force a new page before environments.

**for HTML output:**

```
102 \begin{warpHTML}
103 \RequirePackage[paperheight=190in,paperwidth=20in,%
104 left=7in,right=7in,%
105 top=1in,bottom=1in,%
106 ]{geometry}
107 \end{warpHTML}
```

**for HTML & PRINT:**

```
108 \begin{warpall}
```

Pkg **xparse**

L<sup>A</sup>T<sub>E</sub>X3 command argument parsing

```
109 \RequirePackage{xparse}
```

```
110 \end{warpall}
```

**for HTML output:** 111 \begin{warpHTML}

Pkg **expl3**

L<sup>A</sup>T<sub>E</sub>X3 programming

```
112 \RequirePackage{expl3}
```

Pkg **getttitlestring**

Used to emulate \nameref.

```
113 \RequirePackage{getttitlestring}
```

Pkg **everyhook**

everyhook is used to patch paragraph handling.

```
114 \RequirePackage{everyhook}
```

```
115 \end{warpHTML}
```

**for HTML & PRINT:** 116 \begin{warpall}

Pkg **fancyvrb**

Used for Verbatim, verse.

```
117 \RequirePackage{fancyvrb}
```

```
118 \end{warpall}
```

**for HTML output:** 119 \begin{warpHTML}

Pkg **xifthen**

```
120 \RequirePackage{xifthen}
```

Pkg **xstring**

```
121 \RequirePackage{xstring}
```



Pkg **makeidx**

```
122 \RequirePackage{makeidx}  
123 \makeindex
```

Pkg **calc**

```
124 \RequirePackage{calc}
```

Pkg **newfloat**

```
125 \RequirePackage{newfloat}
```

Pkg **caption**

```
126 \RequirePackage{caption}
```

Pkg **subcaption**

```
127 \RequirePackage{subcaption}
```

Pkg **enumitem**

enumitem is patched to support `\newlist` with HTML.

```
128 \RequirePackage{enumitem}  
129 \setlist[itemize]{leftmargin=0em}  
130 \setlist[enumerate]{leftmargin=0em}  
131 \setlist[description]{leftmargin=0em}
```

Pkg **pagenote**

Used to convert footnotes into pagenotes.

```
132 \RequirePackage[continuous]{pagenote}  
133 \makepagenote  
134 \let\LWR@origmakepagenote\makepagenote  
135 \let\LWR@origprintnotes\printnotes  
136 \let\LWR@origpagenote\pagenote  
137 \end{warpHTML}
```

**for HTML & PRINT:** 138 \begin{warpall}

Pkg **titling**

Used for `\maketitle` and the title page. See section 41.

```
139 \RequirePackage{titling}
```

```
140 \end{warpall}
```

for HTML output: 141 \begin{warpHTML}

Pkg    **zref**

Used for cross-references.

```
142 \RequirePackage{zref}
```

Pkg    **amsmath**

Equation numbers are placed to the left for HTML.

`newtxmath` automatically loads `amsmath`, so the options `leqno` and `fleqn` are passed beforehand to be picked up both here and by `newtxmath` if it is used.

```
143 \PassOptionsToPackage{leqno,fleqn}{amsmath}
```

```
144 \RequirePackage{amsmath}
```

Pkg    **environ**

Used to encapsulate math environments for re-use in HTML ALT text.

```
145 \RequirePackage{environ}
```

Pkg    **titleps**

Used to place an HTML comment into the footer of a page below the footnotes. This comment is used for `lateximage` environments, including `math`.

The `nopatches` option prevents `titleps` from trying to patch sectioning commands.

`\pagestyle` and `\thispagestyle` are nullified for HTML output.

```
146 \RequirePackage[nopatches]{titleps}
```

```
\pagestyle    {\style}
```

```
147 \let\LWR@origpagestyle\pagestyle
```

```
148 \renewcommand*\pagestyle}[1]{} 
```

```
\thispagestyle    {\style}
```

```
149 \let\LWR@origthispagestyle\thispagestyle
```

```
150 \renewcommand*\thispagestyle}[1]{} 
```

`\pagenumbering`  $\{\langle commands \rangle\}$

```
151 \let\LWR@origpagenumbering\pagenumbering
152 \renewcommand*\pagenumbering[1]{}
```

Pkg `xfrac`

Patched for HTML use. See section 114.

```
153 \RequirePackage{xfrac}
```

Used to convert lengths for image width/height options.

```
154 \RequirePackage{printlen}
```

```
155 \end{warpHTML}
```

## 16 Loading Packages

for HTML output: 156 `\begin{warpHTML}`

Remember the original `\RequirePackage`:

```
157 \let\LWR@origRequirePackage\RequirePackage
```

`\LWR@requirepackagenames` Stores the list of required package names.

```
158 \newcommand*\LWR@requirepackagenames{}
```

`\LWR@findword`  $[\langle 1: separator \rangle] \{\langle 2: list \rangle\} \{\langle 3: index \rangle\} [\langle 4: destination \rangle]$

Note that argument 4 is passed directly to `\StrBetween`.

```
159 \newcommand*\LWR@findword[3][,]{%
160   \StrBetween[#3,\numexpr#3+1]{#1#2#1}{#1}{#1}%
161 }
```

`\LWR@lookforpackagename`  $\{\langle index \rangle\}$  If this is a package name, re-direct it to the lwarp version by renaming it `lwarp-` followed by the original name.

```
162 \newcommand*\LWR@lookforpackagename[1]{
```

Find the  $n$ 'th package name from the list:

```
163 \LWR@findword{\LWR@requirepackagenames}{#1}[\LWR@strresult]
```

See if the package name was found:

```
164 \IfStrEq{\LWR@strresult}{}  
165 {}% no filename  
166 {}% yes filename
```

If found, and if an `lwarp`-equivalent name exists, use `lwarp-*` instead.

```
167 \IfFileExists{lwarp-\LWR@strresult.sty}  
168 {}% latex_html_ file found  
169 \StrSubstitute  
170 {\LWR@requirepackagenames}  
171 {\LWR@strresult}  
172 {lwarp-\LWR@strresult}[\LWR@requirepackagenames]  
173 }  
174 {}% no latex_html_* file  
175 {}% yes filename  
176 }
```

`\RequirePackage` [ $\langle 1: options \rangle$ ] [ $\langle 2: package names \rangle$ ] [ $\langle 3: version \rangle$ ]

For each of many package names in a comma-separated list, if an `lwarp` version of a package exists, select it instead of the L<sup>A</sup>T<sub>E</sub>X version.

```
177 \RenewDocumentCommand{\RequirePackage}{o m o}{%
```

Redirect up to nine names:

```
178 \renewcommand*\LWR@requirepackagenames{#2}  
179 \LWR@lookforpackagename{1}  
180 \LWR@lookforpackagename{2}  
181 \LWR@lookforpackagename{3}  
182 \LWR@lookforpackagename{4}  
183 \LWR@lookforpackagename{5}  
184 \LWR@lookforpackagename{6}  
185 \LWR@lookforpackagename{7}  
186 \LWR@lookforpackagename{8}  
187 \LWR@lookforpackagename{9}
```

`\RequirePackage` depending on the options and version:

```
188 \IfValueTF{#1}  
189 {}% options given  
190 \IfValueTF{#3}% version given?
```

```

191 {\LWR@origRequirePackage[#1]{\LWR@requirepackagenames}[#3]}
192 {\LWR@origRequirePackage[#1]{\LWR@requirepackagenames}}
193 }
194 {% no options given
195 \IfValueTF{#3}% version given?
196 {\LWR@origRequirePackage{\LWR@requirepackagenames}[#3]}
197 {\LWR@origRequirePackage{\LWR@requirepackagenames}}
198 }
199 }
200 \let\usepackage\RequirePackage

```

`\LWR@ProvidesPackagePass`     $\{\langle pkgname \rangle\}$  [ $\langle version \rangle$ ]

Provides a new package which passes its options to the original.

```

201 \NewDocumentCommand{\LWR@ProvidesPackagePass}{m o}{
202 \PackageInfo{lwarp}{Using the lwarp version of package ‘#1’, including options,}%
203 \IfValueTF{#2}
204 {\ProvidesPackage{lwarp-#1}[#2]}
205 {\ProvidesPackage{lwarp-#1}}
206 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{#1}}
207 \ProcessOptions\relax
208
209 \IfValueTF{#2}
210 {\LWR@origRequirePackage{#1}[#2]}
211 {\LWR@origRequirePackage{#1}}
212 }

```

`\LWR@ProvidesPackageDrop`     $\{\langle pkgname \rangle\}$  [ $\langle version \rangle$ ]

Provides a new package which drops/discards its options.

```

213 \NewDocumentCommand{\LWR@ProvidesPackageDrop}{m o}{
214 \PackageInfo{lwarp}{Using the lwarp version of package ‘#1’, discarding options,}%
215 \IfValueTF{#2}
216 {\ProvidesPackage{lwarp-#1}[#2]}
217 {\ProvidesPackage{lwarp-#1}}
218 \DeclareOption*{}
219 \ProcessOptions\relax
220 }

221 \end{warpHTML}

```

## 17 Copying a file

for HTML output: 222 \begin{warpHTML}

`\LWR@copyfile`  $\{\langle source\ filename\rangle\} \{\langle destination\ filename\rangle\}$

Used to copy the .toc file to .sidetoc to re-print the TOC in the sideTOC navigation pane.

```

223 \newcommand*\LWR@copyfile}[2]{%
224 \newwrite\copyfile % open the file to write to
225 \immediate\openout\copyfile=#2
226 \newread\file % open the file to read from
227 \openin\file=#1
228 \begingroup\endlinechar=-1
229 \makeatletter
230 \loop\unless\ifeof\file
231 \read\file to\fileline % Read one line and store it into \fileline
232 % \fileline\par % print the content into the pdf
233 % print the content:
234 \immediate\write\copyfile{\unexpanded\expandafter{\fileline}}%
235 \repeat
236 \closeout\copyfile
237 \endgroup
238 }

239 \end{warpHTML}

```

## 18 Debugging messages

Bool `LWR@tracinglwarp` True if tracing is turned on.

```
240 \newbool{LWR@tracinglwarp}
```

`\tracinglwarp` Turns on the debug tracing messages.

```
241 \newcommand{\tracinglwarp}{\booltrue{LWR@tracinglwarp}}
```

`\LWR@traceinfo`  $\{\langle text\rangle\}$  If tracing is turned on, writes the text to the .log file.

```

242 \newcommand{\LWR@traceinfo}[1]{%
243 \ifbool{LWR@tracinglwarp}%
244 {\PackageInfo{lwarp}{#1 : }}%
245 {}%
246 }

```

## 19 Remembering original formatting macros

for HTML output: 247 \begin{warpHTML}

Remember original definitions of formatting commands. Will be changed to HTML commands for most uses. Will be temporarily restored to original meaning inside any lateximage environment. Also nullify unused commands.

```

248 \let\LWR@origtextit\textit
249 \let\LWR@origtextbf\textbf
250 \let\LWR@origtexttt\texttt
251 \let\LWR@origtextsc\textsc
252 \let\LWR@origtextsf\textsf
253 \let\LWR@origtextrm\textrm
254 \let\LWR@origbfseries\bfseries
255 \let\LWR@origrmfamily\rmfamily
256 \let\LWR@origttfamily\ttfamily
257 \let\LWR@orignormalfont\normalfont
258
259 \let\LWR@origraggedright\raggedright
260 \let\LWR@origonecolumn\onecolumn
261
262 \let\LWR@origtextsuperscript\textsuperscript
263 \let\LWR@origtextsubscript\textsubscript
264
265 \let\LWR@origscriptsize\scriptsize
266
267 \let\LWR@orignewpage\newpage
268
269 \let\LWR@origminipage\minipage
270 \let\LWR@origendminipage\endminipage
271
272 \let\LWR@orignewline\newline
273
274 \let\LWR@origitem\item
275
276 \let\LWR@origpar\par
277
278
279 \let\LWR@origfootnote\footnote
280 \let\LWR@orig@mpfootnotetext\@mpfootnotetext
281
282 \let\LWR@origclearpage\clearpage
283 \let\clearpage\relax
284 \let\cleardoublepage\relax
285 \end{warpHTML}

```

## 20 Operating-System portability

lwarp tries to detect which operating system is being used.

Prog MS-Windows Prog Windows \warpOSWindows	If MS-WINDOWS is not correctly detected, use the command <code>\warpOSWindows</code> in the document preamble after lwarp is loaded. This modifies the creation of the <code>lateximages.sh</code> or <code>lateximages.cmd</code> batch file, which is an operating-system-specific shell script used to take individual pages of the L <sup>A</sup> T <sub>E</sub> X PDF output and convert and store them in individual SVG files.
---	---

for HTML & PRINT: 286 `\begin{warpall}`

### 20.1 Common portability code

Bool usingOSWindows	Set if <code>\warpOSWindows</code> .  287 <code>\newbool{usingOSWindows}</code> 288 <code>\boolfalse{usingOSWindows}</code>
---------------------	--

### 20.2 Unix, Linux, and Mac OS

\OSPathSymbol	Symbol used to separate directories in a path.  289 <code>\newcommand*{\OSPathSymbol}{/}</code>
\OSmkdir	The make-directory command.  290 <code>\newcommand*{\OSmkdir}{mkdir } % trailing space is required</code>
\OSrm	The remove-file command.  291 <code>\newcommand*{\OSrm}{rm }</code>
\OSmv	The move-file command.  292 <code>\newcommand*{\OSmv}{mv }</code>
\OSShellPath	Path to the command-line shell.  293 <code>\newcommand*{\OSShellPath}{/bin/sh}</code>



## 20.3 MS-Windows

For MS-Windows:

`\warpOSwindows` Set defaults for the MS-Windows operating system. `lwarp` attempts to auto-detect the operating system, so `\warpOSwindows` may not be necessary.

```
294 \newcommand*\warpOSwindows{
295 {
296 \booltrue{usingOSwindows}
297 \renewcommand*\OSPathSymbol{\@backslashchar}
298 \renewcommand*\OSmkdir}{MD }
299 \renewcommand*\OSrm}{DEL }
300 \renewcommand*\OSmv}{MOVE }
301 }
```

Test for windows during compile. The user may also specify `\warpOSwindows` later in case this test fails.

```
302 \ifwindows
303 \warpOSwindows
304 \fi

305 \end{warpall}
```

## 21 Stacks

for HTML output: 306 `\begin{warphTML}`



Stacks are used to remember how to close sections and list items. Before a new section is started, previously nested sections and items must be closed out (unnested) in proper order. Note that starting a new section may close several levels of previously nested items at the same time. For example, starting a new `\section` would close any currently open subsection, subsubsection, and paragraph. General environments are not nested on the stack since they have their own close mechanism. List environments are nested, and items inside those environments are nested one level deeper still. List environments may be nested inside other list environments, and list items are nested inside list environments as well. Thus, the stack may have items which are not necessarily in order, since a description may contain an enumerate, for example. Depths to be recorded in `\LWR@closedepth`, etc.

## 21.1 Assigning depths

initial depths for empty stack entries:

```
307 \newcommand*{\LWR@depthnone}{-5}
```

all sectioning depths are deeper than LWR@depthfinished:

```
308 \newcommand*{\LWR@depthfinished}{-4}
309 \newcommand*{\LWR@depthpart}{-1}
310 \newcommand*{\LWR@depthchapter}{0}
311 \newcommand*{\LWR@depthsection}{1}
312 \newcommand*{\LWR@depthsubsection}{2}
313 \newcommand*{\LWR@depthsubsubsection}{3}
314 \newcommand*{\LWR@depthparagraph}{4}
315 \newcommand*{\LWR@depthsubparagraph}{5}
```

used by \itemize, \enumerate, \description:

```
316 \newcommand*{\LWR@depthlist}{6}
```

used by \item:

```
317 \newcommand*{\LWR@depthlistitem}{7}
```

## 21.2 Closing actions

A stack to record the action to take to close each nesting level: Add more levels of stack if necessary for a very deeply nested document, adding to \pushclose and \popclose as well.

```
318 \newcommand*{\LWR@closeone}{}% top of the stack
319 \newcommand*{\LWR@closetwo}{}
320 \newcommand*{\LWR@closethree}{}
321 \newcommand*{\LWR@closefour}{}
322 \newcommand*{\LWR@closefive}{}
323 \newcommand*{\LWR@closesix}{}
324 \newcommand*{\LWR@closeseven}{}
325 \newcommand*{\LWR@closeeight}{}
326 \newcommand*{\LWR@closenine}{}
327 \newcommand*{\LWR@closeten}{}
328 \newcommand*{\LWR@closeeleven}{}
329 \newcommand*{\LWR@closetwelve}{}%
```

## 21.3 Closing depths

A stack to record the depth of each level:



Note that nested LaTeX structures may push depths which are non-sequential.

Ex:

---

```
\begin{itemize}
  \item{A}
  \begin{description}
    \item{B}
  \end{description}
\end{itemize}
```

---

```
330 \newcommand*{\LWR@closedepthone}{\LWR@depthnone}% top of the stack
331 \newcommand*{\LWR@closedepthtwo}{\LWR@depthnone}
332 \newcommand*{\LWR@closedepththree}{\LWR@depthnone}
333 \newcommand*{\LWR@closedepthfour}{\LWR@depthnone}
334 \newcommand*{\LWR@closedepthfive}{\LWR@depthnone}
335 \newcommand*{\LWR@closedepthsix}{\LWR@depthnone}
336 \newcommand*{\LWR@closedepthseven}{\LWR@depthnone}
337 \newcommand*{\LWR@closedeptheight}{\LWR@depthnone}
338 \newcommand*{\LWR@closedepthnine}{\LWR@depthnone}
339 \newcommand*{\LWR@closedepthten}{\LWR@depthnone}
340 \newcommand*{\LWR@closedeptheleven}{\LWR@depthnone}
341 \newcommand*{\LWR@closedepthtwelve}{\LWR@depthnone}
```

## 21.4 Pushing and popping the stack

`\pushclose`  $\{\langle action \rangle\} \{\langle depth \rangle\}$

Pushes one return action and its LaTeX depth onto the stacks.

```
342 \NewDocumentCommand{\pushclose}{m m}
343 {
344 \let\LWR@closetwelve\LWR@closeeleven
345 \let\LWR@closeeleven\LWR@closeten
346 \let\LWR@closeten\LWR@closenine
347 \let\LWR@closenine\LWR@closeeight
348 \let\LWR@closeeight\LWR@closeseven
349 \let\LWR@closeseven\LWR@closesix
350 \let\LWR@closesix\LWR@closefive
351 \let\LWR@closefive\LWR@closefour
352 \let\LWR@closefour\LWR@closethree
```

```

353 \let\LWR@closethree\LWR@closetwo
354 \let\LWR@closetwo\LWR@closeone
355 \let\LWR@closeone#1
356 \let\LWR@closedepthtwelve\LWR@closedeptheleven
357 \let\LWR@closedeptheleven\LWR@closedepthten
358 \let\LWR@closedepthten\LWR@closedepthnine
359 \let\LWR@closedepthnine\LWR@closedeptheight
360 \let\LWR@closedeptheight\LWR@closedepthseven
361 \let\LWR@closedepthseven\LWR@closedepthsix
362 \let\LWR@closedepthsix\LWR@closedepthfive
363 \let\LWR@closedepthfive\LWR@closedepthfour
364 \let\LWR@closedepthfour\LWR@closedepththree
365 \let\LWR@closedepththree\LWR@closedepthtwo
366 \let\LWR@closedepthtwo\LWR@closedepthone
367 \let\LWR@closedepthone#2
368 }

```

`\popclose` Pops one action and its depth off the stacks.

```

369 \newcommand*{\popclose}
370 {
371 \let\LWR@closeone\LWR@closetwo
372 \let\LWR@closetwo\LWR@closethree
373 \let\LWR@closethree\LWR@closefour
374 \let\LWR@closefour\LWR@closefive
375 \let\LWR@closefive\LWR@closesix
376 \let\LWR@closesix\LWR@closeseven
377 \let\LWR@closeseven\LWR@closeeight
378 \let\LWR@closeeight\LWR@closenine
379 \let\LWR@closenine\LWR@closeten
380 \let\LWR@closeten\LWR@closeeleven
381 \let\LWR@closeeleven\LWR@closetwelve
382 \let\LWR@closedepthone\LWR@closedepthtwo
383 \let\LWR@closedepthtwo\LWR@closedepththree
384 \let\LWR@closedepththree\LWR@closedepthfour
385 \let\LWR@closedepthfour\LWR@closedepthfive
386 \let\LWR@closedepthfive\LWR@closedepthsix
387 \let\LWR@closedepthsix\LWR@closedepthseven
388 \let\LWR@closedepthseven\LWR@closedeptheight
389 \let\LWR@closedeptheight\LWR@closedepthnine
390 \let\LWR@closedepthnine\LWR@closedepthten
391 \let\LWR@closedepthten\LWR@closedeptheleven
392 \let\LWR@closedeptheleven\LWR@closedepthtwelve
393 }

394 \end{warpHTML}

```

## 22 Data arrays

These macros are similar to the `arrayjobx` package, except that `\LWR@setexparray`'s argument is expanded only once when assigned.

`name` has no backslash, `index` can be a number or a text name, and an empty `value` must be `\relax` instead of empty.

To assign an empty value:

```
\LWR@setexparray{name}{index}{\relax}
```

for HTML output: 395 \begin{warpHTML}

```
\LWR@setexparray {\langle name \rangle} {\langle index \rangle} {\langle contents \rangle}
```

```
396 \NewDocumentCommand{\LWR@setexparray}{m m m}{%
397 \expandafter\edef\csname #1#2\endcsname{\expandonce#3}%
398 }
```

```
\LWR@getexparray {\langle name \rangle} {\langle index \rangle}
```

```
399 \newcommand*{\LWR@getexparray}[2]{\csuse{#1#2}}
```

```
400 \end{warpHTML}
```

## 23 HTML entities

for HTML output: 401 \begin{warpHTML}

HTML entites and HTML Unicode entities:

```
402 \let\LWR@origampersand\&
```

```
\HTMLentity {\langle entitytag \rangle}
```

```
403 \newcommand*{\HTMLentity}[1]{\LWR@origampersand#1;}
```

```
\HTMLunicode {\langle hex_unicode \rangle}
```

```
404 \newcommand*{\HTMLunicode}[1]{\HTMLentity{\#x#1}}
```

```
\&
```

```
405 \renewcommand*{\&}{\HTMLentity{amp}}
```

```

\textless
\textgreater
406 \let\LWR@origtextless\textless
407 \renewcommand*{\textless}{\HTMLentity{lt}}
408
409 \let\LWR@origtextgreater\textgreater
410 \renewcommand*{\textgreater}{\HTMLentity{gt}}

411 \end{warpHTML}

```

## 24 HTML filename generation

The filename of the homepage is set to `\HomeHTMLFileName.html`. The filenames of additional sections start with `\HTMLFileName`, to which is appended a section number or a simplified section name, depending on `FileSectionNames`.

**for HTML & PRINT:** 412 `\begin{warpall}`

`\BaseJobname` The `\jobname` of the printed version, even if currently compiling the HTML version. I.e. this is the `\jobname` without `_html` appended. This is used to set `\HomeHTMLFileName` if the user did not provide one.

```
413 \providecommand*{\BaseJobname}{\jobname}
```

`\HTMLFileName` The prefix for all generated HTML files other than the home page, defaulting to empty. See section 5.7.

```
414 \providecommand*{\HTMLFileName}{}
```

`\HomeHTMLFileName` The filename of the home page, defaulting to the `\BaseJobname`. See section 5.7.

```
415 \providecommand*{\HomeHTMLFileName}{\BaseJobname}
```

`\SetHTMLFileNumber` `{\langle number \rangle}`

Sets the file number for the next file to be generated. 0 is the home page. Use just before the next sectioning command, and set it to one less than the desired number of the next section. May be used to generate numbered groups of nodes such as 100+ for one chapter, 200+ for another chapter, etc.

```
416 \newcommand*{\SetHTMLFileNumber}[1]{%
417 \setcounter{LWR@htmlfilenumber}{#1}%
418 }
```

Bool FileSectionNames Selects how to create HTML file names.

Defaults to use section names in the filenames.

```
419 \newbool{FileSectionNames}
420 \booltrue{FileSectionNames}

421 \end{warpall}
```

for HTML output: 422 \begin{warpHTML}

Ctr LWR@htmlfilenumber Records the number of each HTML file as it is being created. Number 0 is the home page.

```
423 \newcounter{LWR@htmlfilenumber}
424 \setcounter{LWR@htmlfilenumber}{0}
```

\LWR@htmlsectionfilename *{(htmlfilenumber or name)}*

Prints the filename for a given section: \HTMLFileName{ }filenumber/name.html

```
425 \newcommand*{\LWR@htmlsectionfilename}[1]{%
426 \LWR@traceinfo{LWR@htmlsectionfilename A}%
```

Section 0 or empty is given the home filename. The filename must be detokenized for underscores.

```
427 \LWR@traceinfo{about to assign temp}%
428 \edef\LWR@tempone{#1}%
429 \LWR@traceinfo{about to compare with ??}%
430 \ifthenelse{\equal{\LWR@tempone}{??}}{%
431 {%
432 \LWR@traceinfo{found ??}%
433 }{%
434 \LWR@traceinfo{not found ??}%
435 }%
436 \LWR@traceinfo{about to compare with zero or empty}%
437 \ifthenelse{%
438 \equal{\LWR@tempone}{0}}{%
439 \OR \equal{\LWR@tempone}{}}%
440 \OR \equal{\LWR@tempone}{??}}%
441 }%
442 {%
443 \LWR@traceinfo{LWR@htmlsectionfilename B \HomeHTMLFileName.html}%
444 \HomeHTMLFileName.html%
445 }%
```

For a L<sup>A</sup>T<sub>E</sub>X section named “Index” or “index” without a prefix, create a filename with a leading underscore to avoid colliding with the HTML filename `index.html`:

```

446 {%
447 \LWR@traceinfo{LWR@htmlsectionfilename C}%
448 \ifthenelse{%
449 \equal{\HTMLFileName}{ } \AND \equal{\LWR@tempone}{Index} \OR \equal{\LWR@tempone}{index}%
450 }%
451 {\_#1.html}%

```

Otherwise, create a filename with the chosen prefix:

```

452 {\HTMLFileName#1.html}%
453 }%
454 \LWR@traceinfo{LWR@htmlsectionfilename Z}%
455 }

```

`\LWR@htmlrefsectionfilename`  $\{ \langle label \rangle \}$

Prints the filename for the given label

```

456 \newcommand*{\LWR@htmlrefsectionfilename}[1]{%
457 \LWR@traceinfo{LWR@htmlrefsectionfilename A: !#1!}%
458 \LWR@htmlsectionfilename{\LWR@htmlfileref{#1}}%
459 \LWR@traceinfo{LWR@htmlrefsectionfilename B}%
460 }

461 \end{warpHTML}

```

## 25 Homepage link

**for HTML output:** 462 `\begin{warpHTML}`

`\LinkHome` `\LinkHome` may be used wherever you wish to place a link back to the homepage. The filename must be detokenized for underscores.

```

463 \newcommand*{\LinkHome}{%
464 \LWR@subhyperrefclass{%
465 \HomeHTMLFileName.html}%
466 {Home}{linkhome}%
467 }

```

`\LWR@topnavigation` Creates a link to the homepage at the top of the page for use when the window is too narrow for the sidetoc.



```

468 \newcommand*{\LWR@topnavigation}{
469 \LWR@html@elementclassline{nav}{topnavigation}{\LinkHome}
470 }

```

**\LWR@botnavigation** Creates a link to the homepage at the bottom of the page for use when the window is too narrow for the sidetoc.

```

471 \newcommand*{\LWR@botnavigation}{
472 \LWR@html@elementclassline{nav}{botnavigation}{\LinkHome}
473 }

```

```

474 \end{warpHTML}

```

## 26 \PrintStack diagnostic tool



Diagnostics tool: Prints the LaTeX nesting depth values for the stack levels. Must have **\LWR@startpars** active while printing the stack, so **\PrintStack** may be called from anywhere in the normal text flow.

**for HTML output:** 475 \begin{warpHTML}

**\PrintStack** Prints the closedepth stack.

```

476 \newcommand*{\PrintStack}{
477 \LWR@startpars
478 \LWR@closedepthone{} \LWR@closedepthtwo{} \LWR@closedepththree{}
479 \LWR@closedepthfour{} \LWR@closedepthfive{} \LWR@closedepthsix{}
480 \LWR@closedepthseven{} \LWR@closedeptheight{} \LWR@closedepthnine{}
481 \LWR@closedephten{} \LWR@closedeptheleven{} \LWR@closedephtwelve{}
482 }

```

```

483 \end{warpHTML}

```

## 27 Closing stack levels

**for HTML output:** 484 \begin{warpHTML}

Close one nested level:

```

485 \newcommand*{\LWR@closeoneprevious}{%
486
487 \LWR@closeone{

```

```

488
489 \popclose{}
490 }

```

`\LWR@closeprevious`  $\{ \langle depth \rangle \}$  Close everything up to the given depth:

```

491 \newcommand*{\LWR@closeprevious}[1]{
    Close any pending paragraph:

492 \LWR@stoppars

    Close anything nested deeper than the desired depth:

493 \whiledo{\not\(\LWR@closedepthone<#1\)}{\LWR@closeoneprevious}
494 }

495 \end{warpHTML}

```

## 28 HTML tags, spans, divs, elements

for HTML output: 496 \begin{warpHTML}

### 28.1 Mapping L<sup>A</sup>T<sub>E</sub>X Sections to HTML Sections

```

497 \newcommand*{\LWR@tagpart}{h2}
498 \newcommand*{\LWR@tagpartend}{/h2}
499 \newcommand*{\LWR@tagchapter}{h3}
500 \newcommand*{\LWR@tagchapterend}{/h3}
501 \newcommand*{\LWR@tagsection}{h4}
502 \newcommand*{\LWR@tagsectionend}{/h4}
503 \newcommand*{\LWR@tagsubsection}{h5}
504 \newcommand*{\LWR@tagsubsectionend}{/h5}
505 \newcommand*{\LWR@tagsubsubsection}{h6}
506 \newcommand*{\LWR@tagsubsubsectionend}{/h6}
507 \newcommand*{\LWR@tagparagraph}{span class="paragraph">{}}
508 \newcommand*{\LWR@tagparagraphend}{/span}
509 \newcommand*{\LWR@tagsubparagraph}{span class="subparagraph">{}}
510 \newcommand*{\LWR@tagsubparagraphend}{/span}
511
512 \newcommand*{\LWR@tagregularparagraph}{p}

```

## 28.2 HTML tags

`\LWR@htmltagc`  $\{\langle tag \rangle\}$  Break ligatures in HTML tags.

`\protect` is in case the tag appears in TOC, LOF, LOT.

```

513 \newcommand*{\LWR@htmltagc}[1]{%
514 {%
515 \LWR@origttfamily%
516 % \protect\LWR@origtexttt{%
517 % \protect\LWR@origtextbf{%
518 \LWR@origtextless#1\LWR@origtextgreater%
519 % }%
520 % }%
521 }%
522 }
```

Env `LWR@nestspan` Disable minipage, `\parbox` inside a `span`.

⚠ `\begin{LWR@nestspan}` must follow the opening `<span>` tag to allow a paragraph to start if the span is at the beginning of a new paragraph.

⚠ `\end{LWR@nestspan}` must follow the `/span` or an extra `<p>` may appear.

```

523 \newenvironment*{LWR@nestspan}
524 {%
525 \addtocounter{LWR@spandepth}{1}%
526 \RenewDocumentEnvironment{minipage}{0{t} o 0{t} m}{-}{-}%
527 }%
528 {\addtocounter{LWR@spandepth}{-1}}
```

`\LWR@htmlspan`  $\{\langle tag \rangle\} \{\langle text \rangle\}$

⚠ `\LWR@spandepth` is used to ensure that paragraph tags are not generated inside a `span`. The exact sequence of when to add and subtract the counter is important to correctly handle the paragraph tags before and after the span.

```

529 \NewDocumentCommand{\LWR@htmlspan}{m +m}{%
530 \LWR@ensuredoingapar%
531 \LWR@htmltagc{#1}%
532 \begin{LWR@nestspan}%
533 #2%
534 \LWR@htmltagc{/#1}%
535 \end{LWR@nestspan}%
536 }
```

`\LWR@htmlspanclass`  $\{\langle class \rangle\} [\langle style \rangle] \{\langle text \rangle\}$

```

537 \NewDocumentCommand{\LWR@htmlspanclass}{m o +m}{%
538 \LWR@ensuredoingapar%
539 \LWR@subhtmlmlelementclass{span}{#1}{#2}%
540 \begin{LWR@nestspan}%
541 #3%
542 \LWR@htmltagc{/span}%
543 \end{LWR@nestspan}%
544 }

```

`\LWR@htmltag`    `{\langle tag \rangle}`

print an HTML tag: `<tag>`

```

545 \newcommand*{\LWR@htmltagb}[1]{%
546 \LWR@htmltagc{#1}%
547 \endgroup%
548 }
549
550 \newcommand*{\LWR@htmltag}{%
551 \begingroup\catcode'\_ =12
552 \LWR@htmltagb%
553 }

```

### 28.3 Block tags and comments

In the following, `\origtexttt` breaks ligatures, which may not be used for HTML codes:

`\LWR@htmlopencomment`  
`\LWR@htmlclosecomment`

```

554 \newcommand*{\LWR@htmlopencomment}{%
555 % \LWR@origtexttt{\LWR@origtextbf{\LWR@origtextless{!}{-}{-}}}%
556 \begingroup\LWR@origttfamily\LWR@origtextless{!}{-}{-}\endgroup%
557 }
558
559 \newcommand*{\LWR@htmlclosecomment}{%
560 % \LWR@origtexttt{\LWR@origtextbf{{-}{-}\LWR@origtextgreater{}}}%
561 \begingroup\LWR@origttfamily{-}{-}\LWR@origtextgreater{}\endgroup%
562 }

```

`\LWR@htmlcomment`    `{\langle comment \rangle}`

```

563 \newcommand{\LWR@htmlcomment}[1]{%
564 \LWR@htmlopencomment}%
565 \LWR@origtextrm{#1}%
566 \LWR@htmlclosecomment}%

```

`\LWR@htmlblockcomment`  $\{\langle comment \rangle\}$

```

567 \newcommand{\LWR@htmlblockcommentb}[1]
568 {\LWR@stoppars\LWR@htmlcomment{#1}\LWR@startpars\endgroup}
569
570 \newcommand{\LWR@htmlblockcomment}
571 {%
572 \begingroup\catcode'\_ =12%
573 \LWR@htmlblockcommentb%
574 }
```

`\LWR@htmlblocktag`  $\{\langle tag \rangle\}$  print a stand-alone HTML tag

```

575 \newcommand*\LWR@htmlblocktag[1]{%
576 \LWR@stoppars%
577 \LWR@htmltag{#1}%
578 \LWR@startpars%
579 }
```

`\LWR@htmlblocktagcomment`  $\{\langle tag \rangle\} \{\langle HTML comment \rangle\}$

```

580 \NewDocumentCommand{\LWR@htmlblocktagcomment}{m +m}{%
581 \LWR@stoppars%
582 \LWR@htmltag{#1}\LWR@htmlcomment{#2}\LWR@orignewline%
583 \LWR@startpars%
584 }
```

## 28.4 Div class and element class

`\LWR@subhtml elementclass`  $\{\langle element \rangle\} \{\langle class \rangle\} [\langle style \rangle]$

Factored and reused in several places.

```

585 \NewDocumentCommand{\LWR@subhtml elementclass}{m m o}{%
586 \IfValueTF{#3}%
587 {% option
588 \ifthenelse{\equal{#3}{}}{%
589 {\LWR@htmltag{#1 class="#2"}}% empty option
590 {\LWR@htmltag{#1 class="#2" style="#3"}}% non-empty option
591 }% option
592 {\LWR@htmltag{#1 class="#2"}}% no option
593 }
```

`\LWR@html elementclass`  $\{\langle element \rangle\} \{\langle class \rangle\} [\langle style \rangle]$

```

594 \NewDocumentCommand{\LWR@htmlelementclass}{m m o}{%
595 \LWR@stoppars%
596 \LWR@subhtmlelementclass{#1}{#2}[#3]%
597 \LWR@startpars%
598 }

```

`\LWR@htmlelementclassend`     $\{\langle element \rangle\} \{\langle class \rangle\}$

```

599 \newcommand*{\LWR@htmlelementclassend}[2]{%
600 \LWR@stoppars%
601 \LWR@htmltag{/#1}%
602 \LWR@htmlcomment{End of #1 ‘‘#2’’}%
603 \LWR@startpars%
604 }

```

`\LWR@htmldivclass`     $\{\langle class \rangle\} [\langle style \rangle]$

```

605 \NewDocumentCommand{\LWR@htmldivclass}{m o}{%
606 \LWR@htmlelementclass{div}{#1}[#2]%
607 }

```

`\LWR@htmldivclassend`     $\{\langle class \rangle\}$

```

608 \newcommand*{\LWR@htmldivclassend}[1]{%
609 \LWR@htmlelementclassend{div}{#1}%
610 }

```

## 28.5 Single-line elements

A single-line element, without a paragraph tag for the line of text:

`\LWR@htmlelementclassline`     $\{\langle element \rangle\} \{\langle class \rangle\} [\langle style \rangle] \{\langle text \rangle\}$

```

611 \NewDocumentCommand{\LWR@htmlelementclassline}{m m o +m}{%
612 \LWR@stoppars
613 \LWR@subhtmlelementclass{#1}{#2}[#3]%
614 #4%
615 \LWR@htmltag{/#1}
616 \LWR@startpars
617 }

```

## 28.6 HTML5 semantic elements

`\LWR@htmlelement`  $\{\langle element \rangle\}$

```
618 \newcommand*\LWR@htmlelement}[1]{%
619 \LWR@htmlblocktag{#1}
620 }
```

`\LWR@htmlelementend`  $\{\langle element \rangle\}$

```
621 \newcommand*\LWR@htmlelementend}[1]{%
622 \LWR@stoppars
623 \LWR@htmltag{/#1}
624 \LWR@startpars
625 }
626
627 \end{warpHTML}
```

## 28.7 High-level block and inline classes

These are high-level commands which allow the creation of arbitrary block or inline sections which may be formatted with CSS.

For other direct-formatting commands, see section [62](#).

`Env` `BlockClass`  $\{\langle class \rangle\}$   $[\langle style \rangle]$  High-level interface for div classes.

Ex: `\begin{BlockClass}{class} text \end{BlockClass}`

**for HTML output:**

```
628 \begin{warpHTML}
629 \NewDocumentEnvironment{BlockClass}{m o}
630 {
631 \LWR@htmldivclass{#1}[#2]
632 }
633 {
634 \LWR@htmldivclassend{#1}
635 }
636 \end{warpHTML}
```

**for PRINT output:**

```
637 \begin{warpprint}
638 \NewDocumentEnvironment{BlockClass}{m o}{\}{\}
639 \end{warpprint}
```

`\BlockClassSingle`  $\{\langle class \rangle\}$   $\{\langle text \rangle\}$  A single-line div, without a paragraph tag for the line of text.

**for HTML output:**

```

640 \begin{warpHTML}
641 \newcommand{\BlockClassSingle}[2]{%
642 \LWR@html@element@classline{div}{#1}{#2}%
643 }
644 \end{warpHTML}

```

for PRINT output:

```

645 \begin{warpprint}
646 \newcommand{\BlockClassSingle}[2]{#2}
647 \end{warpprint}

```

`\InlineClass` `{\langle class \rangle} [\langle style \rangle] {\langle text \rangle}` High-level interface for inline span classes.

for HTML output:

```

648 \begin{warpHTML}
649 \NewDocumentCommand{\InlineClass}{m o m}{%
650 \LWR@html@spanclass{#1}{#2}{#3}%
651 }
652 \end{warpHTML}

```

for PRINT output:

```

653 \begin{warpprint}
654 \NewDocumentCommand{\InlineClass}{m o m}{#3}
655 \end{warpprint}

```

## 28.8 Closing HTML tags

for HTML output:

```

656 \begin{warpHTML}

```

Sections H1, H2, etc. do not need a closing HTML tag, but we add a comment for readability:

```

657 \newcommand*{\LWR@printclosepart}
658     {\LWR@htmlcomment{Closing part}}
659 \newcommand*{\LWR@printclosechapter}
660     {\LWR@htmlcomment{Closing chapter}}
661 \newcommand*{\LWR@printclosesection}
662     {\LWR@htmlcomment{Closing section}}
663 \newcommand*{\LWR@printclosesubsection}
664     {\LWR@htmlcomment{Closing subsection}}
665 \newcommand*{\LWR@printclosesubsubsection}
666     {\LWR@htmlcomment{Closing subsubsection}}
667 \newcommand*{\LWR@printcloseparagraph}
668     {\LWR@htmlcomment{Closing paragraph}}
669 \newcommand*{\LWR@printclose subparagraph}
670     {\LWR@htmlcomment{Closing subparagraph}}

```

Lists require closing HTML tags:



```

671 \newcommand*{\LWR@printcloselistitem}
672     {\LWR@htmltag{/li}}
673 \newcommand*{\LWR@printclosedescitem}
674     {\LWR@htmltag{/dd}}
675 \newcommand*{\LWR@printcloseitemize}
676     {\LWR@htmltag{/ul}}
677 \newcommand*{\LWR@printcloseenumerate}
678     {\LWR@htmltag{/ol}}
679 \newcommand*{\LWR@printclosedescription}
680     {\LWR@htmltag{/dl}}

681 \end{warpHTML}

```

## 29 Paragraph handling

These commands generate the HTML paragraph tags when allowed and required.

Paragraph tags are or are not allowed depending on many conditions. Section 30 has high-level commands which allow paragraph-tag generation to start/stop. Even when allowed (`\LWR@doingstartpars`), tags are not generated until a  $\text{\LaTeX}$  paragraph is being used (`\LWR@doingapar`). `LWR@lateximagedepth` is used to prevent nesting tags inside a `lateximage`. `LWR@spandepth` is used to prevent nesting paragraph tags inside a paragraph, which became important inside `\fbox` commands and other spans.

**for HTML output:** 682 `\begin{warpHTML}`

Ctr `LWR@spandepth` Do not create paragraph tags inside of an HTML span.

```

683 \newcounter{LWR@spandepth}
684 \setcounter{LWR@spandepth}{0}

```

Bool `LWR@doingstartpars` Tells whether paragraphs may be generated.

```

685 \newbool{LWR@doingstartpars}
686 \boolfalse{LWR@doingstartpars}

```

Bool `LWR@doingapar` Tells whether have actually generated and are currently processing paragraph text.

```

687 \newbool{LWR@doingapar}
688 \global\boolfalse{LWR@doingapar}

```

`\LWR@ensuredoingapar` If are about to print something visible, and if allowed to start a new paragraph, ensure that are `LWR@doingapar`, so that paragraph tags are placed:

```

689 \newcommand*{\LWR@ensuredoingapar}{%
690 \ifbool{\LWR@doingstartpars}%
691 {\global\booltrue{\LWR@doingapar}}%
692 {}%
693 }

```

\LWR@openparagraph

```

694 \newcommand*{\LWR@openparagraph}
695 {%

```

See if paragraph handling is enabled:

```

696 \ifbool{\LWR@doingstartpars}%
697 {% handling pars

```

See if have already started a `lateximage` or a `span`. If so, do not generate nested paragraph tags.

```

698 \ifthenelse{%
699 \cnttest{\value{\LWR@lateximagedepth}}{>}{0} \OR%
700 \cnttest{\value{\LWR@spandepth}}{>}{0}%
701 }{% nested par tags?

```

If so: Do nothing if already started a `lateximage` page. Cannot nest a `lateximage`. Also do nothing if already inside a `span`. Do not nest paragraph tags inside a `span`.

```

702 {}% no nested par tags

```

Else: No `lateximage` or `span` has been started yet, so it's OK to generate paragraph tags.

```

703 {% yes nest par tags
704 \LWR@htmltagc{\LWR@tagregularparagraph}%

```

Manually indent item list labels to avoid left margin intrusion:

$\text{\LaTeX}$  default list environments use `\@itemdepth` and `\@enumdepth`, but `lwrap` uses the `enumitem` package, which uses `\@listdepth`.

See if are nested inside an item list:

```

705 \ifnumcomp{\@listdepth}{>}{0}%
706 {%

```

If so, leave some horizontal room in the  $\text{\LaTeX}$  PDF output for list labels:

```

707 \LWR@origspace{1in}%
708 }{}%

```

Now have started a paragraph.

```
709 \global\booltrue{LWR@doingapar}%
```

At the end of each paragraph, generate closing tag and do regular /par stuff. (Attempting to use the everyhook cr hook for \LWR@closeparagraph does not work well.)

```
710 \let\par\LWR@closeparagraph%
711 }% end of yes nest par tags
712 }% end of handling pars
713 {}% not handling pars
714 }
```

\LWR@closeparagraph

```
715 \newcommand*{\LWR@closeparagraph}
716 {%
```

See if paragraph handling is enabled:

```
717 \ifbool{LWR@doingapar}%
```

If currently in paragraph mode:

```
718 {% handling pars
```

See if already started a lateximage or a span:

```
719 \ifthenelse{%
720 \cnttest{\value{LWR@lateximagedepth}}{>}{0} \OR%
721 \cnttest{\value{LWR@spandepth}}{>}{0}%
722 }%
```

Do nothing if already started a lateximage or a span, but add a parbreak if in a span but not a lateximage.

```
723 {% no nested par tags
724 \ifthenelse{%
725 \cnttest{\value{LWR@spandepth}}{>}{0}%
726 \AND%
727 \cnttest{\value{LWR@lateximagedepth}}{=}{0}%
728 }%
729 {%
730 \ifbool{LWR@intabularmetadata}{\unskip\LWR@htmltagc{br /}}%
731 }%
732 {}%
733 }% no nested par tags
```

If have not already started a `lateximage` or a `span`:

```
734 {% yes nest par tags
```

Print a closing tag:

```
735 \unskip%
736 \LWR@htmltagc{/LWR@tagregularparagraph}%
```

No longer doing a paragraph:

```
737 \global\boolfalse[LWR@doingapar]%
738 % Disable the special \env{minipage} \& \cs{hspace} interaction
739 % until a new minipage is found:
740 %   \begin{macrocode}
741 \global\boolfalse[LWR@minipagethispar]%
742 }% end of yes nest par tags
743 }% end of handling pars
```

Add a parbreak if in a span, but not in a table outside a row:

```
744 {% not handling pars
745 \ifthenelse{\cnttest{\value{LWR@spandepth}}{>}{0}}%
746 {\ifbool{LWR@intabularmetadata}{\unskip\LWR@htmltagc{br /}}}%
747 {}%
748 }% not handling pars
```

Finish with regular paragraph processing

```
749 \LWR@origpar%
750 }
```

```
751 \end{warpHTML}
```

## 30 Paragraph start/stop handling

These commands allow/disallow the generation of HTML paragraph tags.

Section 29 has the commands which actually generate the tags.

The `everyhook` package is used to generate the opening paragraph tags. The closing tags are generated by `\par`.

for HTML output: 752 \begin{warpHTML}

`\LWR@startpars` Begin handling HTML paragraphs. This allows an HTML paragraph to start, but one has not yet begun.

```
753 \newcommand*{\LWR@startpars}%
```

```
754 {%
```

See if currently handling HTML paragraphs:

```
755 \ifbool{LWR@doingstartpars}%
```

If already in paragraph mode, do nothing.

```
756 {}%
```

If not currently in paragraph mode:

```
757 {%
```

At the start of each paragraph, generate an opening tag:

```
758 \PushPreHook{par}{\LWR@openparagraph}%
```

At the end of each paragraph, generate closing tag and do regular `/par` actions:

```
759 \let\par\LWR@closeparagraph
```

```
760
```

```
761 }% an intentionally blank line
```

Are now handling paragraphs, but have not yet actually started one:

```
762 \global\setbool{LWR@doingstartpars}{true}%
```

No `<par>` tag yet to undo:

```
763 \global\boolfalse{LWR@doingapar}%
```

```
764 }
```

`\LWR@stoppars` Stop handling HTML paragraphs. Any currently open HTML paragraph is closed, and no more will be opened.

```
765 \newcommand*{\LWR@stoppars}%
```

```
766 {%
```

See if currently handling HTML paragraphs:

```
767 \ifbool{LWR@doingapar}%
```

if currently in an HTML paragraph:

768 {%

Print a closing tag:

769 \unskip%

770 \LWR@htmltagc{/LWR@tagregularparagraph}%

No longer have an open HTML paragraph:

771 \global\boolfalse{LWR@doingapar}%

Disable the special `minipage` & `\hspace` interaction until a new minipage is found:

772 \global\boolfalse{LWR@minipagethispar}

773

774 }% an intentionally blank line

If was not in an HTML paragraph:

775 {%}

See if currently allowing HTML paragraphs:

776 \ifbool{LWR@doingstartpars}%

If so: clear the `par` hook to no longer catch paragraphs:

777 {%

778 \ClearPreHook{par}%

779 }%

Else: do nothing

780 {}%

no longer in paragraph mode

781 \global\setbool{LWR@doingstartpars}{false}%

no `<p>` tag to undo:

782 \global\boolfalse{LWR@doingapar}%

783 }

784 \end{warpHTML}

## 31 Page headers and footers

for HTML & PRINT: 785 \begin{warpall}

In the following, catcode is manually changes back and forth without groups, since new macros are being defined which must not be contained within the groups.

```
786 \newcommand{\LWR@firstpagetop}{ } % for the home page alone
787 \newcommand{\LWR@pagetop}{ } % for all other pages
788 \newcommand{\LWR@pagebottom}{ }
789
790 \newcommand{\LWR@setfirstpagetop}[1]{%
791 \renewcommand{\LWR@firstpagetop}{#1}
792 \catcode'\_ =8
793 }
```

\SetFirstPageTop {*{text and logos}*}

```
794 \newcommand{\SetFirstPageTop}{%
795 \catcode'\_ =12
796 \LWR@setfirstpagetop
797 }
```

```
798 \newcommand{\LWR@setpagetop}[1]{%
799 \renewcommand{\LWR@pagetop}{#1}
800 \catcode'\_ =8
801 }
```

\SetPageTop {*{text and logos}*}

```
802 \newcommand{\SetPageTop}{%
803 \catcode'\_ =12
804 \LWR@setpagetop
805 }
```

```
806 \newcommand{\LWR@setpagebottom}[1]{%
807 \renewcommand{\LWR@pagebottom}{#1}
808 \catcode'\_ =8
809 }
```

\SetPageBottom {*{text and logos}*}

```
810 \newcommand{\SetPageBottom}{%
811 \catcode'\_ =12
812 \LWR@setpagebottom
813 }
```

```
814 \end{warpall}
```

## 32 CSS

**for HTML output:** 815 \begin{warpHTML}

**\LWR@currentcss** The CSS filename to use. This may be changed mid-document using **\NewCSS**, allowing different CSS files to be used for different sections of the document.

```
816 \newcommand*{\LWR@currentcss}{lwarp.css}
```

**\NewCSS**  $\{ \langle new-css-filename.css \rangle \}$

Assigns the CSS file to be used by the following HTML pages.

```
817 \newcommand*{\LWR@newcssb}[1]{%
818 \renewcommand*{\LWR@currentcss}{#1}
819 \catcode'\_ =8
820 }
821
822 \newcommand*{\NewCSS}{
823 \catcode'\_ =12
824 \LWR@newcssb
825 }
826 \end{warpHTML}
```

**for PRINT output:** 827 \begin{warpprint}  
828 \newcommand\*{\NewCSS}[1]{}  
829 \end{warpprint}

## 33 HTML meta description

**for HTML output:** 830 \begin{warpHTML}

**\LWR@currentHTMLdescription** The HTML meta description to use.

This is placed inside an HTML meta tag at the start of each file. This may be changed mid-document using **\NewHTMLdescription**, allowing different HTML descriptions to be used for different sections of the document.

Do not use double quotes, and do not exceed 150 characters.



```
831 \newcommand{\LWR@currentHTMLdescription}{}
```

```
\NewHTMLdescription {⟨New HTML meta description.⟩}
```

Assigns the HTML file's description meta tag

```
832 \newcommand{\NewHTMLdescription}[1]{%
833 \renewcommand{\LWR@currentHTMLdescription}{#1}
834 }
835
836 \end{warpHTML}
```

```
for PRINT output: 837 \begin{warpprint}
838 \newcommand{\NewHTMLdescription}[1]{%
839 \end{warpprint}
```

## 34 Footnotes

Several kinds of footnotes are used: in a regular page, in a minipage, or as thanks in the titlepage. Each of these is handle differently.

### 34.1 Regular page footnotes

In L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> printed documents, footnotes are placed at the bottom of the printed page. For HTML, footnotes will be replaced by page notes using the `pagenote` package. Footnotes will then appear at the bottom of the HTML page.

See section [34.4](#) for the implementation.

### 34.2 Minipage footnotes

See section [61.2](#) for minipage footnotes.

### 34.3 Titlepage thanks

See section [41.6](#) for titlepage footnotes.

### 34.4 Regular page footnote implementation

for HTML output: 840 `\begin{warpHTML}`

Delayed in case the user's preamble loaded and adjusted `pagenote`:

841 `\AtEndPreamble{`

`\notedivision` Prints the footnote heading at the end of the page.

842 `\renewcommand*{\notedivision}`

843 `{\BlockClassSingle{footnoteheader}{}}`

To print the word "Notes:" at the top of the footnotes:

---

`\renewcommand*{\notedivision}| \`  
`{\BlockClassSingle{footnoteheader}{Notes:}}`

---

`\pagenotesubhead` `{\langle num \rangle} {\langle title \rangle}`

Don't print page note headings in HTML page notes.

844 `\renewcommand{\pagenotesubhead}[2]{}`

`\footnote` `[\langle id \rangle] {\langle text \rangle}` Footnotes now become page notes.

845 `\let\footnote\LWR@origpagenote`

`\notenuminnotes` `{\langle number \rangle}`

Sets the page notes to use superscripts to match standard classes.

846 `\renewcommand{\notenuminnotes}[1]{%`

847 `\textsuperscript{\#1} }`

`\noteentry` `{\langle 1: note number \rangle} {\langle 2: id \rangle} {\langle 3: text \rangle} {\langle 4: pagenum \rangle}`

848 `\let\LWR@orignoteentry\noteentry`

849 `\renewcommand{\noteentry}[4]{%`

`\@currentlabel` A label may be used inside a note, so the proper `\@currentlabel` must be set  
`\cref@currentlabel`

inside a group. Likewise for `cleveref`'s `\cref@currentlabel`.

```
850 \begingroup%
851 \renewcommand*{\@currentlabel}{#1}%
852 \renewcommand*{\cref@currentlabel}{[pagenote] [#1] [] {#1}}%
```

Re-use `pageref`'s code, with an extra `endgroup`:

```
853 \LWR@orignoteentry{#1}{#2}{#3}{#4}%
854 \endgroup%
855 }
```

`\LWR@printpendingnotes` Prints the page notes at the end of the section.

```
856 \newcommand*{\LWR@printpendingnotes}{
```

If there are footnotes to print, open a new `div` and print the notes.

```
857 \ifthenelse{\cnttest{\value{pagenote}}{>}{0}}{%
858 {
859 \begin{BlockClass}{footnotes}
```

Print any pending endnotes, allowing new endnotes to accumulate.

```
860 \LWR@origprintnotes*
```

Start over with new page notes.

```
861 \setcounter{pagenote}{0}
862 \end{BlockClass}
863 }
864 {}
865 }
```

```
866 }% \AtEndPreamble
```

```
867 \end{warpHTML}
```

## 35 Marginpars

for HTML output: 868 \begin{warpHTML}

```
\marginpar [⟨left⟩] {⟨right⟩}
```

```
869 \end{warpHTML}
```

## 36 Splitting HTML files

- Files are split according to `FileDepth` and `CombineHigherDepths`.
- Filenames are sanitized by `\LWR@filenamenoblanks`.
- `\LWR@newhtmlfile` finishes an HTML page, adds a comment to tell where and how to split the file, then starts a new HTML page.

**for HTML & PRINT:** 870 `\begin{warpall}`

`Ctr FileDepth` `{\section depth}` determines how deeply to break into new HTML files, similar to `tocdepth`. The default of -5 produces one large HTML file.

```
871 \newcounter{FileDepth}
872 \setcounter{FileDepth}{-5}
```

`Bool CombineHigherDepths` Combile higher-level sections together into one file?

```
873 \newbool{CombineHigherDepths}
874 \booltrue{CombineHigherDepths}
```

```
875 \end{warpall}
```

**for HTML output:** 876 `\begin{warpHTML}`

`\LWR@thisfilename` The currently-active filename or number.

```
877 \newcommand*{\LWR@thisfilename}{}
```

`\LWR@thisnewfilename` The filename being sanitized.

```
878 \newcommand*{\LWR@thisnewfilename}{}
```

`\LWR@filenamenoblanks` `{\filename}`

Convert blanks into dashes, removes short words, store result in `\LWR@thisfilename`.



Be sure that this does not result in filename collisions! Use the optional TOC caption entry parameter for formatting. Remember to `\protect` L<sup>A</sup>T<sub>E</sub>X commands which appear in section names and TOC captions.

```
879 \newcommand*{\LWR@filenamenoblanks}[1]{%
880 \begingroup
```

Locally temporarily disable direct-formatting commands, not used in filenames:

```
881 \renewcommand*\HTMLUnicode}[1]{}
882 \renewcommand*\HTMLentity}[1]{##1}
883 \renewcommand*\LWR@htmltagc}[1]{}
884 \DeclareExpandableDocumentCommand{\InlineClass}{m o m}{##3}
```

Ampersand becomes “and”, which is a short word and is then removed from the filename.

```
885 \renewcommand*\&{and}
886 \renewcommand{\textit}[1]{##1}
887 \renewcommand{\textsc}[1]{##1}
888 \renewcommand{\textsl}[1]{##1}
889 \renewcommand{\textbf}[1]{##1}
890 \renewcommand{\texttt}[1]{##1}
891 \renewcommand{\textsf}[1]{##1}
892 \renewcommand{\textrm}[1]{##1}
893 \renewcommand{\textsuperscript}[1]{##1}
894 \renewcommand{\textsubscript}[1]{##1}
```

Replaces common symbols and short words with hyphens:

```
895 \edef\LWR@thisnewfilename{#1}
896 \fullexpandarg
```

Convert spaces into hyphens:

```
897 \StrSubstitute{\LWR@thisnewfilename}{ }{-}[\LWR@thisnewfilename]
```

Convert punctuation into hyphens:

```
898 \StrSubstitute{\LWR@thisnewfilename}{,}{-}[\LWR@thisnewfilename]
899 \StrSubstitute{\LWR@thisnewfilename}{'}{-}[\LWR@thisnewfilename]
900 \StrSubstitute{\LWR@thisnewfilename}%
901 {\LWR@origampersand}{-}[\LWR@thisnewfilename]
902 \StrSubstitute{\LWR@thisnewfilename}{+}{-}[\LWR@thisnewfilename]
903 \StrSubstitute{\LWR@thisnewfilename}{,}{-}[\LWR@thisnewfilename]
904 \StrSubstitute{\LWR@thisnewfilename}{/}{-}[\LWR@thisnewfilename]
905 \StrSubstitute{\LWR@thisnewfilename}{:}{-}[\LWR@thisnewfilename]
906 \StrSubstitute{\LWR@thisnewfilename}{;}{-}[\LWR@thisnewfilename]
907 \StrSubstitute{\LWR@thisnewfilename}{=}{-}[\LWR@thisnewfilename]
908 \StrSubstitute{\LWR@thisnewfilename}{?}{-}[\LWR@thisnewfilename]
909 \StrSubstitute{\LWR@thisnewfilename}{@}{-}[\LWR@thisnewfilename]
910 \StrSubstitute{\LWR@thisnewfilename}{"}{-}[\LWR@thisnewfilename]
911 \StrSubstitute{\LWR@thisnewfilename}%
912 {\textless}{-}[\LWR@thisnewfilename]
913 \StrSubstitute{\LWR@thisnewfilename}%
914 {\textgreater}{-}[\LWR@thisnewfilename]
```

```

915 \StrSubstitute{\LWR@thisnewfilename}{\#}{-}[\LWR@thisnewfilename]
916 \StrSubstitute{\LWR@thisnewfilename}{\%}{-}[\LWR@thisnewfilename]
917 \StrSubstitute{\LWR@thisnewfilename}{\}{-}[\LWR@thisnewfilename]
918 \StrSubstitute{\LWR@thisnewfilename}{\}{-}[\LWR@thisnewfilename]
919 \StrSubstitute{\LWR@thisnewfilename}{|}{-}[\LWR@thisnewfilename]
920 \StrSubstitute{\LWR@thisnewfilename}%
921 {\textbackslash}{-}[\LWR@thisnewfilename]
922 \StrSubstitute{\LWR@thisnewfilename}{^}{-}[\LWR@thisnewfilename]
923 \StrSubstitute{\LWR@thisnewfilename}{~}{-}[\LWR@thisnewfilename]
924 %      "~{" for babel
925 \StrSubstitute{\LWR@thisnewfilename}{[]}{-}[\LWR@thisnewfilename]
926 \StrSubstitute{\LWR@thisnewfilename}{[]}{-}[\LWR@thisnewfilename]
927 \StrSubstitute{\LWR@thisnewfilename}{' }{-}[\LWR@thisnewfilename]

```

Convert short words:

```

928 \StrSubstitute{\LWR@thisnewfilename}{-s-}{-}[\LWR@thisnewfilename]
929 \StrSubstitute{\LWR@thisnewfilename}{-S-}{-}[\LWR@thisnewfilename]
930 \StrSubstitute{\LWR@thisnewfilename}{-a-}{-}[\LWR@thisnewfilename]
931 \StrSubstitute{\LWR@thisnewfilename}{-A-}{-}[\LWR@thisnewfilename]
932 \StrSubstitute{\LWR@thisnewfilename}{-an-}{-}[\LWR@thisnewfilename]
933 \StrSubstitute{\LWR@thisnewfilename}{-AN-}{-}[\LWR@thisnewfilename]
934 \StrSubstitute{\LWR@thisnewfilename}{-to-}{-}[\LWR@thisnewfilename]
935 \StrSubstitute{\LWR@thisnewfilename}{-TO-}{-}[\LWR@thisnewfilename]
936 \StrSubstitute{\LWR@thisnewfilename}{-by-}{-}[\LWR@thisnewfilename]
937 \StrSubstitute{\LWR@thisnewfilename}{-BY-}{-}[\LWR@thisnewfilename]
938 \StrSubstitute{\LWR@thisnewfilename}{-of-}{-}[\LWR@thisnewfilename]
939 \StrSubstitute{\LWR@thisnewfilename}{-OF-}{-}[\LWR@thisnewfilename]
940 \StrSubstitute{\LWR@thisnewfilename}{-and-}{-}[\LWR@thisnewfilename]
941 \StrSubstitute{\LWR@thisnewfilename}{-AND-}{-}[\LWR@thisnewfilename]
942 \StrSubstitute{\LWR@thisnewfilename}{-for-}{-}[\LWR@thisnewfilename]
943 \StrSubstitute{\LWR@thisnewfilename}{-FOR-}{-}[\LWR@thisnewfilename]
944 \StrSubstitute{\LWR@thisnewfilename}{-the-}{-}[\LWR@thisnewfilename]
945 \StrSubstitute{\LWR@thisnewfilename}{-THE-}{-}[\LWR@thisnewfilename]

```

Convert multiple hyphens:

```

946 \StrSubstitute{\LWR@thisnewfilename}{-----}{-}[\LWR@thisnewfilename]
947 \StrSubstitute{\LWR@thisnewfilename}{----}{-}[\LWR@thisnewfilename]
948 \StrSubstitute{\LWR@thisnewfilename}{---}{-}[\LWR@thisnewfilename]
949 \StrSubstitute{\LWR@thisnewfilename}{--}{-}[\LWR@thisnewfilename]
950 \StrSubstitute{\LWR@thisnewfilename}{--}{-}[\LWR@thisnewfilename]
951 %      emdash
952 \StrSubstitute{\LWR@thisnewfilename}{-}{-}[\LWR@thisnewfilename]
953 %      endash
954 \global\let\LWR@thisfilename\LWR@thisnewfilename% return a global result
955 \endgroup
956 }

```

`\LWR@newhtmlfile` `{\section name}`}

Finishes the current HTML page with footnotes, footer, navigation, then starts a new HTML page with an HTML comment telling where to split the page and what the new filename and CSS are, then adds navigation, side TOC, header, and starts the text body.

```
957 \newcommand*{\LWR@newhtmlfile}[1]{
```

At the bottom of the ending file:

```
958 \LWR@htmlelementclassend{section}{textbody}
959
960 \LWR@printpendingnotes
961
962 \LWR@htmlelement{footer}
963
964 \LWR@pagebottom
965
966 \LWR@htmlelementend{footer}
```

No bottom navigation if are finishing the home page: Presumably has a table-of-contents printed.

```
967 \ifnumcomp{\value{LWR@htmlfilenumber}}{>}{0}{\LWR@botnavigation}{}
```

End of this HTML file:

```
968 \LWR@stoppars
969 \LWR@htmltag{/body}\LWR@orignewline
970 \LWR@htmltag{/html}\LWR@orignewline
971
972 \addtocounter{LWR@htmlfilenumber}{1}%
```

If using a filename, create a version without blanks. The filename without blanks will be placed into `\LWR@thisfilename`. If not using a filename, the file number will be used instead.

```
973 \ifbool{FileSectionNames}%
974 {\LWR@filenamenooblanks{#1}}
975 {\renewcommand*{\LWR@thisfilename}{\theLWR@htmlfilenumber}}
```

Include an HTML comment to instruct `lwarpmk` where to split the files apart. Uses pipe-separated fields for `split_html.gawk`. Uses monospaced font with ligatures disabled for everything except the title.

```
976 \LWR@htmlblockcomment{%
977 \LWR@origtexttt{|Start file|%
```

```

978 \LWR@htmlsectionfilename{\LWR@thisfilename}}}%
979 }

```

At the top of the starting file:

```

980 \LWR@stoppars
981
982 \LWR@filestart{ -- #1}% there is an EMDash in front of the #1
983
984 \LWR@topnavigation
985
986 \LWR@html element{header}
987
988 \LWR@pagetop
989
990 \LWR@html elementend{header}
991
992 \LWR@printthetitle
993
994 \LWR@sidetoc
995
996 \LWR@html elementclass{section}{textbody}
997
998 \LWR@stoppars
999
1000 \setcounter{LWR@latestautopage}{\value{page}}
1001 }

1002 \end{warpHTML}

```

## 37 Sectioning

Sectioning and cross-references have been emulated from scratch, rather than try to patch several layers of existing L<sup>A</sup>T<sub>E</sub>X code and packages. Formatting is handled by CSS, so the emulated code has much less work to do than the print versions.

**Unicode** Section names and the resulting filenames with accented characters are partially supported, depending on the ability of **pdf<sub>l</sub>at<sub>e</sub>x** to generate characters and **pdf<sub>t</sub>ot<sub>e</sub>xt** to read them. If extra symbols appear in the text, it may be that **pdf<sub>l</sub>at<sub>e</sub>x** is actually producing a symbol over or under a character, resulting in **pdf<sub>t</sub>ot<sub>e</sub>xt** picking up the accent symbol separately.




X<sub>Y</sub>L<sup>A</sup>T<sub>E</sub>X and LuaL<sup>A</sup>T<sub>E</sub>X directly support accented section and file names.

**for HTML output:** 1003 \begin{warpHTML}



### 37.1 Book class commands

`\mainmatter`  Declare the main matter section of the document. Does not reset the page number, which must be consecutive arabic numbers for the HTML conversion.

```
1004 \newbool{LWR@mainmatter}
1005 \DeclareDocumentCommand{\mainmatter}{}{%
1006 \booltrue{LWR@mainmatter}%
1007 }
```

`\frontmatter` Declare the front matter section of the document, using arabic numbering for the internal numbering. Does not reset the page number.

```
1008 \DeclareDocumentCommand{\frontmatter}{}{%
1009 \boolfalse{LWR@mainmatter}%
1010 }
```

`\backmatter` Declare the back matter section of the document. Does not reset the page number.

```
1011 \DeclareDocumentCommand{\backmatter}{}{%
1012 \boolfalse{LWR@mainmatter}
1013 }
```

### 37.2 Sectioning support macros

`\LWR@sectionnumber`  $\{\langle section type \rangle\}$

Typeset a section number and its trailing space with CSS formatting:

```
1014 \newcommand*{\LWR@sectionnumber}[1]{%
1015 \InlineClass{sectionnumber}{#1} %
1016 }
```

`autosec` A tag used by the TOC and index.

`\LWR@createautosec`  $\{\langle section type \rangle\}$

Create an autosection tag.

```
1017 \newcommand*{\LWR@createautosec}[1]{%
1018 \LWR@htmltag{#1 id="autosec-\thepage"{} }%
1019 }
```

`\LWR@pushoneclose`  $\{\langle depth \rangle\} \{\langle printclose \rangle\}$  Stacks the new sectioning level's closing tag, to be used when this section is closed some time later.

⚠ `\LWR@stoppars` must be executed first.

```
1020 \NewDocumentCommand{\LWR@pushoneclose}{m m}{\pushclose{#2}{#1}}
```

`\LWR@startnewdepth`  $\{\langle depth \rangle\} \{\langle printclose \rangle\}$

Closes currently stacked tags of a lesser level, then opens the new nesting level by saving this new sectioning level's closing tag for later use.

⚠ `\LWR@stoppars` must be executed first.

```
1021 \NewDocumentCommand{\LWR@startnewdepth}{m m}{%
```

Close any stacked sections up to this new one.

```
1022 \LWR@closeprevious{#1}%
```

Push a new section depth:

```
1023 \LWR@pushoneclose{#1}{#2}%
```

```
1024 }
```

Ctr `LWR@prevFileDepth` Remembers the previous `LWR@FileDepth`.

Initialized to a deep level so that any section will trigger a new HTML page after the home page.

```
1025 \newcounter{LWR@prevFileDepth}
```

```
1026 \setcounter{LWR@prevFileDepth}{\LWR@depthsubparagraph}
```

`\LWR@section` \*  $[\langle TOC name \rangle] \{\langle name \rangle\} \{\langle sectiontype \rangle\}$

The common actions for the high-level sectioning commands.

```
1027 \DeclareDocumentCommand{\LWR@section}{m m m m}{%
```

```
1028 \LWR@traceinfo{LWR@section}%
```

```
1029 \LWR@stoppars%
```

Cancel special `minipage` horizontal space interaction:

```
1030 \global\boolfalse{LWR@minipagethispar}%
```

Generate a new LaTeX page so that TOC and index page number points to the section:

```
1031 \LWR@orignewpage%
1032
```

Start a new HTML file if not starred, and is a shallow sectioning depth:

```
1033 \LWR@traceinfo{LWR@section: testing about to start a new HTML file}%
1034 \IfBooleanTF{#1}{}{% not starred
1035 \ifthenelse{%
1036 \cnttest{\csuse{LWR@depth#4}}{<=}{\value{FileDepth}}}%
1037 \AND%
1038 \(%
1039 \NOT\boolean{CombineHigherDepths}\OR%
1040 \cnttest{\csuse{LWR@depth#4}}{<=}{\value{LWR@prevFileDepth}}}%
1041 \)%
1042 }%
```

If so: start a new HTML file:

```
1043 {% new file
1044 \LWR@traceinfo{LWR@section: new HTML file}%
```

See if there was an optional TOC name entry:

```
1045 \IfNoValueTF{#2}%
```

If no optional entry

```
1046 {\LWR@newhtmlfile{#3}}%
```

If yes an optional entry

```
1047 {\LWR@newhtmlfile{#2}}%
1048 }% new file
```

Else: No new HTML file:

```
1049 {}% not new file
1050 }% not starred
```

Remember this section's name for \nameref:

```
1051 \LWR@traceinfo{LWR@section: about to LWR@setlatestname}%
1052 \IfValueTF{#2}{\LWR@setlatestname{#2}}{\LWR@setlatestname{#3}}%
```

Print an opening comment with the level and the name; ex: “section” “Introduction”

```
1053
1054 \LWR@htmlcomment{Opening #4 ‘‘#3’’{}}
1055
```

For inline sections paragraph and subparagraph, start a new paragraph now:

```
1056 \ifthenelse{%
1057 \cnttest{\csuse{LWR@depth#4}}{>=}{\LWR@depthparagraph}%
1058 }%
1059 {\LWR@startpars}
1060 {}
```

Create the opening tag with an autosec:

```
1061 \LWR@createautosec{\csuse{LWR@tag#4}}%
```

If not starred, step counter and add to TOC:

```
1062 \IfBooleanTF{#1}%
1063 {}% starred
1064 {% not starred
```

Only add a numbered TOC entry if section number is not too deep:

```
1065 \ifthenelse{%
1066 \cnttest{\csuse{LWR@depth#4}}{<=}{\value{secnumdepth}}%
1067 }%
1068 {% if secnumdepth
```

If in the main matter, step the counter and add the TOC entry. For **article** class, **lwarp** assumes that all is mainmatter.

```
1069 \LWR@traceinfo{LWR@section: about to test main matter}%
1070 \ifbool{LWR@mainmatter}%
1071 {%
1072 \LWR@traceinfo{LWR@section: yes mainmatter}%
1073 \refstepcounter{#4}%
```

Add main matter numbered TOC entry with the TOC name or the regular name:

```
1074 \LWR@traceinfo{LWR@section: about to addcontentsline}%
1075 \addcontentsline{toc}{#4}%
1076 {\protect\LWR@sectionnumber{\csuse{the#4}}}%
1077 \IfValueTF{#2}{#2}{#3}%
1078 \LWR@traceinfo{LWR@section: finished addcontentsline}%
1079 }% end of if main matter
```

If not main matter, add unnumbered TOC name or regular name:

```
1080 {% not main matter
1081 \LWR@traceinfo{LWR@section: no main matter}%
1082 \addcontentsline{toc}{#4}{\IfValueTF{#2}{#2}{#3}}%
1083 }% end of not main matter
```

1084 }% end of secnumdepth

Deeper than secnumdepth, so add an unnumbered TOC entry:

```
1085 {%
1086 \addcontentsline{toc}{#4}{\IfValueTF{#2}{#2}{#3}}%
1087 }%
```

For part, print the section type:

```
1088 \ifbool{LWR@mainmatter}%
1089 {%
1090 \ifthenelse{%
1091 \(\cnttest{\csuse{LWR@depth#4}}{<=}}{%
1092 {\value{secnumdepth}}\)\ AND
1093 \(\cnttest{\csuse{LWR@depth#4}}{<=}{\LWR@depthpart}\)
1094 }%
1095 {\csuse{#4name}~{}}%
1096 {}}%
```

Print the section number:

```
1097 \LWR@traceinfo{LWR@section: about to print section number}%
1098 \ifthenelse{%
1099 \cnttest{\csuse{LWR@depth#4}}{<=}{\value{secnumdepth}}%
1100 }%
1101 {\protect\LWR@sectionnumber{\csuse{the#4}}}%
1102 {}%
1103 \LWR@traceinfo{LWR@section: finished print section number}%
1104 }{}%
1105 }% end of not starred
```

Print the section name:

1106 #3

close the heading tag, such as /H2

```
1107 \LWR@htmltag{\csuse{LWR@tag#4end}}%
```

Generate a L<sup>A</sup>T<sub>E</sub>X label:

```
1108 \label{autopage-\thepage}%
```

Start paragraph handing unless is an inline paragraph or subparagraph:

```
1109 \ifthenelse{%
1110 \cnttest{\csuse{LWR@depth#4}}{<}{\LWR@depthparagraph}}%
1111 {\LWR@startpars}%
```

```
1112 {}%
```

If not starred, remember the previous depth to possibly trigger a new HTML page.

A starred section does not trigger a new HTML page at the beginning of this macro, so it should not affect it here at the end either. This became an issue when a `\listoftables` was tested in the middle of the document. The `\chapter*` for the list was not allowing a new HTML page for the section following it while `CombineHigherDepths` was true.

```
1113 \IfBooleanTF{#1}{}{% not starred
1114 \setcounter{LWR@prevFileDepth}{\csuse{LWR@depth#4}}%
1115 }% not starred
1116 \LWR@traceinfo{LWR@section: done}%
1117 }
```

### 37.3 `\section` and friends

`\part` \* [*TOC name*] {*name*}

```
1118 \@ifundefined{chapter}
1119 {}
1120 {%
1121 \DeclareDocumentCommand{\part}{s o m}{%
1122 \LWR@stoppars%
1123
1124 \LWR@startnewdepth{\LWR@depthpart}{\LWR@printclosepart}%
1125
1126 \LWR@section{#1}{#2}{#3}{part}%
1127 }
1128 }
```

`\chapter` \* [*TOC name*] {*name*}

```
1129 \@ifundefined{chapter}
1130 {}
1131 {%
1132 \DeclareDocumentCommand{\chapter}{s o m}{%
1133 \LWR@traceinfo{chapter #3}%
1134 \LWR@stoppars%
1135 \LWR@startnewdepth{\LWR@depthchapter}{\LWR@printclosechapter}%
1136
1137 \LWR@traceinfo{chapter: about to LWR@section}%
1138 \LWR@section{#1}{#2}{#3}{chapter}%
1139 \LWR@traceinfo{chapter: done}%
1140 }
1141 }
```

`\section * [⟨TOC name⟩] {⟨name⟩}`

```
1142 \DeclareDocumentCommand{\section}{s o m}{%
1143 \LWR@stoppars%
1144
1145 \LWR@startnewdepth{\LWR@depthsection}{\LWR@printclosesection}%
1146
1147 \LWR@section{#1}{#2}{#3}{section}%
1148 }
```

`\subsection * [⟨TOC name⟩] {⟨name⟩}`

```
1149 \DeclareDocumentCommand{\subsection}{s o m}{%
1150 \LWR@stoppars%
1151
1152 \LWR@startnewdepth{\LWR@depthsubsection}{\LWR@printclosesubsection}%
1153
1154 \LWR@section{#1}{#2}{#3}{subsection}%
1155 }
```

`\subsubsection * [⟨TOC name⟩] {⟨name⟩}`

```
1156 \DeclareDocumentCommand{\subsubsection}{s o m}{%
1157 \LWR@stoppars%
1158
1159 \LWR@startnewdepth{\LWR@depthsubsubsection}{%
1160 {\LWR@printclosesubsubsection}%
1161
1162 \LWR@section{#1}{#2}{#3}{subsubsection}%
1163 }
```

`\paragraph * [⟨TOC name⟩] {⟨name⟩}`

```
1164 \DeclareDocumentCommand{\paragraph}{s o m}{%
1165 \LWR@stoppars%
1166
1167 \LWR@startnewdepth{\LWR@depthparagraph}{\LWR@printcloseparagraph}%
1168
1169 \LWR@section{#1}{#2}{#3}{paragraph}%
1170 }
```

`\subparagraph * [⟨TOC name⟩] {⟨name⟩}`

```
1171 \DeclareDocumentCommand{\subparagraph}{s o m}{%
1172 \LWR@stoppars%
1173
1174 \LWR@startnewdepth{\LWR@depthsubparagraph}{\LWR@printclosesubparagraph}%
1175 }
```

```

1175
1176 \LWR@section{#1}{#2}{#3}{subparagraph}%
1177 }

1178 \end{warpHTML}

```

## 38 Starting a new file

for HTML output: 1179 \begin{warpHTML}

\MetaLanguage Default language for the HTML lang tag.

```
1180 \newcommand*{\MetaLanguage}{en-US}
```

\LWR@filestart  $\{\langle title\_suffix \rangle\}$  Creates the opening HTML tags.

```
1181 \newcommand*{\LWR@filestart}[1]{
```

Locally temporarily disable direct-formatting commands:

```

1182 \begingroup
1183 \renewcommand{\textit}[1]{##1}% not used in filenames
1184 \renewcommand{\textsc}[1]{##1}
1185 \renewcommand{\textsl}[1]{##1}
1186 \renewcommand{\textbf}[1]{##1}
1187 \renewcommand{\texttt}[1]{##1}
1188 \renewcommand{\textsf}[1]{##1}
1189 \renewcommand{\textrm}[1]{##1}
1190 \renewcommand{\textsuperscript}[1]{##1}
1191 \renewcommand{\textsubscript}[1]{##1}
1192 \renewcommand*{\HTMLunicode}[1]{ }
1193 \renewcommand*{\HTMLentity}[1]{ }
1194 \RenewDocumentCommand{\LWR@htmlspanclass}{m o +m}{##3}
1195 \DeclareExpandableDocumentCommand{\InlineClass}{m o m}{##3}

```

Create the page's HTML header:

```
1196 \LWR@htmltag{!DOCTYPE html}\LWR@orignewline
```

The language is user-adjustable:

```
1197 \LWR@htmltag{html lang="\MetaLanguage"}\LWR@orignewline
```

Start of the meta data:



```
1198 \LWR@htmltag{head}\LWR@orignewline
```

Charset is fixed at UTF-8:

```
1199 \LWR@htmltag{meta charset="UTF-8" /\LWR@orignewline
```

lwarp is the generator:

```
1200 \LWR@htmltag{meta name="generator" content="LaTeX lwarp package" /\%
1201 \LWR@orignewline
```

If there is a description, add it now:

```
1202 \ifdefempty{\LWR@currentHTMLdescription}{-}{-}%
1203 \LWR@htmltag{%
1204 meta name="description" content="\LWR@currentHTMLdescription" /\%
1205 \LWR@orignewline
1206 }%
```

Mobile-friendly viewport:

```
1207 \LWR@htmltag{meta name="viewport" %
1208 content="width=device-width, initial-scale=1.0" /\%
1209 \LWR@orignewline
```

IE patch:

```
1210 \LWR@htmltag{!{-}{-}[if lt IE 9]}\LWR@orignewline
1211 \LWR@htmltag{%
1212 script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"{}
1213 \LWR@htmltag{/script}\LWR@orignewline
1214 \LWR@htmltag{![endif]{-}{-}}\LWR@orignewline
```

The page's title:

```
1215 \LWR@htmltag{title}\thetitle#1\LWR@htmltag{/title}\LWR@orignewline
```

The page's stylesheet:

```
1216 \LWR@htmltag{%
1217 link rel="stylesheet" type="text/css" href="\LWR@currentcss" /\%
1218 \LWR@orignewline
```

Optional MathJax support. The HTML tags must be turned off during the verbatim input, and the paragraph handling which was turned on at the end of verbatim input must be immediately turned off again.

```
1219 \ifbool{mathjax}%
```

```

1220 {%
1221 \boolfalse{LWR@verbtags}
1222 \VerbatimInput{lwarp_mathjax.txt}%
1223 \booltrue{LWR@verbtags}
1224 \LWR@stoppars
1225 }% end of mathjax
1226 {}%

```

End of the header:

```
1227 \LWR@htmltag{/head}\LWR@orignewline
```

Start of the body:

```

1228 \LWR@htmltag{body}\LWR@orignewline
1229 \endgroup
1230 }

```

```
1231 \end{warpHTML}
```

## 39 Starting HTML output

**for HTML output:** 1232 \begin{warpHTML}

**\LWR@LwarpStart** Executed at the beginning of the entire document.

```

1233 \newcommand*{\LWR@LwarpStart}
1234 {%

```

Expand and detokenize \HomeHTMLFileName and \HTMLFileName:

```

1235 \edef\LWR@strresult{\HomeHTMLFileName}
1236 \edef\HomeHTMLFileName{\detokenize\expandafter{\LWR@strresult}}
1237 \edef\LWR@strresult{\HTMLFileName}
1238 \edef\HTMLFileName{\detokenize\expandafter{\LWR@strresult}}

```

Force onecolumn:

```
1239 \LWR@origonecolumn%
```

Reduce chance of line overflow in verbatim environments:

```
1240 \LWR@origscriptsize%
```

In PDF output, don't allow line breaks to interfere with HTML tags:

```
1241 \LWR@origraggedright%
1242 \let\\\LWR@endofline%
```

Spread the lines for `pdftotext` to read them well:

```
1243 \linespread{1.3}%
```

For `pdftotext` to reliably identify paragraph splits:

```
1244 \setlength{\parindent}{0pt}
1245 \setlength{\parskip}{2ex}
```

For the `lateximages` command file:

```
1246 \ifbool{usingOSwindows}
1247 {% Windows:
1248 \immediate\openout\LWR@file=lateximages.cmd
1249 }
1250 {% Unix:
1251 \immediate\openout\LWR@file=lateximages.sh
1252 \immediate\write\LWR@file{\LWR@hashmark!\OSShellPath}
1253 }
```

The first action of the command file is to create the `lateximages` directory:

```
1254 \immediate\write\LWR@file{\OSmkdir lateximages\OSPathSymbol}
```

Removes space after the caption in the HTML:

```
1255 \setlength{\belowcaptionskip}{-3ex}
```

Redefine the plain page style to be empty when used by index pages:

```
1256 \renewcommand{\ps@plain}{}
```

```
\centering Not used in the HTML environment:
\raggedleft
\raggedright 1257 \renewcommand*{\centering}{}
1258 \renewcommand*{\raggedleft}{}
1259 \renewcommand*{\raggedright}{}%
```

Plug in some new actions. This is done just before the document start so that they won't be over-written by some other package.

Tabular:

```

1260 \let\LWR@origtabular\tabular
1261 \let\LWR@origendtabular\endtabular
1262 \let\tabular\LWR@tabular
1263 \let\endtabular\endLWR@tabular

```

Float captions:

```

1264 \let\LWR@origcaption\caption

```

Labels: `\ltx@label` is used in `amsmath` environments and is also patched by `cleveref`.

```

1265 \let\ltx@label\LWR@htmlmathlabel

```

Logos:

```

1266 \let\TeX\LWR@TeX
1267 \let\LaTeX\LWR@LaTeX
1268 \let\LuaTeX\LWR@LuaTeX
1269 \let\LuaLaTeX\LWR@LuaLaTeX
1270 \let\XeTeX\LWR@XeTeX
1271 \let\XeLaTeX\LWR@XeLaTeX
1272 \let\ConTeXt\LWR@ConTeXt

```

Graphics:

```

1273 \let\textcolor\LWR@textcolor
1274 \let\pagecolor\LWR@pagecolor
1275 \let\colorbox\LWR@colorbox
1276 \let\fcolorbox\LWR@fcolorbox
1277 \let\rotatebox\LWR@rotatebox
1278 \let\scalebox\LWR@scalebox
1279 \let\reflectbox\LWR@reflectbox

```

Not yet started any paragraph handling:

```

1280 \global\boolfalse{LWR@doingapar}
1281 \global\boolfalse{LWR@doingstartpars}

```

Start a new HTML file and a header:

```

1282 \LWR@filestart{}
1283 \LWR@htmltag{header}\LWR@orignewline
1284 \LWR@startpars
1285 \LWR@firstpagetop
1286 \LWR@stoppars
1287 \LWR@htmltag{/header}\LWR@orignewline
1288 \LWR@htmltag{section class="textbody"{} }
1289 \LWR@origpagestyle{empty}

```

Document and page settings:

```
1290 \mainmatter
1291 \LWR@origpagenumbering{arabic}
```

Set default titlepage thanks footnote marks. See section 41.6.

```
1292 \if@titlepage
1293   \thanksmarkseries{arabic}
1294 \else
1295   \thanksmarkseries{fnsymbol}
1296 \fi
```

Initial default patch for fancyvrb:

```
1297 \fvset{frame=none}%
```

The ampersand is redefined active, and acts depending on whether it is inside a tabular.

```
1298 \catcode'\&=\active
```

Allow HTML paragraphs to begin:

```
1299 \LWR@startpars
1300 }
```

```
1301 \end{warpHTML}
```

## 40 Ending HTML output

**for HTML output:** 1302 \begin{warpHTML}

`\LWR@requesttoc`  $\{\langle boolean \rangle\}$   $\{\langle suffix \rangle\}$  Requests that a toc, lof, or lot be generated.

```
1303 \newcommand*\LWR@requesttoc[2]{%
1304 \ifbool{#1}
1305 {
1306 \expandafter\newwrite\csuse{tf@#2}
1307 \immediate\openout \csuse{tf@#2} \jobname.#2\relax
1308 }{}
1309 }
```

`\LWR@LwarpEnd` Final stop of all HTML output:

```
1310 \newcommand*{\LWR@LwarpEnd}  
1311 {  
1312 \LWR@stoppars  
1313 \LWR@closeprevious{\LWR@depthfinished}
```

At the bottom of the ending file:

Close the textbody:

```
1314 \LWR@htmlclassend{section}{textbody}
```

Print any pending footnotes:

```
1315 \LWR@printpendingnotes
```

Create the footer:

```
1316 \LWR@htmlclassend{footer}  
1317  
1318 \LWR@pagebottom  
1319  
1320 \LWR@htmlclassend{footer}
```

No bottom navigation if are finishing the home page:  
Presumably has a table-of-contents printed.

```
1321 \ifnumcomp{\value{LWR@htmlfilenumber}}{>}{0}{\LWR@botnavigation}{}
```

```
1322 \LWR@stoppars% final stop of all paragraphs
```

Finish the HTML file:

```
1323 \LWR@htmltag{/body}\LWR@orignewline  
1324 \LWR@htmltag{/html}\LWR@orignewline
```

Seems to be required sometimes:

```
1325 \LWR@orignewpage
```

For lateximage commands:

```
1326 \immediate\closeout\LWR@file  
1327 }
```

```
1328 \end{warpHTML}
```

## 41 Titles and the titling package

Supports and extends the titling package.

Additional functions include `\published` and `\subtitle`, and the `\author` command has an additional `\affiliation` command to provide an affiliation and other additional information for each author in the title page. The affiliation information is removed when using `\theauthor` in the main text.

The `titling` package maintains the definitions of `\thetitle`, `\theauthor`, etc., after the title has been typeset. These commands are to be used to refer to the document's title and author, etc., in the main text. These definitions have the `\thanks` and `\affiliation` removed, and for author the `\and` is replaced to generate a simple inline list of authors separated by commas.

`\printtitle`, `\printauthor`, etc., are provided for use inside the `titlepage` or `titlingpage` environments, and these retain the `\thanks` and `\affiliation`.

Several additional hooks are provided in addition to titling:

<code>\maketitlehookaa</code>	<code>\maketitlehookaa</code> : Between “published” and the title.
<code>\maketitlehookaaa</code>	<code>\maketitlehookaaa</code> : Between the title and the subtitle.
<code>\prepublished</code>	<code>\prepublished</code> : Before the “published” field.
<code>\postpublished</code>	<code>\postpublished</code> : After the “published” field.
<code>\presubtitle</code>	<code>\presubtitle</code> : Before the subtitle.
<code>\postsubtitle</code>	<code>\postsubtitle</code> : After the subtitle.
<code>\printthanks</code>	<code>\printthanks</code> has been added to force the printing of thanks inside a <code>titlingpage</code> environment when <code>\maketitle</code> is not used.



**No footnotes!**

Inside a `\titlepage` or `\titlingpage` environment, use `\thanks` for footnotes, do not use `\footnote`.

At the end of the `titlingpage` environment, footnote marks are forced to reset to zero.

Inside a `titlingpage` environment with the `article` document class, thanks marks will be `fnsymbol` instead of `arabic`. `arabic` is usually used when inside `titlepage` environments where the title page is on its own page, but is not automatically used inside a `titlingpage` environment.

To force the thanks marks to be `arabic`:

---

```
\begin{titlingpage}
\thanksmarkseries{arabic}
...
```

---

## 41.1 Setting the title, etc.

The following provide setting commands for both HTML and print outputs.

<pre>\published   \title   \subtitle   \author   \date</pre>	<pre>\@title, \@subtitle, \@author, etc. store the values as originally assigned, including any \thanks, \and, or \affiliation. These are low-level macros intended to be used by other macros only inside a titlepage or titlingpage, and are used by \maketitle. The author is printed inside a single-column table, which becomes multiple single-column tables if multiples authors are included.</pre>
<pre>\printpublished   \printtitle   \printsubtitle   \printauthor   \printdate</pre>	<pre>\printtitle, \printsubtitle, etc. are user-level macros intended to be used in titlepage and titlingpage environments in cases where \maketitle is not desired. These commands preserve the \thanks, etc., and should not be used in the main text. The author is printed inside a single-column table, which becomes multiple single-column tables if multiples authors are included.</pre>
<pre>\thepublished   \thetitle   \thesubtitle   \theauthor   \thedate</pre>	<pre>\thetitle, \thesubtitle, \theauthor, etc. are user-level sanitized versions which have removed the \thanks and \affiliation, and \and is changed for inline text usage. The author is printed inline without \affiliation or \thanks, with \and placing commas between multiple authors. Thus, these commands are to be used in the main text whenever the user wishes to refer to the document's title and such. One practical use for this is to place the authors at the bottom of each HTML page, such as:</pre>
<pre>\SetPageBottom</pre>	<pre>{\text}</pre>

---

```
\SetPageBottom{
\begin{center}\textcopyright~2016 \theauthor\end{center}
}
```

---

<pre>\author   \and</pre>	<pre>{\author}</pre> While using <code>\maketitle</code> , the author is treated as a single-column table and the <code>\and</code> feature finishes the current table then starts a new one for the next author. Each author thus is placed into its own table, and an affiliation may be placed on its own line such as
---------------------------	---

```
\author{Name \\ Affiliation \and Second Name \\ Second Affiliation}
```

After `\maketitle` has completed, `\theauthor` retains the definition of the author, but `\and` is changed to become a comma and a space, intending to print the authors



names separated by spaces. This fails when affiliations are included on their own table rows.

A solution, provide here, is to define a macro `\affiliation` which during `\maketitle` starts a new table row and adds the affiliation, but after `\maketitle` is finished `\affiliation` is re-defined to throw away its argument, thus printing only the author names when `\author` is later used inline.

```
\affiliation {{text}}
```

Adds the affiliation to the author for use in `\maketitle`. Nullified when later used for inline use of `\theauthor`.

```
for HTML output: 1329 \begin{warpHTML}
                  1330 \newrobustcmd{\affiliation}[1]{\ \ \ \InlineClass{affiliation}{#1}}
                  1331 \end{warpHTML}
```

```
for PRINT output: 1332 \begin{warpprint}
                  1333 \newrobustcmd{\affiliation}[1]{\ \ \ \textsc{\small#1}}
                  1334 \end{warpprint}
```

The following are based on the original titling code:

```
for HTML & PRINT: 1335 \begin{warpall}
```

```
\author {{text}}
```

Redefined to nullify `\affiliation`, etc. before printing the authors inline.

`\@author` retains the entire author with its `\thanks`, while `\theauthor` will have `\thanks` removed and `\and` simplified.

```
1336 \renewcommand{\author}[1]{%
1337 \gdef\@author{#1}
1338 \begingroup
1339   \renewcommand{\thanks}[1]{ }
1340   \renewcommand{\and}{\unskip, }
1341   \renewcommand{\thanksmark}[1]{ }
1342   \renewcommand{\thanksgap}[1]{ }
1343   \renewcommand{\affiliation}[1]{ }
1344   \protected@xdef\theauthor{#1}
1345 \endgroup}
```

```
\published {{text}}
```

```
1346 \newcommand{\published}[1]{%
1347 \gdef\@published{#1}
```

```

1348 \begingroup
1349   \renewcommand{\thanks}[1]{%
1350     \renewcommand{\thanksmark}[1]{%
1351       \renewcommand{\thanksgap}[1]{%
1352         \protected@xdef\thepublished{#1}
1353       }
1354     }
1355   \newcommand{\@published}{%
1356     \newcommand{\thepublished}{%

\subtitle {<text>}

1357 \newcommand{\subtitle}[1]{%
1358   \gdef\@subtitle{#1}
1359   \begingroup
1360     \renewcommand{\thanks}[1]{%
1361       \renewcommand{\thanksmark}[1]{%
1362         \renewcommand{\thanksgap}[1]{%
1363           \protected@xdef\thesubtitle{#1}
1364         }
1365       }
1366     \newcommand{\@subtitle}{%
1367       \newcommand{\thesubtitle}{%

1368 \end{warpall}

```

## 41.2 Changes to HTML titlepage and titlingpage

for HTML output: 1369 \begin{warpHTML}

Env **titlepage** Sets up a titlepage div with a L<sup>A</sup>T<sub>E</sub>X PDF minipage inside.

```

1370 \renewenvironment*{titlepage}
1371   {\BlockClass{titlepage}\LWR@subminipage}
1372   {\LWR@endsubminipage\endBlockClass}

```

Env **titlingpage**

```

1373 \renewenvironment*{titlingpage}
1374 {%

```

Start an HTML titlepage div:

```

1375 \begin{titlepage}

```

Prepare for a custom version of `\maketitle` inside the `titlingpage`:

```
1376 \LWR@maketitlesetup
1377 \let\maketitle\LWR@titlingmaketitle
1378 }
1379 {
```

At the end of the environment, end the HTML titlepage div:

```
1380 \end{titlepage}
```

Reset the footnote counter:

```
1381 \@bscontmark
1382 }
```

```
1383 \end{warpHTML}
```

**for HTML & PRINT:** 1384 `\begin{warpall}`

`\printthanks` Forces the `\thanks` to be printed.

This is necessary in a `titlingpage` environment when `\maketitle` was not used.

```
1385 \newcommand*{\printthanks}{\@thanks}
```

Env `titlingpage` At the end of the `titlingpage` for both print and HTML, reset footnote markers to zero.

```
1386 \AtEndEnvironment{titlingpage}{\@bscontmark}
```

```
1387 \end{warpall}
```

### 41.3 Printing the title, etc. in HTML

The following are for printing the title, etc. in a `titlepage` or a `titlingpage` in HTML:

**for HTML output:** 1388 `\begin{warpHTML}`

Patch the pre/post title/author/date to add HTML tags, then initialize:

```
1389 \newcommand{\prepublished}[1]{%
1390 \def\@bspublished{\BlockClass{published}#1}%
1391 }
1392
```

```
1393 \newcommand{\postpublished}[1]{%
1394 \def\@bspostpublished{#1\endBlockClass}%
1395 }
1396
1397 \renewcommand{\pretitle}[1]{%
1398 \def\@bspretitle{#1\LWR@stoppars\LWR@htmltag{h1}}}%
1399 }
1400
1401 \renewcommand{\posttitle}[1]{%
1402 \def\@bsposttitle{\LWR@htmltag{/h1}\LWR@startpars#1}%
1403 }
1404
1405 \newcommand{\presubtitle}[1]{%
1406 \def\@bspresubtitle{\BlockClass{subtitle}#1}%
1407 }
1408
1409 \newcommand{\postsubtitle}[1]{%
1410 \def\@bspostsubtitle{#1\endBlockClass}%
1411 }
1412
1413 \renewcommand{\preauthor}[1]{%
1414 \def\@bspreauthor{\BlockClass{author}#1}%
1415 }
1416
1417 \renewcommand{\postauthor}[1]{%
1418 \def\@bspostauthor{#1\endBlockClass}%
1419 }
1420
1421 \renewcommand{\predate}[1]{%
1422 \def\@bspredate{#1\BlockClass{titledate}}}%
1423 }
1424
1425 \renewcommand{\postdate}[1]{%
1426 \def\@bspostdate{\endBlockClass#1}%
1427 }
1428
1429 \prepublished{\begin{center}}
1430 \postpublished{\par\end{center}}
1431
1432 \pretitle{\begin{center}}
1433 \posttitle{\par\end{center}}
1434
1435 \presubtitle{\begin{center}}
1436 \postsubtitle{\par\end{center}}
1437
1438 \preauthor{\begin{center}}%
1439 \begin{tabular}[t]{c}%
1440 }
1441 \postauthor{\end{tabular}\par\end{center}}
1442
```

```

1443 \predatex{\begin{center}}
1444 \postdatex{\par\end{center}}

```

### \printpublished

```

1445 \newcommand*\printpublished{
1446 \ifthenelse{\equal{\thepublished}{}}
1447 {}
1448 {
1449 \begin{BlockClass}{published}
1450 \@published
1451 \end{BlockClass}
1452 }
1453 }

```

### \printtitle

```

1454 \newcommand*\printtitle
1455 {
1456 \LWR@stoppars
1457 \LWR@htmltag{h1}%
1458 \@title%
1459 \LWR@htmltag{/h1}
1460 \LWR@startpars
1461 }

```

**\LWR@printthetitle** A private version which prints the title without footnotes, used to title each HTML page.

```

1462 \newcommand*\LWR@printthetitle
1463 {
1464 \LWR@stoppars
1465 \LWR@htmltag{h1}%
1466 \thetitle%
1467 \LWR@htmltag{/h1}
1468 \LWR@startpars
1469 }

```

### \printsubtitle

```

1470 \newcommand*\printsubtitle{
1471 \ifthenelse{\equal{\thesubtitle}{}}
1472 {}
1473 {
1474 \begin{BlockClass}{subtitle}
1475 \@subtitle
1476 \end{BlockClass}

```

```
1477 }
1478 }
```

`\printauthor`

```
1479 \newcommand*{\printauthor}{
1480 \begin{BlockClass}{author}
1481 \begin{tabular}{c}\@author\end{tabular}
1482 \end{BlockClass}
1483 }
```

`\printdate`

```
1484 \newcommand*{\printdate}{%
1485 \begin{BlockClass}{titledate}
1486 \@date
1487 \end{BlockClass}
1488 }
```

```
1489 \end{warpHTML}
```

## 41.4 Printing the title, etc. in print form

The following are for printing the title, etc. in a `titlepage` or a `titlingpage` in print form:

for **PRINT** output: 1490 `\begin{warpprint}`

`\printpublished`

```
1491 \newcommand*{\printpublished}{\Large\scshape\@published}
```

`\printtitle`

```
1492 \newcommand*{\printtitle}{\Huge\@title}
```

`\printsubtitle`

```
1493 \newcommand*{\printsubtitle}{\Large\itshape\@subtitle\bigskip}
```

`\printauthor`

```
1494 \newcommand*{\printauthor}
1495 {\large\begin{tabular}{t}{c}\@author\end{tabular}}
```

\printdate

```
1496 \newcommand*{\printdate}{\small\textit{\@date}}
```

## 41.5 \maketitle for print output

\maketitle From the titling package, patched to add the publisher and subtitle.

```
1497 \providecommand{\maketitle}{}
1498 \if@titlepage
1499   \renewcommand{\maketitle}{\begin{titlepage}%
1500     \let\footnotesize\small
1501     \let\footnoterule\relax
1502     \let \footnote \thanks
1503     \@bsmarkseries
1504     \def\@makefnmark{\rlap{\@textsuperscript{%
1505       \normalfont\@bsthanksheadpre \tamark \@bsthanksheadpost}}}%
1506       \long\def\@makefntext##1{\makethanksmark ##1}
1507     \null\vfil
1508     \vskip 60\p@
1509     \vspace*{\droptitle}
1510     \maketitlehooka
1511     \ifcseempty{@published}
1512 {}
1513 {\@bsprepublished \@published \@bspostpublished}\maketitlehookaa}
1514 {\@bsprettitle \@title \@bsposttitle}
1515 \ifcseempty{@subtitle}
1516 {}
1517 {\maketitlehookaaa{\@bspresubtitle \@subtitle \@bspostsubtitle}}
1518 \maketitlehookb
1519 {\@bspreauthor \@author \@bspostauthor}
1520 \maketitlehookc
1521 {\@bspredate \@date \@bspostdate}
1522 \maketitlehookd
1523 \par
1524 \@thanks
1525 \vfil\null
1526 \end{titlepage}%
1527 \@bscontmark % \setcounter{footnote}{0}%
1528 %% \bsmtitleempty
1529 } % end titlepage defs
1530 \else
1531 \renewcommand{\maketitle}{\par
1532 \begingroup
1533 \@bsmarkseries
1534 \def\@makefnmark{\rlap{\@textsuperscript{%
1535   \normalfont\@bsthanksheadpre \tamark \@bsthanksheadpost}}}%
1536   \long\def\@makefntext##1{\makethanksmark ##1}
1537 \null\vfil
1538 \vskip 60\p@
1539 \vspace*{\droptitle}
1540 \par
1541 \@thanks
1542 \vfil\null
1543 \endgroup
1544 }
```

```

1536     \long\def\@makefntext##1{\makethanksmark ##1}
1537     \if@twocolumn
1538         \ifnum \col@number=\@ne
1539             \@maketitle
1540         \else
1541             \twocolumn[\@maketitle]%
1542         \fi
1543     \else
1544         \newpage
1545         \global\@topnum\z@
1546         \@maketitle
1547     \fi
1548     \thispagestyle{plain}\@thanks
1549 \endgroup
1550 \@bscontmark % \setcounter{footnote}{0}%
1551 %%% \bsmtitleempty
1552 } % end non-titlepage
1553
1554 \def\@maketitle{%
1555     \newpage
1556     \null
1557     \vskip 2em%
1558     \vspace*{\droptitle}
1559     \maketitlehooka
1560     \ifcsempy{@published}
1561 {}
1562 {\@bsprepublished \@published \@bspostpublished}\maketitlehookaa}
1563     {\@bspretitle \@title \@bsposttitle}
1564     \ifcsempy{@subtitle}
1565 {}
1566 {\maketitlehookaaa{\@bspresubtitle \@subtitle \@bspostsubtitle}}
1567     \maketitlehookb
1568     {\@bspreadauthor \@author \@bspostauthor}
1569     \maketitlehookc
1570     {\@bspredate \@date \@bspostdate}
1571     \maketitlehookd
1572     \par
1573     \vskip 1.5em}
1574 \fi
1575
1576 \providecommand{\maketitlehookaa}{}
1577
1578 \providecommand{\maketitlehookaaa}{}
1579
1580 \newcommand{\prepublished}[1]{%
1581 \def\@bsprepublished{#1}%
1582 }
1583
1584 \newcommand{\postpublished}[1]{%
1585 \def\@bspostpublished{#1}%

```



```

1586 }
1587
1588 \newcommand{\presubtitle}[1]{%
1589 \def\@bspresubtitle{#1}%
1590 }

```

`\presubtitle` Hook after printing the subtitle.

```

1591 \newcommand{\postsubtitle}[1]{%
1592 \def\@bspostsubtitle{#1}%
1593 }

```

Initial settings:

```

1594 \if@titlepage
1595 \prepublished{
1596 \vspace*{-\baselineskip}\vspace*{-\medskipamount}\vspace*{-2em}
1597 \begin{center}}
1598 \postpublished{\par\end{center}\vskip 2em}
1599
1600 \presubtitle{\unskip\begin{center}\unskip}
1601 \postsubtitle{\par\end{center}\vskip 2em}
1602 \else
1603 \prepublished{\begin{center}}
1604 \postpublished{\par\end{center}\vskip 0.5em}
1605
1606 \presubtitle{\begin{center}\unskip}
1607 \postsubtitle{\par\end{center}\vskip 0.5em}
1608 \fi
1609 \end{warpprint}

```

## 41.6 \maketitle for HTML output

An HTML div of class `titlepage` is created, inside of which a L<sup>A</sup>T<sub>E</sub>X PDF minipage is generated (without HTML tags), allowing the `\thanks` footnotes to be generated immediately at the end of the title page during HTML output. This is desirable when a large table of contents immediately follows the title.

`\thanks` are a form of footnotes used in the title page. See section 34 for other kinds of footnotes.

See `\thanksmarkseries{series}`, below, to set the style of the footnote marks.

for HTML output: 1610 \begin{warppHTML}

`\LWR@maketitlesetup` Patches `\thanks` macros to use L<sup>A</sup>T<sub>E</sub>X minipage footnotes.

```
1611 \newcommand*{\LWR@maketitlesetup}{%
```

Select which kind of footnote marks to use:

```
1612 \@bsmarkseries
```

```
1613 \@mpbsmarkseries
```

Redefine the footnote mark:

```
1614 \def\@makefnmark{\textsuperscript{\thefootnote}}%
```

```
\thefootnote ⇒ \nameuse{arabic}{footnote}, or
\thefootnote ⇒ \nameuse{fnsymbol}{footnote}
```

Redefine the footnote text:

```
1615 \long\def\@makefntext##1{%
```

Make the footnote mark and some extra horizontal space for the tags:

```
1616 \makethanksmark \LWR@origspace{1in}
```

```
\makethanksmark ⇒ \thanksfootmark ⇒ \tamark ⇒
\@thefnmark ⇒ \itshape a (or similar)
```

Print the text and a closing paragraph tag:

```
1617 ##1\LWR@htmltagc{/LWR@tagregularparagraph}%
```

```
1618 }%
```

```
1619 }
```

`\@fnsymbol`  $\{\langle counter \rangle\}$

Re-defined to use an HTML entity for the double vertical bar symbol. The original definition used `\|` which was not being found by `pdftotext`.

```
1620 \def\@fnsymbol#1{\ensuremath{\ifcase#1\or *\or \dagger\or \ddagger\or
```

```
1621 \mathsection\or \mathparagraph\or \text{\HTMLUnicode{2016}}\or
```

```
1622 **\or \dagger\dagger \or \ddagger\ddagger \else\@ctrerr\fi}}
```

`\maketitle` Creates an HTML titlepage div and typesets the title, etc.

Code from the titling package is adapted, simplified, and modified for HTML output.

```
1623 \renewcommand*{\maketitle}{%
```

An HTML titlepage div is used for all classes.

```
1624 \begin{titlepage}
```

Set up special patches:

```
1625 \LWR@maketitlesetup
```

Typeset the title, etc:

```
1626 \@maketitle
```

Immediately generate any \thanks footnotes:

```
1627 \@thanks
```

Close the HTML titlepage div:

```
1628 \end{titlepage}
```

Reset the footnote counter:

```
1629 \@bscontmark
```

```
1630 }
```

\@maketitle   Typesets the title, etc. for HTML:

```
1631 \DeclareDocumentCommand{\@maketitle}{-}{%
1632 \maketitlehooka
1633 \ifcempty{@published}
1634 {}
1635 {\@bsprepublished \@published \@bspostpublished}\maketitlehookaa}
1636 {\@bsprettitle \@title \@bsposttitle}
1637 \ifcempty{@subtitle}
1638 {}
1639 {\maketitlehookaaa{\@bspresubtitle \@subtitle \@bspostsubtitle}}
1640 \maketitlehookb
1641 {\@bspreauthor \@author \@bspostauthor}
1642 \maketitlehookc
1643 {\@bspredate \@date \@bspostdate}
1644 \maketitlehookd
1645 }
```

```
1646 \providecommand{\maketitlehookaa}{}
1647 \providecommand{\maketitlehookaaa}{}
1648 \providecommand{\maketitlehookb}{}
1649 \providecommand{\maketitlehookc}{}
1650 \providecommand{\maketitlehookd}{}
1651 }
```

\LWR@titlingmaketitle   \maketitle for use inside an HTML titlingpage environment.

```
1648 \newcommand*{\LWR@titlingmaketitle}{%
```

Typeset the title, etc:

```
1649 \@maketitle
```

Immediately generate any \thanks footnotes:

```
1650 \@thanks
```

```
1651 }
```

```
\thanksmarkseries {<series>}
```

Sets the type of footnote marks used by \thanks, where type is ‘arabic’, ‘roman’, ‘fnsymbol’, etc. Modified to use the L<sup>A</sup>T<sub>E</sub>X PDF minipage which is included with the title page.

```
1652 \let\LWR@origthanksmarkseries\thanksmarkseries
```

```
1653 \renewcommand{\thanksmarkseries}[1]{%
```

```
1654 \def\@mpbsmarkseries{%
```

```
1655 \renewcommand*{\thempfootnote}{\@nameuse{#1}{mpfootnote}}}
```

```
1656 \LWR@origthanksmarkseries{#1}
```

```
1657 }
```

```
1658 \end{warpHTML}
```

## 42 Abstract

for HTML output: 1659 \begin{warpHTML}

\abstractname User-redefinable title for the abstract.

Also over-written by the babel package.

```
1660 \providecommand*{\abstractname}{Abstract}
```

Env abstract

```
1661 \DeclareDocumentEnvironment{abstract}{}%
```

```
1662 {
```

```
1663 \BlockClass{abstract}
```

```
1664 \BlockClassSingle{abstracttitle}{\abstractname}
```

```
1665 }
```

```
1666 {
```

```

1667 \endBlockClass
1668 }

1669 \end{warpHTML}

```

## 43 Quote and verse

### 43.1 Citations and attributions

`\attribution` for use inside quote, quotation, verse:

ex: `\attribution{author name} --- \citetitle{book name}`

**for HTML output:**

```

1670 \begin{warpHTML}
1671 \newcommand{\attribution}[1]{%
1672 \InlineClass{attribution}{--\,#1}}% emdash
1673 \end{warpHTML}

```

**for PRINT output:**

```

1674 \begin{warpprint}
1675 \newcommand{\attribution}[1]{\textsc{---\,#1}}
1676 \end{warpprint}

```

`\citetitle` for use inside quote, quotation, verse:

**for HTML output:**

```

1677 \begin{warpHTML}
1678 \newcommand{\citetitle}[1]{%
1679 \InlineClass{citetitle}{--\,#1}}% emdash
1680 \end{warpHTML}

```

**for PRINT output:**

```

1681 \begin{warpprint}
1682 \newcommand{\citetitle}[1]{\textsl{---\,#1}}
1683 \end{warpprint}

```

### 43.2 Quotes, quotations

**for HTML output:**

```

1684 \begin{warpHTML}

```

Env `quote`

```

1685 \renewenvironment*{quote}
1686 {\LWR@htmlblocktag{blockquote}}
1687 {\LWR@htmlblocktag{/blockquote}}

```

```

1688
1689 \renewenvironment*{quotation}
1690 {\LWR@htmlblocktag{blockquotation}}
1691 {\LWR@htmlblocktag{/blockquotation}}

1692 \end{warpHTML}

```

### 43.3 Verse

**\attrib** The documentation for the **verse** and **memoir** packages suggest defining an **\attrib** command, which may already exist in current documents, but it will only work for print output. **lwarp** provides **\attribution**, which works for both print and HTML output. To combine the two so that **\attrib** is used for print and **\attribution** is used for HTML:

---

```

\begin{warpHTML}
\let\attrib\attribution
\end{warpHTML}

```

---

<p>Len <b>\leftskip</b></p> <p>Len <b>\leftmargini</b></p> <p>Len <b>\TMLvleftskip</b></p> <p>Len <b>\TMLleftmargini</b></p>	<p>These lengths are used by <b>verse</b> and <b>memoir</b> to control the left margin, and they may already be set by the user for print output. New lengths <b>\HTMLvleftskip</b> and <b>\HTMLleftmargini</b> are provided to control the margins in HTML output. These new lengths may be set by the user before any <b>verse</b> environment, and persist until they are manually changed again. One reason to change <b>\HTMLleftmargini</b> is if there is a wide <b>\flagverse</b> in use, such as the word “Chorus”, in which case the value of <b>\HTMLleftmargini</b> should be set to a wide enough length to contain “Chorus”. The default is wide enough for a stanza number.</p>
--	--

Horizontal spacing relies on **pdftotext**’s ability to discern the **-layout** of the text in the HTML-tagged PDF output. For some settings of **\HTMLleftmargini** or **\HTMLleftskip** the horizontal alignment may not work out exactly, in which case a label may be shifted by one space.

**for HTML & PRINT:** 1693 **\begin{warpall}**

The following lengths may be set in either print or HTML output, but are only used in HTML. This allows the user to set **\vleftskip** and **\leftmargini** for print output, and optionally select different values for HTML.

Len **\TMLvleftskip** Sets **\vleftskip** inside a **verse** environment in HTML.

```

1694 \newlength{\HTMLvleftskip}
1695 \setlength{\HTMLvleftskip}{1em}

```

Len \TMLleftmargini Sets \leftmargini inside a verse environment in HTML.

```
1696 \newlength{\HTMLleftmargini}
1697 \setlength{\HTMLleftmargini}{4.5em}

1698 \end{warpall}
```

for HTML output: 1699 \begin{warpHTML}

Env verse The verse environment will be placed inside a HTML pre.

```
1700 \AfterEndPreamble{
```

At the beginning of the verse environment:

```
1701 \AtBeginEnvironment{verse}
1702 {%
```

Pkg verse The verse or memoir packages can place stanza numbers to the left with their  
 Pkg memoir \flagverse command. Do not allow them to go into the left margin, which would  
 \flagverse cause pdfcrop to crop the entire page further to the left:

```
Len \leftskip 1703 \ifdef{\vleftskip}{%
1704 \setlength{\vleftskip}{\HTMLvleftskip}
1705 \setlength{\leftmargini}{\HTMLleftmargini}
1706 }{}
1707 \LWR@atbeginverbatim{verse}
1708 \unskip\vspace{-\baselineskip}
1709 }
```

After the end of the verse environment, which places the pre tag at the regular left margin:

```
1710 \AfterEndEnvironment{verse}{
1711 \unskip\vspace{-\baselineskip}
1712 \LWR@afterendverbatim
1713 }
```

Patch to place poemtitle inside an HTML span of class poemtitle:

```
1714 \ifdef{\poemtitle}{
1715 \DeclareDocumentCommand{\@vstypeptitle}{m}{%
1716 \vspace{\beforepoemtitleskip}%
1717 {\InlineClass{poemtitle}{\poemtitlefont #1}\par}%
1718 \vspace{\afterpoemtitleskip}%
1719 }
1720 }{}
1721
```

```

1722 }
1723 \end{warpHTML}

```

## 44 Verbatim

**for HTML output:** 1724 \begin{warpHTML}

Env **verbatim**

```

1725 \AfterEndPreamble{
1726 \AtBeginEnvironment{verbatim}{\LWR@atbeginverbatim{verbatim}\unskip\vspace*{-\baselineskip}}
1727 \AfterEndEnvironment{verbatim}{\unskip\vspace*{-\baselineskip}\LWR@afterendverbatim}
1728 }

1729 \end{warpHTML}

```

## 45 Fancyvrb

**for HTML & PRINT:** 1730 \begin{warpall}

Len **\VerbatimHTMLWidth** Width to use in HTML Verbatim environment.

This width is used when placing line numbers to the right. Ignored during print output.

```

1731 \newlength{\VerbatimHTMLWidth}
1732 \setlength{\VerbatimHTMLWidth}{4in}
1733 \end{warpall}

```

**for HTML output:** 1734 \begin{warpHTML}

Bool **LWR@verbtags** Used to temporarily turn off verbatim tags while doing **VerbatimInput** in the HTML head.

```

1735 \newbool{LWR@verbtags}
1736 \booltrue{LWR@verbtags}

```

For **\VerbatimFootnotes**:

```

1737 \renewcommand{\VerbatimFootnotes}{
1738 \PackageError{lwarp}
1739 {Verbatim footnotes are not yet supported by lwarp.}

```



1740 {This may be improved when lwarp uses LaTeX footnotes instead of pagenotes.}  
 1741 }

`\LWR@atbeginverbatim` `{\langle class\rangle}`

Encloses a verbatim environment with the given CSS class.

1742 `\newcommand*\LWR@atbeginverbatim}[1]`  
 1743 `{%`

Avoid excessive space between lines:

1744 `\setlength{\parskip}{0ex}%`

Stop generating HTML paragraph tags:

1745 `\LWR@stoppars%`

Create a new `pre` of the given class:

1746 `\ifbool\LWR@verbtags{\LWR@htmltag{pre class="#1"{} }{}}%`

Use a mono-spaced font to preserve horizontal positioning. If horizontal alignment is important for the user, use a mono-spaced font in the CSS for the `verse` class.

1747 `\LWR@origttfamily%`

Do not produce HTML tags for `\hspace` inside a `verse par`. Restore plain L<sup>A</sup>T<sub>E</sub>X `\hspace` functionality:

1748 `\let\hspace\LWR@orighspace%`  
 1749 }

`\LWR@afterendverbatim` Finishes enclosing a verbatim environment.

1750 `\newcommand*\LWR@afterendverbatim}{%`

Remove excess vertical space at the end of the `pre`:

1751 `\unskip%`

At the end of the environment, close the `pre`:

1752 `\ifbool\LWR@verbtags{\noindent\LWR@htmltag{/pre}}`  
 1753 `}`  
 1754 `{}}%`

Resume regular paragraph handling:

```
1755 \LWR@startpars%
1756 }
```

**\LWR@Verbatimclass** Holds the class of the following verbatim.

```
1757 \newcommand*{\LWR@Verbatimclass}{fancyvrb}
```

**Env VerbatimClass** `{\class}` [*Verbatim options*]

Creates a **Verbatim** enclosed in a **DIV** of the given class.

```
1758 \NewDocumentEnvironment{VerbatimClass}{m 0{}}
1759 {%
1760 \renewcommand*{\LWR@Verbatimclass}{#1}%
1761 \LWR@origVerbatim[#2]%
1762 }
1763 {\endVerbatim}
```

After the preamble is loaded, after any patches to **Verbatim**:

```
1764 \AfterEndPreamble{
```

Remember the original definition of **Verbatim**:

```
1765 \let\LWR@origVerbatim\Verbatim
```

**Env Verbatim** Patched to place the environment in a **fancyvrb** div, and the label in a **fancyvrblabel** div. Also corrects the left margin for line numbers. Also uses **VerbatimHTMLWidth** to control placement of line numbers on the right. Aligning the right margin requires knowing the width.

```
1766 \renewcommand*{\Verbatim}{%
1767 \renewcommand*{\LWR@Verbatimclass}{fancyvrb}%
1768 \LWR@origVerbatim%
1769 }
```

The following patches to **Verbatim** are executed at the start and end of the environment, depending on the choice of **frame**. Original code is from the **fancyvrb** package.

```
1770 \newcommand*{\LWR@fvstartnone}{%
1771 % \typeout{fvstartnone}%
1772 \ifbool{\LWR@verbtags}{\hbox to\z@{\LWR@htmltagc{div class="\LWR@Verbatimclass"}}}{%}
```

```

1773 \hbox to\z@\{\LWR@atbeginverbatim{verbatim}}}%
1774 }
1775
1776 \newcommand*\{\LWR@fvendnone}{%
1777 % \typeout{fvendnone}%
1778 \hbox to\z@\{\LWR@afterendverbatim}%
1779 \ifbool{\LWR@verbtags}{\hbox to\z@\{\LWR@htmltagc{/div}}}{}}}%
1780 }
1781
1782 \newcommand*\{\LWR@fvstartsingle}{%
1783 % \typeout{fvstartsingle}%
1784 \LWR@fvstartnone%
1785 \FV@BeginListFrame@Single%
1786 }
1787
1788 \newcommand*\{\LWR@fvendsingle}{%
1789 % \typeout{fvendsingle}%
1790 \FV@endListFrame@Single%
1791 \LWR@fvendnone%
1792 }
1793
1794 \newcommand*\{\LWR@fvstartline}{%
1795 % \typeout{fvstartline}%
1796 \LWR@fvstartnone%
1797 \FV@BeginListFrame@Lines%
1798 }
1799
1800 \newcommand*\{\LWR@fvendline}{%
1801 % \typeout{fvendline}%
1802 \FV@endListFrame@Lines%
1803 \LWR@fvendnone%
1804 }

```

The following patches select the start/left/right/end behaviors depending on **frame**.  
Original code is from the fancyvrb package.

```

1805 \def\FV@Frame@none{%
1806 \let\FV@BeginListFrame\LWR@fvstartnone%
1807 \let\FV@LeftListFrame\relax%
1808 \let\FV@RightListFrame\relax%
1809 \let\FV@endListFrame\LWR@fvendnone}
1810
1811 \def\FV@Frame@single{%
1812 \let\FV@BeginListFrame\LWR@fvstartsingle%
1813 \let\FV@LeftListFrame\FV@LeftListFrame@Single%
1814 \let\FV@RightListFrame\FV@RightListFrame@Single%
1815 \let\FV@endListFrame\LWR@fvendsingle}
1816
1817 \def\FV@Frame@lines{%

```

```

1818 \let\FV@BeginListFrame\LWR@fvstartline%
1819 \let\FV@LeftListFrame\relax%
1820 \let\FV@RightListFrame\relax%
1821 \let\FV@EndListFrame\LWR@fvendline}
1822
1823 \def\FV@Frame@topline{%
1824 \let\FV@BeginListFrame\LWR@fvstartline%
1825 \let\FV@LeftListFrame\relax%
1826 \let\FV@RightListFrame\relax%
1827 \let\FV@EndListFrame\LWR@fvendnone}
1828
1829 \def\FV@Frame@bottomline{%
1830 \let\FV@BeginListFrame\LWR@fvstartnone%
1831 \let\FV@LeftListFrame\relax%
1832 \let\FV@RightListFrame\relax%
1833 \let\FV@EndListFrame\LWR@fvendline}
1834
1835 \def\FV@Frame@leftline{%
1836 % To define the \FV@FrameFillLine macro (from \FV@BeginListFrame)
1837 \ifx\FancyVerbFillColor\relax%
1838 \let\FV@FrameFillLine\relax%
1839 \else%
1840 \@tempdima\FV@FrameRule\relax%
1841 \multiply\@tempdima-\tw@%
1842 \edef\FV@FrameFillLine{%
1843 {\noexpand\FancyVerbFillColor{\vrule\@width\number\@tempdima sp}%
1844 \kern-\number\@tempdima sp}}%
1845 \fi%
1846 \let\FV@BeginListFrame\LWR@fvstartnone%
1847 \let\FV@LeftListFrame\FV@LeftListFrame@Single%
1848 \let\FV@RightListFrame\relax%
1849 \let\FV@EndListFrame\LWR@fvendnone}

```

Adds the optional label to the top and bottom edges. Original code is from the fancyvrb package.

```

1850 \def\FV@SingleFrameLine#1{%
1851 \hbox to\z@{%
1852 % \kern\leftmargin
1853 \ifnum#1=\z@\relax
1854 \let\FV@Label\FV@LabelBegin
1855 \else
1856 \let\FV@Label\FV@LabelEnd
1857 \fi
1858 \ifx\FV@Label\relax
1859 % \FancyVerbRuleColor{\vrule \@width\linewidth \@height\FV@FrameRule}%
1860 \else
1861 \ifnum#1=\z@
1862 % \setbox\z@\hbox{\strut\enspace\FV@LabelBegin\enspace\strut}%

```

```

1863      \ifx\FV@LabelPositionTopLine\relax
1864 \else
1865      \LWR@htmltagc{div class="fancyvrblabel"}
1866 \LWR@origtextrm{\FV@LabelBegin}% \textrm preserves emdash
1867 \LWR@htmltagc{/div}
1868 \fi
1869      \else
1870 %      \setbox\z@\hbox{\strut\enspace\FV@LabelEnd\enspace\strut}%
1871      \ifx\FV@LabelPositionBottomLine\relax
1872 \else
1873      \LWR@htmltagc{div class="fancyvrblabel"}
1874 \LWR@origtextrm{\FV@LabelEnd}
1875 \LWR@htmltagc{/div}
1876 \fi
1877      \fi
1878
1879 \fi
1880 \hss
1881 }
1882 }

```

Processes each line, adding optional line numbers. Original code is from the fancyvrb package.

```

1883 \def\FV@ListProcessLine#1{%
1884   \hbox to \hsize{%
1885     %      \kern\leftmargin
1886     \hbox to \VerbatimHTMLWidth {%
1887       \ifcvoid\FV@LeftListNumber}{\kern 2.5em}%
1888 \FV@LeftListNumber%
1889 %      \FV@LeftListFrame
1890 \FancyVerbFormatLine{#1}%
1891 \hss%
1892 %      \FV@RightListFrame
1893 \FV@RightListNumber%
1894 }%
1895 \hss% required to avoid underfull hboxes
1896 }
1897 }

```

Env BVerbatim

```

1898 \AtBeginEnvironment{BVerbatim}
1899 {
1900 \LWR@atbeginverbatim{bverbatim}
1901
1902 }
1903
1904 \AfterEndEnvironment{BVerbatim}

```

```

1905 {
1906 \leavevmode\par\vspace{-\baselineskip}
1907 \LWR@afterendverbatim
1908 }

```

Env LVerbatim No changes required.

End of the modifications to make at the end of the preamble:

```

1909 } % \AfterEndPreamble

```

```
\UseVerbatim {\text}
```

No changes required.

```

1910 \end{warpHTML}

```

## 46 Theorems

```
\newtheorem {\text} [\textit{counter}] -or- [\textit{oldname}] {\text}
```

A few minor changes are made to supply HTML tags.

- The entire theorem is placed into a div of class `theorem`.
- The label for each theorem is placed inside a span of class `theoremlabel`.
- The contents are placed inside a div of class `theoremcontents`.

for HTML output: 1911 \begin{warpHTML}

```
\@begintheorem {\name} {\number}
```

```

1912 \renewcommand{\@begintheorem}[2]{%
1913 \LWR@htmldivclass{theorem}
1914 \InlineClass{theoremlabel}{#1\ #2\ }
1915 \LWR@htmldivclass{theoremcontents}
1916 }

```

```
\@opargbegintheorem {\name} {\number} {\oparg}
```

```

1917 \renewcommand{\@opargbegintheorem}[3]{%

```

```

1918 \BlockClass{theorem}
1919 \InlineClass{theoremlabel}{#1\ #2\ (#3)\ }
1920 \BlockClass{theoremcontents}
1921 }

```

\@endtheorem

```

1922 \renewcommand*{\@endtheorem}{%
1923 \endBlockClass% theoremcontents
1924 \endBlockClass% theorem
1925 }

```

```


1926 \end{warpHTML}

```

## 47 Lists

If using babel with French, use

```
\frenchbsetup{StandardLists=true}
```

 **French** to preserve the special HTML and enumitem list handling.

**enumitem** enumitem is pre-loaded during HTML output. Many of the spacing options are rendered irrelevant by pdftotext and HTML. Numbering, labels, and \newlist function correctly.

### 47.1 Itemize

**for HTML output:** 1927 \begin{warpHTML}

```

1928 \let\LWR@origitem\item

```

\LWR@itemizeitem [*label*]

Handles \item inside an itemize or enumerate.

See \LWR@openparagraph where extra \hspace is used to leave room for the label while inside a list during paragraph construction.

```

1929 \newcommand*{\LWR@itemizeitem}{%
1930 \LWR@stoppars%
1931 \LWR@startnewdepth{\LWR@depthlistitem}{\LWR@printcloselistitem{}}%
1932 \LWR@htmltag{li}%
1933 \LWR@startpars%
1934 \LWR@origitem%
1935 }

```

To have a blank item, use `\mbox{}`. This forces a new line in print output, matching the new line which will appear in HTML output. Ex:

---

```
begin{itemize}
item \mbox{}
  \begin{itemize}
...

```

---

Env **itemize** [*enumitem options*]

```
1936 \AtBeginEnvironment{itemize}{\LWR@itemizestart}
1937
1938 \newcommand*{\LWR@itemizestart}{%
1939 \LWR@stoppars%
1940 \LWR@pushoneclose{\LWR@depthlist}{\LWR@printcloseitemize{}}%
1941 \LWR@htmltag{ul style="list-style-type:none">{}}%
1942 \LWR@startpars%
1943 \let\item\LWR@itemizeitem%
1944 }
1945
1946 \AtEndEnvironment{itemize}{\LWR@itemizeend}
1947
1948 \newcommand*{\LWR@itemizeend}{%
1949 \LWR@stoppars%
1950 \LWR@closeprevious{\LWR@depthlistitem}%
1951 \LWR@closeoneprevious{}}%
1952 \LWR@startpars%
1953 }
```

## 47.2 Enumerate

An HTML unordered list is used with customized L<sup>A</sup>T<sub>E</sub>X-generated labels.

Env **enumerate** [*enumitem options*]

```
1954 \AtBeginEnvironment{enumerate}{\LWR@enumeratestart}
1955
1956 \newcommand*{\LWR@enumeratestart}{%
1957 \LWR@stoppars%
1958 \LWR@pushoneclose{\LWR@depthlist}{\LWR@printcloseitemize{}}%
1959 \LWR@htmltag{ul style="list-style-type:none">{}}%
1960 \LWR@startpars%
1961 \let\item\LWR@itemizeitem%
1962 }
```



```

1963
1964
1965 \AtEndEnvironment{enumerate}{\LWR@enumerateend}
1966
1967 \newcommand*{\LWR@enumerateend}{%
1968 \LWR@stoppars%
1969 \LWR@closeprevious{\LWR@depthlistitem}%
1970 \LWR@closeoneprevious{}}%
1971 \LWR@startpars%
1972 }

```

### 47.3 Description

`\LWR@descitem` [*(label)*] Handles an `\item` inside a description.

```

1973 \newcommand*{\LWR@descitem}[1][]{%
1974 {%
1975 \LWR@stoppars%
1976 \LWR@setlatestname{#1}%
1977 \LWR@startnewdepth{\LWR@depthlistitem}{\LWR@printclosedescitem{}}%
1978 \LWR@origitem[]%

```

Be sure the label doesn't print to the left of the rest of the file:

```

1979 \LWR@origspace{1in}
1980 \LWR@htmltag{dt}#1\LWR@htmltag{/dt}%
1981 \LWR@orignewline%
1982 \LWR@htmltag{dd}%
1983 \LWR@startpars%
1984 }

```

Env `description` [*(enumitem options)*]

```

1985 \AtBeginEnvironment{description}{\LWR@descriptionstart}
1986
1987 \newcommand*{\LWR@descriptionstart}{%
1988 \LWR@stoppars%
1989 \LWR@pushoneclose{\LWR@depthlist}{\LWR@printclosedescription{}}%
1990 \LWR@htmltag{dl}%
1991 \LWR@startpars%
1992 \let\item\LWR@descitem%
1993 }
1994
1995 \AtEndEnvironment{description}{\LWR@descriptionend}
1996
1997 \newcommand*{\LWR@descriptionend}{%

```

```

1998 \LWR@stoppars%
1999 \LWR@closeprevious{\LWR@depthlistitem}%
2000 \LWR@closeoneprevious{}%
2001 \LWR@startpars%
2002 }

```

```
\newlist {<name>} {<type>} {<maxdepth>}
```

```
\renewlist {<name>} {<type>} {<maxdepth>}
```

For `enumitem` lists, new lists must have the start and end actions assigned to the new environment. Renewed lists already have their actions assigned, and thus need no changes.

```

2003 \let\LWR@orignewlist\newlist
2004
2005 \renewcommand*\newlist[3]{%
2006 \LWR@orignewlist{#1}{#2}{#3}%
2007 \AtBeginEnvironment{#1}{\csuse{LWR@#2start}}}%
2008 \AtEndEnvironment{#1}{\csuse{LWR@#2end}}}%
2009 }

2010 \end{warpHTML}

```

## 48 Tabular

This is arguably the most complicated part of the entire package. Numerous tricks are employed to handle the syntax which is involved.

Limitations:

column types

- Vertical rules are not yet supported.
- `*` in a column specification is not used (so far). Repeat the column type the correct number of times.

 `\multirow &`  
`\multicolumn`



- `\multirow` and `\multicolumn` cannot be used at the same time. (No rectangular holes wider than one column or taller than one row.)

 `\multirow`


- For `\multirow`, insert `\mrowcell` into any empty multi-row cells. This will be a null function for the print output, and is a placeholder for parsing the table for HTML output.

`\multirow with rules`

- If a `\multirow` reaches to the bottom of a table, and `\bottomrule` does not go over to that edge, try adding a line of empty cells below the `\bottomrule`. This may be a browser bug.

- rule at last row
  - If a `\midrule` is desired after the last row, an additional row of blank cells must be used.
-  paragraphs
  - Multiple paragraphs in one cell of a p, b, m column must have `\newline` between paragraphs.
- `\cmidrule` width, trim
  - `\cmidrule` does not support width or trim options due to CSS limitations.
- `longtable` headings
  - For `longtable`, place headings and footings which do not apply to HTML inside `\warpprintonly{}`.
-  `\warpprintonly`
  - For `\toprule` and `\bottomrule`, when combined with a `warpprint` or `warpHTML` environment, if a “misplaced `\noalign`” error occurs, change
 

`This & That \endhead`  
 to  
`\warpprintonly{This & That \endhead}`

 and likewise with the other `\end` headings. Keep the `\endfirsthead` row unchanged, as it is still relevant to HTML output.
-  S columns
  - For S columns (from the `siunitx` package), while producing print output, anything non-numeric must be placed inside `{ }` braces, including commands such as `\multirow`. While producing HTML output, though, anything placed inside braces is not seen by `lwarp`’s tabular handling algorithm. To resolve this problem, make a copy of the row, with one version for print output, containing the extra braces, and another version for HTML output, without the extra braces, such as:
 

`\warpprintonly{1 & 2 & {\multirow{2}{2cm}{Text}} & 3 \\}`  
`\warpHTMLonly{1 & 2 & \multirow{2}{2cm}{Text} & 3 \\}`

## 48.1 Token lookahead

Used by `\LWR@futurenonSPACElet` to look at the next token.

for HTML output: 2011 `\begin{warpHTML}`

`\LWR@mynexttoken`

2012 `\newcommand\LWR@mynexttoken\relax`

`\futurelet` copies the next token then executes a function to analyze

`\LWR@futurenonSPACElet` does the same, but ignores intervening white space

Based on the `booktabs` style:

`\LWR@futurenonpacelet`

```
2013 \def\LWR@futurenonpacelet#1{\def\LWR@cs{#1}%
2014 \afterassignment\LWR@fnslone\let\nexttoken= }
2015 \def\LWR@fnslone{\expandafter\futurelet\LWR@cs\LWR@fnsltwo}
2016 \def\LWR@fnsltwo{%
2017 \expandafter\ifx\LWR@cs\@sptoken\let\next=\@BTfnslthree%
2018 \else\let\next=\nexttoken\fi\next}
2019 \def\@BTfnslthree{\afterassignment\LWR@fnslone\let\next= }
```

`\LWR@getmynexttoken` Looks ahead and copies the next token into `\LWR@mynexttoken`.

```
2020 \newcommand*{\LWR@getmynexttoken}{%
2021 % nothing must follow this next line
2022 \LWR@futurenonpacelet\LWR@mynexttoken\LWR@tabledatacolumnstag
2023 }
```

## 48.2 Booleans

Bool `LWR@startedrow` True if should print a row tag before this column.

```
2024 \newbool{LWR@startedrow}
2025 \boolfalse{LWR@startedrow}
```

Bool `LWR@doinghline` True if the next row will have an hline above it.

```
2026 \newbool{LWR@doinghline}
2027 \boolfalse{LWR@doinghline}
```

Bool `LWR@doingtbrule` True if the next row will have a top/bottom rule above it.

```
2028 \newbool{LWR@doingtbrule}
2029 \boolfalse{LWR@doingtbrule}
```

Bool `LWR@tableparcell` True if are handling a paragraph inside a table cell, so must close the paragraph tag before moving on.

```
2030 \newbool{LWR@tableparcell}
```

Bool `LWR@skippingmrowcell` True if are doing an empty multi-row cell, and thus there is no data tag to close.

```
2031 \newbool{LWR@skippingmrowcell}
```

Bool `LWR@intabularmetadata` True if are in a tabular but not in a data cell. Used to prevent extra HTML breaks if not inside table data.

```
2032 \newbool{LWR@intabularmetadata}
2033 \boolfalse{LWR@intabularmetadata}
```

### 48.3 Handling &

For technical discussion regarding problems redefining `\&`, See:  
<http://tex.stackexchange.com/questions/11638/where-do-i-find-futurelets-nasty-behaviour-documented/11860#11860>

`\LWR@closetabledatacell` If `LWR@skippingmrowcell` then there is no data tag to close. Otherwise, close any paragraphs, then close the data tag.

```
2034 \newcommand*{\LWR@closetabledatacell}{%
2035 \global\booltrue{LWR@intabularmetadata}%
2036 \ifbool{LWR@exitingtabular}{}%
2037 {% not exiting tabular
2038 \ifbool{LWR@skippingmrowcell}{}%
2039 {% not skippingmrowcell
```

Insert any `<` then any `@` column contents:

```
2040 \unskip%
2041 \LWR@getexparray{LWR@colafterspec}{\theLWR@tablecolspos}%
2042 \LWR@getexparray{LWR@colatspec}{\theLWR@tablecolspos}%
```

Close paragraphs:

```
2043 \ifbool{LWR@tableparcell}{\LWR@stoppars}{}%
2044 \global\boolfalse{LWR@tableparcell}%
```

Close the table data cell:

```
2045 \unskip\LWR@htmltag{/td}\LWR@orignewline%
2046 }% not skipping mrowcell
2047 }% not exiting tabular
2048 \global\boolfalse{LWR@skippingmrowcell}%
2049 }
```

`LWR@tabulardepth` tracks whether `&` is being used inside a `tabular`.

```
2050 \newcounter{LWR@tabulardepth}
2051 \setcounter{LWR@tabulardepth}{0}
2052
```

When not used inside a `tabular`, `&` performs its original function as recorded here ( with catcode 4 ).

```
2053 \def\LWR@origampmacro{&}
```

See below for why the group is used.

```
2054 \begingroup
```

`&` Will behave depending on whether it is being used inside `tabular`.

`&` is redefined to test whether it is inside a tabular environment, in which case it performs special processing for HTML conversion. If not, it behaves normally.

The `\catcode` allows the `&` character to be redefined.

```
2055 \catcode'\&=\active
2056
2057 \gdef&{%
2058 \ifthenelse{\cnttest{\value{LWR@tabulardepth}}{>}{0}}%
2059 {%
```

If not skipping a multirow cell, close the current data cell.

```
2060 \unskip%
2061 \LWR@closetabledatacell%
```

Move to the next column.

```
2062 \addtocounter{LWR@tablecolspos}{1}%
```

Look at the next token to decide multi or single column data tag.

```
2063 \LWR@getmynexttoken%
2064 }%
```

If not inside a tabular, performs the original action:

```
2065 {\LWR@origampmacro}%
2066 }
2067 \endgroup
```

Outside the group, `&` is left its original catcode for now.

`tikz` package seems to require `&` be left alone until after `tikz` has been loaded.

`\LWR@lwarpStart` finally makes `&` active at the beginning of the HTML conversion.

## 48.4 Handling \\

Inside tabular, \\ is redefined to \LWR@tabularendoffline

Throws away options \\[dim] or \\\*

\LWR@tabularendoffline

```
2068 \NewDocumentCommand{\LWR@tabularendoffline}{s o}
2069 {%
2070 \LWR@closetabledatacell%
```

Finish the previous row:

```
2071 \LWR@htmltag{/tr}\LWR@orignewline
2072 \global\booltrue{\LWR@intabularmetadata}
```

Not yet started a table row:

```
2073 \global\boolfalse{\LWR@startedrow}
```

Additional setup:

```
2074 \global\boolfalse{\LWR@doinghline}%
2075 \global\boolfalse{\LWR@doingtbrule}%
2076 \LWR@clearmidrules%
```

Start at first column:

```
2077 \setcounter{\LWR@tablecolspos}{1}
```

Look at the next token to decide between single column data tag or a special case:

```
2078 \LWR@getmynexttoken%
2079 }
```

## 48.5 Variables

```
2080 \newcommand*{\LWR@colsresult}{}%temp storage for column format results
2081 \newcommand*{\LWR@pposition}{}
2082 \newcommand*{\LWR@pleft}{}
2083 \newcommand*{\LWR@pright}{}%
```

\LWR@tablecolspec Holds the parsed column specification, of total width \LWR@tabletotalcols.

Will contain a string such as `llrrccpc`, exactly one letter per column, without `@`, `>`, `<`, or the vertical pipe.

2084 `\newcommand*{\LWR@tablecolspec}{}`

`\LWR@strresult` Holds the result of `Str` functions.

2085 `\newcommand*{\LWR@strresult}{}{}`

`\LWR@origcolspec` Holds the original column specs given to `tabular`.

2086 `\newcommand*{\LWR@origcolspec}{}{}`

Ctr `LWR@tablecolwidth` Holds the width of the table specification.

(This is not the total # columns.)

2087 `\newcounter{LWR@tablecolwidth}`

Ctr `LWR@tablecolspos` Where are currently looking into the table column specification.

2088 `\newcounter{LWR@tablecolspos}`

Ctr `LWR@tabletotalcols` Holds the final number of table columns.

2089 `\newcounter{LWR@tabletotalcols}`

Ctr `LWR@tabletotalcolsnext` Holds the next column while parsing. Is one more than `LWR@tabletotalcols`.

2090 `\newcounter{LWR@tabletotalcolsnext}`

`LWR@colatspec` A data array of specifications for `@` columns. The leftmost's index is `leftedge`, the others are counter values. See section 22.

`LWR@colbeforespec` A data array of specifications for `>` columns.

`LWR@colafterspec` A data array of specifications for `<` columns.

## 48.6 Parsing @, >, <, ! columns

`\LWR@parseatcolumn` Handles `@{text}` columns.

2091 `\newcommand*{\LWR@parseatcolumn}{%`



Move to the next token after the '@':

```
2092 \LWR@traceinfo{at column}%
2093 \addtocounter{LWR@tablecolspos}{1}%
```

Read the next token into \LWR@strresult, expanding once:

```
2094 \LWR@traceinfo{about to read the next token:}%
2095 \expandarg%
2096 \StrChar{\LWR@origcolspec}{\theLWR@tablecolspos}[\LWR@strresult]
2097 \fullexpandarg%
```

Store the result into a data array, expanding once out of \LWR@strresult:

```
2098 \LWR@traceinfo{have now read the next token}%
2099 \ifthenelse{\cnttest{\value{LWR@tabletotalcols}}=0}{
2100 {% left edge of the table:
2101 \LWR@traceinfo{at the left edge}%
2102 \LWR@setexparray{LWR@colatspec}{\leftedge}{\LWR@strresult}%
2103 \LWR@traceinfo{finished with the def}%
2104 }%
2105 {% not at the left edge:
2106 \LWR@traceinfo{not at the left edge}%
2107 \LWR@setexparray{LWR@colatspec}{\theLWR@tabletotalcols}{\LWR@strresult}%
2108 \LWR@traceinfo{at \theLWR@tabletotalcols: \LWR@colatspec(\theLWR@tabletotalcols)!}%
2109 }%
2110 \let\LWR@strresult\relax%
2111 }
```

\LWR@parsebeforecolumn Handles >{text} columns.

```
2112 \newcommand*{\LWR@parsebeforecolumn}{%
```

Move to the next token after the '>':

```
2113 \addtocounter{LWR@tablecolspos}{1}%
```

Read the next token, expanding once into \LWR@strresult:

```
2114 \expandarg%
2115 \StrChar{\LWR@origcolspec}{\theLWR@tablecolspos}[\LWR@strresult]%
2116 \fullexpandarg%
```

Store the result into a data array, expanding once out of \LWR@strresult:

```
2117 \LWR@setexparray{LWR@colbeforespec}{\theLWR@tabletotalcolsnext}{\LWR@strresult}%
2118 \let\LWR@strresult\relax%
2119 }
```

`\LWR@parseaftercolumn` Handles `<{text}` columns.

```
2120 \newcommand*{\LWR@parseaftercolumn}{%
```

Move to the next token after the '`<`':

```
2121 \addtocounter{LWR@tablecolspos}{1}%
```

Read the next token, expanding once into `\LWR@strresult`:

```
2122 % \StrChar{#1}{\theLWR@tablecolspos}[\LWR@strresult]
```

```
2123 \expandarg%
```

```
2124 \StrChar{\LWR@origcolspec}{\theLWR@tablecolspos}[\LWR@strresult]%
```

```
2125 \fullexpandarg%
```

Store the result into a data array, expanding once out of `\LWR@strresult`:

```
2126 \LWR@setexparray{LWR@colafterspec}{\theLWR@tabletotalcols}{\LWR@strresult}%
```

```
2127 \let\LWR@strresult\relax%
```

```
2128 }
```

`\LWR@parseskipcolumn` Handles columns to skip, such as `!`.

```
2129 \newcommand*{\LWR@parseskipcolumn}{%
```

```
2130 }
```

## 48.7 Parsing 'l', 'c', or 'r' columns

`\LWR@parsenormalcolumn` `{(thiscolumn)}`

Add to the accumulated column specs, advance counters, and pre-clear another column of at, before, and after specs.

```
2131 \newcommand*{\LWR@parsenormalcolumn}[1]{%
```

```
2132 \appto\LWR@tablecolspec{#1}%
```

```
2133 \addtocounter{LWR@tabletotalcols}{1}%
```

```
2134 \addtocounter{LWR@tabletotalcolsnext}{1}%
```

```
2135 \LWR@setexparray{LWR@colatspec}{\theLWR@tabletotalcolsnext}{\relax}%
```

```
2136 \LWR@setexparray{LWR@colbeforespec}{\theLWR@tabletotalcolsnext}{\relax}%
```

```
2137 \LWR@setexparray{LWR@colafterspec}{\theLWR@tabletotalcolsnext}{\relax}%
```

```
2138 }
```

Table 5: Tabular baseline

l	p	m	b	r
			bot	
		mid	bot	
l	par	mid	bot	r
	par	mid		
	par			

48.8 Parsing ‘p’, ‘m’, or ‘b’ columns

```
\LWR@parsepcolumn {<thiscolumn>} the width will be ignored

2139 \newcommand*{\LWR@parsepcolumn}[1]{%

converts to the given column type

2140 \LWR@parsenormalcolumn{#1}%

skips the following width

2141 \addtocounter{LWR@tablecolspos}{1}%
2142 }
```

48.9 Parsing the column specifications


 HTML CSS cannot exactly match the L<sup>A</sup>T<sub>E</sub>X concept of a baseline for a table row. Table 5 shows the L<sup>A</sup>T<sub>E</sub>X results for various vertical-alignment choices, with the baseline of the first column drawn across all the columns for comparison. See the p column specification in table 6 for details.

Table 6 describes how each kind of column is converted to HTML.

```
\LWR@parsetablecols {<colspecs>}

Scans the column specification left to right.

Builds \LWR@tablecolspec with the final specification, one column per entry. The
number of final columns is stored in LWR@tabletotalcols.

2143 \newcommand*{\LWR@parsetablecols}[1]{%
2144 \LWR@traceinfo{LWR@parsetablecols started}%
```

Table 6: Tabular HTML column conversions

- 
- l, r, c:** Converted to table cells without paragraph tags.  
 Uses CSS `vertical-align:middle` so that top or bottom-aligned cells may go above or below this cell.
- p:** Converted to table cells with paragraph tags. Ref: Table 5,  $\text{\LaTeX}$  places the top line of a parbox aligned with the rest of the text line, so CSS `vertical-align:bottom` is used to have the HTML result appear with the paragraph extending below the L, R, C cells at the middle, if possible. This may be confusing as a P cell may not top-align with an L,R,C cell in the HTML conversion, especially in the presence of a B cell, and two P cells side-by-side will be aligned at the bottom instead of the top. Some adjustment of the CSS may be desired, changing `td.tdp`, `td.tdP`, `td.tdprule`, and `td.tdPrule` to `vertical-align: middle`. Another possibility is to change L,R,C, and P to `vertical-align: top` and not worry about the alignment of B and M cells or trying to approximate  $\text{\LaTeX}$  baselines.
- m:** With paragraph tags, CSS `vertical-align:middle`.
- b:** With paragraph tags, CSS `vertical-align:top` so that the bottom of the text is closest to the middle of the text line.
- P, M, B:** Horizontally-centered versions,
- S:** Converted to 'r'. From the `siunitx` package.
- @, >, <:** Ignored so far.
-

Remember the original supplied column spec:

```
2145 \renewcommand*{\LWR@origcolspec}{#1}%
```

Clear the parsed resulting column spec:

```
2146 \renewcommand*{\LWR@tablecolspec}{}%
```

Total number of columns found so far. Also pre-initialize the first several columns of specs:

```
2147 \setcounter{LWR@tabletotalcols}{0}%
2148 \setcounter{LWR@tabletotalcolsnext}{1}%
2149 \LWR@setexparray{LWR@colatspec}{leftedge}{\relax}%
2150 \LWR@setexparray{LWR@colatspec}{1}{\relax}%
2151 \LWR@setexparray{LWR@colatspec}{2}{\relax}%
2152 \LWR@setexparray{LWR@colatspec}{3}{\relax}%
2153 \LWR@setexparray{LWR@colbeforespec}{1}{\relax}%
2154 \LWR@setexparray{LWR@colbeforespec}{2}{\relax}%
2155 \LWR@setexparray{LWR@colbeforespec}{3}{\relax}%
2156 \LWR@setexparray{LWR@colafterspec}{1}{\relax}%
2157 \LWR@setexparray{LWR@colafterspec}{2}{\relax}%
2158 \LWR@setexparray{LWR@colafterspec}{3}{\relax}%
```

Starting at the first column specification:

```
2159 \setcounter{LWR@tablecolspos}{1}%
```

Place the colspecs string length into \LWR@strresult, and remember the number of characters in the column specification:

```
2160 \LWR@traceinfo{about to StrLen}%
2161 \noexpandarg%
2162 \StrLen{#1}[\LWR@strresult]%
2163 \fullexpandarg%
2164 \LWR@traceinfo{finished StrLen}%
2165 \setcounter{LWR@tablecolswidth}{\LWR@strresult}%
```

Scan through the column specifications:

```
2166 \whileof{\not\value{LWR@tablecolspos}>\value{LWR@tablecolswidth}}{%
```

Place the next single-character column type into \LWR@strresult:

```
2167 \noexpandarg%
2168 \StrChar{#1}{\theLWR@tablecolspos}[\LWR@strresult]%
2169 \fullexpandarg%
```



Note that the parameter for a `p{spec}` column is a token list which will NOT match `l,c,r,p`.

```

2170 \IfStrEq{\LWR@strresult}{l}{\LWR@parsenormalcolumn{l}}{}%
2171 \IfStrEq{\LWR@strresult}{c}{\LWR@parsenormalcolumn{c}}{}%
2172 \IfStrEq{\LWR@strresult}{r}{\LWR@parsenormalcolumn{r}}{}%
2173 \IfStrEq{\LWR@strresult}{S}{\LWR@parsenormalcolumn{r}}{}%
2174 \IfStrEq{\LWR@strresult}{\detokenize{@}}{\LWR@parseatcolumn}{}%
2175 \IfStrEq{\LWR@strresult}{>}{\LWR@parsebeforecolumn}{}%
2176 \IfStrEq{\LWR@strresult}{<}{\LWR@parseaftercolumn}{}%
2177 \IfStrEq{\LWR@strresult}{!}{\LWR@parseskipcolumn}{}%
2178 \IfStrEq{\LWR@strresult}{p}{\LWR@parsepcolumn{p}}{}%
2179 \IfStrEq{\LWR@strresult}{m}{\LWR@parsepcolumn{m}}{}%
2180 \IfStrEq{\LWR@strresult}{b}{\LWR@parsepcolumn{b}}{}%
```

---

Many people define centered versions “P”, “M”, and “B”:

`\newcolumn{P}[1]{>\centering\arraybackslash}p{#1}`

---

```

2181 \IfStrEq{\LWR@strresult}{P}{\LWR@parsepcolumn{P}}{}%
2182 \IfStrEq{\LWR@strresult}{M}{\LWR@parsepcolumn{M}}{}%
2183 \IfStrEq{\LWR@strresult}{B}{\LWR@parsepcolumn{B}}{}%
2184 \addtocounter{LWR@tablecolspos}{1}%
2185 }%
2186 }%
```

## 48.10 Starting a new row

`\LWR@maybenewtablerow` If have not yet started a new table row, begin one now. Creates a new row tag, adding a class for `hline` or `tbrule` if necessary.

```

2187 \newcommand*{\LWR@maybenewtablerow}
2188 {%
2189 \ifbool{LWR@startedrow}%
2190 {}% started the row
2191 {% not started the row
```

Remember that now have started the row:

```

2192 \global\booltrue{LWR@startedrow}%
```

Create the row tag, with a class if necessary.

```

2193 \global\booltrue{LWR@intabularmetadata}%
2194 \ifbool{LWR@doinghline}%
```

```

2195 {\LWR@htmltag{tr class="hline"}}\LWR@orignewline}%
2196 {% not doing hline
2197 \ifbool{LWR@doingtbrule}%
2198 {\LWR@htmltag{tr class="tbrule"}}\LWR@orignewline}%
2199 {\LWR@htmltag{tr}\LWR@orignewline}%
2200 }% end of not doing hline
2201 }% end of not started the row
2202 }

```

## 48.11 Data opening tag

`\LWR@tabledatasinglecolumnstag` Print a table data opening tag with style for alignment

```

2203 \newcommand*{\LWR@tabledatasinglecolumnstag}%
2204 {%
2205 \LWR@maybe newtablerow%

```

If have found the end of tabular command, do not create the next data cell:

```

2206 \ifbool{LWR@exitingtabular}{}%
2207 {% not exiting tabular

```

Fetch the current column's alignment character into `\LWR@strresult`:

```

2208 \StrChar{\LWR@tablecolspec}{\the\LWR@tablecolspec}{\LWR@strresult}%

```

print the start of a new table data cell:

```

2209 \LWR@htmltag{td class="td%

```

append this column's spec:

```

2210 \LWR@strresult%

```

If this column has a `cmidrule`, add “rule” to the end of the HTML class tag:

```

2211 \ifthenelse{\equal{\LWR@getexparray{LWR@midrules}{\the\LWR@tablecolspec}}{Y}}{rule}{}%
2212 "}}%

```

If this is a p, m, or b column, allow paragraphs:

```

2213 \ifthenelse{%
2214 \equal{\LWR@strresult}{p}\OR%
2215 \equal{\LWR@strresult}{m}\OR%
2216 \equal{\LWR@strresult}{b}\OR%
2217 \equal{\LWR@strresult}{P}\OR%

```

```

2218 \equal{\LWR@strresult}{M}\OR%
2219 \equal{\LWR@strresult}{B}%
2220 }%
2221 {% allow pars
2222 \LWR@startpars%
2223 \global\booltrue{\LWR@tableparcell}%
2224 }% allow pars
2225 {}% no pars

```

Print the @ contents before first column, and then the > contents:

```

2226 \ifthenelse{\cnttest{\value{\LWR@tablecolspos}}=1}%
2227 {\LWR@getexparray{\LWR@colatspec}{leftedge}}% left edge
2228 {}% not left edge
2229 \LWR@getexparray{\LWR@colbeforespec}{\the\LWR@tablecolspos}%
2230 \global\boolfalse{\LWR@intabularmetadata}%
2231 }% not exiting tabular
2232 }%

```

## 48.12 Midrules

**LWR@midrules** LWR@midrules is a data array (section 22) of columns containing Y if a midrule should be created for each column.

**Ctr LWR@midrulecounter** Indexes across the LWR@midrules data array.

```

2233 \newcounter{\LWR@midrulecounter}

```

**\LWR@clearmidrules** Start new midrules. Called at beginning of tabular and also at \.

Clears all LWR@midrules markers for this line.

```

2234 \newcommand*{\LWR@clearmidrules}
2235 {%
2236 \setcounter{\LWR@midrulecounter}{1}%
2237 \whiledo{%
2238 \cnttest{\value{\LWR@midrulecounter}}{<=}{\value{\LWR@tablecolwidth}}%
2239 }%
2240 {%
2241 \LWR@setexparray{\LWR@midrules}{\the\LWR@midrulecounter}{\relax}%
2242 \addtocounter{\LWR@midrulecounter}{1}%
2243 }%
2244 }

```

**\LWR@subcmidrule** [*<width>*] [*<trim>*] [*<leftcolumn>*] [*<rightcolumn>*]



Marks LWR@midrules data array elements to be “Y” from left to right columns.

```

2245 \newcommand*{\LWR@subcmidrule}[4]{%
2246 \setcounter{LWR@midrulecounter}{#3}%
2247 \whileof{\cnttest{\value{LWR@midrulecounter}}{<=}{#4}}%
2248 {%
2249 \LWR@setexparray{LWR@midrules}{\theLWR@midrulecounter}{Y}%
2250 \addtocounter{LWR@midrulecounter}{1}%
2251 }% end of the whiledo
2252 }

```

`\LWR@docmidrule` [*width*] [*trim*] [*leftcolumn-rightcolumn*]

Marks LWR@midrules} array elements to be “Y” from left to right columns.

```

2253 \NewDocumentCommand{\LWR@docmidrule}{o d() >{\SplitArgument{1}{-}}m}%
2254 {\LWR@subcmidrule{#1}{#2}{#3}}

```

## 48.13 Multicolumns

### 48.13.1 Parsing multicolumns

```

2255 \newcounter{LWR@tablemulticolwidth}
2256 \newcounter{LWR@tablemulticolspos}

```

`\LWR@printmccoltype` [*colspec*] Print any valid column type found. Does not print @, >, or < columns or their associated tokens.

```

2257 \newcommand*{\LWR@printmccoltype}[1]{%
2258 \LWR@traceinfo{lwarp@printmccoltype -#1-}%

```

Get one token of the column spec:

```

2259 \StrChar{#1}{\theLWR@tablemulticolspos}[\LWR@strresult]%

```

Add to the HTML tag depending on which column type is found:

```

2260 \IfStrEq{\LWR@strresult}{l}{l}{}%
2261 \IfStrEq{\LWR@strresult}{c}{c}{}%
2262 \IfStrEq{\LWR@strresult}{r}{r}{}%
2263 \IfStrEq{\LWR@strresult}{p}{p}{}%
2264 \IfStrEq{\LWR@strresult}{m}{m}{}%
2265 \IfStrEq{\LWR@strresult}{b}{b}{}%
2266 \IfStrEq{\LWR@strresult}{P}{P}{}%
2267 \IfStrEq{\LWR@strresult}{M}{M}{}%
2268 \IfStrEq{\LWR@strresult}{B}{B}{}%

```

```

2269 \IfStrEq{\LWR@strresult}{S}{r}{}%
2270 \LWR@traceinfo{lwr@printmccoltype done}%
2271 }

```

`\LWR@multicolpartext` Print the data with paragraph tags:

```

2272 \newcommand*{\LWR@multicolpartext}{%
2273 \LWR@startpars%
2274 \LWR@multicoltext%
2275 \LWR@stoppars%
2276 }

```

`\LWR@multicolother` `{<colspec>}` For @, >, <, print the next token without paragraph tags:

```

2277 \newcommand*{\LWR@multicolother}[1]{%
2278 \addtocounter{LWR@tablemulticolspos}{1}%
2279 \StrChar{#1}{\theLWR@tablemulticolspos}[\LWR@strresult]%
2280 \LWR@strresult%
2281 }

```

`\LWR@printmccoldata` `{<colspec>}` Print the data for any valid column type found.

```

2282 \newcommand*{\LWR@printmccoldata}[1]{%
2283 \LWR@traceinfo{lwr@printmccoldata -#1}%

```

Get one token of the column spec:

```

2284 \StrChar{#1}{\theLWR@tablemulticolspos}[\LWR@strresult]%

```

Print the text depending on which column type is found. Also handles @, >, < as it comes to them.

```

2285 \IfStrEq{\LWR@strresult}{l}{\LWR@multicoltext}{}%
2286 \IfStrEq{\LWR@strresult}{c}{\LWR@multicoltext}{}%
2287 \IfStrEq{\LWR@strresult}{r}{\LWR@multicoltext}{}%
2288 \IfStrEq{\LWR@strresult}{p}{\LWR@multicolpartext}{}%
2289 \IfStrEq{\LWR@strresult}{m}{\LWR@multicolpartext}{}%
2290 \IfStrEq{\LWR@strresult}{b}{\LWR@multicolpartext}{}%
2291 \IfStrEq{\LWR@strresult}{P}{\LWR@multicolpartext}{}%
2292 \IfStrEq{\LWR@strresult}{M}{\LWR@multicolpartext}{}%
2293 \IfStrEq{\LWR@strresult}{B}{\LWR@multicolpartext}{}%
2294 \IfStrEq{\LWR@strresult}{S}{\LWR@multicolpartext}{}%
2295 \IfStrEq{\LWR@strresult}{\detokenize{@}}{\LWR@multicolother{#1}}{}%
2296 \IfStrEq{\LWR@strresult}{\detokenize{>}}{\LWR@multicolother{#1}}{}%
2297 \IfStrEq{\LWR@strresult}{\detokenize{<}}{\LWR@multicolother{#1}}{}%
2298 % \addtocounter{LWR@tablemulticolspos}{1}%
2299 % \StrChar{#1}{\theLWR@tablemulticolspos}[\LWR@strresult]%

```

```

2300 % \LWR@strresult%
2301 % }{}%
2302 \LWR@traceinfo{lw@printmccoldata done}%
2303 }

```

```
\parsemulticolumnalignment {\langle 1: colspec \rangle} {\langle 2: printresults \rangle}
```

Scan the multicolumn specification and execute the printfunction for each entry.

Note that the spec for a `p{spec}` column, or `@`, `>`, `<`, is a token list which will NOT match `l`, `c`, `r`, or `p`.

```

2304 \newcommand*{\LWR@parsemulticolumnalignment}[2]{%
2305 \setcounter{\LWR@tablemulticolspos}{1}%
2306 \StrLen{#1}[\LWR@strresult]%
2307 \setcounter{\LWR@tablemulticolwidth}{\LWR@strresult}%

```

Scan across the tokens in the column spec:

```

2308 \whiledo{%
2309 \not\value{\LWR@tablemulticolspos}>\value{\LWR@tablemulticolwidth}%
2310 }%
2311 {%

```

Execute the assigned print function for each token in the column spec:

```
2312 #2{#1}%
```

Move to the next token in the column spec:

```

2313 \addtocounter{\LWR@tablemulticolspos}{1}%
2314 }%
2315 }

```

### 48.13.2 High-level multicolumn interface

```
\LWR@domulticolumn {\langle 1: numcols \rangle} {\langle 2: colspec \rangle} {\langle 3: text \rangle}
```

```

2316 \newcommand{\LWR@multicoltext}{}
2317
2318 \NewDocumentCommand{\LWR@domulticolumn}{m m +m}{%
2319 \LWR@traceinfo{lw@domulticolumn -#1- -#2-}%
2320 \renewcommand{\LWR@multicoltext}{#3}%
2321 \LWR@maybenewtablerow%

```

Begin the opening table data tag:

```
2322 \LWR@htmltag{td colspan="#1"
2323 class="td%
```

Print the column type:

```
2324 \LWR@parsemulticolumnalignment{#2}{\LWR@printmccoltype}%
```

If this column has a `cmidrule`, add “rule” to the end of the HTML class tag.

If this position had a “Y” then add “rule”.

```
2325 \ifthenelse{\equal{\LWR@gettexpparray{\LWR@midrules}{\theLWR@tablecolspos}}{Y}}{rule}{}%
```

Close the class tag’s opening quote:

```
2326 "%
2327 }% end of the opening table data tag
2328 \global\boolfalse{\LWR@intabularmetadata}%
2329 \LWR@parsemulticolumnalignment{#2}{\LWR@printmccoldata}%
2330 }
```

### 48.13.3 Longtable captions

`Bool LWR@starredlongtable` Per the caption package, step the counter if `longtable*`.

```
2331 \newbool{\LWR@starredlongtable}
2332 \boolfalse{\LWR@starredlongtable}
```

Per the caption package. User-redefinable float type.

```
2333 \providecommand*\LTcapttype{table}
```

`\LWR@longtabledatacaptiontag` \* [*toc entry*] {*caption*}

```
2334 \NewDocumentCommand{\LWR@longtabledatacaptiontag}{s o +m}
2335 {%
```

Remember the latest name for `\nameref`:

```
2336 \IfValueTF{#2}{% optional given?
2337 \ifthenelse{\equal{#2}{}}{% optional empty?
2338 {\LWR@setlatestname{#3}}% empty
2339 {\LWR@setlatestname{#2}}% given and non-empty
2340 }% optional given
2341 {\LWR@setlatestname{#3}}% no optional
```

create a multicolumn across all the columns

```

2342 \LWR@domulticolumn{\theLWR@tabletotalcols}{P}{% \LWR@domulticolumn
2343 % \IfBooleanTF{#1}% star?
2344 % {\IfValueTF{#2}{\LWR@origcaption*{#2}{#3}}{\LWR@origcaption*{#3}}}
2345 % {\IfValueTF{#2}{\LWR@origcaption{#2}{#3}}{\LWR@origcaption{#3}}}
2346 \IfBooleanTF{#1}% star?

```

Star version, show a caption but do not make a LOT entry:

```

2347 {% yes star
2348 \LWR@htmlblocktag{figcaption}%
2349 #3%
2350 \LWR@htmlblocktag{/figcaption}%
2351 }%
2352 {% No star:

```

Not the star version:

Don't step the counter if \caption[] {A caption.}

```

2353 \ifbool{LWR@starredlongtable}%
2354 {%
2355 \ifthenelse{\equal{#2}{}}% TOC entry
2356 {}%
2357 {%
2358 \refstepcounter{\LTcapttype}%
2359 \protected@edef\@currentlabel{%
2360 \csuse{p@\LTcapttype}\csuse{the\LTcapttype}}%
2361 }%
2362 }{}%

```

Create an HTML caption. Afterwards, maybe make a LOT entry.

```

2363 \LWR@htmlblocktag{figcaption}%
2364 \csuse{fnum@\LTcapttype}\CaptionSeparator#3%
2365 \LWR@htmlblocktag{/figcaption}%

```

See if an optional caption was given:

```

2366 \ifthenelse{\equal{#2}{}}% TOC entry empty

```

if the optional caption was given, but empty, do not form a TOC entry

```

2367 {}%

```

If the optional caption was given, but might only be []:

```

2368 {% TOC entry not empty

```

2369 \IfNoValueTF{#2}% No TOC entry?

The optional caption is []:

```
2370 {% No TOC entry
2371 \addcontentsline%
2372 {\csuse{ext@LTcapttype}}%
2373 {\LTcapttype}%
2374 {%
2375 \protect\numberline%
2376 {\csuse{p@LTcapttype}\csuse{theLTcapttype}}%
2377 {\ignorespaces #3\protect\relax}%
2378 }%
2379 }% end of No TOC entry
```

The optional caption has text enclosed:

```
2380 {% yes TOC entry
2381 \addcontentsline%
2382 {\csuse{ext@LTcapttype}}%
2383 {\LTcapttype}%
2384 {%
2385 \protect\numberline%
2386 {\csuse{p@LTcapttype}\csuse{theLTcapttype}}%
2387 {\ignorespaces #2\protect\relax}%
2388 }%
2389 }% end of yes TOC entry
2390 }% end of TOC entry not empty
2391 }% end of no star
2392 }% end of \LWR@domulticolumn
2393
2394 \addtocounter{LWR@tablecolspos}{\theLWR@tabletotalcols}
2395 \addtocounter{LWR@tablecolspos}{-1}
2396
2397 }
```

#### 48.13.4 \tabledatamulticolumntag

\LWR@tabledatamulticolumntag {<numcols>} {<alignment>} {<text>}

```
2398 \NewDocumentCommand{\LWR@tabledatamulticolumntag}{m m +m}%
2399 {%
2400 \LWR@domulticolumn{#1}{#2}{#3}%
2401 \addtocounter{LWR@tablecolspos}{#1}%
2402 \addtocounter{LWR@tablecolspos}{-1}%
2403 }
```

## 48.14 Multirow

Pkg multirow

```
\LWR@tabledatamultirowtag {<numrows>} [<bigstruts>] {<width>} [<fixup>] {<text>}
```

```
2404 \NewDocumentCommand{\LWR@tabledatamultirowtag}{m o m o m}%
```

```
2405 {%
```

```
2406 \LWR@maybenewtablerow%
```

Print the start of a new table data cell:

```
2407 \LWR@htmltag{td rowspan="#1" class="td%
```

Append this column's spec:

```
2408 \StrChar{\LWR@tablecolspec}{\theLWR@tablecolspos}%
```

If this column has a cmidrule, add “rule” to the end of the HTML class tag:

```
2409 \ifthenelse{\equal{\LWR@gettexpparray{\LWR@midrules}{\theLWR@tablecolspos}}{Y}}{rule}{}%
```

```
2410 "{}}
```

While printing the text, redefine \\\ to generate a new line

```
2411 \begingroup \let\\\LWR@endoffline #5 \endgroup
```

```
2412 \LWR@stoppars%
```

```
2413 \global\boolfalse{\LWR@intabularmetadata}%
```

```
2414 }%
```

## 48.15 Utility macros inside a table

```
2415 \newcommand*{\LWR@donothing}{}%
```

```
2416 \newcommand*{\LWR@domidrule}{\booltrue{\LWR@doinghline}}%
```

```
2417 \newcommand*{\LWR@dotbrule}{\booltrue{\LWR@doingtbrule}}%
```

## 48.16 Checking for a new table cell

`\LWR@tabledatacolumnntag` Open a new HTML table cell unless the next token is for a macro which does not create data, such as `\hline`, `\toprule`, etc:

```
2418 \newbool{\LWR@exitingtabular}
```

```
2419 \newcommand*{\LWR@tabledatacolumnntag}%
```

```
2420 {%
```

`\show\LWR@mynexttoken` to see what tokens to look for

If not any of the below, start a new table cell:

```
2421 \let\mynext\LWR@tabledatasinglecolumnntag%
```

If exiting the tabular:

```
2422 \ifthenelse{\isequivalentto{\LWR@mynexttoken}{\end}}{%
2423 {\booltrue\LWR@exitingtabular}}{ }%
```

`longtable` can have a caption in a cell

```
2424 \ifthenelse{\isequivalentto{\LWR@mynexttoken}{\caption}}{%
2425 {\let\mynext\LWR@donothing}}{ }%
```

Look for other things which would not start a table cell:

```
2426 \ifthenelse{\isequivalentto{\LWR@mynexttoken}{\multicolumn}}{%
2427 {\let\mynext\LWR@donothing}}{ }%
2428 \ifthenelse{\isequivalentto{\LWR@mynexttoken}{\multirow}}{%
2429 {\let\mynext\LWR@donothing}}{ }%
```

if come to an `\mrowcell`, this is a cell to be skipped over

```
2430 \ifthenelse{\isequivalentto{\LWR@mynexttoken}{\mrowcell}}{%
2431 {\let\mynext\LWR@donothing}}{ }%
2432 %
2433 \ifthenelse{\isequivalentto{\LWR@mynexttoken}{\hline}}{%
2434 {\let\mynext\LWR@donothing}}{ }%
2435 %
2436 \ifthenelse{\isequivalentto{\LWR@mynexttoken}{\toprule}}{%
2437 { }%
2438 \let\mynext\LWR@donothing}}{ }%
2439 %
2440 \ifthenelse{\isequivalentto{\LWR@mynexttoken}{\midrule}}{%
2441 {\let\mynext\LWR@donothing}}{ }%
2442 %
2443 \ifthenelse{\isequivalentto{\LWR@mynexttoken}{\cmidrule}}{%
2444 {\let\mynext\LWR@donothing}}{ }%
2445 %
2446 \ifthenelse{\isequivalentto{\LWR@mynexttoken}{\cline}}{%
2447 {\let\mynext\LWR@donothing}}{ }%
2448 %
2449 \ifthenelse{\isequivalentto{\LWR@mynexttoken}{\bottomrule}}{%
2450 {\let\mynext\LWR@donothing}}{ }%
2451 %
2452 \ifthenelse{\isequivalentto{\LWR@mynexttoken}{\warpprintonly}}
```



```

2453 {\let\mynext\LWR@donothing}{}%
2454 %
2455 \ifthenelse{\isequivalentto{\LWR@mynexttoken}{\warpHTMLonly}}
2456 {\let\mynext\LWR@donothing}{}%

```

no action for an `\end` token

Add similar to the above for any other non-data tokens which might appear in the table.

Start the new table cell if was not any of the above:


```

2457 \mynext%
2458 }

2459 \end{warpHTML}

```

## 48.17 `\mrowcell`

`\mrowcell` The user must insert `\mrowcell` into any multirow cells which must be skipped.  
 This command has no action during print output.

**for HTML & PRINT:**

```

2460 \begin{warpall}
2461 \newcommand*{\mrowcell}{}
2462 \end{warpall}

```

## 48.18 New `\tabular` definition

**for HTML output:**

```

2463 \begin{warpHTML}

```

Env `LWR@tabular` [*⟨verticalposition⟩*] {*⟨colspecs⟩*}

The new tabular environment will be `\let` in `\LWR@LwarpStart`, since `siunitx` might redefine `tabular` in the user's document.

```

2464 \newenvironment*{LWR@tabular}[2]{}
2465 {%
2466 \LWR@traceinfo{tabular started}%
2467 \begingroup%
2468 \addtocounter{LWR@tabulardepth}{1}%

```

Not yet started a table row:

```

2469 \global\boolfalse{LWR@startedrow}%

```

Not yet doing an hline:

```
2470 \global\boolfalse{LWR@doinghline}%
```

Not yet doing a top/bottom rule:

```
2471 \global\boolfalse{LWR@doingtbrule}%
```

Have not yet found the end of tabular command:

```
2472 \boolfalse{LWR@exitingtabular}%
```

Create the `table` tag:

```
2473 \global\booltrue{LWR@intabularmetadata}%
```

```
2474 \LWR@htmlblocktag{table}%
```

Parse the table columns:

```
2475 \LWR@parsetablecols{#2}%
```

Table col spec is: `\LWR@tablecolspec` which is a string of `llccrr`, etc.

Do not place the table inside a paragraph:

```
2476 \LWR@stoppars%
```

Track column # for setting text-align:

```
2477 \setcounter{LWR@tablecolspos}{1}%
```

Start looking for midrules:

```
2478 \LWR@clearmidrules%
```

`\` becomes a macro to end the table row:

```
2479 \let\\\LWR@tabularendoffline%
```

The following may appear before a data cell is created, so after doing their actions, we look ahead with `\LWR@getmynexttoken` to see if the next token might create a new data cell:

```
2480 \renewcommand*{\hline}{\LWR@domidrule\LWR@getmynexttoken}%
```

```
2481 \newcommand*{\midrule}{\LWR@domidrule\LWR@getmynexttoken}%
```

```
2482 \NewDocumentCommand{\cmidrule}{o d() m}%
```

```
2483 {\LWR@docmidrule[##1](##2){##3}\LWR@getmynexttoken}%
```

```
2484 \RenewDocumentCommand{\cline}{m}%
```

```

2485 {\LWR@docmidrule{##1}\LWR@getmynexttoken}%
2486 \newcommand*{\toprule}{\LWR@dotbrule\LWR@getmynexttoken}%
2487 \newcommand*{\bottomrule}{\LWR@dotbrule\LWR@getmynexttoken}%

```

The following create data cells and will have no more data in this cell, so we do not want to look ahead for a possible data cell, so do not want to use `\LWR@getmynexttoken`.

```

2488 \let\multicolumn\LWR@tabledatamulticolumntag%
2489 \let\multirow\LWR@tabledatamultirowtag%
2490 \renewcommand*{\mrowcell}{\global\booltrue\LWR@skippingmrowcell}}%
2491 \let\caption\LWR@longtabledatacaptiontag%

```

Reset for new processing:

```

2492 \global\boolfalse\LWR@tableparcell}%
2493 \global\boolfalse\LWR@skippingmrowcell}%

```

Look ahead for a possible table data cell:

```

2494 \LWR@getmynexttoken%
2495 }%

```

Ending the environment:

```

2496 {%
2497 \LWR@closetabledatacell%
2498 \LWR@htmlblocktag{/tr}%
2499 \LWR@htmlblocktag{/table}%
2500 \global\boolfalse\LWR@intabularmetadata}%

2501 \addtocounter\LWR@tabulardepth{-1}%
2502 \endgroup%
2503 }

2504 \end{warpHTML}

```

## 48.19 Array

Pkg `array`

`array` is also automatically loaded by `siunitx`.

## 49 Cross-references

Sectioning commands have been emulated from scratch, so the cross-referencing commands are custom-written for them. Emulating both avoids several layers of patches.

The `zref` package is used to remember section name, file, and lateximage depth and number for each label.

Table 7 shows the data structures related to cross-referencing.

for HTML output: 2505 `\begin{warpHTML}`

### 49.1 Setup

`\@currentlabelname` To remember the most recently defined section name, description, or caption, for `\nameref`.

2506 `\newcommand*{\@currentlabelname}{}`

`\LWR@stripperperiod` `{\textit}{[<.]}`

Removes a trailing period.

2507 `\def\LWR@stripperperiod#1.\ltx@empty#2\@nil{#1}%`

`\LWR@setlatestname` `{\object name}`

Removes `\label`, strips any final period, and remembers the result.

2508 `\newcommand*{\LWR@setlatestname}[1]{%`

Remove `\label` and other commands from the name, the strip any final period.  
See `zref-titleref` and `getttitlestring`.

2509 `\GetTitleStringExpand{#1}%`

2510 `\edef\@currentlabelname{\detokenize\expandafter{\GetTitleStringResult}}%`

2511 `\edef\@currentlabelname{%`

2512 `\expandafter\LWR@stripperperiod\@currentlabelname%`

2513 `\ltx@empty.\ltx@empty\@nil%`

2514 `}%`

2515 `}`

Table 7: Cross-referencing data structures

<b>Original L<sup>A</sup>T<sub>E</sub>X:</b>	(print and HTML)
<b>\refstepcounter:</b> Steps the counter and sets \@currentlabel.	
<b>\@currentlabel:</b> \p@<ctr>\the<ctr> Updated by \refstepcounter.	
<b>\label:</b> Writes to the .aux file: \newlabel{<label>}{\@currentlabel}{\thepage}}	
<b>\newlabel:</b> When the .aux file is read, sets \r@<label>.	
<b>\r@&lt;label&gt;:</b> Set to: {\@currentlabel}{\thepage}}	
<b>\ref:</b> Returns the first part of \r@<label>.	
<b>\pageref:</b> Returns the second part of \r@<label>.	
<b>Added by amsmath:</b>	(print and HTML)
<b>\label:</b> Delayed until math is completed.	
<b>Added by lwarp:</b>	(HTML only)
<b>\nameref:</b> Emulated from hyperref for lwarp. See section 49.4.	
<b>\ref and \nameref:</b> Adds HTML tags. See section 49.4.	
<b>Added by cleverref:</b>	(print and HTML)
<b>\refstepcounter:</b> Added: sets \cref@currentlabel.	
<b>\cref@currentlabel:</b> (<type>=<ctr> unless an alias is used): [<type>][\arabic{<ctr>}][<parent ctrs>]{\p@<ctr>\the<ctr>} Also see section 34.4 for use with footnotes.	
<b>\label:</b> Writes to the .aux file: \newlabel{<label>@cref}{\cref@currentlabel}{\thepage}}	
<b>\newlabel:</b> (Unchanged.) When the .aux file is read, sets \r@<label>@cref.	
<b>\r@&lt;label&gt;@cref:</b> Set to: {\cref@currentlabel}{\thepage}}	
<b>Utility functions:</b> See \cref@getlabel, \cref@gettype, \cref@getcounter, \cref@getprefix.	
<b>Cross-referencing names:</b> \crefname and \Crefname assign human-readable names for references to this counter type.	
<b>Additionally patched by lwarp:</b>	(HTML only)
<b>\cref, etc.:</b> Modified for lwarp. See section 58.	
<b>\label:</b> Adds HTML tags (section 49.3), plus \splabel data (section 49.2): <b>zLWR@name:</b> The section name for this label. <b>zLWR@htmlfilenumber:</b> The filenumber or name for this label. <b>zLWR@lateximagedepth:</b> The lateximagedepth for this label. <b>zLWR@lateximagenumber:</b> The lateximagenumber for this label.	
<b>\label inside math:</b> See section 53.4.1.	
<b>Footnotes:</b> See \noteentry in section 34.4.	

## 49.2 Zref setup

See:

[http://tex.stackexchange.com/questions/57194/  
extract-section-number-from-equation-reference](http://tex.stackexchange.com/questions/57194/extract-section-number-from-equation-reference)

Create a new property list called special:

```
2516 \zref@newlist{special}
```

Define a new property which has the name of the most recently declared section:

```
2517 \zref@newprop{zLWR@name}{\@currentlabelname}
```

Define a new property which has either a filename or a file number:

```
2518 \zref@newprop{zLWR@htmlfilenumber}{%
2519 \ifbool{FileSectionNames}{\LWR@thisfilename}{\theLWR@htmlfilenumber}}%
2520 }%
```

Additional properties for lateximages:

```
2521 \zref@newprop{zLWR@lateximagedepth}{\arabic{LWR@lateximagedepth}}
2522 \zref@newprop{zLWR@lateximagenumber}{\arabic{LWR@lateximagenumber}}
```

`zLWR@htmlfilenumber` property holds the file number or name

Add a `LWR@htmlfilenumber` property, and lateximage properties to special:

```
2523 \zref@addprop{special}{zLWR@name}
2524 \zref@addprop{special}{zLWR@htmlfilenumber}
2525 \zref@addprop{special}{zLWR@lateximagedepth}
2526 \zref@addprop{special}{zLWR@lateximagenumber}
```

Returns the selected field:

```
2527 \newcommand*{\LWR@spreff}[2]{%
2528 \zref@extractdefault{#1}{#2}{??}}
```

`\LWR@nameref`  $\{\langle label \rangle\}$  Returns the section name for this label:

```
2529 \newcommand*{\LWR@nameref}[1]{%
2530 \LWR@spreff{#1}{zLWR@name}%
2531 }
```

`\LWR@htmlfileref`  $\{\langle label \rangle\}$  Returns the file number for this label:

```

2532 \newcommand*{\LWR@htmlfileref}[1]{%
2533 \LWR@traceinfo{\LWR@htmlfileref A: !#1!}%
2534 \LWR@sprep{#1}{z\LWR@htmlfilenumber}%
2535 \LWR@traceinfo{\LWR@htmlfileref B}%
2536 }

```

`\LWR@lateximagedepthref` `{\label}` Returns the `lateximagedepth` for this label:

```

2537 \newcommand*{\LWR@lateximagedepthref}[1]{%
2538 \LWR@sprep{#1}{z\LWR@lateximagedepth}%
2539 }

```

`\LWR@lateximagenumberref` `{\label}` Returns the `lateximagenumber` for this label:

```

2540 \newcommand*{\LWR@lateximagenumberref}[1]{%
2541 \LWR@sprep{#1}{z\LWR@lateximagenumber}%
2542 }

```

`\LWR@splabel` `{\label}` Sanitize the name and then creates the label:

```

2543 \newcommand*{\LWR@splabel}[1]{%
2544 \LWR@setlatestname{@currentlabelname}%
2545 \zref@labelbylist{#1}{special}}

```

### 49.3 Labels

`\LWR@subsublabel` `{\label}` Creates an HTML id tag.

```

2546 \newcommand*{\LWR@subsublabel}[1]{%

```

Create an HTML id tag unless are inside a `lateximage`, since it would appear in the image:

```

2547 \ifthenelse{\cnttest{\value{\LWR@lateximagedepth}}{>}{0}}{%
2548 {}%
2549 {% not lateximage

```

If not doing a `lateximage`, create an HTML ID tag: (To be factored...)

```

2550 \ifbool{\LWR@doingstartpars}%
2551 {% pars allowed
2552 \ifbool{\LWR@doingapar}
2553 {% par started
2554 \LWR@htmltag{a id="#1"}{\LWR@htmltag{/a}%
2555 }% par started

```

```

2556 {% par not started
2557 \LWR@stoppars%
2558 \LWR@htmltag{a id="#1"{}}\LWR@htmltag{/a}%
2559 \LWR@startpars%
2560 }% par not started
2561 }% pars allowed
2562 {% pars not allowed
2563 \LWR@htmltag{a id="#1"{}}\LWR@htmltag{/a}%
2564 }% pars not allowed
2565 }% not lateximage
2566 }

```

`\LWR@sublabel`  $\{\langle label \rangle\}$

`\label` during HTML output. Creates a traditional L<sup>A</sup>T<sub>E</sub>X label, adds a special label with additional information, and creates an HTML id tag.

Used by `\LWR@htmllabel` and `\LWR@htmlmathlabel`.

```

2567 \newcommand*{\LWR@sublabel}[1]{%

```

Create a traditional LaTeX label, as modified by `cleveref`:

```

2568 \LWR@traceinfo{\LWR@sublabel: !#1!}%
2569 \LWR@origlabel{#1}%

```

Create a special label which holds the section number, `LWR@htmlfilenumber`, `LWR@lateximagedepth`, and `LWR@lateximagenumber`:

```

2570 \LWR@traceinfo{about to splabel}%
2571 \LWR@traceinfo{filesectionnames is \ifbool{FileSectionNames}{true}{false}}%
2572 \LWR@traceinfo{\LWR@thisfilename is !\LWR@thisfilename!}%
2573 \LWR@traceinfo{\LWR@htmlfilenumber is \the\LWR@htmlfilenumber}%
2574 \LWR@splabel{#1}%
2575 \LWR@subsublabel{#1}%
2576 \LWR@traceinfo{finished with \LWR@sublabel}%
2577 }

```

`\LWR@newlabel`  $\{\langle label \rangle\}$

`\label` during HTML output when not in math mode, removing extra spaces around the label, as done by regular L<sup>A</sup>T<sub>E</sub>X `\label`.

```

2578 \newcommand*{\LWR@newlabel}[1]{%
2579 \LWR@traceinfo{\LWR@newlabel: !#1!}%
2580 % \@bsphack%
2581 \LWR@sublabel{#1}%
2582 % \@esphack%

```



```
2583 \LWR@traceinfo{lwr@newlabel: done}%
2584 }
```

## 49.4 References

`\LWR@startref` `{\label}` (Common code for `\ref` and `\nameref`.)

Open an HTML tag reference to a filename, # character, and a label.

```
2585 \newcommand*{\LWR@startref}[1]
2586 {%
2587 \edef\LWR@lhref{\LWR@lateximagedepthref{#1}}
2588 % \typeout{***LWR@startref A: !#1!}%
```

Create the tag:

```
2589 \LWR@htmltag{a href="%
2590 % \typeout{***LWR@startref B}%
2591 \LWR@htmlrefsectionfilename{#1}%
2592 % \typeout{***LWR@startref C}%
2593 \#%
```

See if `LWR@lateximagedepth` is unknown:

```
2594 % \typeout{***LWR@startref D: !#1!}%
2595 \ifthenelse{\equal{\LWR@lhref}{??}}%
```

“??” if `LWR@lateximagedepth` is unknown:

```
2596 {??}%
```

If `LWR@lateximagedepth` is known:

```
2597 {%
2598 % \typeout{***LWR@startref D1}%
2599 % \typeout{***depthref is \LWR@lhref}%
2600 \ifthenelse{\cinttest{\LWR@lhref}{>}{0}}%
2601 {%
2602 % \typeout{***LWR@startref D2}%
2603 lateximage\LWR@lateximagenumberref{#1}%
2604 }%
2605 {%
2606 % \typeout{***LWR@startref D3}%
2607 #1%
2608 }%
2609 }%
2610 % \typeout{***LWR@startref E}%
```

closing quote:

```
2611 "{}}%
2612 % \typeout{***LWR@startref F}%
2613 }
```

`\ref * {<label>}` Create an internal document reference link, or without a link if starred per hyperref.

```
2614 \NewDocumentCommand{\LWR@newref}{s m}{%
2615 \LWR@traceinfo{\LWR@newref #2}%
2616 \IfBooleanTF{#1}%
2617 {\LWR@origref{#2}}%
2618 {%
2619 \LWR@startref{#2}%
2620 \LWR@origref{#2}%
2621 \LWR@htmltag{/a}%
2622 }%
2623 }
2624
```

`\pagerefPageFor` Text for starred page references.

```
2625 \newcommand*{\pagerefPageFor}{page for}
```

`\pageref * {<label>}` Create an internal document reference, or just the unlinked number if starred, per hyperref.

```
2626 \NewDocumentCommand{\LWR@newpageref}{s m}{%
2627 \IfBooleanTF{#1}%
2628 {\(\pagerefPageFor\ \LWR@origref{#2})}%
2629 {\(\cpageref{#2})}%
2630 }
```

`\nameref {<label>}`

```
2631 \newcommand*{\nameref}[1]{%
2632 \LWR@traceinfo{\nameref A}%
2633 \LWR@startref{#1}%
2634 \LWR@traceinfo{\nameref B}%
2635 \LWR@nameref{#1}%
2636 \LWR@traceinfo{\nameref C}%
2637 \LWR@htmltag{/a}%
2638 \LWR@traceinfo{\nameref D}%
2639 }
```

## 49.5 Hyper-references

⚠ Note that the code currently only sanitizes the underscore character. Additional characters should be rendered inert as well. See the `hyperref.sty` definition of `\gdef\hyper@normalise` for an example.

Pkg **hyperref**

⚠ Do not tell other packages that `hyperref` is emulated. Some packages patch various commands if `hyperref` is present, which will probably break something, and the emulation already handles whatever may be emulated anyhow.

⚠ Any reference to `\usepackage{hyperref}` must be placed inside a `warpprint` environment.

```
2640 % DO NOT TELL OTHER PACKAGES TO ASSUME HYPERREF:
```

```
2641 % \EmulatesPackage{hyperref}[2015/08/01]% Disabled. Do not do this.
```

Create a link with a text name:

```
\LWR@subhyperref {\URL} {\text}
```

```
2642 \NewDocumentCommand{\LWR@subhyperref}{m +m}{%
```

```
2643 \LWR@htmltag{a href="#1" target="_{blank}"\LWR@orignewline}#2\LWR@htmltag{/a}%
```

```
2644 \LWR@ensuredoingapar%
```

```
2645 }
```

```
\LWR@subhyperrefclass {\URL} {\text} {\htmlclass}
```

```
2646 \NewDocumentCommand{\LWR@subhyperrefclass}{m +m m}{%
```

```
2647 \LWR@htmltag{a href="#1"
```

```
2648 class="#3"\LWR@orignewline}#2\LWR@htmltag{/a}%
```

```
2649 \LWR@ensuredoingapar%
```

```
2650 }
```

```
\href [{options}] {\URL} {\text}
```

Create a link with accompanying text:

```
2651 \NewDocumentCommand{\LWR@hrefb}{0{} m +m}{%
```

```
2652 \LWR@subhyperref{#2}{#3}%
```

```
2653 \endgroup%
```

```
2654 \LWR@ensuredoingapar%
```

```
2655 }
```

```
2656
```

```
2657 \newcommand{\href}{%
```

```
2658 \LWR@ensuredoingapar%
```

```

2659 \begingroup%
2660 \catcode'\_ =12
2661 \LWR@hrefb%
2662 }

```

`\nolinkurl`  $\{\langle URL \rangle\}$

Print the name of the link without creating the link:

```

2663 \newcommand*\LWR@nolinkurlb}[1]{#1\endgroup\LWR@ensuredoingapar%
2664
2665 \newcommand{\nolinkurl}{%
2666 \LWR@ensuredoingapar%
2667 \begingroup\catcode'\_ =12
2668 \LWR@nolinkurlb%
2669 }

```

`\url`  $\{\langle URL \rangle\}$

Create a link whose text name is the address of the link:

```

2670 \newcommand*\LWR@urlb}[1]{%
2671 \href{#1}{#1}%
2672 \endgroup%
2673 \LWR@ensuredoingapar%
2674 }
2675
2676 \newcommand{\url}{%
2677 \LWR@ensuredoingapar%
2678 \begingroup\catcode'\_ =12
2679 \LWR@urlb%
2680 }

```

`\LWR@subinlineimage`  $[\langle alttag \rangle] \{\langle class \rangle\} \{\langle filename \rangle\} \{\langle extension \rangle\} \{\langle style \rangle\}$

```

2681 \newcommand*\LWR@subinlineimage}[5][]{%
2682 \ifthenelse{\equal{#1}{}}{%
2683 {\LWR@htmltag{img src="#3.#4" alt="#3" style="#5" class="#2"{}}}%
2684 {\LWR@htmltag{img src="#3.#4" alt="#1" style="#5" class="#2"{}}}%
2685 }
2686 \end{warpHTML}

```

Table 8: Float data structures

---

For each `<type>` of float (figure, table, etc.) there exists the following:

---

**counter `<type>`:** A counter called `<type>`, such as `figure`, `table`.

**`\<type>name`:** Name. `\figurename` prints “Figure”, etc.

**`\ext@<type>`:** File extension. `\ext@figure` prints “lof”, etc.

**`\fps@<type>`:** Placement.

**`\the<type>`:** Number. `\thetable` prints the number of the table, etc.

**`\p@<type>`:** Parent’s number. Prints the number of the [within] figure, etc.

**`\fnum@<type>`:** Prints the figure number for the caption.

`\<type>name \the<type>`, “Figure 123”.

**`\<type>`:** Starts the float environment. `\figure` or `\begin{figure}`

**`\end<type>`:** Ends the float environment. `\endfigure` or `\end{figure}`

**`\tf@<ext>`:** The L<sup>A</sup>T<sub>E</sub>X file identifier for the output file.

**`LWR@have<type>`:** A boolean remembering whether a `\listof` was requested for a float of this type.

**File with extension `lo<f,t,a-z>`:** An output file containing the commands to build the `\listof<type><name>` “table-of-contents” structure.

**Cross-referencing names:** For `cleveref`’s `\cref` and related, `\crefname` and `\Crefname` assign human-readable names for references to this float type.

---

## 50 Floats

Floats are supported, although partially through emulation.

Table 8 shows the data structure associated with each `<type>` of float.

`\@makecaption` is redefined to print the float number and caption text, separated by `\CaptionSeparator`, which works with the `babel` package to adjust the caption separator according to the language. French, for example, uses an en-dash instead of a colon: “Figure 123 – Caption text”.

## 50.1 Float captions

for HTML output: 2687 \begin{warpHTML}

\LWR@floatbegin {<type>} [<placement>]

Begins a \newfloat environment.

```
2688 \NewDocumentCommand{\LWR@floatbegin}{m o}{%
2689 \LWR@stoppars
```

There is a new float, so increment the unique float counter:

```
2690 \addtocounter{LWR@thisfloat}{1}%
2691 \booltrue{LWR@freezethisfloat}%
```

```
2692 \begingroup
```

Settings while inside the environment:

```
2693 \LWR@origraggedright
```

Open an HTML figure tag:

```
2694 \LWR@htmltag{figure id="autofloat-\arabic{LWR@thisfloat}" class="#1"}
```

```
2695 \renewcommand*{\@capttype}{#1}
2696 \caption@settype{#1}
2697 \LWR@startpars
2698 }
```

\@float Support packages which create floats directly.  
 \@dblfloat

```
2699 \let\@float\LWR@floatbegin
2700 \let\@dblfloat\LWR@floatbegin
```

\LWR@floatend Ends a \newfloat environment.

```
2701 \newcommand*{\LWR@floatend}{%
2702 \LWR@stoppars%
2703 \LWR@htmllementend{figure}%
2704 \endgroup%
2705 \boolfalse{LWR@freezethisfloat}%
2706 \LWR@startpars%
2707 }
```

`\end@float` Support packages which create floats directly.  
`\end@dblfloat`

2708 `\let\end@float\LWR@floatend`  
 2709 `\let\end@dblfloat\LWR@floatend`

**Ctr** `LWR@thisfloat` A sequential counter for all floats. This is used to identify the float then reference it from the List of Figures and List of Tables.

2710 `\newcounter{LWR@thisfloat}`

**Bool** `LWR@freezethisfloat` Prevents multiple increments of `\LWR@thisfloat` inside a float.

2711 `\newbool{LWR@freezethisfloat}`  
 2712 `\boolfalse{LWR@freezethisfloat}`

`\LWR@maybeinthisfloat`

2713 `\newcommand*{\LWR@maybeinthisfloat}{%`  
 2714 `\ifbool{LWR@freezethisfloat}{\addtocounter{LWR@thisfloat}{1}}%`  
 2715 `}`

`\@capttype` Remembers which float type is in use.

2716 `\newcommand*{\@capttype}{}`

### 50.1.1 Caption inside a float environment

`\CaptionSeparator` How to separate the float number and the caption text.

2717 `\AtBeginDocument{\providecommand*\CaptionSeparator}{:~}`

`\@makecaption` `{\langle name and num \rangle}{\langle text \rangle}`

Prints the float type and number, the caption separator, and the caption text.

2718 `\AtBeginDocument{\renewcommand{\@makecaption}[2]{\#1\CaptionSeparator\#2}}`

### 50.1.2 Caption and LOF linking and tracking

When a new HTML file is marked in the  $\text{\LaTeX}$  PDF file, the  $\text{\LaTeX}$  page number at that point is stored in `LWR@latestautopage`, (and the associated filename is remembered by the special  $\text{\LaTeX}$  labels). This page number is used to generate an `autofloat` HTML ID in the HTML output at the start of the new HTML file.

Meanwhile, there is a float counter used to generate an HTML `autofloat` IDs at the start of the float itself in the HTML file. The `autopage` and `autofloat` values to use for each float are written to the `.lof`, etc. files just before each float's entry. These values are used by `\l@figure`, etc. to create the HTML links in the List of Figures, etc.

Ctr `LWR@nextautofloat` Tracks autofloat for floats. Tracks autopage for floats.

Ctr `LWR@nextautopage` These are updated per float as the `.lof` file is read.

```
2719 \newcounter{LWR@nextautofloat}
2720 \newcounter{LWR@nextautopage}
```

`\LWRsetnextfloat`  $\{\langle autopage \rangle\} \{\langle autofloat \rangle\}$

This is written to the `.lof` file just before each float's usual entry. The `autopage` and `autofloat` are remembered for `\l@figure` to use when creating the HTML links.

```
2721 \newcommand*{\LWRsetnextfloat}[2]{%
2722 \setcounter{LWR@nextautopage}{#1}%
2723 \setcounter{LWR@nextautofloat}{#2}%
2724 }
```

Ctr `LWR@latestautopage` Updated each time a new HTML file is begun. `\LWRsetnextfloat` is written with this and the `autofloat` by the modified `\addcontentsline` just before each float's entry.

```
2725 \newcounter{LWR@latestautopage}% updated each new HTML file
2726 \setcounter{LWR@latestautopage}{1}
```

```
2727 \let\LWR@origcaption@begin\caption@begin
2728 \let\LWR@origcaption@end\caption@end
2729 \let\LWR@orig@@par\@@par
```

`\LWR@caption@begin` Low-level patches to create HTML tags for captions.

```
2730 \newcommand{\LWR@caption@begin}
2731 {
2732 \LWR@traceinfo{LWR@caption@begin}%
```

Keep `par` and `minipage` changes local:

```
2733 \begingroup%
```

The `caption` code was not allowing the closing `par` tag:

```
2734 \renewcommand{\@@par}{\LWR@closeparagraph\LWR@orig@@par}%
```



No need for a minipage inside the caption:

```
2735 \RenewDocumentEnvironment{minipage}{0{t} o 0{t} m}{-}{-}%
```

Enclose the original caption code inside an HTML tag:

```
2736 \LWR@htmlblocktag{figcaption}%
2737 \LWR@origcaption@begin%
2738 }
```

`\LWR@caption@end` Low-level patches to create HTML tags for captions.

```
2739 \newcommand{\LWR@caption@end}
2740 {%
2741 \LWR@origcaption@end%
```

Subcaptions were being over-written by the closing HTML tag:

```
2742 \vspace*{\baselineskip}%
```

Closing tag:

```
2743 \LWR@htmlblocktag{/figcaption}%
2744 \endgroup%
2745 % \leavevmode% avoid bad space factor (0) error
2746 \LWR@traceinfo{\LWR@caption@end: done}%
2747 }
```

`\caption@begin` Low-level patches to create HTML tags for captions.

```
\caption@end
2748 \let\caption@begin\LWR@caption@begin
2749 \let\caption@end\LWR@caption@end
```

`\captionlistentry` Tracks the float number for this caption used outside a float. Patched to create an HTML anchor.

```
2750 \let\LWR@origcaptionlistentry\captionlistentry
2751
2752 \renewcommand*{\captionlistentry}{%
2753 \LWR@maybeinthisfloat%
2754 \LWR@ensuredoingapar%
2755 \LWR@htmltag{a id="autofloat-\arabic{\LWR@thisfloat}"}{\LWR@htmltag{/a}%
2756 \LWR@origcaptionlistentry%
2757 }
2758
2759 \def\LWR@LTcaptionlistentry{%
2760 \LWR@ensuredoingapar%
```

```

2761 \LWR@htmltag{a id="autofloat-\arabic{LWR@thisfloat}"{}}\LWR@htmltag{/a}%
2762 \bgroup
2763 \@ifstar{\egroup\LWR@LT@captionlistentry}% gobble *
2764 {\egroup\LWR@LT@captionlistentry}}%
2765 \def\LWR@LT@captionlistentry#1{%
2766 \caption@listentry\@firstoftwo[\LTcaptype]{#1}}%

```

`\addcontentsline` Patched to write the autopage and autofloat before each float's entry. No changes if writing `.toc`.

```

2767 \let\LWR@origaddcontentsline\addcontentsline
2768
2769 \renewcommand*\addcontentsline[3]{%
2770 \ifthenelse{\equal{#1}{toc}}{ }{%
2771 \addtocontents{\@nameuse{ext@#2}}{%
2772 \protect\LWRsetnextfloat%
2773 {\arabic{LWR@latestautopage}}%
2774 {\arabic{LWR@thisfloat}}%
2775 }%
2776 }%
2777 \LWR@origaddcontentsline{#1}{#2}{#3}%
2778 }

```

`\captionof` Patched to track the float number since this is used outside a float, and also create an HTML anchor for the virtual float.

```

2779 \AtBeginDocument{
2780 \let\LWR@origcaptionof\captionof
2781
2782 \renewcommand*\captionof{%
2783 \LWR@maybeinthisfloat%
2784 \LWR@stoppars
2785 \LWR@htmltag{a id="autofloat-\arabic{LWR@thisfloat}"{}}\LWR@htmltag{/a}%
2786 \LWR@origcaptionof%
2787 }
2788 }
2789 \end{warpHTML}

```

## 51 Table of Contents, LOF, LOT

This section controls the generation of the TOC, LOF, LOT.

The `.toc`, `.lof`, and `.lot` files are named by the source code `\jobname`.

In HTML, the printed tables are placed inside a div of class `.toc`, `.lof`, or `.lot`.

A “sidetoc” is provided which prints a subset of the TOC on the side of each page other than the homepage.

The regular L<sup>A</sup>T<sub>E</sub>X infrastructure is used for TOC, along with some patches to generate HTML output.

for HTML output: 2790 \begin{warpHTML}

## 51.1 Reading and printing the TOC

\LWR@myshorttoc {\<toc/lof/lot\}}

Reads in and prints the TOC/LOF/LOT at the current position. While doing so, makes the @ character into a normal letter to allow formatting commands in the section names.

Unlike in regular L<sup>A</sup>T<sub>E</sub>X, the file is not reset after being read, since the TOC may be referred to again in each HTML page, and is used for the sideTOC.

```
2791 \newcommand*\LWR@myshorttoc}[1]{
2792 \LWR@ensuredoingapar
```

Only if the file exists:

```
2793 \IfFileExists{\jobname.#1}{
```



Make @ a regular letter. Many of the commands in the file will have @ characters in them, so @ must be made a regular letter.



For pdf<sub>l</sub>at<sub>e</sub>x, also change to latin1 encoding. When reading back a file with accented characters, the encoding change seems to be required, rather than leaving it utf8.

```
2794 \begingroup
2795 % \ifxetexorluatex%
2796 % \else
2797 % \inputencoding{latin1}% currently disabled
2798 % \fi
2799 \makeatletter
```

Read in the TOC file:

```
2800 \@input{\jobname.#1}
2801 % \makeatother
2802 \endgroup
2803 }%
2804 {}%
```

2805 }

`\LWR@subtableofcontents`  $\{\langle toc/lof/lot \rangle\} \{\langle sectionstarname \rangle\}$

Places a TOC/LOF/LOT at the current position.

2806 `\NewDocumentCommand{\LWR@subtableofcontents}{m m}{%`

Closes previous levels:

2807 `\@ifundefined{chapter}`

2808 `{\LWR@closeprevious{\LWR@depthsection}}`

2809 `{\LWR@closeprevious{\LWR@depthchapter}}`

Prints any pending footnotes so that they appear above the potentially large TOC:

2810 `\LWR@printpendingnotes`

Place the list into its own chapter (if defined) or section:

2811 `\@ifundefined{chapter}{\section*{#2}}{\chapter*{#2}}`

Create a new HTML `nav` containing the TOC/LOF/LOT:

2812 `\LWR@htmlclass{nav}{#1}`

Create the actual list:

2813 `\LWR@myshorttoc{#1}`

Close the `nav`:

2814 `\LWR@htmlclassend{nav}{#1}`

2815 }

Patch `\@starttoc` to encapsulate the TOC inside HTML tags:

2816 `\let\LWR@orig@starttoc\@starttoc`

2817

2818 `\renewcommand{\@starttoc}[1]{`

2819 `\LWR@htmlclass{nav}{#1}`

2820 `\LWR@orig@starttoc{#1}`

2821 `\LWR@htmlclassend{nav}{#1}`

2822 }

Patch `\tableofcontents`, etc. to print footnotes first. `newfloat` uses `\listoffigures` for all future float types.

```

2823 \let\LWR@origtableofcontents\tableofcontents
2824 \let\LWR@origlistoffigures\listoffigures
2825 \let\LWR@origlistoftables\listoftables
2826 \renewcommand*{\tableofcontents}{%

```

Copy the .toc file to .sidetoc for printing the sidetoc. The original .toc file is renewed when \tableofcontents is finished.

```

2827 \LWR@copyfile{\jobname.toc}{\jobname.sidetoc}%
2828 \LWR@printpendingnotes%
2829 \LWR@origtableofcontents%
2830 }
2831 \renewcommand*{\listoffigures}{\LWR@printpendingnotes\LWR@origlistoffigures}
2832 \renewcommand*{\listoftables}{\LWR@printpendingnotes\LWR@origlistoftables}

```

## 51.2 High-level TOC commands

`\listof`  $\{\langle type \rangle\} \{\langle title \rangle\}$

Emulate the `\listof` command from the `float` package (section 81). Used to create lists of custom float types. Also used to redefine the standard L<sup>A</sup>T<sub>E</sub>X `\listoffigures` and `\listoftables` commands.

```

2833 \NewDocumentCommand{\listof}{m +m}{%
2834 \LWR@subtableofcontents{\@nameuse{ext@#1}}{#2}
2835 \expandafter\newwrite\csname tf@\csname ext@#1\endcsname\endcsname
2836 \immediate\openout \csname tf@\csname ext@#1\endcsname\endcsname
2837 \jobname.\csuse{ext@#1}\relax
2838 }

```

## 51.3 Side TOC

The “side TOC” is a table-of-contents positioned to the side.

It may be renamed by redefining `\sidetocname`, and may contain paragraphs.

CSS may be used to format the sideTOC:

CSS related to sideTOC:

---

**nav.sidetoc:** The entire sideTOC.

**div.sidetoctitle:** The title.

**div.sidetoccontents:** The table of contents.

---

```
2839 \end{warpHTML}
```

for HTML & PRINT: 2840 \begin{warpall}

Ctrl SideTOCDepth Controls how deep the side-TOC gets. Use a standard L<sup>A</sup>T<sub>E</sub>X section level similar to tocdepth.

```
2841 \newcounter{SideTOCDepth}
2842 \setcounter{SideTOCDepth}{1}
```

\sidetocname Holds the default name for the sidetoc.

```
2843 \newcommand{\sidetocname}{Contents}
```

```
2844 \end{warpall}
```

for HTML output: 2845 \begin{warpHTML}

\LWR@sidetoc Creates the actual side-TOC.

```
2846 \newcommand*{\LWR@sidetoc}{
2847 \LWR@stoppars
2848
```

The entire sideTOC is placed into a nav of class sidetoc.

```
2849 \LWR@htmlclass{nav}{sidetoc}
2850
2851 \setcounter{tocdepth}{\value{SideTOCDepth}}
2852
```

The title is placed into a div of class sidetoctitle, and may contain paragraphs.

```
2853 \begin{BlockClass}{sidetoctitle}
2854 \sidetocname
2855 \end{BlockClass}
```

The table of contents is placed into a div of class sidetoccontents.

```
2856 \begin{BlockClass}{sidetoccontents}
2857 \LinkHome
2858
2859 \LWR@myshorttoc{sidetoc}
2860 \end{BlockClass}
2861 \LWR@htmlclassend{nav}{sidetoc}
2862 }
```

## 51.4 Low-level TOC line formatting

`\numberline`  $\{\langle number \rangle\}$

(Called from each line in the `.aux`, `.lof` files.)

Record this section number for further use:

```
2863 \renewcommand*\numberline[1]{%
2864 \LWR@sectionnumber{#1}%
2865 }
```

`\hypertoc`  $\{\langle 1: depth \rangle\} \{\langle 2: class \rangle\} \{\langle 3: name \rangle\} \{\langle 4: page \rangle\}$

Called by `\l@section`, etc. to create a hyperlink to a section.

The autopage label is always created just after the section.

**#1** is depth

**#2** HTML CSS class is `tocsection`, `tocsubsection`, etc.

**#3** the text of the caption

**#4** page number

```
2866 \NewDocumentCommand{\hypertoc}{m m +m m}{%
```

Respond to `tocdepth`:

```
2867 \ifthenelse{\cnttest{#1}{<=}{\value{tocdepth}}}{%
2868 \LWR@startpars%
```

Create an HTML link to `filename#autosec-(page)`, with text of the caption, of the given HTML class.

```
2869 \LWR@subhyperrefclass{%
2870 \LWR@htmlrefsectionfilename{autopage-#4}\#autosec-#4}{#3}{#2}%
2871 \LWR@stoppars%
2872 }
2873 {}
2874 }
```

`\hypertocfloat`  $\{\langle 1: depth \rangle\} \{\langle 2: class \rangle\} \{\langle 3: caption \rangle\} \{\langle 4: page \rangle\}$

**#1** is depth

**#2** HTML CSS class is tocfigure, toctable

**#3** the text of the caption

**#4** page number

```
2875 \newcommand{\hypertocfloat}[4]{%
2876 \LWR@startpars
```

Create an HTML link to filename#autofloat-(float number), with text of the caption, of the given HTML class.

```
2877 \LWR@subhyperrefclass{%
2878 \LWR@htmlrefsectionfilename{autopage-\arabic{\LWR@nextautopage}}%
2879 \#autofloat-\arabic{\LWR@nextautofloat}}%
2880 {#3}{#2}%
2881 \LWR@stoppars%
2882 }
```

Automatically called by \contentsline:

```
2883 \renewcommand{\l@part}[2]{\hypertoc{-1}{tocpart}{#1}{#2}}
2884 \DeclareDocumentCommand{\l@chapter}{m m}
2885 {\hypertoc{0}{tocchapter}{#1}{#2}}
2886 \renewcommand{\l@section}[2]{\hypertoc{1}{tocsection}{#1}{#2}}
2887 \renewcommand{\l@subsection}[2]{\hypertoc{2}{tocsubsection}{#1}{#2}}
2888 \renewcommand{\l@subsubsection}[2]
2889 {\hypertoc{3}{tocsubsubsection}{#1}{#2}}
2890 \renewcommand{\l@paragraph}[2]{\hypertoc{4}{tocparagraph}{#1}{#2}}
2891 \renewcommand{\l@subparagraph}[2]{\hypertoc{5}{tocsubparagraph}{#1}{#2}}
2892 \renewcommand{\l@figure}[2]{\hypertocfloat{1}{tocfigure}{#1}{#2}}
2893 \renewcommand{\l@table}[2]{\hypertocfloat{1}{toctable}{#1}{#2}}
2894 \end{warpHTML}
```

## 52 Index

See:

[http://tex.stackexchange.com/questions/187038/  
how-to-mention-section-number-in-index-created-by-imakeidx](http://tex.stackexchange.com/questions/187038/how-to-mention-section-number-in-index-created-by-imakeidx)

**for HTML output:** 2895 \begin{warpHTML}

**\LWR@indexsection** Controls whether the index will be in a section or a chapter, depending on the documentclass.



```

2896 \@ifundefined{chapter}
2897 {\newcommand*{\LWR@indexsection}{\section{\indexname}}}
2898 {\newcommand*{\LWR@indexsection}{\chapter{\indexname}}}

```

`\printindex`

```

2899 \let\LWR@origprintindex\printindex
2900
2901 \renewcommand*{\printindex}
2902 {
2903   \LWR@indexsection
2904   \LWR@startpars
2905   \LWR@origprintindex
2906 }

```

Env `theindex`

```

2907 \renewenvironment*{theindex}{%
2908 \let\item\LWR@indexitem%
2909 \let\subitem\LWR@indexsubitem%
2910 \let\subsubitem\LWR@indexsubsubitem%
2911 }{}

```

`\LWR@indexitem`

```

2912 \newcommand{\LWR@indexitem}{
2913
2914 \InlineClass{indexitem}{}
2915 }

```

`\LWR@indexitem`

```

2916 \newcommand{\LWR@indexsubitem}{
2917
2918 \InlineClass{indexsubitem}{}
2919 }

```

`\LWR@indexitem`

```

2920 \newcommand{\LWR@indexsubsubitem}{
2921
2922 \InlineClass{indexsubsubitem}{}
2923 }

```

`\hyperindexref`  $\{(\textit{autosecnumber})\}$

`\hyperindexref{web address}` is inserted into `*.ind` by the xindy style file `lwarp_html.xdy`

```
2924 \newcommand*\hyperindexref}[1]{
2925 \LWR@htmltag{a href="%
```

Create an HTML reference to the autosection:

```
2926 \LWR@htmlrefsectionfilename{autopage-#1}\#autosec-#1"{}%}
```

Add the section number and section name, then close the tag:

```
2927 \LWR@origref{autopage-#1}\enskip\LWR@nameref{autopage-#1}%
2928 \LWR@htmltag{/a}%
2929 }
```

```
2930 \end{warpHTML}
```

**for PRINT output:** A null command for print mode, in case `hyperref` was not used:

```
2931 \begin{warpprint}
2932 \newcommand*\hyperindexref}[1]{#1}
2933 \end{warpprint}
```

## 53 Math

Math may be rendered as SVG graphics or using the MATHJAX JavaScript display engine.

**SVG math option** For SVG math, math is rendered as usual by  $\text{\LaTeX}$  into the initial PDF file using the current font<sup>8</sup>, then is captured from the PDF and converted to SVG graphics via a number of utility programs. The SVG format is a scalable-vector web format, so math may be typeset by  $\text{\LaTeX}$  with its fine control and precision, then displayed or printed at any size, depending on (sometimes broken) browser support. An HTML ALT tag carries the  $\text{\LaTeX}$  code which generated the math, allowing copy/paste of the  $\text{\LaTeX}$  math expression into other documents.

**SVG image font size** The size of the math and text used in the SVG image may be adjusted by setting `\LateximageFontSizeName` to a font size name — *without the backslash*, for ex:

```
\renewcommand{\LateximageFontSizeName}{large}
```

**SVG files** As currently implemented, each instance of math creates a new SVG file. In text with many references to math variables, this can result in a large number of files

<sup>8</sup>See section 114 regarding fonts and fractions.

with duplicate content. In the future, some method of content-based naming and checksumming may be used to remove the need for duplicate files.

**SVG inline** Another approach would be to in-line the SVG files directly into the HTML. This avoids having a large number of files and potentially speeds loading the images, but dis-allows the possibility of sharing one file among many instances without user intervention.

**PNG files** Others have used PNG files, sometimes pre-scaled for print resolution but displayed on-screen at a scaled down size. This allows high-quality print output at the expense of larger files, but SVG files are also larger as well.

**MathML** Conversion to MathML might be a better approach, among other things allowing a more compact representation of math than SVG drawings. Problems with MathML include limited browser support and some issues with the fine control of the appearance of the result.

**MathJax math option** The popular MathJax alternative ([mathjax.org](http://mathjax.org)) may be used to display math.

Prog **MathJax**

When MathJax is enabled, math is rendered twice:

1. As regular  $\text{\LaTeX}$  PDF output placed inside an HTML comment, allowing equation numbering and cross referencing to be almost entirely under the control of  $\text{\LaTeX}$ , and
2. As detokenized printed  $\text{\LaTeX}$  commands placed directly into the HTML output for interpretation by the MathJax display scripts. An additional script is used to pre-set the equation number format and value according to the current  $\text{\LaTeX}$  values, and the MathJax cross-referencing system is ignored in favor of the  $\text{\LaTeX}$  internal system, seamlessly integrating with the rest of the  $\text{\LaTeX}$  code.

**MathJax limitations** Limitations when using MathJax include:

Prog **MathJax**

**chapter numbers**

- In document classes which have chapters,  $\text{\tagged}$  equations have the chapter number prepended in HTML output, unlike  $\text{\LaTeX}$ .  $\text{\tag*}$  equations (correctly) do not. This may be improved with future versions of the MathJax support script.

<https://groups.google.com/forum/#!topic/mathjax-users/jUtewUcE2bY>

**subequations**

- MathJax itself does not support subequations. This may be improved by parsing the  $\text{\LaTeX}$  math expression to manually insert tags, but this has not yet been done.

**footnotes in math**

- Footnotes inside equations are not yet supported while using MathJax.

- `lateximage`
  - Math appearing inside a `lateximage`, and therefore also inside a `Tikz` or `picture` environment, is rendered as SVG math even if MathJax is used in the rest of the document.
- `siunitx`
  - Usage of `siunitx` inside a math equation is supported via a third-party MathJax extension. While inside a math expression, do not use `\SI` or `\si` inside `\text`, where it will be rendered as normal text.  
<https://github.com/mathjax/MathJax-third-party-extensions/tree/master/siunitx>
- $\text{\LaTeX}$  macros
  - MathJax does not automatically support custom  $\text{\LaTeX}$  macros, but they may be set up by the user.
- custom MathJax macros
 

As an example of using custom  $\text{\LaTeX}$  macros with MathJax, place the following at the start of the document, after `\begin{document}`:

---

```
\begin{warpHTML} % Only for HTML output,
\ifbool{mathjax} % and only for MathJax output:
{
  \(\ % New macros for MathJax are placed inside a math expression:
  \newcommand{\expval}[1]{\langle#1\rangle}
  \newcommand{\abs}[1]{\lvert#1\rvert}
  \)
}{}
\end{warpHTML}
```

---

for HTML output: 2934 `\begin{warpHTML}`

`\$` Plain dollar signs appearing in the HTML output may be interpreted by MathJax to be math shifts. For a plain text dollar `\$`, print it inside a `span` to avoid it being interpreted by MathJax.

```
2935 \let\LWR@origtextdollar\$
2936
2937 \renewcommand*{\$}{\%
2938 \LWR@htmltagc{span}%
2939 \LWR@origtextdollar%
2940 \LWR@htmltagc{/span}%
2941 }
```

Ctr `LWR@externalfilecnt` Counter for the external files which are generated and then referenced from the HTML:

```
2942 \newcounter{LWR@externalfilecnt}
```

### 53.1 Inline and display math

```

2943 \let\LWR@origdollar=$
2944 \let\secondorigdollar=$% balance for editor syntax highlighting

2945 \let\LWR@origopenparen\l
2946 \let\LWR@origcloseparen\r

```

**\$** Redefine the dollar sign to place math inside a `lateximage`, or use MathJax:  
**\$\$**

```

2947 \begingroup
2948 \catcode'\$=\active%
2949 \protected\gdef$ {\ifnextchar$\LWR@doubledollar\LWR@singledollar}%

```

**\LWR@doubledollar** Redefine the double dollar sign to place math inside a `lateximage`, or use MathJax:

```

2950 \gdef\LWR@doubledollar$#1$$ {
2951 \ifbool{mathjax}%

```

For MathJax, print the math between `\[` and `\]`:

```

2952 {\textbackslash[\LWR@HTMLsanitize{#1}\textbackslash]}

```

For SVG, print the math inside a `lateximage`, with an ALT tag of the  $\LaTeX$  code:

```

2953 {% not mathjax
2954
2955 \begin{lateximage}%
2956 [\textbackslash{[] \LWR@HTMLsanitize{#1} \textbackslash{[]}]%
2957 \LWR@origdollar\LWR@origdollar#1\LWR@origdollar\LWR@origdollar%
2958 \end{lateximage}
2959
2960 }
2961 }%

```

**\LWR@singledollar** Redefine the single dollar sign to place math inside a `lateximage`, or use MathJax:

```

2962 \gdef\LWR@singledollar#1$ {%
2963 \ifbool{mathjax}%

```

For MathJax, print the math between `\(` and `\)`:

```

2964 {\textbackslash(\LWR@HTMLsanitize{#1}\textbackslash)}

```

For SVG, print the math inside a `lateximage`, with an ALT tag of the  $\LaTeX$  code:

```

2965 {% not mathjax
2966 \begin{lateximage}%
2967 [\textbackslash( \LWR@HTMLsanitize{#1} \textbackslash)]%

```

```

2968 \LWR@origdollar#1\LWR@origdollar%
2969 \end{lateximage}%
2970 }%
2971 }%

```

\( Redefine to the above dollar macros.  
 \)

```

2972 \gdef\(#1\){$#1$}
2973 \gdef\[#1\]{$$#1$$}
2974
2975 \endgroup

```

Remove the old `math` and `displaymath` environments:

```

2976 \let\math\relax
2977 \let\endmath\relax
2978 \let\displaymath\relax
2979 \let\enddisplaymath\relax

```

Env `math` Set math mode then typeset the body of what was between the begin/end. See the `environ` package for `\BODY`.

```

2980 \NewEnviron{math}{\expandafter\(\BODY\)}

```

Env `displaymath` Set math mode then typeset the body of what was between the begin/end. See the `environ` package for `\BODY`.

```

2981 \NewEnviron{displaymath}{\expandafter\[\BODY\]\@ignoretrue}

```

When the document begins, the dollar sign must be made active to trigger the new math macros:

```

2982 \AtBeginDocument{\catcode'\$=\active}

```

## 53.2 MathJax support

Ctrl `LWR@nextequation` Used to add one to compute the next equation number.

```

2983 \newcounter{LWR@nextequation}

```

`\LWR@syncmathjax` Sets the MathJax equation format and number for the following equations.

These MathJax commands are printed inside “\( $\)$ ” and “\( $\)$ ” characters. They are printed to HTML output, not interpreted by  $\text{\LaTeX}$ .

```
2984 \newcommand*{\LWR@syncmathjax}{%
```

If using chapters, place the chapter number in front of the equation. Otherwise, use the simple equation number.

```
2985 \ifcsdef{thechapter}{
2986 \BlockClassSingle{hidden}{
2987 \textbackslash(
2988 \textbackslash{}seteqsection \{\thechapter\}
2989 \textbackslash)
2990 }
2991 }
2992 {}% not using chapters
2993
```

MathJax doesn't allow setting the equation number to 1:

```
2994 \ifthenelse{\cnttest{\value{equation}}>0}
2995 {
```

Tell MathJax that the next set of equations begins with the current L<sup>A</sup>T<sub>E</sub>X equation number, plus one.

```
2996 \setcounter{\LWR@nextequation}{\value{equation}}
2997 \addtocounter{\LWR@nextequation}{1}
```

Place the MathJax command inside “\ (“ and “\)” characters, to be printed to HTML, not interpreted by L<sup>A</sup>T<sub>E</sub>X.

```
2998 \BlockClassSingle{hidden}{
2999 \textbackslash(
3000 \textbackslash{}seteqnumber \{\arabic{\LWR@nextequation}\}
3001 \textbackslash)
3002 }
3003 }{}% not eq > 1
3004 }
```

**\LWR@restoremathlatexformatting** While producing math, use regular L<sup>A</sup>T<sub>E</sub>X formatting instead of HTML tags.

```
3005 \newcommand*{\LWR@restoremathlatexformatting}{%
3006 \let\hspace\LWR@origspace%
3007 \let\rule\LWR@origrule%
3008 \let\,\LWR@origcomma% disable HTML short unbreakable space
3009 \let\textit\LWR@origtextit%
3010 \let\textbf\LWR@origtextbf%
3011 \let\texttt\LWR@origtexttt%
3012 \let\textsc\LWR@origtextsc%
3013 \let\textsf\LWR@origtextsf%
```

```

3014 \let\textrm\LWR@origtextrm%
3015 \renewcommand*{\thefootnote}{\fnsymbol{footnote}}%
3016 \let\textsuperscript\LWR@origtextsuperscript%
3017 \let\textsubscript\LWR@origtextsubscript%
3018 \let~\LWR@origtilde%
3019 \let\enskip\LWR@origenskip%
3020 \let\quad\LWR@origquad%
3021 \let\qquad\LWR@origqquad%
3022 }

```

`\LWR@hidelatexequation`  $\{\langle environment \rangle\}$   $\{\langle contents \rangle\}$

Creates the L<sup>A</sup>T<sub>E</sub>X version of the equation inside an HTML comment.

```

3023 \NewDocumentCommand{\LWR@hidelatexequation}{m +m}{%

```

Stop HTML paragraph handling and open an HTML comment:

```

3024 \LWR@stoppars
3025 \LWR@htmlopencomment
3026

```

Start the LaTeX math environment inside the HTML comment:

```

3027 \begingroup
3028 \csuse{\LWR@orig#1}

```

While in the math environment, restore various commands to their L<sup>A</sup>T<sub>E</sub>X meanings.

```

3029 \LWR@restoremathlatexformatting

```

When a label is encountered, the HTML comment is closed, the label is generated in an HTML-output context, then the HTML comment is opened again. See `\LWR@htmlmathlabel` in section 53.4.1.

Print the contents of the equation:

```

3030 #2

```

End the L<sup>A</sup>T<sub>E</sub>X math environment inside the HTML comment:

```

3031 \csuse{\LWR@origend#1}
3032 \endgroup
3033

```

Close the HTML comment and resume HTML paragraph handling:

```

3034 \LWR@htmlclosecomment

```



```
3035 \LWR@startpars
3036 }
```

```
\LWR@addmathjax {<environment>} {<contents>}
```

Given the name of a math environment and its contents, create a MathJax instance. The contents are printed to HTML output, not interpreted by L<sup>A</sup>T<sub>E</sub>X.

```
3037 \NewDocumentCommand{\LWR@addmathjax}{m +m}{%
3038
```

Enclose the MathJax environment inside printed “\ (“ and “\)” characters.

```
3039 \textbackslash{}begin\{#1\}
```

Print the contents, sanitizing for HTML special characters.

```
3040 \LWR@HTMLsanitizeexpand{\detokenize\expandafter{#2}}
```

Close the MathJax environment:

```
3041 \textbackslash{}end\{#1\}
3042
3043 }
```

### 53.3 Equation environment

Remember existing `equation` environment:

```
3044 \let\LWR@origequation\equation
3045 \let\LWR@origendequation\endequation
```

Remove existing `equation` environment:

```
3046 \let\equation\relax
3047 \let\endequation\relax
```

**Env**   **equation**   The new `equation` environment is created with `\NewEnviron` (from the `environ` package), which stores the contents of its environment in a macro called `\BODY`.

For SVG math output, the contents are typeset using the original `equation` inside a `lateximage`, along with an ALT tag containing a detokenized copy of the L<sup>A</sup>T<sub>E</sub>X source for the math.

For MathJax output, the contents are typeset in an original `equation` environment placed inside a HTML comment, with special processing for `\labels`. The contents are also printed to the HTML output for processing by the MathJax script.

```
3048 \NewEnviron{equation}{%
3049
3050 \ifbool{mathjax}
```

MathJax output:

```
3051 {
```

Print commands to synchronize MathJax's equation number and format to the current  $\LaTeX$  chapter/section and equation number:

```
3052 \LWR@syncmathjax
```

Print the  $\LaTeX$  math inside an HTML comment:

```
3053 \LWR@hidelatexequation{equation}{\BODY}
3054 }
```

SVG output: Create the `lateximage` along with an HTML ALT tag having an equation number, the  $\LaTeX$  equation environment commands, and the contents of the environment's `\BODY`.

```
3055 {% not mathjax
```

Begin the `lateximage` with an ALT tag containing the math source:

```
3056 \begin{lateximage}[(\theequation) \textbackslash{begin\{equation\}}}%
3057 \LWR@HTMLsanitizeexpand{\detokenize\expandafter{\BODY}}}%
3058 \textbackslash{end\{equation\}}}% alt tag
```

Create the actual  $\LaTeX$ -formatted equation inside the `lateximage` using the contents of the environment.

```
3059 \LWR@origequation
3060 \BODY% contents collected by NewEnviron
3061 \LWR@origendequation
3062 \end{lateximage}%
3063 }
3064
```

After the environment, if MathJax, print the math to the HTML output for MathJax processing:

```
3065 }[\ifbool{mathjax}{\LWR@addmathjax{equation}{\BODY}}{}}
```

## 53.4 AMS Math environments

### 53.4.1 Support macros

Bool `LWR@amsmultline` True if processing a multiline environment.

To compensate for `multline`-specific code, `LWR@amsmultline` is used to add extra horizontal space in `\LWR@htmlmathlabel` if is used in an `amsmath` environment which is not a `multline` environment and not an `equation`.

```
3066 \newbool{LWR@amsmultline}
3067 \boolfalse{LWR@amsmultline}
```

`\LWR@htmlmathlabel`  $\{ \langle label \rangle \}$

`\label` when called inside a  $\text{\LaTeX}$  AMS math environment's math mode which is encased inside an HTML comment while creating a HTML document.

```
3068
3069 \newcommand*{\LWR@htmlmathlabel}[1]{%
3070 \LWR@traceinfo{\LWR@htmlmathlabel #1}%
3071 \ifbool{mathjax}{%
```

The combined  $\text{\LaTeX}$  & HTML label is printed in a `\text` field:

```
3072 \text{
```

Shift the label over to the right side of the environment to avoid over-printing the math:

```
3073 \ifbool{LWR@amsmultline}{\hspace*{\totwidth0}}
```

Temporarily end the HTML comment, insert the  $\text{\LaTeX}$  & HTML label, then resume the HTML comment. `\@firstofone` is required to remove extra braces introduced by the `amsmath` package.)

```
3074 \LWR@htmlclosecomment\LWR@sublabel{\@firstofone#1}\LWR@htmlopencomment}%
3075 }%
3076 {\LWR@sublabel{\@firstofone#1}}%
3077 }
```

`\LWR@beginhideamsmath` Starts hiding  $\text{\LaTeX}$  math inside an HTML comment.

```
3078 \newcommand*{\LWR@beginhideamsmath}{
3079 \LWR@stoppars
3080
3081 \LWR@htmlopencomment
```

```

3082
3083 \begingroup
3084 \LWR@restoremathlatexformatting
3085 }

```

`\LWR@endhideamsmath` Ends hiding  $\text{\LaTeX}$  math inside an HTML comment.

```

3086 \newcommand*{\LWR@endhideamsmath}{
3087 \endgroup
3088
3089 \LWR@htmlclosecomment
3090
3091 \LWR@startpars
3092 }

```

### 53.4.2 Environment patches

The following `amsmath` environments already collect their contents in `\@envbody` for further processing.

For SVG math: Each environment is encapsulated inside a `lateximage` environment, along with a special `LWRAMSMATHBODY` argument telling `lateximage` to use as the HTML ALT tag the environment's contents which were automatically captured by the  $\mathscr{A}\mathscr{M}\mathscr{S}$  environment.

For MathJax: Each environment is syched with  $\text{\LaTeX}$ 's equation numbers, typeset with  $\text{\LaTeX}$  inside an HTML comment, then printed to HTML output for MathJax to process.

Env `multline`

```

3093 \BeforeBeginEnvironment{multline}{
3094 \ifbool{mathjax}
3095 {
3096 \LWR@syncmathjax
3097 \booltrue{LWR@amsmultline}
3098 \LWR@beginhideamsmath
3099 }
3100 {
3101 \lateximage[LWRAMSMATHBODY]
3102 }
3103 }
3104
3105 \AfterEndEnvironment{multline}{
3106
3107 \ifbool{mathjax}
3108 {

```

```

3109 \LWR@endhideamsmath
3110 \boolfalse{LWR@amsmultline}
3111 \LWR@addmathjax{multline}{\the\@envbody}
3112 }
3113 {\endlateximage}
3114
3115 }

```

Env **multline\***

```

3116 \BeforeBeginEnvironment{multline*}{
3117 \ifbool{mathjax}
3118 {
3119 \LWR@syncmathjax
3120 \booltrue{LWR@amsmultline}
3121 \LWR@beginhideamsmath
3122 }
3123 {
3124 \lateximage[LWRAMSMATHBODY]
3125 }
3126 }
3127
3128 \AfterEndEnvironment{multline*}{
3129
3130 \ifbool{mathjax}
3131 {
3132 \LWR@endhideamsmath
3133 \boolfalse{LWR@amsmultline}
3134 \LWR@addmathjax{multline*}{\the\@envbody}
3135 }
3136 {\endlateximage}
3137
3138 }
3139

```

Env **gather**

```

3140 \BeforeBeginEnvironment{gather}{
3141 \ifbool{mathjax}
3142 {
3143 \LWR@syncmathjax
3144 \boolfalse{LWR@amsmultline}
3145 \LWR@beginhideamsmath
3146 }
3147 {
3148 \lateximage[LWRAMSMATHBODY]
3149 }
3150 }
3151

```

```

3152 \AfterEndEnvironment{gather}{
3153
3154 \ifbool{mathjax}
3155 {
3156 \LWR@endhideamsmath
3157 \LWR@addmathjax{gather}{\the\@envbody}
3158 }
3159 {\endlateximage}
3160
3161 }

```

Env **gather\***

```

3162 \BeforeBeginEnvironment{gather*}{
3163 \ifbool{mathjax}
3164 {
3165 \LWR@syncmathjax
3166 \boolfalse{LWR@amsmultline}
3167 \LWR@beginhideamsmath
3168 }
3169 {
3170 \lateximage[LWRAMSMATHBODY]
3171 }
3172 }
3173
3174 \AfterEndEnvironment{gather*}{
3175
3176 \ifbool{mathjax}
3177 {
3178 \LWR@endhideamsmath
3179 \LWR@addmathjax{gather*}{\the\@envbody}
3180 }
3181 {\endlateximage}
3182
3183 }

```

Env **align**

```

3184 \BeforeBeginEnvironment{align}{
3185 \ifbool{mathjax}
3186 {
3187 \LWR@syncmathjax
3188 \boolfalse{LWR@amsmultline}
3189 \LWR@beginhideamsmath
3190 }
3191 {
3192 \lateximage[LWRAMSMATHBODY]
3193 }
3194 }

```

```

3195
3196 \AfterEndEnvironment{align}{
3197
3198 \ifbool{mathjax}
3199 {
3200 \LWR@endhideamsmath
3201 \LWR@addmathjax{align}{\the\@envbody}
3202 }
3203 {\endlateximage}
3204
3205 }

```

Env **align\***

```

3206 \BeforeBeginEnvironment{align*}{
3207 \ifbool{mathjax}
3208 {
3209 \LWR@syncmathjax
3210 \boolfalse{LWR@amsmultiline}
3211 \LWR@beginhideamsmath
3212 }
3213 {
3214 \lateximage[LWRAMSMATHBODY]
3215 }
3216 }
3217
3218 \AfterEndEnvironment{align*}{
3219
3220 \ifbool{mathjax}
3221 {
3222 \LWR@endhideamsmath
3223 \LWR@addmathjax{align*}{\the\@envbody}
3224 }
3225 {\endlateximage}
3226
3227 }

```

Env **flalign**

```

3228 \BeforeBeginEnvironment{flalign}{
3229 \ifbool{mathjax}
3230 {
3231 \LWR@syncmathjax
3232 \boolfalse{LWR@amsmultiline}
3233 \LWR@beginhideamsmath
3234 }
3235 {
3236 \lateximage[LWRAMSMATHBODY]
3237 }

```

```

3238 }
3239
3240 \AfterEndEnvironment{flalign}{
3241
3242 \ifbool{mathjax}
3243 {
3244 \LWR@endhideamsmath
3245 \LWR@addmathjax{flalign}{\the\@envbody}
3246 }
3247 {\endlateximage}
3248
3249 }

```

Env **flalign\***

```

3250 \BeforeBeginEnvironment{flalign*}{
3251 \ifbool{mathjax}
3252 {
3253 \LWR@syncmathjax
3254 \boolfalse{LWR@amsmultline}
3255 \LWR@beginhideamsmath
3256 }
3257 {
3258 \lateximage[LWRAMSMATHBODY]
3259 }
3260 }
3261
3262 \AfterEndEnvironment{flalign*}{
3263
3264 \ifbool{mathjax}
3265 {
3266 \LWR@endhideamsmath
3267 \LWR@addmathjax{flalign*}{\the\@envbody}
3268 }
3269 {\endlateximage}
3270
3271 }

3272 \end{warpHTML}

```

## 54 Lateximages

A `\lateximage` is typeset on its own PDF page inside an HTML comment starting on the preceeding page and ending on following page, and instructions are written to `lateximage.sh` or `lateximages.cmd` to extract the `\lateximage` from the page



of the PDF file then generate an accompanying .svg file image file. Meanwhile, instructions to show this image are placed into the HTML file after the comment.

Since the `\lateximage` is placed onto its own PDF page, which `pdftotext` will read and try to convert to text, an HTML open comment is placed onto the bottom of the current page, but it must appear below any footnotes on the current page, so a special page style (`LWR@commentfooterstyle`) is used.

An HTML span is created to hold both the HTML comment which will have the `pdftotext` conversion, and also the link to the final .svg image.

**SVG image font size** The size of the math and text used in the SVG image may be adjusted by setting `\LateximageFontSizeName` to a font size name — *without the backslash*, for ex:  
`\renewcommand{\LateximageFontSizeName}{large}`

**for HTML output:** 3273 `\begin{warpHTML}`

`LWR@commentfooterstyle`

```
3274 \newpagestyle{LWR@commentfooterstyle}{
3275 \sethead{}{}{}
3276 \setfoot{\LWR@htmlopencomment}{}{}
3277 }
```

**Ctrl LWR@lateximagenumber** Sequence the images.

```
3278 \newcounter{LWR@lateximagenumber}
3279 \setcounter{LWR@lateximagenumber}{0}
```

**Ctrl LWR@lateximagedepth** Do not create `\lateximage` inside of `\lateximage`.

```
3280 \newcounter{LWR@lateximagedepth}
3281 \setcounter{LWR@lateximagedepth}{0}
```

Declare the `\LWR@file` for writing to generate file `lateximages.sh` or `lateximages.cmd`:

```
3282 \ifcsdef{\LWR@file}{}{\newwrite{\LWR@file}}
```

A few utility macros to write special characters:

```
3283 \edef\LWR@hashmark{\string#} % for use in \write
3284 \edef\LWR@percent{\@percentchar} % for use in \write
```

**Ctrl LWR@nextpage** Used to count ahead to the next page:

```
3285 \newcounter{LWR@nextpage}

3286 \end{warpHTML}
```

for HTML & PRINT: 3287 \begin{warpall}

`\LateximageFontSizeName` Declares how large to write text in the `\lateximage`. The `.svg` file text size should blend well with the surrounding HTML text size.

*Do not include the leading backslash in the name.*

```
3288 \newcommand*{\LateximageFontSizeName}{normalsize}
```

```
3289 \end{warpall}
```

for HTML output: 3290 \begin{warpHTML}

`\LWR@HTMLsanitize`  $\{ \langle text \rangle \}$

Math expressions are converted to `lateximages`, and some math environments may contain “&”, “<”, or “>”, which should not be allowed inside an HTML ALT tag, so must convert them to HTML entities.

Two versions follow, depending on expansion needs. There may be a better way...

```
3291 \newcommand{\LWR@HTMLsanitize}[1]{%
3292 \protect\StrSubstitute{\detokenize{#1}}{%
3293 {\detokenize{&}}{%
3294 {\detokenize{&}}{\LWR@strresult}}%
3295 \protect\StrSubstitute{\detokenize\expandafter{\LWR@strresult}}{%
3296 {\detokenize{<}}{%
3297 {\detokenize{<}}{\LWR@strresult}}%
3298 [\LWR@strresult]}%
3299 \protect\StrSubstitute{\detokenize\expandafter{\LWR@strresult}}{%
3300 {\detokenize{>}}{%
3301 {\detokenize{>}}{\LWR@strresult}}%
3302 [\LWR@strresult]}%
3303 \protect\StrSubstitute{\detokenize\expandafter{\LWR@strresult}}{%
3304 {\detokenize{##}}{%
3305 {\detokenize{##}}{\LWR@strresult}}%
3306 [\LWR@strresult]}%
3307 \LWR@strresult%
3308 }
```

`\LWR@HTMLsanitizeexpand`  $\{ \langle text \rangle \}$

This version expands the argument before sanitizing it.

```
3309 \newcommand{\LWR@HTMLsanitizeexpand}[1]{%
3310 \protect\StrSubstitute{\detokenize\expandafter{#1}}{%
3311 {\detokenize{&}}{%
```

```

3312 {\detokenize{&}}}%
3313 [\LWR@strresult]%
3314 \protect\StrSubstitute{\detokenize\expandafter{\LWR@strresult}}}%
3315 {\detokenize{<}}}%
3316 {\detokenize{&lt;}}}%
3317 [\LWR@strresult]%
3318 \protect\StrSubstitute{\detokenize\expandafter{\LWR@strresult}}}%
3319 {\detokenize{>}}}%
3320 {\detokenize{&gt;}}}%
3321 [\LWR@strresult]%
3322 \LWR@strresult%
3323 }

```

Env `lateximage` [*alttag*]

```

3324 \NewDocumentEnvironment{lateximage}{0{image}}{%
3325 \ifthenelse{\cnttest{\value{LWR@lateximagedepth}}{>}{0}}%

```

If nesting inside an already-existing `lateximage`, simply record one more level:

```

3326 {%
3327 \addtocounter{LWR@lateximagedepth}{1}%
3328 }%

```

Otherwise, this is the outer-most `lateximage`:

```

3329 {% start of outer-most lateximage

```

Starting a new `lateximage`:

```

3330 \addtocounter{LWR@lateximagenumber}{1}%

```

While inside a `lateximage`, do not use `mathjax`:

```

3331 \boolfalse{mathjax}

```

Be sure that are doing a paragraph:

```

3332 \LWR@ensuredoingapar%

```

Next file:

```

3333 \addtocounter{LWR@externalfilecnt}{1}%

```

Figure out what the next page number will be:

```

3334 \setcounter{LWR@nextpage}{\thepage}%
3335 \addtocounter{LWR@nextpage}{1}%

```

Create an HTML span which will hold the comment which contains the `pdftotext` translation of the image's page, and also will hold the link to the `.svg` file:

```
3336 \LWR@htmltag{span id="lateximage\arabic{LWR@lateximagenumber}" %
3337 class="lateximagesource"{} } \LWR@orignewline
```

Write instructions to the `lateximage.sh` file:

```
3338 \immediate\write\LWR@file{%
3339 pdfseparate -f \theLWR@nextpage \space -l %
3340 \theLWR@nextpage \space %
3341 \jobname.pdf lateximagetemp-\LWR@percent d.pdf}%
3342 % keep the space before 'd.pdf'
3343 \immediate\write\LWR@file{%
3344 pdfcrop --hires lateximagetemp-\theLWR@nextpage.pdf
3345 lateximage-\theLWR@externalfilecnt.pdf}%
3346 \immediate\write\LWR@file{%
3347 pdftocairo -svg lateximage-\theLWR@externalfilecnt.pdf
3348 lateximage-\theLWR@externalfilecnt.svg}%
3349 \immediate\write\LWR@file{%
3350 \OSmv lateximage-\theLWR@externalfilecnt.svg %
3351 lateximages\OSPathSymbol}%
3352 \immediate\write\LWR@file{%
3353 \OSrm lateximage-\theLWR@externalfilecnt.pdf
3354 lateximagetemp-\theLWR@nextpage.pdf}%

```

Place an open comment tag at the bottom of page; footnotes will be above this tag. This will hide any traces of the `lateximage` PDF page which were picked up by `pdftotext`.

```
3355 \LWR@origthispagestyle{LWR@commentfooterstyle}%
3356 \addtocounter{LWR@lateximagedepth}{1}%

```

Start the new PDF page:

```
3357 \LWR@orignewpage%
```

Typeset the image in a “standard” width page and font size:

```
3358 \LWR@origminipage{6in}%
3359 \csuse{LWR@orig\LateximageFontSizeName}%

```

Temporarily restore formatting to its PDF definitions: Do not produce HTML tags for `\hspace`, etc. inside a `lateximage`.

```
3360 \let\hspace\LWR@orighspace%
3361 \let\rule\LWR@origrule%
3362 \let\,\LWR@origcomma% disable HTML short unbreakable space
3363 \let\textit\LWR@origtextit%

```

```

3364 \let\textbf\LWR@origtextbf%
3365 \let\texttt\LWR@origtexttt%
3366 \let\textsc\LWR@origtextsc%
3367 \let\textsf\LWR@origtextsf%
3368 \let\textrm\LWR@origtextrm%
3369 \renewcommand*{\thefootnote}{\fnsymbol{footnote}}}%
3370 \let\textsuperscript\LWR@origtextsuperscript%
3371 \let\textsubscript\LWR@origtextsubscript%
3372 \let~\LWR@origtilde%
3373 \let\enskip\LWR@origenskip%
3374 \let\quad\LWR@origquad%
3375 \let\qqquad\LWR@origqqquad%
3376 \let\tabular\LWR@origtabular%
3377 \let\endtabular\LWR@origendtabular%
3378 \let\newline\LWR@orignewline%
3379 }% end of outer-most lateximage
3380 }% end of \begin{lateximage}
3381 {% start of \end{lateximage}
3382 \ifthenelse{\cnttest{\value{LWR@lateximagedepth}}{>}{1}}{

```

If nesting inside an already-existing lateximage, simply record one more level:

```

3383 {%
3384 \addtocounter{LWR@lateximagedepth}{-1}%
3385 }%

```

if this is the outer-most lateximage:

```

3386 {% end of outer-most lateximage

```

Finish the lateximage minipage and start a new PDF page:

```

3387 \LWR@origendminipage%
3388 \LWR@orignewpage%
3389 \LWR@origscriptsize%

```

Close the HTML comment which encapsulated any traces of the lateximage picked up by `pdftotext`:

```

3390 \LWR@htmlclosecomment{}\LWR@orignewline%

```

Create a link to the lateximage, allowing its natural height:

If the alt tag is given as “LWRAMSMATHBODY”, then use the text collected by the `amsmath` `multline`, `gather`, or `align` environments.

```

3391 \ifthenelse{\equal{#1}{LWRAMSMATHBODY}}{
3392 {%
3393 \LWR@subinlineimage[%

```

```

3394 \LWR@HTMLsanitizeexpand{\detokenize\expandafter{\the\@envbody}}}%
3395 ]%
3396 {lateximage}%
3397 {lateximages\OSPathSymbol{}}lateximage-\theLWR@externalfilecnt}%
3398 {svg}%
3399 {}%
3400 }%
3401 {%
3402 \LWR@subinlineimage[#1]{lateximage}%
3403 {lateximages\OSPathSymbol{}}lateximage-\theLWR@externalfilecnt}{svg}{}%
3404 }%
3405 % \LWR@orignewline% Removed to prevent extra space.

```

Be sure that are doing a paragraph:

```

3406 \LWR@ensuredoingapar%

```

Close the HTML span which has the `pdftotext` comment and also the link to the `.svg` image:

```

3407 \LWR@htmltag{/span}%
3408 \LWR@htmlcomment{End of lateximage}%
3409 % \LWR@orignewline% Removed to prevent extra space.

```

Undo one `lateximage` level:

```

3410 \addtocounter{LWR@lateximagedepth}{-1}%
3411 }% end of outer-most lateximage
3412 }%
3413 \end{warpHTML}

```

for PRINT output:

```

3414 \begin{warpprint}
3415 \newenvironment{lateximage}[1][\begingroup]{\endgroup}
3416 \end{warpprint}

```

## 55 center, flushleft, flushright

for HTML output:

```

3417 \begin{warpHTML}

```

Env `center` Replace `center` functionality with CSS tags:

```

3418 \renewenvironment*{center}
3419 {\BlockClass{center}}
3420 {\endBlockClass}

```

Env `flushright`

```
3421 \renewenvironment*{flushright}
3422 {\BlockClass{flushright}}
3423 {\endBlockClass}
```

Env `flushleft`

```
3424 \renewenvironment*{flushleft}
3425 {\BlockClass{flushleft}}
3426 {\endBlockClass}

3427 \end{warpHTML}
```

## 56 Siunitx

Pkg `siunitx`



`per-mode`

Do not use `per-mode=fraction`, which cannot be seen by the final `pdftotext` conversion.

for HTML output: 3428 \begin{warpHTML}

Options for `siunitx`:

```
3429 \PassOptionsToPackage{
3430 detect-mode=true,
3431 per-mode=symbol,% fraction is not seen by pdftotext
3432 text-celsius = {\HTMLentity{deg}C},
3433 text-degree = {\HTMLentity{deg}},
3434 }{siunitx}

3435 \end{warpHTML}
```

## 57 Graphics

Pkg `graphics`


Pkg `graphicx`

`\graphicspath` `\graphicspath` only works for a single directory; all graphics must be in this directory.


**units** For `\includegraphics`, avoid `px` and `%` units for width and height, or enclose them inside `warpHTML` environments. For font-proportional image sizes, use `ex` or `em`. For fixed-sized images, use `cm`, `mm`, `in`, `pt`, or `pc`. Using the keys `width=.5\linewidth`, or similar for `\textwidth` or `\textheight` to give fixed-sized images proportional to a 6 by 9 inch text area.

**options** `\includegraphics` accepts `width` and `height`, `origin`, `rotate` and `scale`, plus a new `class` key.

**HTML class** With HTML output, `\includegraphics` accepts an optional `class=xyz` keyval combination, and if this is given then the HTML output will include that class for the image. The class is ignored for print output.

 **image file types** For `\includegraphics` the user should provide both `.pdf` and `.svg` images, but always refer to `.pdf` images in the document source. All `\includegraphics` references to `.pdf` will automatically be changed to `.svg` for HTML output, and will be left as `.pdf` for print output. Images may also be `.jpg` and `.png`, and will be used as-is for either output.

**\rotatebox** `\rotatebox` accepts the optional `origin` key.

 **browser support** `\rotatebox`, `\scalebox`, and `\reflectbox` depend on modern browser support. The CSS3 standard declares that when an object is transformed the whitespace which they occupied is preserved, unlike  $\text{\LaTeX}$ , so expect some ugly results for scaling and rotating.

**for HTML output:** 3436 `\begin{warpHTML}`

## 57.1 `\graphicspath`

`\graphicspath`  $\{\langle path \rangle\}$

3437 `\newcommand*\thisgraphicspath{\{}}`

3438 `\renewcommand*\graphicspath[1]{\renewcommand*\thisgraphicspath{\#1}}`

`\DeclareGraphicsExtensions`  $\{\langle list \rangle\}$

`\DeclareGraphicsRule`  $\{\langle \rangle\} \{\langle \rangle\} \{\langle \rangle\} \{\langle \rangle\}$

3439 `\renewcommand*\DeclareGraphicsExtensions[1]{}`

3440 `\renewcommand*\DeclareGraphicsRule[4]{}`



## 57.2 Length conversions and graphics options



whitespace

A scaled image in L<sup>A</sup>T<sub>E</sub>X by default takes only as much space on the page as it requires, but HTML browsers use as much space as the original unscaled image would have taken, with the scaled image over- or under-flowing the area.

```
3441 \renewcommand*{\unitspace}{}

```

Used to store the user's selected dimensions and HTML class.

The class defaults to “inlineimage” unless changed by a `class=xyx` option.

```
3442 \newlength{\LWR@igwidth}
3443 \newlength{\LWR@igheight}
3444 \newcommand*{\LWR@igwidthstyle}{}
3445 \newcommand*{\LWR@igheightstyle}{}
3446 \newcommand*{\LWR@igorigin}{}
3447 \newcommand*{\LWR@igangle}{}
3448 \newcommand*{\LWR@igxscale}{1}
3449 \newcommand*{\LWR@igyyscale}{1}
3450 \newcommand*{\LWR@igclass}{inlineimage}

```

Set the actions of each of the key/value combinations for `\includegraphics`. Many are ignored.

If an optional width was given, set an HTML style:

```
3451 \define@key{igraph}{width}{%
3452 \setlength{\LWR@igwidth}{#1}%
3453 \ifthenelse{\lengthtest{\LWR@igwidth > 0pt}}{}%
3454 {}}

```

Default to use the converted fixed length given:

```
3455 \uselengthunit{PT}%
3456 \renewcommand*{\LWR@igwidthstyle}{width:\rndprintlength{\LWR@igwidth}}%

```

If ex or em dimensions were given, use those instead:

```
3457 \IfEndWith{#1}{ex}%
3458 {\renewcommand*{\LWR@igwidthstyle}{width:#1}}% yes ex
3459 {}% not ex
3460 \IfEndWith{#1}{em}%
3461 {\renewcommand*{\LWR@igwidthstyle}{width:#1}}% yes em
3462 {}% not em
3463 \IfEndWith{#1}{\}%
3464 {\renewcommand*{\LWR@igwidthstyle}{width:#1}}% yes percent
3465 {}% not percent

```

```

3466 \IfEndWith{#1}{px}%
3467 {\renewcommand*{\LWR@igwidthstyle}{width:#1}}% yes px
3468 {}% not px
3469 }{}% end of length > Opt
3470 }

```

If an optional height was given, set an HTML style:

```

3471 \define@key{igraph}{height}{%
3472 \setlength{\LWR@igheight}{#1}%
3473 \ifthenelse{\lengthtest{\LWR@igheight > Opt}}%
3474 {%

```

Default to use the converted fixed length given:

```

3475 \uselengthunit{PT}%
3476 \renewcommand*{\LWR@igheightstyle}{%
3477 height:\rndprintlength{\LWR@igheight} %
3478 }%

```

If ex or em dimensions were given, use those instead:

```

3479 \IfEndWith{#1}{ex}%
3480 {\renewcommand*{\LWR@igheightstyle}{height:#1}}% yes ex
3481 {}% not ex
3482 \IfEndWith{#1}{em}%
3483 {\renewcommand*{\LWR@igheightstyle}{height:#1}}% yes em
3484 {}% not em
3485 \IfEndWith{#1}{\}%
3486 {\renewcommand*{\LWR@igheightstyle}{height:#1}}% yes percent
3487 {}% not percent
3488 \IfEndWith{#1}{px}%
3489 {\renewcommand*{\LWR@igheightstyle}{height:#1}}% yes px
3490 {}% not px
3491 }{}% end of length > Opt
3492 }

```

Handle origin key:

```

3493 \define@key{igraph}{origin}{%
3494 \renewcommand*{\LWR@igorigin}{#1}%
3495 }

```

Handle angle key:

```

3496 \define@key{igraph}{angle}{\renewcommand*{\LWR@igangle}{#1}}

```

Handle class key:

```

3497 \define@key{igraph}{class}{\renewcommand*{\LWR@igclass}{#1}}
3498

```

It appears that `graphicx` does not have separate keys for `xscale` and `yscale`. `scale` adjusts both at the same time.

```

3499 \define@key{igraph}{scale}{%
3500 \renewcommand*{\LWR@igxscale}{#1}%
3501 \renewcommand*{\LWR@igyscale}{#1}}

```

Numerous ignored keys:

```

3502 \define@key{igraph}{bb}{}
3503 \define@key{igraph}{bbllx}{}
3504 \define@key{igraph}{bbllx}{}
3505 \define@key{igraph}{bburx}{}
3506 \define@key{igraph}{bbury}{}
3507 \define@key{igraph}{natwidth}{}
3508 \define@key{igraph}{natheight}{}
3509 \define@key{igraph}{hiresbb}{}
3510 \define@key{igraph}{viewport}{}
3511 \define@key{igraph}{trim}{}
3512 \define@key{igraph}{totalheight}{}
3513 \define@key{igraph}{keepaspectratio}{}
3514 \define@key{igraph}{clip}{}
3515 \define@key{igraph}{draft}{}
3516 \define@key{igraph}{type}{}
3517 \define@key{igraph}{ext}{}
3518 \define@key{igraph}{read}{}
3519 \define@key{igraph}{command}{}

```

`\LWR@rotstyle`  $\{\langle prefix \rangle\} \{\langle degrees \rangle\}$

Prints the rotate style with the given prefix.

`prefix` is `-ms-` or `-webkit-` or nothing, and is used to generate three versions of the `transform:rotate` style.

```

3520 \newcommand*{\LWR@rotstyle}[2]{%
3521 #1transform:rotate(-#2deg);
3522 }

```

`\LWR@scalestyle`  $\{\langle prefix \rangle\} \{\langle xscale \rangle\} \{\langle yscale \rangle\}$

Prints the scale style with the given prefix.

`prefix` is `-ms-` or `-webkit-` or nothing, and is used to generate three versions of the `transform:scale` style.

```

3523 \newcommand*{\LWR@scalestyle}[3]{%
3524   #1transform:scale(#2,#3);
3525 }

```

### 57.3 \includegraphics

Bool `LWR@infloatrow` Used to compute `\linewidth`.

```

3526 \newbool{LWR@infloatrow}
3527 \boolfalse{LWR@infloatrow}

3528 \newcommand*{\LWR@imageextension}{}
3529 \newcommand*{\LWR@expgraphicsfilename}{}

```

`\LWR@includegraphicsb` [*options*] {*filename*}

```

3530
3531 \newcommand*{\LWR@includegraphicsb}[2]{}
3532 {}

```

Start the image tag on a new line, allow PDF output word wrap:

```

3533 \LWR@origtilde \LWR@orignewline%

```

Temporarily compute `\linewidth`, `\textwidth`, `\textheight` arguments with a 6x9 inch size until the next `\endgroup`.

```

3534 \ifthenelse{\cinttest{\value{LWR@minipagedepth}}{=}{0}}{%
3535   \ifbool{LWR@infloatrow}%
3536   {}
3537   {% not in a minipage or a floatrow:
3538     \setlength{\linewidth}{6in}%
3539     \setlength{\textwidth}{6in}%
3540     \setlength{\textheight}{9in}%
3541   }%
3542 }{}%

```

See if can find the image by adding an extension:

Preference is `svgz`, then `svg`, `gif`, `png`, and `jpg`.

`\detokenize\expandafter` allows underscore characters in filenames.

```

3543 \edef\LWR@expgraphicsfilename{#2}
3544 \renewcommand*{\LWR@imageextension}{}%
3545 \IfFileExists{\detokenize\expandafter\thisgraphicspath\LWR@expgraphicsfilename.jpg}%

```

```

3546 {\renewcommand*{\LWR@imageextension}{.jpg}}{}%
3547 \IfFileExists{\detokenize\expandafter\thisgraphicspath\LWR@expgraphicsfilename.JPG}%
3548 {\renewcommand*{\LWR@imageextension}{.JPG}}{}%
3549 \IfFileExists{\detokenize\expandafter\thisgraphicspath\LWR@expgraphicsfilename.png}%
3550 {\renewcommand*{\LWR@imageextension}{.png}}{}%
3551 \IfFileExists{\detokenize\expandafter\thisgraphicspath\LWR@expgraphicsfilename.PNG}%
3552 {\renewcommand*{\LWR@imageextension}{.PNG}}{}%
3553 \IfFileExists{\detokenize\expandafter\thisgraphicspath\LWR@expgraphicsfilename.gif}%
3554 {\renewcommand*{\LWR@imageextension}{.gif}}{}%
3555 \IfFileExists{\detokenize\expandafter\thisgraphicspath\LWR@expgraphicsfilename.GIF}%
3556 {\renewcommand*{\LWR@imageextension}{.GIF}}{}%
3557 \IfFileExists{\detokenize\expandafter\thisgraphicspath\LWR@expgraphicsfilename.svg}%
3558 {\renewcommand*{\LWR@imageextension}{.svg}}{}%
3559 \IfFileExists{\detokenize\expandafter\thisgraphicspath\LWR@expgraphicsfilename.SVG}%
3560 {\renewcommand*{\LWR@imageextension}{.SVG}}{}%
3561 \IfFileExists{\detokenize\expandafter\thisgraphicspath\LWR@expgraphicsfilename.svgz}%
3562 {\renewcommand*{\LWR@imageextension}{.svgz}}{}%
3563 \IfFileExists{\detokenize\expandafter\thisgraphicspath\LWR@expgraphicsfilename.SVGZ}%
3564 {\renewcommand*{\LWR@imageextension}{.SVGZ}}{}%

```

Convert a PDF extension to SVG, leave the result in \LWR@strresult:

Must also \detokenize .pdf and .svg comparison strings.

```

3565 \StrSubstitute{\detokenize\expandafter{\LWR@expgraphicsfilename}}%
3566 {\detokenize{.pdf}}{\detokenize{.svg}}[\LWR@strresult]%
3567 %
3568 \StrSubstitute{\LWR@strresult}%
3569 {\detokenize{.PDF}}{\detokenize{.SVG}}[\LWR@strresult]%

```

For correct em sizing during the width and height conversions:

```

3570 \large%

```

Reset some defaults, possibly will be changed below if options were given:

```

3571 \setlength{\LWR@igwidth}{0pt}%
3572 \setlength{\LWR@igheight}{0pt}%
3573 \renewcommand*{\LWR@igwidthstyle}{}%
3574 \renewcommand*{\LWR@igheightstyle}{}%
3575 \renewcommand*{\LWR@igorigin}{}%
3576 \renewcommand*{\LWR@igangle}{}%
3577 \renewcommand*{\LWR@igxscale}{1}%
3578 \renewcommand*{\LWR@igyscale}{1}%
3579 \renewcommand*{\LWR@igclass}{inlineimage}%

```

Options are in #1

```

3580 \setkeys{igraph}{#1}%

```

Create the HTML reference with the graphicspath, filename, extension, alt tag, style, and class.

The `\LWR@origtilde` adds space between tags in case this is being done inside a `\savebox` where `\newline` has no effect.

```

3581 \href{\thisgraphicspath\LWR@strresult\LWR@imageextension}%
3582 {% start of href
3583 \LWR@htmltag{% start of image tags
3584 img src="\thisgraphicspath\LWR@strresult\LWR@imageextension" \LWR@originewline
3585 \LWR@origtilde{} alt="\LWR@strresult" \LWR@originewline

```

Only include a style tag if a width, height, angle, or scale was given:

```

3586 \ifthenelse{
3587 \NOT\equal{\LWR@igwidthstyle}{}} \OR
3588 \NOT\equal{\LWR@igheightstyle}{}} \OR
3589 \NOT\equal{\LWR@igorigin{}} \OR
3590 \NOT\equal{\LWR@igangle{}} \OR
3591 \NOT\equal{\LWR@igxscale}{1}} \OR
3592 \NOT\equal{\LWR@igyscale}{1}}
3593 }%
3594 {\LWR@origtilde{} style="%
3595 \ifthenelse{\NOT\equal{\LWR@igwidthstyle}{}}%
3596 {\LWR@igwidthstyle;}}}%
3597 \ifthenelse{\NOT\equal{\LWR@igheightstyle}{}}%
3598 {\LWR@igheightstyle;}}}%
3599 \ifthenelse{\NOT\equal{\LWR@igorigin{}}}%
3600 {\LWR@origtilde{} transform-origin: \LWR@originnames{\LWR@igorigin}; \LWR@originewline}{}}%
3601 \ifthenelse{\NOT\equal{\LWR@igangle{}}}%
3602 {%
3603 \LWR@rotstyle{-ms-}{\LWR@igangle}%
3604 \LWR@rotstyle{-webkit-}{\LWR@igangle}%
3605 \LWR@rotstyle{}{\LWR@igangle}%
3606 }{}%
3607 \ifthenelse{\NOT\equal{\LWR@igxscale}{1}}\OR%
3608 \NOT\equal{\LWR@igyscale}{1}}%
3609 {\LWR@scalestyle{-ms-}{\LWR@igxscale}{\LWR@igyscale}%
3610 \LWR@scalestyle{-webkit-}{\LWR@igxscale}{\LWR@igyscale}%
3611 \LWR@scalestyle{}{\LWR@igxscale}{\LWR@igyscale}}{}%
3612 " \LWR@originewline){}}%

```

Set the class:

```

3613 \LWR@origtilde{} class="\LWR@igclass" \LWR@originewline%
3614 }% end of image tags
3615 }% end of href
3616 \endgroup

```

Return to small-sized output:

```
3617 \LWR@origscriptsize
3618 }
```

`\includegraphics` [*<key=val>*] {*<filename>*}

Handles width and height, converted to fixed width and heights.

Converts any `.pdf` references to `.svg` for HTML

The user should always refer to `.pdf` in the document source.

```
3619 \renewcommand*{\includegraphics}
3620 {%
```

This graphic should trigger an HTML paragraph even if alone, so ensure that are doing paragraph handling:

```
3621 \LWR@ensuredoingapar%
3622 \begin{group}%
3623 \LWR@includegraphicsb%
3624 }
```

```
3625 \end{warpHTML}
```

**for PRINT output:** For print output, accept and then discard the new `class` key:

```
3626 \begin{warpprint}
3627 \define@key{Gin}{class}{}
3628 \end{warpprint}
```

## 57.4 `\rotatebox`, `\scalebox`, `\reflectbox`

**for HTML output:** 3629 `\begin{warpHTML}`

`\LWR@rotboxorigin` Holds the origin key letters.

```
3630 \newcommand*{\LWR@rotboxorigin}{}%
```

`\LWR@originname` {*<letter>*}

Given one `LATEX` origin key value, translate into an HTML origin word:

```
3631 \newcommand*{\LWR@originname}[1]{%
```

```

3632 \ifthenelse{\equal{#1}{t}}{top}{}%
3633 \ifthenelse{\equal{#1}{b}}{bottom}{}%
3634 \ifthenelse{\equal{#1}{c}}{center}{}%
3635 \ifthenelse{\equal{#1}{l}}{left}{}%
3636 \ifthenelse{\equal{#1}{r}}{right}{}%
3637 }

```

`\LWR@originnames`  $\{ \langle letters \rangle \}$

Given one- or two-letter L<sup>A</sup>T<sub>E</sub>X origin key values, translate into HTML origin words:

```

3638 \newcommand*{\LWR@originnames}[1]{%
3639 \StrChar{#1}{1}[\LWR@strresult]%
3640 \LWR@originname{\LWR@strresult}%
3641 \StrChar{#1}{2}[\LWR@strresult]%
3642 \LWR@originname{\LWR@strresult}%
3643 }

```

Handle the origin key for `\rotatebox`:

```

3644 \define@key{krotbox}{origin}{%
3645 \renewcommand*{\LWR@rotboxorigin}{#1}%
3646 }

```

These keys are ignored:

```

3647 \define@key{krotbox}{x}{}
3648 \define@key{krotbox}{y}{}
3649 \define@key{krotbox}{units}{}

```

`\rotatebox`  $[ \langle keyval list \rangle ] \{ \langle angle \rangle \} \{ \langle text \rangle \}$

Will `\let\rotatebox\LWR@rotatebox` at `\LWR@LwarpStart`, in case `\rotatebox` was over-written by a later package load.

```

3650 \NewDocumentCommand{\LWR@rotatebox}{0{} m +m}{%

```

Reset the origin to “none-given”:

```

3651 \renewcommand*{\LWR@rotboxorigin}{}

```

Process the optional keys, which may set `\LWR@rotateboxorigin`:

```

3652 \setkeys{krotbox}{#1}%

```

Select `inline-block` so that HTML will transform this span:

```

3653 \LWR@htmltagc{span style="display: inline-block;

```



If an origin was given, translate and print the origin information:

```
3654 \ifthenelse{\NOT\equal{\LWR@rotboxorigin}{}}%
3655 {transform-origin: \LWR@originnames{\LWR@rotboxorigin};\LWR@origtilde}{}%
```

Print the rotation information:

```
3656 \LWR@rotstyle{-ms-}{#2} %
3657 \LWR@rotstyle{-webkit-}{#2} %
3658 \LWR@rotstyle{}{#2} %
3659 "{}\LWR@orignewline%
```

Print the text to be rotated:

```
3660 \begin{LWR@nestspan}%
3661 #3%
```

Close the span:

```
3662 \LWR@htmltagc{/span}%
3663 \end{LWR@nestspan}%
3664 }
```

`\scalebox`  $\{\langle h\text{-scale}\rangle\} [\langle v\text{-scale}\rangle] \{\langle text\rangle\}$

Will `\let\scalebox\LWR@scalebox` at `\LWR@LwarpStart`, in case `\scalebox` was over-written by a later package load.

```
3665 \NewDocumentCommand{\LWR@scalebox}{m o m}{%
```

Select `inline-block` so that HTML will transform this span:

```
3666 \LWR@htmltagc{span style="display: inline-block; %
```

Print the scaling information:

```
3667 \LWR@scalestyle{-ms-}{#1}{\IfNoValueTF{#2}{#1}{#2}} %
3668 \LWR@scalestyle{-webkit-}{#1}{\IfNoValueTF{#2}{#1}{#2}} %
3669 \LWR@scalestyle{}{#1}{\IfNoValueTF{#2}{#1}{#2}} %
3670 "{}}%
```

Print the text to be scaled:

```
3671 \begin{LWR@nestspan}%
3672 #3%
```

Close the span:

```

3673 \LWR@htmltagc{/span}%
3674 \end{LWR@nestspan}%
3675 }

```

`\reflectbox`  $\{\langle text \rangle\}$

Will `\let\reflectbox\LWR@reflectbox` at `\LWR@LwarpStart`, in case `\reflectbox` was over-written by a later package load.

```

3676 \newcommand{\LWR@reflectbox}[1]{\LWR@scalebox{-1}[1]{#1}}
3677 \end{warpHTML}

```

## 57.5 Null functions

These functions are not supported by `lwarp`'s HTML conversion.

**for HTML output:** 3678 `\begin{warpHTML}`

`\resizebox`  $\{\langle h-length \rangle\} \{\langle v-length \rangle\} \{\langle text \rangle\}$

Simply prints its text argument.

```

3679 \renewcommand{\resizebox}[3]{#3}
3680 \end{warpHTML}

```

## 58 Cleverref

**Pkg** `cleveref` `cleveref` package is used as-is with minor patches.

**loading order** `cleveref` and the following associated macro patches are automatically preloaded at the end of the preamble via `\AtEndPreamble` and `\AfterEndPreamble`. This is done because the HTML conversion requires `cleveref`. The user's document may not require `cleveref`, thus the user may never explicitly load it, so during HTML output `lwarp` loads it last. If the user's document preamble uses `cleveref` options, or functions such as `\crefname`, then `cleveref` may be loaded in the user's preamble near the end, and `lwarp`'s additional loading of `cleveref` will have no effect.

Table 7 on 181 shows the data structure of the label/reference system as revised by `lwarp` and `cleveref`.

A few patches allow `cleveref` to work as-is:

for HTML output: 3681 \begin{warpHTML}

`\AtEndPreamble` forces `cleveref` to be loaded last:

```

3682 \AtEndPreamble{
3683 \RequirePackage{cleveref}
3684 \crefname{pagenote}{note}{notes}
3685 }

```

The following patches are applied after `cleveref` has loaded, and after `\AtBeginDocument`:

```
3686 \AfterEndPreamble{
```

$$\backslash@@setcref \{\langle kindofref \rangle\} \{\langle label \rangle\}$$

```
3687 \renewcommand*{\@@setcref}[2]{#1{\ref{#2}}}{}}{}}
```

$$\backslash@@setcrefrange \quad \{\langle text \rangle\} \quad \{\langle label \rangle\} \quad \{\langle label \rangle\}$$

```
3688 \renewcommand{\@@setcrefrange}[3]{%
3689 #1{\ref{#2}}{\ref{#3}}{}{}{}}}
```

`\cpagerefFor` Redefinable word between “page(s)” and the page numbers.

```
3690 \newcommand*{\cpagerefFor}{for}
```

`\@@setcpageref`  $\{\langle typeofref \rangle\} \{\langle label \rangle\}$ , where `typeofref` is “page” or “pages”

```

3691 \renewcommand*{\@@setcpageref}[2]{%
3692 #1{\cpagerefFor\ \cref{#2}}{}}%
3693 }

```

```
3694 \renewcommand{\@@setcpagerefrange}[3]{%
3695 #1{\cpagerefFor\ \cref{#2}}{\cref{#3}}{}{}{}{}}
```

Remember and patch some label-related definitions.

```

3696 \let\LWR@origlabel\label
3697 \let\label\LWR@newlabel
3698 \let\LWR@origref\ref
3699 \let\ref\LWR@newref% \end{ syntax highlighting
3700 \let\LWR@origpageref\pageref
3701 \let\pageref\LWR@newpageref

```

```

3702
3703
3704 }% AfterEndPreamble
3705
3706 \end{warpHTML}

```

## 59 Picture

Env **picture** The **picture** environment is enclosed inside a `\lateximage`.

for HTML output: 3707 \begin{warpHTML}

Env **picture**

```

3708 \BeforeBeginEnvironment{picture}{
3709
3710 \lateximage}
3711
3712 \AfterEndEnvironment{picture}{\endlateximage
3713
3714 }

3715 \end{warpHTML}


```

## 60 Tikzpicture

Pkg **tikz**

Env **tikzpicture** May be used as-is, and its contents will be converted to an image.

For recent versions of *Tikz* (v3.0.0 — 2013/12/20 and later), include in the document preamble:

 **preamble** `\usetikzlibrary{babel} % v3.0.0 and up`

For older versions of *Tikz* without the *Tikz babel* library, *lwarp* automatically handles the catcode change for the `$` math shift.

**catcodes** (*lwarp* changes the catcode of `$` for its own use. The *Tikz babel* library temporarily changes catcodes back to normal for *Tikz*'s use, restoring them to their *lwarp* purpose while inside each *Tikz* node.)

for HTML output: 3716 \begin{warpHTML}

Bool `LWR@tikzbabel` `tikz` v3.0.0 introduced the `babel` library which handles catcode changes. For older versions, `lwarp` must change `$`'s catcode itself.

```

3717 \newbool{LWR@tikzbabel}
3718 \boolfalse{LWR@tikzbabel}
3719
3720 \AtBeginDocument{
3721 \ifpackageloaded{tikz}{
3722 \ifpackagelater{tikz}{2013/12/20}% Test for Tikz version v3.0.0
3723 {\booltrue{LWR@tikzbabel}}
3724 {\boolfalse{LWR@tikzbabel}}
3725 }{}
3726 }

```

Env `tikzpicture` `tikzpicture` environment is enclosed inside a `\lateximage`.

```

3727 \BeforeBeginEnvironment{tikzpicture}{%
3728
3729 \lateximage
3730 \ifbool{LWR@tikzbabel}
3731 {}
3732 {\catcode'\$=3} % dollar sign is math shift
3733 }
3734
3735 \AfterEndEnvironment{tikzpicture}{%
3736 \endlateximage
3737
3738 \ifbool{LWR@tikzbabel}
3739 {}
3740 {\catcode'\$=\active}
3741 }
3742 \end{warpHTML}

```

## 61 Boxes and Minipages

A CSS flexbox is used for minipages and parboxes, allowing external and internal vertical positioning.

Minipages and parboxes will be placed side-by-side in HTML unless you place a `\newline` between them.

`inline` A line of text with an inline minipage or parbox will have the minipage or parbox placed onto its own line, because a paragraph is a block element and cannot be made `inline-block`.

- in a span** There is limited support for minipages inside an HTML `|span|`. An HTML `|div|` cannot appear inside a `|span|`. While in a `|span|`, minipages and parboxes are ignored. Use `\newline` or `\par` for an HTML break.
- size** When using `\linewidth`, `\textwidth`, and `\textheight`, widths and heights are scaled proportionally to a 6×9 inch text area.
- full-width minipages** A new macro `\minipagefullwidth` requests that the next minipage be generated without an HTML `width` tag, allowing it to be the full width of the display rather than the fixed width given.
- ⚠ text alignment** Nested minipages adopt their parent’s text alignment in HTML, whereas in regular L<sup>A</sup>T<sub>E</sub>X PDF output they do not. Use a `flushleft` or similar environment in the child minipage to force a text alignment.
- for HTML output:** 3743 `\begin{warpHTML}`

## 61.1 Counters and lengths

**Ctrl** `LWR@minipagedepth` Used to only reset the line width at the outermost minipage.

```
3744 \newcounter{LWR@minipagedepth}
3745 \setcounter{LWR@minipagedepth}{0}
```

**Len** `\WR@minipagewidth` Used to convert the width into printable units.

```
3746 \newlength{\LWR@minipagewidth}
```

**Len** `\WR@minipageheight` Used to convert the height into printable units.

```
3747 \newlength{\LWR@minipageheight}
```

Remember the original definitions:

```
3748 \let\LWR@origminipage\minipage
3749 \let\LWR@origendminipage\endminipage
```

## 61.2 Footnote handling

Also see section 34 for other forms of footnotes.

`\LWR@nullrule` [*raise*] {*width*} {*height*}

Used to cancel the footnotesep rule during HTML output. This rule was causing occasional end paragraph tags to be inserted for some (so far) undetermined reason, as well as adding purposeless code to the HTML output.

```
3750 \NewDocumentCommand{\LWR@nullrule}{o m m}{}
```

`\LWR@makefntext`  $\{ \langle text \rangle \}$

Adds a paragraph closing tag to the end of each footnote.

```
3751 \newcommand{\LWR@makefntext}[1]{%
3752 \begingroup%
3753 \let\rule\LWR@nullrule%
3754 \textsuperscript{\@thefnmark} #1%
3755 \endgroup%
3756 \LWR@htmltagc{/LWR@tagregularparagraph}%
3757 }
```

`\LWR@makefnmark` Creates an HTML superscript footnote mark:

```
3758 \newcommand*{\LWR@makefnmark}{\textsuperscript{\@thefnmark}}
```

### 61.3 Minipage handling

`\LWR@endminipage` Used to close a minipage.

Copied the L<sup>A</sup>T<sub>E</sub>X definition and modified to create a `mpfootnotes` div class:

```
3759 \def\LWR@endminipage{%
3760   \par
3761   \unskip
3762   \ifvoid\@mpfootins\else
3763     \vskip\skip\@mpfootins
3764     \normalcolor
3765   \LWR@htmldivclass{mpfootnotes}
3766     \unvbox\@mpfootins
3767 \LWR@htmldivclassend{mpfootnotes}
3768   \fi
3769   \@minipagefalse
3770   \color@endgroup
3771   \egroup
3772   \expandafter\@iiiparbox\@mpargs{\unvbox\@tempboxa}}
```

`\LWR@subminipage` Used to create a PDF minipage without creating an HTML minipage. This allows footnotes to appear at the bottom of the minipage instead of the bottom of the HTML page.

```
3773 \newcommand*{\LWR@subminipage}{%
3774 \LWR@stoppars
3775 \LWR@origminipage{6in}
```

`\raggedright` cancels hyphenation, which will be done by HTML instead.

```
3776 \LWR@origraggedright%
```

Minipage footnotes appear the bottom of the minipage.

```
3777 \let\footnote\LWR@origfootnote%
3778 \let\@makefnmark\LWR@makefnmark%
3779 \let\@makefnmark\LWR@makefnmark%
```

Resume paragraph tag handling for the contents of the minipage:

```
3780 \LWR@startpars%
3781 }
```

`\LWR@endsubminipage` Closes the subminipage.

```
3782 \newcommand*{\LWR@endsubminipage}{%
3783 \LWR@stoppars%
3784 \LWR@endminipage% The following empty line is required:
3785
3786 }
```

Bool `LWR@minipagefullwidth` Should the next minipage have no HTML width?

```
3787 \newbool{LWR@minipagefullwidth}
3788 \boolfalse{LWR@minipagefullwidth}
```

`\minipagefullwidth` Requests that the next minipage have no width tag in HTML:

**for HTML output:**

```
3789 \newcommand*{\minipagefullwidth}{\booltrue{LWR@minipagefullwidth}}
3790 \end{warpHTML}
```

**for PRINT output:**

```
3791 \begin{warpprint}
3792 \newcommand*{\minipagefullwidth}{}
3793 \end{warpprint}
```

**for HTML output:**

```
3794 \begin{warpHTML}
```



Bool LWR@minipagethispar Has a minipage been seen this paragraph? If true, prevents paragraph tags around horizontal space between minipages.

```
3795 \newbool{LWR@minipagethispar}
3796 \boolfalse{LWR@minipagethispar}
```

Env minipage [*<vert position>*] [*<height>*] [*<inner vert position>*] {*<width>*}

The vertical positions may be 'c', 't', or 'b'. The inner position may also be 's'.

When using \linewidth, \textwidth, or \textheight, these are scaled proportionally to a 6×9 inch text area.

```
3797 \RenewDocumentEnvironment{minipage}{0{t} o 0{t} m}
3798 {%
```

Pre-compute the given width and height:

Reset the text area if are starting the outer-most minipage:

```
3799 \LWR@traceinfo{starting minipage}%
3800 \uselengthunit{in}%
3801 \setlength{\LWR@minipagewidth}{#4}%
3802 \ifthenelse{\cnttest{\value{LWR@minipagedepth}}{=}{0}}{%
3803 \addtolength{\LWR@minipagewidth}{3em}% room for frames
3804 \setlength{\linewidth}{6in}%
3805 \setlength{\textwidth}{6in}%
3806 \setlength{\textheight}{9in}%
3807 }{}%
3808 \addtocounter{LWR@minipagedepth}{1}%
3809 \setlength{\LWR@minipageheight}{\textheight}% default unless specified
3810 \IfValueTF{#2}{\setlength{\LWR@minipageheight}{#2}}{}%
```

L<sup>A</sup>T<sub>E</sub>X wants to start a paragraph for the new minipage, then start a paragraph again for the contents of the minipage, so cancel the paragraph tag handling until the minipage has begun.

```
3811 \LWR@stoppars%
```

Create the <div> tag with optional alignment style:

```
3812 \LWR@traceinfo{minipage: creating div class}%
3813 \LWR@htmltag{div class="minipage" style="%
3814 \ifthenelse{\equal{#1}{t}}{vertical-align: bottom ; }{}%
3815 \ifthenelse{\equal{#1}{c}}{vertical-align: middle ; }{}%
3816 \ifthenelse{\equal{#1}{b}}{vertical-align: top ; }{}%
3817 \ifthenelse{\equal{#3}{t}}{justify-content: flex-start ; }{}%
```

```

3818 \ifthenelse{\equal{#3}{c}}{justify-content: center ; }{}%
3819 \ifthenelse{\equal{#3}{b}}{justify-content: flex-end ; }{}%
3820 \ifthenelse{\equal{#3}{s}}{justify-content: space-between ; }{}%

```

Print the width and optional height styles:

```

3821 \LWR@traceinfo{minipage: about to print the width}%
3822 \uselengthunit{PT}%
3823 \ifbool{LWR@minipagefullwidth}%
3824 {\boolfalse{LWR@minipagefullwidth}}%
3825 {%
3826 \ifthenelse{\lengthtest{#4}=\linewidth}%
3827 {}%
3828 {width:\rndprintlength{LWR@minipagewidth} ; }%
3829 }%
3830 \LWR@traceinfo{minipage: about to print the height}%
3831 \IfValueTF{#2}{height:\rndprintlength{LWR@minipageheight} ; }{}%
3832 "{}%

```

Finish with an empty line to start L<sup>A</sup>T<sub>E</sub>X minipage processing on a new line. Use a large minipage area to avoid the unnecessary wrapping of tags.

```

3833
3834 \LWR@origminipage{6in}% The preceding empty line is required.

```

Set the user-accessible minipage and text width and height values inside the minipage. These do not affect the actual size of the large minipage created by `\LWR@origminipage` above, but are used by any reference to `\linewidth`, etc. inside the PDF minipage being created here.

```

3835 \setlength{\linewidth}{#4}% the original width
3836 \setlength{\textwidth}{6in}%
3837 \setlength{\textheight}{9in}%

```

`\raggedright` cancels hyphenation, which will be done by HTML instead.

```

3838 \LWR@origraggedright%
3839 \let\footnote\LWR@origfootnote%
3840 \let\@makefnmark\LWR@makefnmark%
3841 \let\@makefnmark\LWR@makefnmark%

```

Resume paragraph tag handling for the contents of the minipage:

```

3842 \LWR@startpars%
3843 \LWR@traceinfo{minipage: finished starting the minipage}%
3844 }

```

End the environment with L<sup>A</sup>T<sub>E</sub>X processing and closing tag:

```

3845 {%
3846 \LWR@stoppars%
3847 \LWR@endminipage% The following empty line is required:
3848
3849 \LWR@htmldivclassend{minipage}%
3850 \vspace{1\baselineskip}% required for subcaption
3851 \addtocounter{LWR@minipagedepth}{-1}%
3852 \LWR@startpars%

```

Prevent paragraph tags around horizontal white space until the start of the next paragraph:

```

3853 \global\booltrue{LWR@minipagethispar}%
3854 }

```

## 61.4 Parbox, makebox, framebox, fbox, raisebox

`\parbox` [*<pos>*] [*<height>*] [*<inner-pos>*] [*<width>*] [*<text>*]

A parbox uses the minipage code:

```

3855 \RenewDocumentCommand{\parbox}{O{t} o O{t} m m}
3856 {
3857 \begin{minipage}[#1][#2][#3]{#4}
3858 #5
3859 \end{minipage}
3860 }

```

`\makebox` [*<width>*] [*<pos>*] [*<text>*]

Width and position are ignored.

```

3861 \RenewDocumentCommand{\makebox}{o o m}{%
3862 \mbox{#3}
3863 }

```

`\framebox` [*<width>*] [*<pos>*] [*<text>*]

Width and position are ignored.

```

3864 \RenewDocumentCommand{\framebox}{o o m}{%
3865 \fbox{#3}
3866 }

```

`\fbox` [*<text>*]

```

3867 \let\LWR@origfbox\fbox
3868 %
3869 \renewcommand*{\fbox}[1]{%
3870 \InlineClass[framebox]{#1}%
3871 }

```

`\raisebox`  $\{\langle raiselen \rangle\} [\langle height \rangle] [\langle depth \rangle] \{\langle text \rangle\}$

```

3872 \RenewDocumentCommand{\raisebox}{m o o m}{%
3873 #4%
3874 }

```

```

3875 \end{warpHTML}

```

## 62 Direct formatting

`\textbf`, etc. are supported, but `\bfseries`, etc. are not yet supported.

For high-level block and inline custom CSS classes, see section [28.7](#).

**for HTML output:** 3876 `\begin{warpHTML}`

`\emph`  $\{\langle text \rangle\}$

```

3877 \renewcommand{\emph}[1]{\LWR@htmlspan{em}{#1}}

```

`\textmd`  $\{\langle text \rangle\}$

```

3878 \renewcommand{\textmd}[1]{\LWR@htmlspan{textmd}{#1}}

```

`\textbf`  $\{\langle text \rangle\}$

```

3879 \renewcommand{\textbf}[1]{\LWR@htmlspan{b}{#1}}

```

`\textrm`  $\{\langle text \rangle\}$

```

3880 \renewcommand{\textrm}[1]{\InlineClass{textrm}{#1}}

```

`\textsf`  $\{\langle text \rangle\}$

```

3881 \renewcommand{\textsf}[1]{\InlineClass{textsf}{#1}}

```

`\texttt`  $\{\langle text \rangle\}$

3882 `\renewcommand{\texttt}[1]{\LWR@htmlspan{kbd}{#1}}`

`\textup`  $\{\langle text \rangle\}$

3883 `\renewcommand{\textup}[1]{\LWR@htmlspan{textup}{#1}}`

`\textit`  $\{\langle text \rangle\}$

3884 `\renewcommand{\textit}[1]{\LWR@htmlspan{i}{#1}}`

`\textsc`  $\{\langle text \rangle\}$

3885 `\renewcommand{\textsc}[1]{\InlineClass{textsc}{#1}}`

`\textnormal`  $\{\langle text \rangle\}$

3886 `\renewcommand{\textnormal}[1]{\textmd{\textrm{\textup{#1}}}}`

`\mdseries`

3887 `\renewcommand*{\mdseries}{}`

`\bfseries`

3888 `\renewcommand*{\bfseries}{}`

`\rmfamily`

3889 `\renewcommand*{\rmfamily}{}`

`\sffamily`

3890 `\renewcommand*{\sffamily}{}`

`\ttfamily`

3891 `\renewcommand*{\ttfamily}{}`

`\upshape`

3892 `\renewcommand*{\upshape}{}`

`\itshape`

3893 `\renewcommand*\itshape{}`

`\scshape`

3894 `\renewcommand*\scshape{}`

`\scshape`

3895 `\renewcommand*\normalfont{}`

`\sp {<text>}`

For siunitx. Must work in math mode.

3896 `\renewcommand{\sp}[1]{\text{<sup>#1</sup>}}`

`\sb {<text>}`

For siunitx. Must work in math mode.

3897 `\renewcommand{\sb}[1]{\text{<sub>#1</sub>}}`

`\textsuperscript {<text>}`

3898 `\renewcommand{\textsuperscript}[1]{\LWR@htmlspan{sup}{#1}}`

`\textsubscript {<text>}`

3899 `\renewcommand{\textsubscript}[1]{\LWR@htmlspan{sub}{#1}}`

`\up {<text>}` Prints superscript.

This is `\let` at the beginning of the document in case some other package has changed the definition.

3900 `\AtBeginDocument{\let\up\textsuperscript}`

`\fup {<text>}` Prints superscript.

Supports `fmtcount` package.

This is `\let` at the beginning of the document in case some other package has changed the definition.

3901 `\AtBeginDocument{\let\fup\textsuperscript}`

`\hfill`

```
3902 \renewcommand*{\hfill}{\quad}
```

`\hrulefill`

```
3903 \renewcommand*{\hrulefill}{\rule{1in}{1pt}}
```

`\dotfill`

```
3904 \renewcommand*{\dotfill}{\dots}
```

```
3905 \end{warpHTML}
```

## 63 Skips, spaces, font sizes

for HTML output: 3906 `\begin{warpHTML}`

`\,` must be redefined after `\RequirePackage{printlen}`

```
3907 \let\LWR@origcomma\,
3908 \let\LWR@origtilde~
3909 \let\LWR@origenskip\enskip
3910 \let\LWR@origquad\quad
3911 \let\LWR@origqqquad\qqquad
3912 \let\LWR@orighspace\hspace
3913 \let\LWR@origrule\rule
```

Direct-formatting space commands become HTML entities:

```
3914 \renewcommand*{\,}{\HTMLunicode{202f}} % HTML thin non-breakable space
```

```
3915 \renewcommand*{~}{\HTMLentity{nbsp}}
```

```
3916 \renewcommand*{\textellipsis}{\HTMLunicode{2026}}
```

Direct-formatting font sizes are ignored:

```
3917 \let\LWR@orignormalsize\normalsize
3918 \let\LWR@origsmall\small
3919 \let\LWR@origfootnotesize\footnotesize
3920 \let\LWR@origscriptsize\scriptsize
3921 \let\LWR@origtiny\tiny
3922 \let\LWR@origlarge\large
3923 \let\LWR@origLarge\Large
```

```

3924 \let\LWR@origLARGE\LARGE
3925 \let\LWR@orighuge\huge
3926 \let\LWR@origHuge\Huge
3927 \renewcommand*{\normalsize}{\}
3928 \renewcommand*{\small}{\}
3929 \renewcommand*{\footnotesize}{\}
3930 \renewcommand*{\scriptsize}{\}
3931 \renewcommand*{\tiny}{\}
3932 \renewcommand*{\large}{\}
3933 \renewcommand*{\Large}{\}
3934 \renewcommand*{\LARGE}{\}
3935 \renewcommand*{\huge}{\}
3936 \renewcommand*{\Huge}{\}
3937
3938 \renewcommand*{\onecolumn}{\}
3939
3940 \renewcommand{\twocolumn}[1][\]{
3941
3942 #1
3943
3944 }

```

`\newline` Uses HTML `<br />` tag

```

3945 \newcommand*{\LWR@newlinebr}{\unskip\LWR@htmltag{br /}\LWR@orignewline}%
3946 \let\newline\LWR@newlinebr

```

`\\` Redefined to `\LWR@endoffline` or `\LWR@tabularendoffline`.

`\LWR@endoffline` \* `[{len}]`

`\\` is assigned to `\LWR@endoffline` at `\LWR@LwarpStart`.

Inside `tabular`, `\\` is temporarily changed to `\LWR@tabularendoffline`.

```

3947 \NewDocumentCommand{\LWR@endoffline}{s o}
3948 {%
3949 \newline%
3950 }

```

`\LWR@minipagestartpars` Minipages are often placed side-by-side inside figures, with a bit of horizontal space to separate them. Since HTML does not allow a `div` to be inside a `p`, paragraphs must be turned off during the generation of the minipage, then turned on after the minipage is complete. When this occurs between side-by-side minipages, `lwarp` correctly suppresses the paragraph tags between the minipages, unless some other text is between the minipages. Such text forms its own paragraph, resulting in text after a minipage to be on its own line. Since people often place small horizontal

```

\hspace
\enskip
\quad
\qquad

```



space between minipages, it is desirable to maintain this space if possible. `lwarp` tries to do this by remembering that a minipage has been seen, in which case paragraph tags are suppressed around `\hspace`, `\enskip`, `\quad`, and `\qquad` until the end of the paragraph, when the closing `p` tag is created.

When a minipage is seen, the boolean `LWR@minipagethispar` is set, telling the following horizontal whitespace commands to try to suppress their surrounding paragraph tags. `LWR@minipagethispar` is cleared at the next end of paragraph, when the HTML paragraph closing tag is generated.

Placed just before `\hspace`, `\quad`, or `\qquad`'s HTML output.

```
3951 \newcommand*{\LWR@minipagestartpars}{%
3952 \ifbool{LWR@minipagethispar}%
3953 {%
3954 \LWR@startpars%
3955 }{}%
3956 }
```

`\LWR@minipagestoppars` Placed just after `\hspace`, `\quad`, or `\qquad`'s HTML output.

```
3957 \newcommand*{\LWR@minipagestoppars}{%
3958 \ifbool{LWR@minipagethispar}%
3959 {%
3960 \LWR@stoppars%
3961 }{}%
3962 }
```

`\quad` Handles special minipage & horizontal space interactions.

```
3963 \renewcommand*{\quad}{%
3964 \LWR@minipagestoppars%
3965 \HTMLUnicode{2001}%
3966 \LWR@minipagestartpars%
3967 }
```

`\qquad` Handles special minipage & horizontal space interactions.

```
3968 \renewcommand*{\qquad}{\quad\quad}
```

`\enskip` Handles special minipage & horizontal space interactions.

```
3969 \renewcommand*{\enskip}{%
3970 \LWR@minipagestoppars%
3971 \HTMLUnicode{2000}%
3972 \LWR@minipagestartpars%
3973 }
```

```

Len  \WR@tempwidth  Used to compute span width, height, raise for \hspace and \rule:
Len  \WR@tempheight 3974 \newlength{\LWR@tempwidth}
Len  \WR@tempraise  3975 \newlength{\LWR@tempheight}
                        3976 \newlength{\LWR@tempraise}

```

```
\LWR@hspace * {\length}
```

Handles special minipage & horizontal space interactions.

Prints a span of a given width. Ignores the optional star.

\fill is zero-width, so \hspace{\fill} is ignored.

```

3977
3978 \NewDocumentCommand{\LWR@hspace}{s m}{%
3979 \setlength{\LWR@tempwidth}{#2}%

```

Only if the width is not zero:

```
3980 \ifthenelse{\dimtest{\LWR@tempwidth}={}\Opt}}{\}{%
```

If had a minipage this paragraph, try to inline the white space without generating paragraph tags:

```
3981 \LWR@minipagestoppars%
```

Support the HTML thin wrappable space:

```

3982 \ifthenelse{\dimtest{\LWR@tempwidth}={}\{.16667em}}
3983 {%
3984 \HTMLUnicode{2009}% thin breakable space
3985 }%

```

Print the span with the converted width. Not rounded.

```

3986 {%
3987 \uselengthunit{PT}%
3988 \LWR@htmltagc{%
3989 span style="width:\printlength{\LWR@tempwidth}; display:inline-block;"%
3990 }%
3991 \LWR@htmltagc{/span}%
3992 }%

```

If had a minipage this paragraph, try to inline the white space without generating paragraph tags:

```
3993 \LWR@minipagestartpars%
```

```
3994 }%
3995 }
```

`\hspace` \*  $\{\langle length \rangle\}$

Handles special minipage & horizontal space interactions.

```
3996 \let\hspace\LWR@hspace
```

`\linebreak` [ $\langle num \rangle$ ] Inserts an HTML `br` tag.

```
3997 \renewcommand*{\linebreak}[1] [] {\newline}
```

`\nolinebreak` [ $\langle num \rangle$ ]

```
3998 \renewcommand*{\nolinebreak}[1] [] {}
```

`\pagebreak` [ $\langle num \rangle$ ] Starts a new paragraph.

```
3999 \renewcommand*{\pagebreak}[1] [] {
4000
4001 }
```

`\nopagebreak` [ $\langle num \rangle$ ]

```
4002 \renewcommand*{\nopagebreak}[1] [] {}
```

`\enlargethispage` \*  $\{\langle len \rangle\}$

```
4003 \RenewDocumentCommand{\enlargethispage}{s m}{}%
```

`\LWR@rule` [ $\langle raise \rangle$ ]  $\{\langle width \rangle\}$   $\{\langle height \rangle\}$

Handles special minipage & horizontal space interactions.

Creates a span of a given width and height. Ignores the optional star.

`\fill` is zero-width, so `\hspace{\fill}` is ignored.

```
4004 \NewDocumentCommand{\LWR@rule}{o m m}{}%
```

The width is copied into a temporary L<sup>A</sup>T<sub>E</sub>X length, from which comparisons and conversions may be made:

```
4005 \setlength{\LWR@tempwidth}{#2}%
```

If it's zero-width then skip the entire rule:

```
4006 \ifthenelse{\lengthtest{\LWR@tempwidth=0pt}}
4007 {}% zero- width
4008 {% non-zero width
```

If it's non-zero width, set a minimal thickness so that it more reliably shows in the browser:

```
4009 \ifthenelse{\lengthtest{\LWR@tempwidth>0pt}\AND%
4010 \lengthtest{\LWR@tempwidth<1pt}}%
4011 {\setlength{\LWR@tempwidth}{1pt}}{}%
```

Likewise with height:

```
4012 \setlength{\LWR@tempheight}{#3}%
4013 \ifthenelse{\lengthtest{\LWR@tempheight>0pt}\AND%
4014 \lengthtest{\LWR@tempheight<1pt}}%
4015 {\setlength{\LWR@tempheight}{1pt}}{}%
```

If had a minipage this paragraph, try to inline the rule without generating paragraph tags:

```
4016 \LWR@minipagestoppars%
```

Print the span with the converted width and height. The width and height are NOT rounded, since a height of less than 1pt is quite common in L<sup>A</sup>T<sub>E</sub>X code.

```
4017 \uselengthunit{PT}%
4018 \LWR@htmltagc{%
4019 span
4020 style=" %
```

The background color is used to draw the filled rule. The color may be changed by `\textcolor`.

```
4021 background:\LWR@currenttextcolor; %
```

The width and height are printed, converted to PT:

```
4022 width:\printlength{\LWR@tempwidth}; %
4023 height:\printlength{\LWR@tempheight}; %
```

The raise height is converted to a CSS transform. The `*2` raise multiplier is to approximately match HTML output's X height. Conversion to a L<sup>A</sup>T<sub>E</sub>X length allows a typical L<sup>A</sup>T<sub>E</sub>X expression to be used as an argument for the raise, whereas printing the raise argument directly to HTML output without conversion to a L<sup>A</sup>T<sub>E</sub>X length

limits the allowable syntax. To do: A superior method would compute a ratio of L<sup>A</sup>T<sub>E</sub>X ex height, then print that to HTML with an ex unit.

```

4024 \IfValueTF{#1}%
4025 {%
4026 \setlength{\LWR@tempraise}{Opt-#1}%
4027 \setlength{\LWR@tempraise}{\LWR@tempraise*2}%
4028 \LWR@orignewline%
4029 -ms-transform: translate(0pt,\printlength{\LWR@tempraise}); %
4030 \LWR@orignewline%
4031 -webkit-transform: translate(0pt,\printlength{\LWR@tempraise}); %
4032 \LWR@orignewline%
4033 transform: translate(0pt,\printlength{\LWR@tempraise}); %
4034 \LWR@orignewline%
4035 }{}%
```

Display inline-block to place the span inline with the text:

```

4036 display:inline-block;"%
4037 }%
4038 \LWR@htmltagc{/span}%
```

If had a minipage this paragraph, try to inline the white space without generating paragraph tags:

```

4039 \LWR@minipagestartpars%
4040 }% non-zero width
4041 }
```

`\rule` [*raise*] {*width*} {*height*}

Handles special minipage & horizontal space interactions.

```

4042 \let\rule\LWR@rule
4043 \end{warpHTML}
```

## 64 \phantomsection

for HTML output: 4044 \begin{warpHTML}

`\phantomsection` Emulate the hyperref `\phantomsection` command, often used to insert the bibliography into table of contents:

```

4045 \newcommand*{\phantomsection}{\section*{}}
```

```
4046 \end{warpHTML}
```

## 65 \LaTeX and other logos

Logos for HTML and print modes:

Some of these logos may be redefined in a later package, so after loading other packages, and at the beginning of the document, their definitions are finally `\let` in `\LWR@LwarpStart`.

For CSS conversions, see:

<http://edward.oconnor.cx/2007/08/tex-poshlet>

<http://nitens.org/taraborelli/texlogo>

### 65.1 HTML logos

**for HTML output:** 4047 `\begin{warpHTML}`

`\TeX`    $\mathrm{T}_{\mathrm{E}}\mathrm{X}$

`latexlogo` is a CSS class used to properly typeset the E and A in  $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$  and friends.

`latexlogofont` is a CSS class used to select the font for the rest of the logo in  $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ ,  $\mathrm{L}^{\mathrm{u}}\mathrm{a}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ ,  $\mathrm{C}^{\mathrm{o}}\mathrm{n}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ t, etc.

```
4048 \newcommand*{\LWR@TeX}
4049 {\InlineClass{latexlogofont}%
4050 {\InlineClass{latexlogo}{T\textsubscript{e}X}}}
```

`\LaTeX`    $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ ,  $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}_{2\epsilon}$

`\LaTeXe`

```
4051 \newcommand*{\LWR@LaTeX}
4052 {\InlineClass{latexlogofont}%
4053 {\InlineClass{latexlogo}%
4054 {L\textsuperscript{a}T\textsubscript{e}X}}}%
4055
4056 \renewcommand*{\LaTeXe}
4057 {\LaTeX\InlineClass{latexlogofont}%
4058 {\,2\textsubscript{\textit{\HTMLUnicode{3B5}}}}}
```

`\LuaTeX`    $\mathrm{L}^{\mathrm{u}}\mathrm{a}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ ,  $\mathrm{L}^{\mathrm{u}}\mathrm{a}\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$

`\LuaLaTeX`

```
4059 \newcommand*{\LWR@LuaTeX}{\InlineClass{latexlogofont}{Lua}\TeX}
4060 \newcommand*{\LWR@LuaLaTeX}{\InlineClass{latexlogofont}{Lua}\LaTeX}
```

`\XeTeX`  $X_{\text{E}}\text{TeX}$ ,  $X_{\text{E}}\text{L}^{\text{A}}\text{TeX}$

`\XeLaTeX` `xetexlogo` is a CSS class which aligns the backwards E in  $X_{\text{E}}\text{TeX}$  and spaces  $\text{TeX}$  appropriately.

`xelatexlogo` is a CSS class which aligns the backwards E in  $X_{\text{E}}\text{L}^{\text{A}}\text{TeX}$  and spaces  $\text{L}^{\text{A}}\text{TeX}$  appropriately.

```
4061 \newcommand*{\Xe}
4062   {X\textsubscript{\HTMLUnicode{18e}}}
4063 \newcommand*{\LWR@XeTeX}{\InlineClass{xetexlogo}{\Xe}\TeX}
4064 \newcommand*{\LWR@XeLaTeX}{\InlineClass{xelatexlogo}{\Xe}\LaTeX}
```

`\ConTeXt`  $\text{ConT}_{\text{E}}\text{Xt}$

```
4065 \newcommand*{\LWR@ConTeXt}
4066 {\InlineClass{latexlogofont}{Con}\TeX{}}%
4067 \InlineClass{latexlogofont}{t}}
```

`\BibTeX`  $\text{BIBT}_{\text{E}}\text{X}$ , *MakeIndex*

```
\MakeIndex
4068 \providecommand*{\BibTeX}
4069 {\InlineClass{latexlogofont}{B\textsc{ib}}\TeX}
4070
4071 \newcommand*{\MakeIndex}
4072 {\InlineClass{latexlogofont}{\textit{MakeIndex}}}
```

`\AmS`  $\mathcal{A}\mathcal{M}\mathcal{S}$

`amslogo` is a CSS class used for the  $\mathcal{A}\mathcal{M}\mathcal{S}$ logo.

```
4073 \AtBeginDocument{\DeclareDocumentCommand{\AmS}{}}
4074 {\InlineClass{amslogo}{\textit{A\textsubscript{M}S}}}
```

`\MiKTeX`  $\text{MiKT}_{\text{E}}\text{X}$

```
4075 \newcommand*{\MiKTeX}{\InlineClass{latexlogofont}{MiK}\TeX}
```

`\LyX`  $\text{LyX}$

`lyxlogo` is a CSS class used for the  $\text{LyX}$ logo.

```
4076 \newcommand*{\LyX}{\InlineClass{lyxlogo}{LyX}}
```

```
4077 \end{warpHTML}
```

## 65.2 Print logos

for PRINT output: 4078 `\begin{warpprint}`  
 4079 `\newcommand*{\XeTeXrevE}`  
 4080 `{\hspace{-.1667em}\raisebox{-.5ex}{\reflectbox{E}}\hspace{-.125em}}`  
 4081 `\providecommand*{\XeTeX}{\mbox{X\XeTeXrevE\TeX}}`  
 4082 `\providecommand*{\XeLaTeX}{\mbox{X\XeTeXrevE\LaTeX}}`  
 4083 `\providecommand*{\AmS}{\%`  
 4084 `\leavevmode\hbox{$\mathcal A\kern-.2em\lower.376ex%`  
 4085 `\hbox{$\mathcal M$}\kern-.2em\mathcal S$}}`  
 4086 `\newcommand*{\LyX}{\textsf{LyX}}`  
 4087 `\providecommand*{\LuaTeX}{\mbox{Lua\TeX}}`  
 4088 `\providecommand*{\LuaLaTeX}{\mbox{Lua\LaTeX}}`  
 4089 `\providecommand*{\BibTeX}{\mbox{B\textsc{ib}\TeX}}`  
 4090 `\providecommand*{\MakeIndex}{\mbox{\textit{MakeIndex}}}`  
 4091 `\providecommand*{\ConTeXt}{\mbox{Con\TeXt}}`  
 4092 `\providecommand*{\MiKTeX}{\mbox{MiK\TeX}}`  
 4093 `\end{warpprint}`

## 66 `\AtBeginDocument`, `\AtEndDocument`

for HTML output: 4094 `\begin{warppHTML}`

`\LWR@LwarpStart` Automatically sets up the HTML-related actions for the start and end of the  
`\LWR@LwarpEnd` document.

4095 `\AfterEndPreamble{\LWR@LwarpStart}`  
 4096 `\AtEndDocument{\LWR@LwarpEnd}`  
 4097 `\end{warppHTML}`



## 67 lwarpmk.conf

File `lwarpmk.conf` `lwarpmk.conf` is automatically (re-)created by the `lwarp` package when executing `pdflatex <project.tex>`, or similar for `xelatex` or `lualatex`, in print-document generation mode, which is the default unless the `warpHTML` option is given. `lwarpmk.conf` is then used by the utility `lwarpmk`.

An example `lwarpmk.conf`:

---

```
opsystem = "Unix"    -- or "Windows"
latexname = "pdflatex" -- or "lualatex" or "xelatex"
sourcename = "projectname" -- your .tex source
homehtmlfilename = "index" -- or "projectname"
htmlfilename = ""    -- or "projectname" if numbered HTML files
```

---

Defaults unless already over-ridden by the user:

```
4098 \providecommand*\HomeHTMLFileName{\BaseJobname}
4099 \providecommand*\HTMLFileName{}
4100 \providecommand*\UseLatexmk{false}
```

for PRINT output:

```
4101 \begin{warpprint}
4102 \ifcsdef{LWR@file}{\newwrite{LWR@file}}
4103 \immediate\openout{LWR@file}=lwarpmk.conf
4104 \ifwindows
4105 \immediate\write{LWR@file}{opsystem = "Windows"}
4106 \else
4107 \immediate\write{LWR@file}{opsystem = "Unix"}
4108 \fi
4109 \ifPDFTeX
4110 \immediate\write{LWR@file}{latexname = "pdflatex"}
4111 \fi
4112 \ifXeTeX
4113 \immediate\write{LWR@file}{latexname = "xelatex"}
4114 \fi
4115 \ifLuaTeX
4116 \immediate\write{LWR@file}{latexname = "lualatex"}
4117 \fi
4118 \immediate\write{LWR@file}{sourcename = "\jobname"}
4119 \immediate\write{LWR@file}{%
4120 homehtmlfilename = "\HomeHTMLFileName"%
4121 }
4122 \immediate\write{LWR@file}{htmlfilename = "\HTMLFileName"}
4123 \immediate\write{LWR@file}{uselatexmk = "\UseLatexmk"}
4124 \immediate\closeout{LWR@file}
4125 \end{warpprint}
```

## 68 Trademarks

- Adobe® and Adobe Framemaker® are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.
- Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.
- Mac OS® is a trademark of Apple Inc.
- MadCap Flare™ is the property of MadCap Software, Inc.
- MathJax is copyright 2009 and later. The MathJax Consortium is a joint venture of the American Mathematical Society (AMS) and the Society for Industrial and Applied Mathematics (SIAM) to advance mathematical and scientific content on the web.
- Microsoft®, Encarta, MSN, and Windows® are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
- UNIX® is a registered trademark of The Open Group.

## Package 2

# lwarp-newproject.sty

## 69 lwarp-newproject

Pkg	<b>lwarp-newproject</b>	lwarp-newproject is used to create lwarp-specific system files in a new project.
Opt lwarp-newproject	<b>warpprint</b>	Configuration files are only created if the <b>warpprint</b> option was used. When this is the case, lwarp-newproject knows that <code>\jobname</code> is the source code's filename (as opposed to the <code>&lt;project&gt;_html.tex</code> filename).
Opt lwarp-newproject	<b>warpHTML</b>	Does not create configuration files. <b>warpprint</b> or <b>warpHTML</b> are passed to both lwarp and lwarp-newproject by lwarpmk, depending on whether lwarpmk <b>print</b> or <b>html</b> was used.
Opt lwarp-newproject	<b>BaseJobName</b>	<code>BaseJobname=somename</code> sets <code>\BaseJobname</code> to <code>somename</code> , which is used to set <code>\HomeHTMLFileName</code> , which is then written to the <code>lwarpmk.conf</code> and <code>&lt;project&gt;.lwarpmkconf</code> configuration files.  <code>\BaseJobname</code> is also written to <code>&lt;project&gt;_html.tex</code> as an option given for the lwarp and lwarp-newproject packages during the creation of HTML output.  <code>\BaseJobname</code> is the <code>\jobname</code> of the printed version, even while compiling the HTML version, where <code>\jobname</code> has <code>_html</code> appended.
Opt lwarp-newproject	<b>lwarpmk</b>	Tells lwarp-newproject to generate a local copy of lwarpmk called <code>lwarpmk.lua</code> . Useful for archiving for future use. This file may be made executable and acts just like <code>lwarpmk</code> .

In the document source:

---

```

\documentclass{article} % or book, report
...
(font selection, input encoding)
...
\newcommand*{\HomeHTMLFileName}{index}, or {projectname}
\newcommand*{\HTMLFileName}{}
\usepackage{lwarp-newproject}
\usepackage{lwarp}
...
(load other packages, the rest of the preamble)
...
\NewCSS{project.css}
\NewHTMLdescription{Summarize this webpage.}

```

---

```

...
\begin{document}
...
\end{document}

```

---

Place `\usepackage{lwarp-newproject}` just before `\usepackage{lwarp}` in the document source. The operating system and  $\TeX$  engine will be auto-detected, and a customized `lwarpmk.conf` file will be generated, along with the other support files (`*.css`, `lwarp_html.xdy`, `lwarp_mathjax.txt`). Rename `sample_project.css` to your own custom `projectname.css` and edit it if desired.

After the first compile, `\usepackage{lwarp-newproject}` may be commented out unless the configuration changes. For example, it may be re-enabled to switch between `pdflatex`, `xelatex`, and `lualatex`, in which case a new manual compile (without using `lwarpmk`) will auto-configure and regenerate the files. It may also be left enabled at all times, in which case the configuration files will be regenerated each time. Note that your own `projectname.css` file will not be over-written by the newly regenerated `sample_project.css`.

Discard all options for `lwarp-newproject`:

```

1 \ProvidesPackage{lwarp-newproject}
2
3 \RequirePackage{etoolbox}
4 \RequirePackage{comment}
5 \RequirePackage{fancyvrb}
6 \RequirePackage{ifplatform}% sense op-system platform
7 \RequirePackage{iftex}% sense pdflatex/lualatex/xelatex
8 \RequirePackage{kvoptions}
9 \SetupKeyvalOptions{family=LWRNP,prefix=LWRNP@}

```

Generate config files?

```

10 \newbool{LWRNP@genconf}
11 \boolfalse{LWRNP@genconf}

```

Optionally generate a local copy of `lwarpmk`. Default to no:

```

12 \excludecomment{LWR@createlwarpmk}

```

Process options:

```

13 \DeclareVoidOption{warpprint}{\booltrue{LWRNP@genconf}}
14 \DeclareVoidOption{warpHTML}{\boolfalse{LWRNP@genconf}}
15 \DeclareVoidOption{lwarpmk}{\includecomment{LWR@createlwarpmk}}
16 \DeclareStringOption[\jobname]{BaseJobname}
17

```

```

18 \booltrue{LWRNP@genconf}% warpprint
19
20 \ProcessKeyvalOptions*\relax

```

Assign the `\BaseJobname` if the user hasn't provided one:

```

21 \providecommand*{\BaseJobname}{\LWRNP@BaseJobname}

```

Defaults unless already over-ridden by the user:

```

22 % \ifbool{LWRNP@genconf}{
23 \providecommand*{\HomeHTMLFileName}{\BaseJobname}
24 % }{}
25 \providecommand*{\HTMLFileName}{%
26 \providecommand*{\UseLatexmk}{false}

```

## 69.1 project\_html.tex

File `project_html.tex` Used to allow an HTML version of the document to exist alongside the print version.

Only write `\jobname_html.tex` if generating the print version.

```

27 \ifbool{LWRNP@genconf}{
28 \ifcsdef{LWR@file}{\newwrite{LWR@file}}
29 \immediate\openout{LWR@file}=\jobname_html.tex
30 \immediate\write{LWR@file}{%
31 \detokenize{\PassOptionsToPackage}%
32 {warpHTML,BaseJobname=\jobname}{lwarp}%
33 }
34 \immediate\write{LWR@file}{%
35 \detokenize{\PassOptionsToPackage}%
36 {warpHTML,BaseJobname=\jobname}{lwarp-newproject}%
37 }
38 \immediate\write{LWR@file}{%
39 \detokenize{\input}\string{\jobname.tex}\string }%
40 }
41 \immediate\closeout{LWR@file}
42 }{}

```

## 69.2 project.lwarpmkconf

File `project.lwarpmkconf` The configuration file for `lwarpmk`.

```

43 \ifbool{LWRNP@genconf}{
44 \ifcsdef{LWR@file}{\newwrite{LWR@file}}

```

```

45 \immediate\openout\LWR@file=\jobname.lwarpmkconf
46 \ifwindows
47 \immediate\write\LWR@file{opsystem = "Windows"}
48 \else
49 \immediate\write\LWR@file{opsystem = "Unix"}
50 \fi
51 \ifPDFTeX
52 \immediate\write\LWR@file{latexname = "pdflatex"}
53 \fi
54 \ifXeTeX
55 \immediate\write\LWR@file{latexname = "xelatex"}
56 \fi
57 \ifLuaTeX
58 \immediate\write\LWR@file{latexname = "lualatex"}
59 \fi
60 \immediate\write\LWR@file{sourcename = "\jobname"}
61 \immediate\write\LWR@file{%
62 homehtmlfilename = "\HomeHTMLFileName"%
63 }
64 \immediate\write\LWR@file{htmlfilename = "\HTMLFileName"}
65 \immediate\write\LWR@file{uselatemk = "\UseLatexmk"}
66 \immediate\closeout\LWR@file
67 }{}

```

### 69.3 lwarp.css

File `lwarp.css` This is the base CSS layer used by lwarp.

This must be present both when compiling the project and also when distributing the HTML files.

```

68 \begin{VerbatimOut}{lwarp.css}
69 /*
70  CSS stylesheet for the LaTeX lwarp package
71  Copyright 2016 Brian Dunn -- BD Tech Concepts LLC
72 */
73
74
75 /* a fix for older browsers: */
76 header, section, footer, aside, nav, main,
77  article, figure { display: block; }
78
79
80 A:link {color:#000080 ; text-decoration: none ; }
81 A:visited {color:#800000 ; }
82 A:hover {color:#000080 ; text-decoration: underline ;}
83 A:active {color:#800000 ; }

```

```
84
85 a.tocpart {display: inline-block ; margin-left: 0em ;
86     font-weight: bold ;}
87 a.tocchapter {display: inline-block ; margin-left: 0em ;
88     font-weight: bold ;}
89 a.tocsection {display: inline-block ; margin-left: 1em ;
90     text-indent: -.5em ; font-weight: bold ; }
91 a.tocsubsection {display: inline-block ; margin-left: 2em ;
92     text-indent: -.5em ; }
93 a.tocsubsubsection {display: inline-block ; margin-left: 3em ;
94     text-indent: -.5em ; }
95 a.tocparagraph {display: inline-block ; margin-left: 4em ;
96     text-indent: -.5em ; }
97 a.tocsubparagraph {display: inline-block ; margin-left: 5em ;
98     text-indent: -.5em ; }
99 a.tocfigure {margin-left: 0em}
100 a.tocsubfigure {margin-left: 2em}
101 a.toctable {margin-left: 0em}
102 a.tocsubtable {margin-left: 2em}
103 a.toclstlisting {margin-left: 0em}
104
105
106 body {
107     font-family: "DejaVu Serif", "Bitstream Vera Serif",
108         "Lucida Bright", Georgia, serif;
109     background: #FAF7F4 ;
110     color: black ;
111     margin: 0em ;
112     padding: 0em ;
113     font-size: 100% ;
114     line-height: 1.2 ;
115 }
116
117 p {margin: 1.5ex 0em 1.5ex 0em ;}
118
119 /* Holds a section number to add space between it and the name */
120 span.sectionnumber { margin-right: .6em }
121
122 /* Inserted in front of index lines */
123 span.indexitem {margin-left: 0em}
124 span.indexsubitem {margin-left: 2em}
125 span.indexsubsubitem {margin-left: 4em}
126
127 div.hidden { display: none ; }
128
129 kbd {
130     font-family: "DejaVu Mono", "Bitstream Vera Mono", "Lucida Console",
131         "Nimbus Mono L", "Liberation Mono", "FreeMono", "Andale Mono",
132         "Courier New", monospace;
133     font-size: 100% ;
```

```
134 }
135
136 span.strong { font-weight: bold; }
137
138 span.textmd { font-weight: normal; }
139
140 span.textsc { font-variant: small-caps; }
141
142 span.textup { font-variant: normal; }
143
144 span.textrm {
145     font-family: "DejaVu Serif", "Bitstream Vera Serif",
146     "Lucida Bright", Georgia, serif;
147 }
148
149 span.textsf {
150     font-family: "DejaVu Sans", "Bitstream Vera Sans",
151     Geneva, Verdana, sans-serif ;
152 }
153
154 span.attribution {
155     margin-left: 1em ; font-size: 80% ; font-variant: small-caps;
156 }
157
158 span.citetitle {
159     margin-left: 1em ; font-size: 80% ; font-style: oblique;
160 }
161
162 span.poemtitle {
163     font-size: 120% ; font-weight: bold;
164 }
165
166 blockquote {
167     margin-left: 0px ;
168     margin-right: 0px ;
169 }
170
171 blockquote p {
172     line-height: 1.5;
173     text-align: left ;
174     font-size: .85em ;
175     margin-left: 3em ;
176     margin-right: 3em ;
177 }
178
179 blockquotation {
180     margin-left: 0px ;
181     margin-right: 0px ;
182 }
183
```



```
184 blockquotation p {
185   line-height: 1.5;
186   text-align: left ;
187   font-size: .85em ;
188   margin-left: 3em ;
189 margin-right: 3em ;
190 }
191
192 div.epigraph {
193   line-height: 1.2;
194   text-align: left ;
195   padding: 3ex 1em 0ex 1em ;
196 /*   margin: 3ex auto 3ex auto ; */ /* Epigraph centered */
197   margin: 3ex 1em 3ex auto ; /* Epigraph to the right */
198 /*   margin: 3ex 1em 3ex 1em ; */ /* Epigraph to the left */
199   font-size: .85em ;
200   max-width: 27em ;
201 }
202
203
204
205 div.epigraphsource{
206   text-align:right ;
207   margin-left:auto ;
208 /*   max-width: 50% ; */
209   border-top: 1px solid #A0A0A0 ;
210   padding-bottom: 3ex ;
211   line-height: 1.2;
212 }
213
214 div.epigraph p { padding: .5ex ; margin: 0ex ;}
215 div.epigraphsource p { padding: .5ex 0ex 0ex 0ex ; margin: 0ex ;}
216
217
218
219
220 html body {
221   margin: 0 ;
222   line-height: 1.2;
223 }
224
225
226 body div {
227   margin: 0ex;
228 }
229
230
231 h1, h2, h3, h4, h5, h6, span.paragraph, span.subparagraph
232 {
233   font-family: "Linux Libertine O", "Hoefler Text", "Garamond",
```

```
234         "Bembo", "Janson", "TeX Gyre Pagella", "Palatino",
235         "Liberation Serif", "Nimbus Roman No 9 L", "FreeSerif", Times,
236         "Times New Roman", serif;
237     font-style: normal ;
238     font-weight: bold ;
239     text-align: left ;
240 }
241
242 h1 { /* title of the entire website, used on each page */
243     text-align: center ;
244     font-size: 2.5em ;
245     padding: .4ex 0em 0ex 0em ;
246 }
247 h2 { font-size: 2.25em }
248 h3 { font-size: 2em }
249 h4 { font-size: 1.75em }
250 h5 { font-size: 1.5em }
251 h6 { font-size: 1.25em }
252 span.paragraph {font-size: 1em ; font-variant: normal ;
253     margin-right: 1em ; }
254 span.subparagraph {font-size: 1em ; font-variant: normal ;
255     margin-right: 1em ; }
256
257
258
259 /* Title of the file */
260 h1 {
261     margin: 0ex 0em 0ex 0em ;
262     line-height: 1.3;
263     text-align: center ;
264 }
265
266 /* Part */
267 h2 {
268     margin: 1ex 0em 1ex 0em ;
269     line-height: 1.3;
270     text-align: center ;
271 }
272
273 /* Chapter */
274 h3 {
275     margin: 3ex 0em 1ex 0em ;
276     line-height: 1.3;
277 }
278
279 /* Section */
280 h4 {
281     margin: 3ex 0em 1ex 0em ;
282     line-height: 1.3;
283 }
```

```
284
285 /* Sub-Section */
286 h5 {
287     margin: 3ex 0em 1ex 0em ;
288     line-height: 1.3;
289 }
290
291 /* Sub-Sub-Section */
292 h6 {
293     margin: 3ex 0em 1ex 0em ;
294     line-height: 1.3;
295 }
296
297
298 div.titlepage {
299     text-align: center ;
300 }
301
302 .footnotes {
303     font-size: .85em ;
304     margin: 3ex 1em 0ex 1em ;
305     padding-bottom: 1ex ;
306 }
307
308 .marginpar {
309     width:20%; float:right;
310     text-align:left;
311     margin: 1ex 0.5em 1ex 1em ;
312     padding: 1ex 0.5em 1ex 0.5em ;
313     font-size: 85% ;
314     border-top: 1px solid silver ;
315     border-bottom: 1px solid silver ;
316     overflow-x: auto;
317 }
318
319 .marginpar br { margin-bottom: 2ex ; }
320
321 div.marginblock {
322     width:50%; float:right;
323     text-align:left;
324     margin: 1ex 0.5em 1ex 1em ;
325     padding: 1ex 0.5em 1ex 0.5em ;
326     overflow-x: auto;
327 }
328
329 div.marginblock div.minipage p { font-size: 85%}
330
331 div.marginblock br { margin-bottom: 2ex ; }
332
333
```

```
334 section.textbody div.footnotes{
335     margin: 3ex 0em 0ex 0em ;
336     border-bottom: 2px solid silver ;
337 }
338
339 .footnoteheader {
340     border-top: 2px solid silver ;
341     margin-top: 3ex ;
342     padding-top: 1ex ;
343     font-weight: bold ;
344 }
345
346 .mpfootnotes {
347     text-align: left ;
348     font-size: .85em ;
349     margin: 3ex 0em 3ex 0em ;
350     border-top: 1px solid silver ;
351     border-bottom: 1px solid silver ;
352 /*     padding-top: 1ex ; */
353 }
354
355 /* Remove footnote top border in the title page. */
356 div.titlepage div.mpfootnotes {
357     border-top: none ;
358 }
359
360
361
362 ol {
363     margin: 1ex 1em 1ex 0em;
364     line-height: 1.2;
365 }
366
367 ul, body dir, body menu {
368     margin: 1ex 1em 1ex 0em;
369     line-height: 1.2;
370 }
371
372 li { margin: 0ex 0em 1ex 0em; }
373
374 html {
375     margin: 0;
376     padding: 0;
377 }
378
379 .programlisting {
380     font-family: "DejaVu Mono", "Bitstream Vera Mono", "Lucida Console",
381                 "Nimbus Mono L", "Liberation Mono", "FreeMono", "Andale Mono",
382                 "Courier New", monospace;
383     margin: 1ex 0ex 1ex 0ex ;
```

```
384 padding: .5ex 0pt .5ex 0pt ;
385 overflow-x: auto;
386 }
387
388 section.textbody>pre.programlisting {
389 border-top: 1px solid silver ;
390 border-bottom: 1px solid silver ;
391 }
392
393
394 .inlineprogramlisting {
395 font-family: "DejaVu Mono", "Bitstream Vera Mono", "Lucida Console",
396             "Nimbus Mono L", "Liberation Mono", "FreeMono", "Andale Mono",
397             "Courier New", monospace;
398 overflow-x: auto;
399 }
400
401
402 div.abstract {
403 margin: 2em 5% 2em 5% ;
404 padding: 1ex 1em 1ex 1em ;
405 /* font-weight: bold ; */
406 font-size: 90% ;
407 }
408
409 div.abstract dl {line-height:1.5;}
410 div.abstract dt {color:#304070;}
411
412 div.abstracttitle{
413 font-family: "URW Classico", Optima, "Linux Biolinum O",
414             "Linux Libertine O", "Liberation Serif", "Nimbus Roman No 9 L",
415             "FreeSerif", "Hoefler Text", Times, "Times New Roman", serif;
416 font-weight:bold;
417 font-size:1.25em;
418 text-align: center ;
419 }
420
421
422 .verbatim {
423 overflow-x: auto ;
424 }
425
426 .alltt {
427 overflow-x: auto ;
428 }
429
430
431 .bverbatim {
432 margin: 1ex 0pt 1ex 0pt ;
433 padding: .5ex 0pt .5ex 0pt ;
```

```
434     overflow-x: auto ;
435 }
436
437 .lverbatim {
438     margin: 1ex 0pt 1ex 0pt ;
439     padding: .5ex 0pt .5ex 0pt ;
440     overflow-x: auto ;
441 }
442
443 .fancyvrb {
444     font-size:.85em ;
445     margin: 3ex 0pt 3ex 0pt
446 }
447
448 .fancyvrblabel {
449     font-weight:bold;
450     text-align: center ;
451 }
452
453
454 .verse {
455     font-family: "Linux Libertine Mono O", "Lucida Console",
456                 "Droid Sans Mono", "DejaVu Mono", "Bitstream Vera Mono",
457                 "Liberation Mono", "FreeMono", "Andale Mono",
458                 "Nimbus Mono L", "Courier New", monospace;
459     margin-left: 1em ;
460 }
461
462
463 div.singlespace { line-height: 1.2 ; }
464 div.onehalfspace { line-height: 1.5 ; }
465 div.doublespace { line-height: 2 ; }
466
467
468 .theorem {
469     background: #e0e0e0;
470     padding: 1ex ;
471     margin: 1em ;
472     border: 1px solid silver ;
473 }
474
475 .theoremlabel { font-weight: bold ; }
476
477 .theoremcontents { font-style: italic; margin-left: 1em ; }
478
479
480 /* Minipage environments, vertically aligned to top, center, bottom: */
481 .minipage {
482     /* display: inline-block ; */
483     /* Mini pages which follow each other will be tiled. */
```

```
484     margin: .25em .25em .25em .25em;
485     padding: .25em .25em .25em .25em;
486     display: inline-flex;
487     flex-direction: column ;
488     overflow: auto;
489 }
490
491 /* Paragraphs in the flexbox did not collapse their margins. */
492 /* Have not yet researched this. */
493 .minipage p {margin: .75ex 0em .75ex 0em ;}
494
495
496
497 .framebox {
498     margin: 0ex ;
499     padding: 0ex ;
500     border: 1px solid black;
501     border-radius: 0px ;
502     padding: .3ex .2em 0ex .2em ;
503     margin: .1ex ;
504     display: inline-block ;
505 }
506
507
508 .mdframed {
509 /*     padding: 0ex ; */
510 /*     border: 1px solid black; */
511 /*     border-radius: 0px ; */
512     padding: 0ex ;
513     margin: 3ex 5% 3ex 5% ;
514 /*     display: inline-block ; */
515 }
516
517 .mdframed p { padding: 0ex .5em 0ex .5em ; }
518
519 .mdframed dl { padding: 0ex .5em 0ex .5em ; }
520
521 .mdframedtitle {
522     padding: .5em ;
523     display: block ;
524     font-size: 130%
525 }
526
527 .mdframedsubtitle {
528     padding: 0ex .5em 0ex .5em ;
529     display: block ;
530     font-size: 115% ;
531 }
532
533 .mdframedsubsubtitle {
```

```
534     padding: 0ex .5em 0ex .5em ;
535     display: block ;
536 }
537
538 .mdtheorem {
539     padding: 0ex .5em 0ex .5em ;
540     margin: 3ex 5% 3ex 5% ;
541 /*     display: inline-block ; */
542 }
543
544
545
546
547 dl {
548     margin: 1ex 2em 1ex 0em;
549     line-height: 1.3;
550 }
551
552 dl dt {
553     margin-top: 1ex;
554     font-weight: bold;
555 }
556
557 dl dd p { margin-top: 0em; }
558
559
560 nav.toc, nav.lof, nav.lot, nav.lol {
561     font-family: "URW Classico", Optima, "Linux Biolinum O",
562         "DejaVu Sans", "Bitstream Vera Sans",
563         Geneva, Verdana, sans-serif ;
564     margin-bottom: 4ex ;
565 }
566
567 nav.toc p, nav.lof p, nav.lot p, nav.lol p {
568     line-height: 1.2 ;
569     margin-top:.5ex ;
570     margin-bottom:.5ex;
571     font-size: .9em ;
572 }
573
574
575
576 img, img.hyperimage, img.borderimage {
577     max-width: 600px;
578     border: 1px solid silver;
579     box-shadow: 3px 3px 3px #808080 ;
580     padding: .5% ;
581     margin: .5% ;
582     background: none ;
583 }
```



```
584
585 img.inlineimage{
586     padding: 0px ;
587     box-shadow: none ;
588     border: none ;
589     background: none ;
590     margin: 0px ;
591     display: inline-block ;
592     border-radius: 0px ;
593 }
594
595 img.logoimage{
596     max-width: 300px ;
597     box-shadow: 3px 3px 3px #808080 ;
598     border: 1px solid black ;
599     background:none ;
600     padding:0 ;
601     margin:.5ex ;
602     border-radius: 10px ;
603 }
604
605
606 .section {
607 /*
608     To have each section float relative to each other:
609 */
610 /*
611     display: block ;
612     float: left ;
613     position: relative ;
614     background: white ;
615     border: 1px solid silver ;
616     padding: .5em ;
617 */
618     margin: 0ex .5em 0ex .5em ;
619     padding: 0 ;
620 }
621
622
623 figure {
624     margin: 3ex auto 3ex auto ;
625     padding: 1ex 1em 1ex 1em ;
626     overflow-x: auto ;
627 }
628
629
630 /* To automatically center images in figures: */
631 /*
632 figure img.inlineimage {
633     margin: 0ex auto 0ex auto ;
```

```
634     display: block ;
635 }
636 */
637
638 /* To automatically center minipages in figures: */
639 /*
640 figure div.minipage, figure div.minipage div.minipage {
641     margin: 1ex auto 1ex auto ;
642     display: block ;
643 }
644 */
645
646 figure div.minipage p { font-size: 85% ; }
647
648 figure.subfigure, figure.subtable {
649     display: inline-block ; margin: 3ex 1em 3ex 1em ;
650 }
651
652 div.floatrow { text-align: center; }
653
654 div.floatrow figure { display: inline-block ; margin: 1ex 2% ; }
655
656 div.floatfoot { font-size: .85em ;
657     border-top: 1px solid silver ; line-height: 1.2 ; }
658
659 figcaption , .lstlistingtitle {
660     font-size: .85em ;
661     text-align: center ;
662     font-weight: bold ;
663     margin-top: 1ex ;
664     margin-bottom: 1ex ;
665 }
666
667 figure.subfigure figcaption, figure.subtable figcaption {
668     border-bottom: none ; background: none ;
669 }
670
671 div.nonfloatcaption {
672     margin: 1ex auto 1ex auto ;
673     font-size: .85em ;
674     text-align: center ;
675     font-weight: bold ;
676 }
677
678 /* For a \RawCaption inside a minipage inside a figure's floatrow: */
679 figure div.floatrow div.minipage figcaption {
680     border: none ;
681     background: none ;
682 }
683
```

```

684
685 table {
686     margin: 1ex auto 1ex auto ;
687     border-collapse: collapse ;
688     border-spacing: 0px ;
689     line-height: 1.3 ;
690 }
691
692 tr.hline {border-top: 1px solid silver ; margin-top: 0ex ;
693     margin-bottom: 0ex ; } /* for \hline */
694
695 tr.tbrule {border-top: 1px solid black ; margin-top: 0ex ;
696     margin-bottom: 0ex ; } /* for \toprule, \bottomrule */
697
698 td {padding: 1ex .5em 1ex .5em ;}
699
700 table td.tdl { text-align: left ; vertical-align: middle ; }
701 table td.tdc { text-align: center ; vertical-align: middle ; }
702 table td.tdr { text-align: right ; vertical-align: middle ; }
703 table td.tdp { text-align: left ; vertical-align: bottom ; }
704 table td.tdm { text-align: left ; vertical-align: middle ; }
705 table td.tdb { text-align: left ; vertical-align: top ; }
706 table td.tdP { text-align: center ; vertical-align: bottom ; }
707 table td.tdM { text-align: center ; vertical-align: middle ; }
708 table td.tdB { text-align: center ; vertical-align: top ; }
709 table td.tdlrule { text-align: left ; border-top: 1px solid silver ;
710     vertical-align: middle ; } /* for cmdrule */
711 table td.tdcrule { text-align: center ; border-top: 1px solid silver ;
712     vertical-align: middle ; }
713 table td.tdrrule { text-align: right ; border-top: 1px solid silver ;
714     vertical-align: middle ; }
715 table td.tdprule { text-align: left ; border-top: 1px solid silver ;
716     vertical-align: bottom ; }
717 table td.tdmrule { text-align: left ; border-top: 1px solid silver ;
718     vertical-align: middle ; }
719 table td.tdbrule { text-align: left ; border-top: 1px solid silver ;
720     vertical-align: top ; }
721 table td.tdPrule { text-align: center ; border-top: 1px solid silver ;
722     vertical-align: bottom ; }
723 table td.tdMrule { text-align: center ; border-top: 1px solid silver ;
724     vertical-align: middle ; }
725 table td.tdBrule { text-align: center ; border-top: 1px solid silver ;
726     vertical-align: top ; }
727
728 /* Margins of paragraphs inside table cells: */
729 td.tdp p , td.tdprule p , td.tdP p , td.tdPrule p { padding-top: 1ex ;
730     padding-bottom: 1ex ; margin: 0ex ; }
731 td.tdm p , td.tdrrule p , td.tdM p , td.tdMrule p { padding-top: 1ex ;
732     padding-bottom: 1ex ; margin: 0ex ; }
733 td.tdb p , td.tdbrule p , td.tdB p , td.tdBrule p { padding-top: 1ex ;

```

```
734     padding-bottom: 1ex ; margin: 0ex ; }
735
736 td.tdp , td.tdprule , td.tdP , td.tdPrule
737     { padding: 0ex .5em 0ex .5em ; }
738 td.tdm , td.tdmrule , td.tdM , td.tdMrule
739     { padding: 0ex .5em 0ex .5em ; }
740 td.tdb , td.tdbrule , td.tdB , td.tdBrule
741     { padding: 0ex .5em 0ex .5em ; }
742
743
744 /* table notes: */
745 .tnotes {
746     margin: 0ex 5% 1ex 5% ;
747     padding: 0.5ex 1em 0.5ex 1em;
748     font-size:.85em;
749     text-align: left ;
750 }
751
752 .tnotes dl dt p {margin-bottom:0px;}
753
754 .tnoteitemheader {margin-right: 1em;}
755
756
757
758 /* center, flushleft, flushright environments */
759 div.center{text-align:center;}
760 div.center table {margin-left:auto;margin-right:auto;}
761 div.flushleft{text-align:left;}
762 div.flushleft table {margin-left:0em ; margin-right:auto;}
763 div.flushright{text-align:right;}
764 div.flushright table {margin-left:auto ; margin-right: 0em ;}
765
766
767
768
769 /* program listing callouts: */
770 span.callout {
771     font-family: "DejaVu Sans", "Bitstream Vera Sans",
772         Geneva, Verdana, sans-serif ;
773     border-radius: .5em;
774     background-color:black;
775     color:white;
776     padding:0px .25em 0px .25em;
777     margin: 0 ;
778     font-weight: bold;
779     font-size:.72em ;
780 }
781
782 div.programlisting pre.verbatim span.callout{
783 font-size: .85em ;
```

```
784 }
785
786
787
788
789
790 div.published
791 {
792     text-align: center ;
793     font-variant: normal ;
794     font-style: italic ;
795     font-size: 1em ;
796     margin: 3ex 0em 3ex 0em ;
797 }
798
799 div.subtitle
800 {
801     text-align: center ;
802     font-variant: normal ;
803     font-style: italic ;
804     font-size: 1.25em ;
805     margin: 3ex 0em 3ex 0em ;
806 }
807
808 div.subtitle p { margin: 1ex ; }
809
810 div.author
811 {
812     font-variant: normal ;
813     font-style: normal ;
814     font-size: 1em ;
815     margin: 3ex 0em 3ex 0em ;
816 }
817
818 div.author table {
819     margin: 3ex auto 0ex auto ;
820     background: none ;
821 }
822
823 div.author table tbody tr td { padding: .25ex ; }
824
825 span.affiliation {font-size: .85em ; font-variant: small-caps; }
826
827 div.titledate {
828     text-align: center ;
829     font-size: .85em ;
830     font-style: italic;
831     margin: 6ex 0em 6ex 0em ;
832 }
833
```

```
834
835 nav.topnavigation{
836     text-align: left ;
837     padding: 0.5ex 1em 0.5ex 1em ;
838 /*     margin: 2ex 0em 3ex 0em ; */
839     margin: 0 ;
840     border-bottom: 1px solid silver ;
841     border-top: 1px solid silver ;
842     clear:right ;
843 }
844
845 nav.botnavigation{
846     text-align: left ;
847     padding: 0.5ex 1em 0.5ex 1em ;
848 /*     margin: 3ex 0em 2ex 0em ; */
849     margin: 0 ;
850     border-top: 1px solid silver ;
851     border-bottom: 1px solid silver ;
852     clear:right ;
853 }
854
855
856 header{
857     line-height: 1.2 ;
858     font-size: 1em ;
859 /*     border-bottom: 2px solid silver ; */
860     margin: 0px ;
861     padding: 0ex 1em 0ex 1em ;
862     text-align:center ;
863 }
864
865 header p {margin:0ex;padding:4ex 0em 2ex 0em ;text-align:center;}
866
867
868 footer{
869     font-size: .85em ;
870     line-height: 1.2 ;
871     margin-top: 1ex ;
872     border-top: 2px solid silver ;
873     padding: 2ex 1em 2ex 1em ;
874     clear:right ;
875     text-align:left ;
876 }
877
878
879 a.linkhome { font-weight:bold ; font-size: 1em ;}
880
881
882 div.lateximagesource { padding: 0px ; margin: 0px ; display: none; }
883
```

```
884 img.lateximage{
885     padding: 0px 0px 0px 0px ;
886     box-shadow: none ;
887     border: none ;
888     background: none ;
889     margin: 0px 0px -.15ex 0px ;
890     /* pdfcrop leaves a slight margin, adjust to baseline */
891     max-width: 100% ;
892     border-radius: 0ex ;
893     border: none ;
894 }
895
896
897
898 nav.sidetoc {
899     font-family: "DejaVu Serif", "Bitstream Vera Serif",
900         "Lucida Bright", Georgia, serif;
901     float:right ;
902     width: 20%;
903     border-left: 1px solid silver;
904     border-top: 1px solid silver;
905     border-bottom: 1px solid silver;
906 /*     border-top: 2px solid #808080 ; */
907     background: #FAF7F4 ;
908     padding: 2ex 0em 2ex 1em ;
909     margin: 0ex 0em 2ex 1em ;
910     font-size:.9em ;
911     border-radius: 20px 0px 0px 20px ;
912 }
913
914 div.sidetoccontents {
915 /*     border-top: 1px solid silver ; */
916     overflow-y: auto ;
917     width: 100% ;
918     text-align: left ;
919 }
920
921 nav.sidetoc p {line-height:1.2 ; margin: 1ex .5em 1ex .5em ;
922     text-indent: 0 ; }
923 nav.sidetoc p a {color:black ; font-size: .7em ;}
924 div.sidetocctitle {font-size: 1.2em; font-weight:bold; text-align:center;
925     border-bottom: 1px solid silver ; }
926 nav.sidetoc a:hover {text-decoration: underline ; }
927
928
929
930 section.textbody { margin: 0ex 1em 0ex 1em ;}
931
932
933 div.multicolsheading { -webkit-column-span: all;
```

```
934     -moz-column-span: all; column-span: all; }
935 div.multicols { -webkit-columns: 3 380px ;
936     -moz-columns: 3 380px ; columns: 3 380px ; }
937 div.multicols p {margin-top: 0ex}
938
939 /* Used to support algorithmicx: */
940 span.floatright { float: right ; }
941
942
943
944 /*
945 For CSS LaTeX and related logos:
946 Based on:
947 http://edward.oconnor.cx/2007/08/tex-poshlet
948 http://nitens.org/taraborelli/texlogo
949 */
950
951 .latexlogofont {
952     font-family: "Linux Libertine O", "Nimbus Roman No 9 L",
953         "FreeSerif", "Hoefler Text", Times, "Times New Roman", serif;
954     font-variant: normal ;
955 }
956
957 .latexlogo {
958     font-family: "Linux Libertine O", "Nimbus Roman No 9 L",
959         "FreeSerif", "Hoefler Text", Times, "Times New Roman", serif;
960     letter-spacing: .03em ;
961     font-size: 1.1em;
962 }
963
964 .latexlogo sup {
965     text-transform: uppercase;
966     letter-spacing: .03em ;
967     font-size: 0.85em;
968     vertical-align: 0.15em;
969     margin-left: -0.36em;
970     margin-right: -0.15em;
971 }
972
973 .latexlogo sub {
974     text-transform: uppercase;
975     vertical-align: -0.5ex;
976     margin-left: -0.1667em;
977     margin-right: -0.125em;
978     font-size: 1em;
979 }
980
981 .xetexlogo {
982     font-family: "Linux Libertine O", "Nimbus Roman No 9 L",
983         "FreeSerif", "Hoefler Text", Times, "Times New Roman", serif;
```



```
984     letter-spacing: .03em ;
985     font-size: 1.1em;
986 }
987
988 /* A smaller gap between Xe and Tex v.s. LaTeX: */
989 .xetexlogo sub {
990     text-transform: uppercase;
991     vertical-align: -0.5ex;
992     margin-left: -0.0667em;
993     margin-right: -0.2em;
994     font-size: 1em;
995     letter-spacing: .03em ;
996 }
997
998 /* A large gap between Xe and LaTeX v.s. TeX: */
999 .xelatexlogo sub {
1000     text-transform: uppercase;
1001     vertical-align: -0.5ex;
1002     margin-left: -0.0667em;
1003     margin-right: -.05em;
1004     font-size: 1em;
1005     letter-spacing: .03em ;
1006 }
1007
1008 .amslogo {
1009     font-family: "TeXGyreChorus","URW Chancery L",
1010         "Apple Chancery","ITC Zapf Chancery","Monotype Corsiva",
1011         "Linux Libertine O", "Nimbus Roman No 9 L", "FreeSerif",
1012         "Hoefler Text", Times, "Times New Roman", serif;
1013     font-style: italic;
1014 }
1015
1016 .lyxlogo {
1017     font-family: "URW Classico", Optima, "Linux Biolinum O",
1018         "DejaVu Sans", "Bitstream Vera Sans", Geneva,
1019     Verdana, sans-serif ;
1020 }
1021
1022
1023
1024
1025 /* Only display top and bottom navigation if a small screen: */
1026 /* Hide the sidetoc if a small screen: */
1027 nav.topnavigation { display:none; }
1028 nav.botnavigation { display:none; }
1029
1030 @media screen and (max-width: 45em) {
1031     /*     nav.sidetoc {display:none;} */
1032     nav.sidetoc {
1033         float: none ;
```

```

1034         width: 100% ;
1035         margin: 5ex 0px 5ex 0px ;
1036         padding: 0 ;
1037         border-radius: 0 ;
1038         border-bottom: 1px solid black ;
1039         border-top: 1px solid black ;
1040         box-shadow: none ;
1041     }
1042 /*     nav.topnavigation { display:block } */
1043     nav.botnavigation { display:block }
1044 }
1045
1046 @media print {
1047     body {
1048         font-family: "Linux Libertine O",
1049             "DejaVu Serif", "Bitstream Vera Serif",
1050             "Liberation Serif", "Nimbus Roman No 9 L",
1051             "FreeSerif", "Hoefler Text", Times, "Times New Roman", serif;
1052     }
1053     nav.sidetoc { display:none; }
1054     nav.topnavigation { display: none; }
1055     nav.botnavigation { display: none; }
1056 }
1057
1058 @media handheld {
1059     nav.sidetoc { display:none; }
1060     nav.topnavigation { display:block }
1061     nav.botnavigation { display:block }
1062 }
1063
1064 @media projection {
1065     nav.sidetoc { display:none; }
1066     nav.topnavigation { display:block }
1067     nav.botnavigation { display:block }
1068 }
1069 \end{VerbatimOut}
1070 % \end{Verbatim}% for syntax highlighting

```

## 69.4 lwarp\_\_sagebrush.css

File `lwarp_sagebrush.css` An optional CSS which may be used for a semi-modern appearance.

If used, this must be present both when compiling the project and also when distributing the HTML files.

```

1071 \begin{VerbatimOut}{lwarp_sagebrush.css}
1072 @import url("lwarp.css") ;

```

```
1073
1074
1075 A:link {color:#105030 ; text-decoration: none ; }
1076 A:visited {color:#705030 ; text-shadow:1px 1px 2px #a0a0a0;}
1077 A:hover {color:#006000 ; text-decoration: underline ; text-shadow:0px 0px 2px #a0a0a0;}
1078 A:active {color:#00C000 ; text-shadow:1px 1px 2px #a0a0a0;}
1079
1080
1081
1082 h1, h2, h3, h4, h5, h6, span.paragraph, span.subparagraph
1083 {
1084     font-family: "URW Classico", Optima, "Linux Biolinum O",
1085                 "Linux Libertine O", "Liberation Serif",
1086                 "Nimbus Roman No 9 L", "FreeSerif",
1087                 "Hoefler Text", Times, "Times New Roman", serif;
1088     font-variant: small-caps ;
1089 font-weight: normal ;
1090     color: #304070 ;
1091     text-shadow: 2px 2px 3px #808080;
1092 }
1093
1094 h1 { /* title of the entire website, used on each page */
1095     font-variant: small-caps ;
1096     color: #304070 ;
1097     text-shadow: 2px 2px 3px #808080;
1098     background-color: #F7F7F0 ;
1099     background-image: linear-gradient(to bottom, #F7F7F0, #C0C0C4);
1100 }
1101
1102 h1 {
1103     border-bottom: 1px solid #304070;
1104     border-top: 2px solid #304070;
1105 }
1106
1107 h2 {
1108     border-bottom: 1px solid #304070;
1109     border-top: 2px solid #304070;
1110     background-color: #F7F7F0 ;
1111     background-image: linear-gradient(to bottom, #F7F7F0, #DAD0C0);
1112 }
1113
1114
1115
1116 div.abstract {
1117     background: #f5f5eb ;
1118     background-image: linear-gradient(to bottom, #f5f5eb, #C8C8B8);
1119
1120     border: 1px solid silver;
1121     border-radius: 1em ;
1122 }
```

```
1123
1124 div.abstract dl {line-height:1.5;}
1125 div.abstract dt {color:#304070;}
1126
1127 div.abstracttitle{
1128     font-family: "URW Classico", Optima, "Linux Biolinum O",
1129         "Linux Libertine O", "Liberation Serif", "Nimbus Roman No 9 L",
1130         "FreeSerif", "Hoefler Text", Times, "Times New Roman", serif;
1131     font-weight:bold;
1132     font-variant: small-caps ;
1133     font-size:1.5em;
1134     border-bottom: 1px solid silver ;
1135     color: #304070 ;
1136     text-align: center ;
1137     text-shadow: 1px 1px 2px #808080;
1138 }
1139
1140
1141 div.epigraph {
1142     background: #f5f5eb ;
1143     background-image: linear-gradient(to bottom, #f5f5eb, #C8C8B8);
1144
1145     border: 1px solid silver ;
1146     border-radius: 1ex ;
1147     box-shadow: 3px 3px 3px #808080 ;
1148 }
1149
1150
1151 .example {
1152     background-color: #f5f5eb ;
1153     background-image: linear-gradient(to bottom, #f5f5eb, #C8C8B8);
1154
1155 }
1156
1157 div.exampletitle{
1158     font-family: "URW Classico", Optima, "Linux Biolinum O",
1159         "Linux Libertine O", "Liberation Serif", "Nimbus Roman No 9 L",
1160         "FreeSerif", "Hoefler Text", Times, "Times New Roman", serif;
1161     font-weight:bold;
1162     font-variant: small-caps ;
1163     border-bottom: 1px solid silver ;
1164     color: #304070 ;
1165     text-align: center ;
1166     text-shadow: 1px 1px 2px #808080;
1167 }
1168
1169
1170 .sidebar {
1171     background-color: #f5f5eb ;
1172     background-image: linear-gradient(to bottom, #f5f5eb, #C8C8B8);
```

```
1173
1174 }
1175
1176 div.sidebartitle{
1177     font-family: "URW Classico", Optima, "Linux Biolinum O",
1178         "Linux Libertine O", "Liberation Serif", "Nimbus Roman No 9 L",
1179         "FreeSerif", "Hoefler Text", Times, "Times New Roman", serif;
1180     font-weight:bold;
1181     font-variant: small-caps ;
1182     border-bottom: 1px solid silver ;
1183     color: #304070 ;
1184     text-align: center ;
1185     text-shadow: 1px 1px 2px #808080;
1186 }
1187
1188
1189 .fancyvrblabel {
1190     font-family: "URW Classico", Optima, "Linux Biolinum O",
1191         "Linux Libertine O", "Liberation Serif", "Nimbus Roman No 9 L",
1192         "FreeSerif", "Hoefler Text", Times, "Times New Roman", serif;
1193     font-weight:bold;
1194     font-variant: small-caps ;
1195     font-size: 1.5em ;
1196     color: #304070 ;
1197     text-align: center ;
1198     text-shadow: 1px 1px 2px #808080;
1199 }
1200
1201
1202
1203 .minipage {
1204     background-color: #eeeeee7 ;
1205     border: 1px solid silver ;
1206     border-radius: 1ex ;
1207 }
1208
1209 figure.figure .minipage { border: none; }
1210
1211 div.marginblock div.minipage { border: none; }
1212
1213 figure , div.marginblock {
1214     background-color: #eeeeee7 ;
1215     border: 1px solid silver ;
1216     border-radius: 1ex ;
1217     box-shadow: 3px 3px 3px #808080 ;
1218 }
1219
1220 figure figure {
1221     border: 1px solid silver ;
1222     margin: 0em ;
```

```
1223 box-shadow: none ;
1224 }
1225
1226 /*
1227 figcaption {
1228     border-top: 1px solid silver ;
1229     border-bottom: 1px solid silver ;
1230     background-color: #e8e8e8 ;
1231 }
1232 */
1233
1234
1235 div.table {
1236     box-shadow: 3px 3px 3px #808080 ;
1237 }
1238
1239 /*
1240 .tnotes {
1241     background: #e8e8e8;
1242     border: 1px solid silver;
1243 }
1244 */
1245
1246
1247 nav.topnavigation{
1248     background-color: #b0b8b0 ;
1249     background-image: linear-gradient(to bottom,#e0e0e0,#b0b8b0) ;
1250 }
1251
1252 nav.botnavigation{
1253     background-color: #b0b8b0 ;
1254     background-image: linear-gradient(to top,#e0e0e0,#b0b8b0) ;
1255 }
1256
1257
1258
1259 header{
1260     background-color: #F7F7F0 ;
1261     background-image: linear-gradient(to top, #F7F7F0, #b0b8b0);
1262 }
1263
1264 footer{
1265     background-color: #F7F7F0 ;
1266     background-image: linear-gradient(to bottom, #F7F7F0, #b0b8b0);
1267 }
1268
1269
1270
1271 nav.sidetoc {
1272     background-color: #F7F7F0 ;
```

```

1273     background-image: linear-gradient(to bottom, #F7F7F0, #C0C0C0);
1274     box-shadow: 3px 3px 3px #808080 ;
1275     border-radius: 0px 0px 0px 20px ;
1276 }
1277
1278 div.sidetoctitle {color: #304070 ; }
1279 nav.sidetoc a:hover {color:#006000 ; text-decoration: none ; text-shadow:0px 0px 2px #a0a0a0;}
1280
1281
1282 @media screen and (max-width: 45em) {
1283     nav.sidetoc { border-radius: 0 ; }
1284 }
1285
1286
1287 \end{VerbatimOut}
1288 % \end{Verbatim}% for syntax highlighting

```

## 69.5 lwarp\_formal.css

File `lwarp_formal.css` An optional CSS which may be used for a more formal appearance.

If used, this must be present both when compiling the project and also when distributing the HTML files.

```

1289 \begin{VerbatimOut}{lwarp_formal.css}
1290 @import url("lwarp.css") ;
1291
1292
1293
1294 A:link {color:#802020 ; text-decoration:none; }
1295 A:visited {color:#802020 ; text-shadow:none ;}
1296 A:hover {color:#400000 ; text-shadow:none ;}
1297 A:active {color:#C00000 ; text-shadow:none ;}
1298
1299
1300 body {
1301     font-family: "Linux Libertine O", "Hoefler Text", "Garamond",
1302         "Bembo", "Janson", "TeX Gyre Pagella", "Palatino",
1303         "Liberation Serif", "Nimbus Roman No 9 L", "FreeSerif", Times,
1304         "Times New Roman", serif;
1305     background: #fffcf5;
1306 }
1307
1308 span.textrm {
1309     font-family: "Linux Libertine O", "Hoefler Text", "Garamond",
1310         "Bembo", "Janson", "TeX Gyre Pagella", "Palatino",
1311         "Liberation Serif", "Nimbus Roman No 9 L", "FreeSerif", Times,

```

```
1312         "Times New Roman", serif;
1313     }
1314
1315     span.textsf {
1316         font-family: "DejaVu Sans", "Bitstream Vera Sans",
1317             Geneva, Verdana, sans-serif ;
1318     }
1319
1320
1321
1322     h1, h2, h3, h4, h5, h6, span.paragraph, span.subparagraph
1323     {
1324         font-family: "Linux Libertine O", "Hoefler Text", "Garamond",
1325             "Bembo", "Janson", "TeX Gyre Pagella", "Palatino",
1326             "Liberation Serif", "Nimbus Roman No 9 L", "FreeSerif", Times,
1327             "Times New Roman", serif;
1328         color: #800000 ;
1329         text-shadow: none ;
1330     }
1331
1332     h1, h2 {
1333         background-color: #fffcf5 ;
1334         background-image: none ;
1335         border-bottom: 1px solid #808080;
1336         border-top: 2px solid #808080;
1337     }
1338
1339     div.abstracttitle {
1340         font-family: "Linux Libertine O", "Hoefler Text", "Garamond",
1341             "Bembo", "Janson", "TeX Gyre Pagella", "Palatino",
1342             "Liberation Serif", "Nimbus Roman No 9 L", "FreeSerif", Times,
1343             "Times New Roman", serif;
1344         color: black ;
1345         text-shadow: none ;
1346     }
1347
1348     div.abstract { font-size: 100% }
1349
1350     .sidebar {
1351         background: #fffcf5;
1352         background-image: none ;
1353         margin: 2em 5% 2em 5%;
1354         padding: 0.5em 1em;
1355         border: none ;
1356         border-top : 1px solid silver;
1357         border-bottom : 1px solid silver;
1358         font-size: 90% ;
1359     }
1360
1361     div.sidebartitle{
```



```
1362     font-family: "Linux Libertine O", "Hoefler Text", "Garamond",
1363         "Bembo", "Janson", "TeX Gyre Pagella", "Palatino",
1364         "Liberation Serif", "Nimbus Roman No 9 L", "FreeSerif", Times,
1365         "Times New Roman", serif;
1366     color: #800000 ;
1367     text-shadow: none ;
1368     border: none ;
1369 }
1370
1371 .example {
1372     background: #fffcf5;
1373     background-image: none ;
1374     margin: 2em 5% 2em 5%;
1375     padding: 0.5em 1em;
1376     border: none ;
1377     border-top : 1px solid silver;
1378     border-bottom : 1px solid silver;
1379 }
1380
1381 div.exampletitle{
1382     font-family: "Linux Libertine O", "Hoefler Text", "Garamond",
1383         "Bembo", "Janson", "TeX Gyre Pagella", "Palatino",
1384         "Liberation Serif", "Nimbus Roman No 9 L", "FreeSerif", Times,
1385         "Times New Roman", serif;
1386     color: #800000 ;
1387     text-shadow: none ;
1388     border: none ;
1389 }
1390
1391 div.fancyvrblabel{
1392     font-family: "Linux Libertine O", "Hoefler Text", "Garamond",
1393         "Bembo", "Janson", "TeX Gyre Pagella", "Palatino",
1394         "Liberation Serif", "Nimbus Roman No 9 L", "FreeSerif", Times,
1395         "Times New Roman", serif;
1396     color: #800000 ;
1397     text-shadow: none ;
1398     border: none ;
1399 }
1400
1401
1402
1403 .verse {
1404     font-family: "Linux Libertine O", "Hoefler Text", "Garamond",
1405         "Bembo", "Janson", "TeX Gyre Pagella", "Palatino",
1406         "Liberation Serif", "Nimbus Roman No 9 L", "FreeSerif", Times,
1407         "Times New Roman", serif;
1408 }
1409
1410
1411 figure {
```

```
1412     margin: 3ex 5% 3ex 5% ;
1413     padding: 1ex 1em 1ex 1em ;
1414     background-color: #fffcf5 ;
1415     overflow-x: auto ;
1416     border: none ;
1417 /*     border-top: 1px solid silver; */
1418 /*     border-bottom: 1px solid silver; */
1419 }
1420
1421
1422 figcaption , .lstlisting {
1423     border: none ;
1424 /*     border-top: 1px solid silver ; */
1425 /*     border-bottom: 1px solid silver ; */
1426     background-color: #fffcf5 ;
1427 }
1428
1429 .tnotes {
1430     background: #fffcf5 ;
1431 }
1432
1433 .theorem {
1434     background: none ;
1435 }
1436
1437 .minipage {
1438     background-color: #fffcf5 ;
1439     border: none ;
1440     margin: 1em ;
1441 }
1442
1443 div.floatrow figure { border: none ; }
1444
1445 figure figure { border: none ; }
1446
1447
1448 nav.toc, nav.lof, nav.lot, nav.lol {
1449     font-family: "Linux Libertine O", "Hoefler Text", "Garamond",
1450         "Bembo", "Janson", "TeX Gyre Pagella", "Palatino",
1451         "Liberation Serif", "Nimbus Roman No 9 L", "FreeSerif", Times,
1452         "Times New Roman", serif;
1453 }
1454
1455 nav.sidetoc {
1456     font-family: "Linux Libertine O", "Hoefler Text", "Garamond",
1457         "Bembo", "Janson", "TeX Gyre Pagella", "Palatino",
1458         "Liberation Serif", "Nimbus Roman No 9 L", "FreeSerif", Times,
1459         "Times New Roman", serif;
1460     background-image: linear-gradient(to bottom, #fffcf5, #C0C0C0);
1461     border-radius: 0px 0px 0px 20px ;
```

```

1462 }
1463
1464 div.sidetocitle{
1465     color: #800000 ;
1466 }
1467
1468 header{
1469     background-color: #e0e0e0 ;
1470     background-image: linear-gradient(to top, #fffcf5, #b0b0b0);
1471     text-align:center ;
1472 }
1473
1474 footer{
1475     background-color: #e0e0e0 ;
1476     background-image: linear-gradient(to bottom, #fffcf5, #b0b0b0);
1477     padding: 2ex 1em 2ex 1em ;
1478     clear:right ;
1479     text-align:left ;
1480 }
1481
1482 nav.botnavigation {
1483     background: #dedcd5 ;
1484     border-top: 1px solid black ;
1485 }
1486 \end{VerbatimOut}
1487 % \end{Verbatim}% for syntax highlighting

```

## 69.6 sample\_project.css

File `sample_project.css` The project-specific CSS file. Use with `\NewCSS`.

If used, this must be present both when compiling the project and also when distributing the HTML files.

```

1488 \begin{VerbatimOut}{sample_project.css}
1489 /* ( --- Start of project.css --- ) */
1490 /* A sample project-specific CSS file for lwarp --- ) */
1491
1492 /* Load default lwarp settings: */
1493 @import url("lwarp.css") ;
1494 /* or lwarp_formal.css, lwarp_sagebrush.css */
1495
1496 /* Project-specific CSS setting follow here. */
1497 /* . . . */
1498
1499 /* ( --- End of project.css --- ) */
1500 \end{VerbatimOut}
1501 % \end{Verbatim}% for syntax highlighting

```

## 69.7 lwarp\_html.xdy

File `lwarp_html.xdy` Used to modify the index for lwarp.

This must be present when compiling the project, but does not need to be present when distributing the resulting HTML files.

```

1502 \begin{VerbatimOut}{lwarp_html.xdy}
1503 (require "tex/inputenc/latin.xdy")
1504 (merge-rule "\\PS *" "Postscript")
1505 (require "texindy.xdy")
1506 (require "page-ranges.xdy")
1507 (require "book-order.xdy")
1508 (markup-locref :open "\hyperindexref{" :close "{")
1509 \end{VerbatimOut}
1510 % \end{Verbatim}% for syntax highlighting

```

## 69.8 lwarp\_mathjax.txt

File `lwarp_mathjax.txt` Used by lwarp when using MathJax.

This must be present when compiling the project, but does not need to be present when distributing the resulting HTML files.

```

1511 \begin{VerbatimOut}{lwarp_mathjax.txt}
1512 <!-- https://groups.google.com/forum/#!topic/
1513                               mathjax-users/jUteWUcE2bY -->
1514 <script type="text/x-mathjax-config">
1515 MathJax.Hub.Register.StartupHook("TeX AMSmath Ready",function () {
1516     var seteqsectionDefault = {name: "", num: 0};
1517     var seteqsections = {}, seteqsection = seteqsectionDefault;
1518     var TEX = MathJax.InputJax.TeX, PARSE = TEX.Parse;
1519     var AMS = MathJax.Extension["TeX/AMSmath"];
1520     TEX.Definitions.Add({
1521     macros: {
1522         seteqsection: "mySection",
1523         seteqnumber: "mySetEqNumber"
1524     }
1525     });
1526
1527     PARSE.Augment({
1528     mySection: function (name) {
1529         seteqsection.num = AMS.number;
1530         var n = this.GetArgument(name);
1531         if (n === "") {
1532             seteqsection = seteqsectionDefault;
1533         } else {

```

```

1534         if (!seteqsections["_"+n])
1535             seteqsections["_"+n] = {name:n, num:0};
1536         seteqsection = seteqsections["_"+n];
1537     }
1538     AMS.number = seteqsection.num;
1539 },
1540 mySetEqNumber: function (name) {
1541     var n = this.GetArgument(name);
1542     if (!n || !n.match(/^ *[0-9]+ *$/))
1543         n = ""; else n = parseInt(n)-1;
1544     <!-- $ syntax highlighting -->
1545     if (n === "" || n < 1)
1546         TEX.Error
1547             ("Argument to "+name+" should be a positive integer");
1548     AMS.number = n;
1549 }
1550 });
1551 MathJax.Hub.Config({
1552   TeX: {
1553     equationNumbers: {
1554       formatTag: function (n)
1555         {return "("+(seteqsection.name+"."+n).replace(/\./,"")+")"},
1556       formatID: function (n) {
1557         n = (seteqsection.name+'.'+n).replace
1558           (/[:"'<>&]/g,"").replace(/\./,"");
1559         return 'mjax-eqn-' + n;
1560       }
1561     }
1562   }
1563 });
1564 });
1565 </script>
1566
1567 <!-- http://docs.mathjax.org/en/latest/options/ThirdParty.html -->
1568 <script type="text/x-mathjax-config">
1569   MathJax.Ajax.config.path["Contrib"] =
1570     "https://cdn.mathjax.org/mathjax/contrib";
1571 </script>
1572
1573 <!-- https://github.com/mathjax/MathJax-third-party-extensions/
1574                                     tree/master/siunitx -->
1575 <script type="text/x-mathjax-config">
1576   MathJax.Hub.Config({
1577     extensions: ["tex2jax.js","[Contrib]/siunitx/siunitx.js"],
1578     jax: ["input/TeX","output/HTML-CSS"],
1579     tex2jax: {inlineMath: [["$","$"],["\\(","\\)"]]},
1580     TeX: {extensions: ["AMSmath.js","AMSsymbols.js", "sinuitx.js"]}
1581   });
1582 </script>
1583

```

```

1584 <script type="text/x-mathjax-config">
1585 MathJax.Hub.Config({
1586     TeX: {
1587         equationNumbers: {
1588             autoNumber: "AMS"
1589         }
1590     }
1591 });
1592 </script>
1593
1594 <script
1595     src="https://cdn.mathjax.org/mathjax/latest/MathJax.js?config=TeX-AMS_HTML-full">
1596 </script>
1597 \end{VerbatimOut}
1598 % \end{Verbatim}% for syntax highlighting

```

## 69.9 Lwarpmk option

The following is only generated if the `lwarpmk` option was given to `lwarp-newproject`.

```

1599 \begin{LWR@createlwarpmk}

```

File `lwarpmk` Creates a local copy of `lwarpmk`:

```

1600 \begin{VerbatimOut}{lwarpmk.lua}
1601 #!/usr/bin/env texlua
1602
1603 -- Copyright 2016 Brian Dunn
1604
1605 -- Print the usage of the lwarpmk command:
1606
1607 printversion = "v0.20"
1608
1609 function printhelp ()
1610 print ("lwarpmk: Use lwarpmk -h or lwarpmk --help for help.") ;
1611 end
1612
1613 function printusage ()
1614 print ( [[
1615
1616 lwarpmk print [project]: Compile a print version.
1617 lwarpmk printindex [project]: Process the index for the print version.
1618 lwarpmk html [project]: Compile an HTML version.
1619 lwarpmk htmlindex [project]: Process the index for the html version.
1620 lwarpmk again [project]: Touch the source code to trigger recompiles.
1621 lwarpmk limages [project]: Process the "lateximages" created by lwarp.sty.
1622 lwarpmk pdftohtml [project]:

```

```

1623     For use with latexmk or a Makefile:
1624     Convert project_html.pdf to project_html.html and
1625     individual HTML files.
1626 lwarpmk clean [project]: Remove project.aux, .toc, .lof, .lot, .idx, .ind, .log
1627 lwarpmk cleanall [project]: Remove auxiliary files and also project.pdf, *.html
1628 lwarpmk -h: Print this help message.
1629 lwarpmk --help: Print this help message.
1630
1631 ]] )
1632 printconf ()
1633 end
1634
1635 -- Print the format of the configuration file lwarpmk.conf:
1636
1637 function printconf ()
1638 print ( [[
1639 An example lwarpmk.conf or <project>.lwarpmkconf project file:
1640 --
1641 opsystem = "Unix"      (or "Windows")
1642 latexname = "pdflatex" (or "lualatex", or "xelatex")
1643 sourcename = "projectname" (the source-code filename w/o .tex)
1644 homehtmlfilename = "index" (or perhaps the project name)
1645 htmlfilename = ""      (or "projectname" - filename prefix)
1646 uselatexmk = "false"   (or "true" to use latexmk to build PDFs)
1647 --
1648 Filenames must contain only letters, numbers, underscore, or dash.
1649 Values must be in "quotes".
1650
1651 ]] ) ;
1652 end
1653
1654 -- Split one large sourcefile into a number of files,
1655 -- starting with destfile.
1656 -- The file is split at each occurrence of <!--|Start file|newfilename|*
1657
1658 function splitfile (destfile,sourcefile)
1659 print ("lwarpmk: Splitting " .. sourcefile .. " into " .. destfile) ;
1660 io.input(sourcefile)
1661 io.output(destfile)
1662 for line in io.lines() do
1663 i,j,copen,cstart,newfilename = string.find (line,"(.*)|(.*)|(.*|") ;
1664 if ( (i~= nil) and (copen == "<!--") and (cstart == "Start file")) then -- split the file
1665 io.output(newfilename) ;
1666 else -- not a splitpoint
1667 io.write (line .. "\n") ;
1668 end
1669 end -- do
1670 end -- function
1671
1672 -- Incorrect value, so print an error and exit.

```

```
1673
1674 function cvalueerror ( line, linenum , cvalue )
1675     print ( linenum .. " : " .. line ) ;
1676     print ("lwarpmk: incorrect variable value \" .. cvalue .. "\"" in lwarpmk.conf.\n" ) ;
1677     printconf ( ) ;
1678     os.exit(1) ;
1679 end
1680
1681 -- Load settings from the project's "lwarpmk.conf" file:
1682
1683 function loadconf ( )
1684 -- Default configuration filename:
1685 local conffile = "lwarpmk.conf"
1686 -- Optional configuration filename:
1687 if arg[2] ~= nil then conffile = arg[2].."lwarpmkconf" end
1688 -- Verify the file exists:
1689 if (lfs.attributes(conffile,"mode")==nil) then -- file not exists
1690 print("lwarpmk: " .. conffile .. " does not exist.")
1691 print("lwarpmk: " .. arg[2] .. " does not appear to be a project name.\n")
1692 printhelp ( ) ;
1693 os.exit(1) -- exit the entire lwarpmk script
1694 else -- file exists
1695 -- Read the file:
1696 print ("lwarpmk: Reading " .. conffile .. ".")
1697 io.input(conffile) ;
1698 -- Scan each line:
1699 local linenum = 0
1700 for line in io.lines() do -- scan lines
1701 linenum = linenum + 1
1702 i,j,cvarname,cvalue = string.find (line,"([%w-_]*)%s*=%s*\"([%w-_]*)\"") ;
1703 -- Error if incorrect enclosing characters:
1704 if ( i == nil ) then
1705 print ( linenum .. " : " .. line ) ;
1706 print ( "lwarpmk: Incorrect entry in " .. conffile .. ".\n" ) ;
1707 printconf ( ) ;
1708 os.exit(1) ;
1709 end
1710 if ( cvarname == "opssystem" ) then
1711     -- Verify choice of opssystem:
1712     if ( (cvalue == "Unix") or (cvalue == "Windows") ) then
1713         opssystem = cvalue
1714     else
1715         cvalueerror ( line, linenum , cvalue )
1716     end
1717 elseif ( cvarname == "latexname" ) then
1718     -- Verify choice of LaTeX compiler:
1719     if (
1720         (cvalue == "pdflatex") or
1721         (cvalue == "xelatex") or
1722         (cvalue == "lualatex")
```



```
1723     ) then
1724         latexname = cvalue
1725     else
1726         cvalueerror ( line, linenum , cvalue )
1727     end
1728 elseif ( cvarname == "sourcename" ) then sourcename = cvalue
1729 elseif ( cvarname == "homehtmlfilename" ) then homehtmlfilename = cvalue
1730 elseif ( cvarname == "htmlfilename" ) then htmlfilename = cvalue
1731 elseif ( cvarname == "uselatexmk" ) then uselatexmk = cvalue
1732 else
1733 print ( linenum .. " : " .. line ) ;
1734 print ("lwarpmk: Incorrect variable name \" .. cvarname .. "\" in " .. conffile .. ".\n" ) ;
1735 printconf ( ) ;
1736 os.exit(1) ;
1737 end
1738 end -- do scan lines
1739 end -- file exists
1740 -- Select some operating-system commands:
1741 if opsystem=="Unix" then -- For Unix / Linux / Mac OS:
1742     rmname = "rm"
1743     touchname = "touch"
1744     chmodcmd = "chmod u+x lateximages.sh"
1745     lateximagesname = "./lateximages.sh"
1746 elseif opsystem=="Windows" then -- For Windows
1747     rmname = "DEL"
1748     touchname = "TOUCH"
1749     chmodcmd = ""
1750     lateximagesname = "lateximages.cmd"
1751 else print ( "lwarpmk: Select Unix or Windows for opsystem" )
1752 end --- for Windows
1753 end -- loadconf
1754
1755 -- Scan the LaTeX log file for the phrase "Rerun to get",
1756 -- indicating that the file should be compiled again.
1757 -- Return true if found.
1758
1759 function reruntoget (filesourse)
1760 io.input(filesourse)
1761 for line in io.lines() do
1762 if ( string.find(line,"Rerun to get") ~= nil ) then return true end
1763 end
1764 return false
1765 end
1766
1767 -- Compile one time, return true if should compile again.
1768 -- fsuffix is "" for print, "_html" for HTML output.
1769
1770 function onetime (fsuffix)
1771 print("lwarpmk: Compiling with " .. latexname .. " " .. sourcename..fsuffix)
1772 err = os.execute(
```

```
1773 --      "echo " ..
1774      latexname .. " " .. sourcename..fsuffix )
1775 if ( err ~= 0 ) then print ( "lwarpmk: Compile error." ) ; os.exit(1) ; end
1776 return (reruntoget(sourcename .. fsuffix .. ".log") ) ;
1777 end
1778
1779 -- Compile up to five times.
1780 -- fsuffix is "" for print, "_html" for HTML output
1781
1782 function manytimes (fsuffix)
1783 if onetime(fsuffix) == true then
1784 if onetime(fsuffix) == true then
1785 if onetime(fsuffix) == true then
1786 if onetime(fsuffix) == true then
1787 if onetime(fsuffix) == true then
1788 end end end end end
1789 end
1790
1791 -- Exit if the given file does not exist.
1792
1793 function verifyfileexists (filename)
1794 if (lfs.attributes ( filename , "modification" ) == nil ) then
1795 print ( "lwarpmk: " .. filename .. " not found." ) ;
1796 os.exit (1) ;
1797 end
1798 end
1799
1800 -- Convert <project>_html.pdf into HTML files:
1801
1802 function pdftohtml ()
1803     -- Convert to text:
1804     print ("lwarpmk: Converting " .. sourcename .. "_html.pdf to " .. sourcename .. "_html.html"
1805     os.execute("pdftotext -enc UTF-8 -nopgbrk -layout " .. sourcename .. "_html.pdf " .. sou
1806     -- Split the result into individual HTML files:
1807     splitfile (homehtmlfilename .. ".html" , sourcename .. "_html.html")
1808 end
1809
1810 -- Remove auxiliary files:
1811
1812 function removeaux ()
1813     os.execute ( rmname .. " " ..
1814         sourcename .. ".aux " .. sourcename .. "_html.aux " ..
1815         sourcename .. ".toc " .. sourcename .. "_html.toc " ..
1816         sourcename .. ".lof " .. sourcename .. "_html.lof " ..
1817         sourcename .. ".lot " .. sourcename .. "_html.lot " ..
1818         sourcename .. ".idx " .. sourcename .. "_html.idx " ..
1819         sourcename .. ".ind " .. sourcename .. "_html.ind " ..
1820         sourcename .. ".log " .. sourcename .. "_html.log "
1821     )
1822 end
```

```
1823
1824 -- lwarpmk --version :
1825
1826 if (arg[1] == "--version") then
1827 print ( "lwarpmk: " .. printversion )
1828
1829 else -- not -- version
1830
1831 -- print intro:
1832
1833 print ("lwarpmk: " .. printversion .. " Automated make for the LaTeX lwarp package.")
1834
1835 -- lwarpmk print:
1836
1837 if arg[1] == "print" then
1838 loadconf ()
1839 if ( uselatexmk == "true" ) then
1840     os.execute ( "latexmk -pdf -dvi- -ps- -pdflatex=\" " .. latexname .. " %0 %S\" " .. sourcenam
1841     print ("lwarpmk: Done.")
1842 else -- not latexmk
1843     verifyfileexists (sourcename .. ".tex") ;
1844     -- See if up to date:
1845     if (
1846         ( lfs.attributes ( sourcename .. ".pdf" , "modification" ) == nil ) or
1847         (
1848             lfs.attributes ( sourcename .. ".tex" , "modification" ) >
1849             lfs.attributes ( sourcename .. ".pdf" , "modification" )
1850         )
1851     ) then
1852         -- Recompile if not yet up to date:
1853         manytimes("")
1854         print ("lwarpmk: Done.") ;
1855     else
1856         print ("lwarpmk: " .. sourcename .. ".pdf is up to date.") ;
1857     end
1858 end -- not latexmk
1859
1860 -- lwarp printindex:
1861 -- Compile the index then touch the source
1862 -- to trigger a recompile of the document:
1863
1864 elseif arg[1] == "printindex" then
1865 loadconf ()
1866 print ("lwarpmk: Processing the index.")
1867 os.execute("texindy -M lwarp_html.xdy " .. sourcename .. ".idx")
1868 print ("lwarpmk: Forcing an update of " .. sourcename .. ".tex.")
1869 os.execute(touchname .. " " .. sourcename .. ".tex")
1870 print ("lwarpmk: " .. sourcename .. ".tex is ready to be recompiled.")
1871 print ("lwarpmk: Done.")
1872
```

```

1873 -- lwarpmk html:
1874
1875 elseif arg[1] == "html" then
1876 loadconf ()
1877 if ( uselatexmk == "true" ) then
1878     -- The recorder option is required to detect changes in <project>.tex
1879     -- while we are loading <project>_html.tex.
1880     err=os.execute ( "latexmk -pdf -dvi- -ps- -recorder "
1881         .. "-e '$makeindex = q/texindy -M lwarp_html.xdy/' "
1882         .. "-pdflatex=\"\" .. latexname .." %O %S\" "
1883         .. sourcename .. "_html.tex" ) ;
1884     if ( err ~= 0 ) then print ( "lwarpmk: Compile error." ) ; os.exit(1) ; end
1885     pdftohtml ()
1886     print ("lwarpmk: Done.")
1887 else -- not latexmk
1888     verifyfileexists ( sourcename .. ".tex" ) ;
1889     -- See if exists and is up to date:
1890     if (
1891         ( lfs.attributes ( homehtmlfilename .. ".html" , "modification" ) == nil ) or
1892         (
1893             lfs.attributes ( sourcename .. ".tex" , "modification" ) >
1894             lfs.attributes ( homehtmlfilename .. ".html" , "modification" )
1895         )
1896     ) then
1897         -- Recompile if not yet up to date:
1898         manytimes("_html")
1899         pdftohtml ()
1900         print ("lwarpmk: Done.")
1901     else
1902         print ("lwarpmk: " .. homehtmlfilename .. ".html is up to date.")
1903     end
1904 end -- not latexmk
1905
1906 elseif arg[1] == "pdftohtml" then
1907     loadconf ()
1908     pdftohtml ()
1909
1910 -- lwarpmk htmlindex:
1911 -- Compile the index then touch the source
1912 -- to trigger a recompile of the document:
1913
1914 elseif arg[1] == "htmlindex" then
1915 loadconf ()
1916 print ("lwarpmk: Processing the index.")
1917 os.execute("texindy -M lwarp_html.xdy " .. sourcename .. "_html.idx")
1918 print ("lwarpmk: Forcing an update of " .. sourcename .. ".tex.")
1919 os.execute(touchname .. " " .. sourcename .. ".tex")
1920 print ("lwarpmk: " .. sourcename .. ".tex is ready to be recompiled.")
1921 print ("lwarpmk: Done.")
1922

```

```
1923 -- lwarpmk limages:
1924 -- Make the lateximages command file executable,
1925 -- execute it to create the images,
1926 -- then touch the source to trigger a recompile.
1927
1928 elseif arg[1] == "limages" then
1929 loadconf ()
1930 print ("lwarpmk: Processing images.")
1931 os.execute(chmodcmd)
1932 os.execute(lateximagesname)
1933 print ("lwarpmk: Forcing an update of " .. sourcename .. ".tex.")
1934 os.execute(touchname .. " " .. sourcename .. ".tex") ;
1935 print ("lwarpmk: " .. sourcename .. ".tex is ready to be recompiled.")
1936 print ("lwarpmk: Done.")
1937
1938 -- lwarpmk again:
1939 -- Touch the source to trigger a recompile.
1940
1941 elseif arg[1] == "again" then
1942 loadconf ()
1943 print ("lwarpmk: Forcing an update of " .. sourcename .. ".tex.")
1944 os.execute(touchname .. " " .. sourcename .. ".tex") ;
1945 print ("lwarpmk: " .. sourcename .. ".tex is ready to be recompiled.")
1946 print ("lwarpmk: Done.")
1947
1948 -- lwarpmk clean:
1949 -- Remove project.aux, .toc, .lof, .lot, .idx, .ind, .log
1950
1951 elseif arg[1] == "clean" then
1952 loadconf ()
1953 removeaux ()
1954 print ("lwarpmk: Done.")
1955
1956 -- lwarpmk cleanall
1957 -- Remove project.aux, .toc, .lof, .lot, .idx, .ind, .log
1958 -- and also project.pdf, *.html
1959
1960 elseif arg[1] == "cleanall" then
1961 loadconf ()
1962 removeaux ()
1963 os.execute ( rmname .. " " ..
1964     sourcename .. ".pdf " .. sourcename .. "_html.pdf " ..
1965     "*.html"
1966     )
1967 print ("lwarpmk: Done.")
1968
1969 -- lwarpmk with no argument :
1970
1971 elseif (arg[1] == nil) then
1972 printhelp ()
```

```

1973
1974 -- lwarpmk -h or lwarpmk --help :
1975
1976 elseif (arg[1] == "-h" ) or (arg[1] == "--help") then
1977 printusage ()
1978
1979 else
1980 print ("lwarpmk: Unknown command \""..arg[1].."\".\n")
1981 printhelp ()
1982 end
1983
1984 end -- not --version
1985 \end{VerbatimOut}
1986 % \end{Verbatim}% for syntax highlighting
1987 \end{LWR@createlwarpmk}

```

## Package 3

# lwarp-afterpage.sty

## 70 Afterpage

Pkg afterpage

for **HTML output**: Discard all options for lwarp-afterpage:

```

1 \LWR@ProvidesPackageDrop{afterpage}

2 \newcommand{\afterpage}[1]{#1}

```

## Package 4


# lwarp-algorithmicx.sty

## 71 Algorithmicx

Pkg algorithmicx algorithmicx is supported with minor adjustments.

for HTML output: 1 \LWR@ProvidesPackagePass{algorithmicx}

Inside the `algorithmic` environment, level indenting is converted to a `span` of the required length, and comments are placed inside a `span` which is floated right.

 package conflicts If using `\newfloat`, `trivfloat`, and/or `algorithmicx` together, see section 110.1.

for HTML output: 2 \begin{warpHTML}

```

3 \AtBeginEnvironment{algorithmic}{%
4 %
5 \let\origALG@doentity\ALG@doentity%
6 %
7 \renewcommand*{\ALG@doentity}{%
8 \origALG@doentity%
9 \uselengthunit{PT}%
10 \LWR@htmltagc{%
11 span style="width:\rndprintlength{\ALG@thistlm}; display:inline-block;"%
12 }%
13 \LWR@htmltagc{/span}%
14 }%
15 %
16 \let\origComment\Comment%
17 %
18 \renewcommand{\Comment}[1]{\InlineClass{floatright}{\origComment{#1}}}%
19 }

20 \end{warpHTML}
```

## Package 5

# lwarp-alltt.sty

## 72 Alltt

Pkg alltt

for **HTML output**: alltt is patched for use by lwarp.

```
1 \LWR@ProvidesPackagePass{alltt}

2 \AfterEndPreamble{
3 \AtBeginEnvironment{alltt}{\LWR@atbeginverbatim{alltt}\unskip\vspace*{-\baselineskip}}
4 \AfterEndEnvironment{alltt}{\unskip\vspace*{-\baselineskip}\LWR@afterendverbatim}
5 }
```



## Package 6

# lwarp-bookmark.sty

## 73 Bookmark

Pkg `bookmark` `bookmark` is emulated during HTML output, and the `bookmark` package is ignored.  
for HTML output: Discard all options for `lwarp-bookmark`:

```
1 \LWR@ProvidesPackageDrop{bookmark}

2 \newcommand*{\bookmarksetup}[1]{}
3 \newcommand*{\bookmarksetupnext}[1]{}
4 \newcommand*{\bookmark}[2] [] {}
5 \newcommand*{\bookmarkdefinestyle}[2]{}
6 \newcommand*{\bookmarkget}[1]{}
7 \newcommand{\BookmarkAtEnd}[1]{}

```

## Package 7

# lwarp-booktabs.sty

## 74 Booktabs

Pkg booktabs booktabs is emulated during HTML output, and the booktabs package is ignored.

for HTML output: 1 \LWR@ProvidesPackageDrop{booktabs}

Booktabs emulation is spread among the tabular code.

Emulated for source compatibility.

```
2 \newcommand*{\addlinespace}[1]{}
3 \newcommand*{\morecmidrules}{}
4 \newcommand*{\specialrule}[3]{}

```

## Package 8

# lwarp-draftwatermark.sty

## 75 Draftwatermark

Pkg draftwatermark draftwatermark is emulated during HTML output, and the draftwatermark package is ignored.

```
1 \LWR@ProvidesPackageDrop{draftwatermark}

```

for HTML output:

```
2 \newcommand{\SetWatermarkAngle}[1]{}
3 \newcommand{\SetWatermarkColor}[1]{}
4 \newcommand{\SetWatermarkLightness}[1]{}
5 \newcommand{\SetWatermarkFontSize}[1]{}
6 \newcommand{\SetWatermarkScale}[1]{}
7 \newcommand{\SetWatermarkHorCenter}[1]{}
8 \newcommand{\SetWatermarkVertCenter}[1]{}
9 \newcommand{\SetWatermarkText}[1]{}

```

## Package 9

# lwarp-epigraph.sty

## 76 Epigraph

Pkg epigraph epigraph is emulated during HTML output, and the epigraph package is ignored.

```

1 \LWR@ProvidesPackageDrop{epigraph}

for HTML output:
2 \newcommand{\qitem}[2]
3 {
4 \begin{BlockClass}{qitem}
5 #1
6 \begin{BlockClass}{epigraphsource}
7 #2
8 \end{BlockClass}
9 \end{BlockClass}
10 }

11 \newcommand{\epigraph}[2]
12 {
13 \begin{BlockClass}{epigraph}
14 \qitem{#1}{#2}
15 \end{BlockClass}
16 }
17
18 \newenvironment*{epigraphs}
19 {\BlockClass{epigraph}}
20 {\endBlockClass}

```

Use CSS to format epigraphs.

The following are null commands for source compatibility:

```

21 \newlength{\epigraphwidth}
22 \setlength{\epigraphwidth}{.5\linewidth}
23 \newenvironment*{flushepinormal}{}{}
24 \newcommand{\textflush}[1]{flushepinormal}
25 \newcommand{\epigraphflush}[1]{flushright}
26 \newcommand{\sourceflush}[1]{flushright}
27 \newcommand*{\epigraphsize}{\small}
28 \newlength{\epigraphrule}
29 \newlength{\beforeepigraphskip}
30 \newlength{\afterepigraphskip}

```

```

31 \newcommand{\epigraphhead}[2][0]{#2}
32 \newcommand{\dropchapter}[1]{}
33 \newcommand*{\undodrop}{}
34 \newcommand{\cleartoevenpage}[1][]{}
```

## Package 10

# lwarp-eso-pic.sty

## 77 Eso-pic

Pkg    **eso-pic**    eso-pic is emulated during HTML output, and the eso-pic package is ignored.

```

1 \LWR@ProvidesPackageDrop{eso-pic}

for HTML output: 2 \newcommand*{\LenToUnit}{}
3 \newcommand{\AtPageUpperLeft}[1]{}
4 \newcommand{\AtPageLowerLeft}[1]{}
5 \newcommand{\AtPageCenter}[1]{}
6 \newcommand{\AtStockLowerLeft}[1]{}
7 \newcommand{\AtStockUpperLeft}[1]{}
8 \newcommand{\AtStockCenter}[1]{}
9 \newcommand{\AtTextUpperLeft}[1]{}
10 \newcommand{\AtTextLowerLeft}[1]{}
11 \newcommand{\AtTextCenter}[1]{}
12 \NewDocumentCommand{\AddToShipoutPictureBG}{s +m}{}
13 \let\AddToShipoutPicture\AddToShipoutPictureBG
14 \NewDocumentCommand{\AddToShipoutPictureFG}{s +m}{}
15 \newcommand*{\ClearShipoutPictureBG}{}
16 \newcommand*{\ClearShipoutPicture}{}
17 \newcommand*{\ClearShipoutPictureFG}{}
18 \newcommand{\gridSetup}[6][]{}
```

## Package 11

# lwarp-everypage.sty

## 78 Everypage

Pkg `everypage` `everypage` is emulated during HTML output, and the `everypage` package is ignored.

```

1 \LWR@ProvidesPackageDrop{everypage}

for HTML output: 2 \newcommand*{\AddEverypageHook}[1]{}
                  3 \newcommand*{\AddThispageHook}[1]{}

```

## Package 12

# lwarp-extramarks.sty

## 79 Extramarks

Pkg `extramarks`

for HTML output: Discard all options for `lwarp-extramarks`:

```

1 \LWR@ProvidesPackageDrop{extramarks}

2 \newcommand*{\extramarks}[2]{}
3 \newcommand*{\firstleftxmark}{}
4 \newcommand*{\lastleftxmark}{}
5 \newcommand*{\firstrightxmark}{}
6 \newcommand*{\lastrightxmark}{}
7 \newcommand*{\firstxmark}{}
8 \newcommand*{\lastxmark}{}
9 \newcommand*{\topxmark}{}
10 \newcommand*{\topleftxmark}{}
11 \newcommand*{\firstleftmark}{}
12 \newcommand*{\lastrightmark}{}

```

## Package 13

# lwarp-fancyhdr.sty

## 80 Fancyhdr

Pkg fancyhdr

for HTML output:

Discard all options for lwarp-fancyhdr:

```

1 \LWR@ProvidesPackageDrop{fancyhdr}

2 \newcommand*{\fancyhead}[2] [] {}
3 \newcommand*{\fancyfoot}[2] [] {}
4 \newcommand*{\fancyhf}[2] [] {}
5 \newcommand*{\fancypagestyle}[2] {}
6 \newcommand*{\lhead}[1] {}
7 \newcommand*{\chead}[1] {}
8 \newcommand*{\rhead}[1] {}
9 \newcommand*{\lfoot}[1] {}
10 \newcommand*{\cfoot}[1] {}
11 \newcommand*{\rfoot}[1] {}
12 \newcommand*{\headrulewidth}{}
13 \newcommand*{\footrulewidth}{}
14 \newcommand*{\fancyheadoffset}[2] [] {}
15 \newcommand*{\fancyfootoffset}[2] [] {}
16 \newcommand*{\fancyhfoffset}[2] [] {}
17 \newcommand*{\iffloatpage}[2] {#2}
18 \newcommand*{\ifftopfloat}[2] {#2}
19 \newcommand*{\iffbotfloat}[2] {#2}

```

## Package 14

# lwarp-float.sty

## 81 Float and \newfloat

Pkg float float is emulated during HTML output, and the float package is ignored.

for HTML output: 1 \LWR@ProvidesPackageDrop{float}[2016/03/04]

Pkg float

The float package is emulated.

See section 51.2 for the \listof command.

\newfloat {<1: type>} {<2: placement>} {<3: ext>} [{<4: within>}]

Emulates the \newfloat command from the float package.

“placement” is ignored.

```
2 \NewDocumentCommand{\newfloat}{m m m o}{%
3 \IfValueTF{#4}
4 {
5 \DeclareFloatingEnvironment[fileext=#3,within=#4]{#1}
6 }
7 {\DeclareFloatingEnvironment[fileext=#3]{#1}}
```

newfloat package automatically creates the \listof command for new floats, but float does not, so remove \listof here in case it is manually created later.

```
8 \cslet{listof#1s}\relax
9 \cslet{listof#1es}\relax
10 }
```

\floatname {<type>} {<name>}

Sets the text name of the float, such as “Figure”.

```
11 \NewDocumentCommand{\floatname}{m +m}{%
12 \SetupFloatingEnvironment{#1}{name=#2}%
13 }
```

\floatplacement {<type>} {<placement>}

Float placement is ignored.

```
14 \newcommand*{\floatplacement}[2]{%  
15 \SetupFloatingEnvironment{#1}{placement=#2}%  
16 }
```

`\floatstyle`     $\{\langle style \rangle\}$

Float styles are ignored.

```
17 \newcommand{\floatstyle}[1]{%  
18 }
```

`\restylefloat`     $* \{\langle style \rangle\}$

Float styles are ignored.

```
19 \NewDocumentCommand{\restylefloat}{s m}{%  
20 }
```



## Package 15

# lwarp-floatrow.sty

## 82 Floatrow

Pkg floatrow floatrow is emulated during HTML output, and the floatrow package is ignored.

for HTML output: 1 \LWR@ProvidesPackageDrop{floatrow}

Pkg floatrow

The floatrow package is emulated.

⚠ \FBwidth, \FBheight The emulation of floatrow does not support \FBwidth or \FBheight. These values are pre-set to .3\linewidth and 2in. Possible solutions include:

- Use fixed lengths. lwarp will scale the HTML lengths appropriately.
- Use warpprint and warpHTML environments to select appropriate values for each case.
- Inside a warpHTML environment, manually change \FBwidth or \FBheight before the \ffigbox or \ttabbox. Use \FBwidth or \FBheight normally afterwards; it will be used as expected in print output, and will use your custom-selected value in HTML output. This custom value will be used repeatedly, until it is manually changed to a new value.

Only parameters for caption and object are used.

LWR@insubfloatrow is true if inside a subfloatrow environment.

```
2 \NewDocumentCommand{\floatbox}{o m o o o +m +m}{%
3 \ifbool{LWR@insubfloatrow}
4 {
```

Inside a subfloatrow. subfigure and subtable take a width argument. Defaults to \linewidth if none is given.

```
5 \IfValueTF{#3}
6 {\@nameuse{sub#2}{#3}}
7 {\@nameuse{sub#2}{\linewidth}}
8 }
9 {
```

Not inside a subfloatrow. `figure` and `table` do not take a width argument.

```

10 \@nameuse{#2}
11 }
12 #6
13
14 #7
15 \ifbool{LWR@insubfloatrow}
16 {\@nameuse{endsub#2}}
17 {\@nameuse{end#2}}
18 }

```

Not used:

```

19 \newcommand*{\nocapbeside}{}
20 \newcommand*{\capbeside}{}
21 \newcommand*{\captop}{}
22 \newlength{\FBwidth}
23 \setlength{\FBwidth}{.3\linewidth}
24 \newlength{\FBheight}
25 \setlength{\FBheight}{2in}
26 \newcommand*{\useFCwidth}{}
27 \newcommand{\floatsetup}[2][{}]{
28 \newcommand{\thisfloatsetup}[1]{
29 \newcommand{\clearfloatsetup}[1]{
30 \newcommand*{\killfloatstyle}{}

```

Preamble and default width are ignored.

```

31 \NewDocumentCommand{\newfloatcommand}{m m o o}{%
32 \@namedef{#1}{
33 \floatbox{#2}
34 }
35 }

```

Preamble and default width are ignored.

```

36 \NewDocumentCommand{\renewfloatcommand}{m m o o}{%
37 \@namedef{#1}{%
38 \floatbox{#2}
39 }
40 }

```

```

41 \newfloatcommand{ffigbox}{figure}[\nocapbeside][

```

```

42 \newfloatcommand{ttabbox}{table}[\captop][\FBwidth]

```

```

43 \newfloatcommand{fcapside}{figure}[\capbeside][

```

The row of floats is placed into a div of class floatrow.

```
44 \newenvironment*{floatrow}[1][2]
45 {
46 \BlockClass{floatrow}
```

While inside the floatrow, divide the \linewidth by the number of floats.

```
47 \booltrue{LWR@infloatrow}
48 \setlength{\linewidth}{6in/#1}
49 }
50 {
51 \boolfalse{LWR@infloatrow}
52 \endBlockClass
53 }
```

Keys for \DeclareNewFloatType:

```
54 \newcommand*{\LWR@frowkeyplacement}{}
55 \newcommand*{\LWR@frowkeyname}{}
56 \newcommand*{\LWR@frowkeyfileext}{}
57 \newcommand*{\LWR@frowkeywithin}{}
58 \newcommand*{\LWR@frowkeycapstyle}{}
59
60 \define@key{frowkeys}{placement}{}%
61 \define@key{frowkeys}{name}{\renewcommand{\LWR@frowkeyname}{#1}}%
62 \define@key{frowkeys}{fileext}{\renewcommand{\LWR@frowkeyfileext}{#1}}%
63 \define@key{frowkeys}{within}{\renewcommand{\LWR@frowkeywithin}{#1}}%
64 \define@key{frowkeys}{relatedcapstyle}{}%
```

Use \listof{type}{Title} to print a list of the floats.

```
65 \newcommand*{\DeclareNewFloatType}[2]{%
```

Reset key values:

```
66 \renewcommand*{\LWR@frowkeyplacement}{}
67 \renewcommand*{\LWR@frowkeyname}{}
68 \renewcommand*{\LWR@frowkeyfileext}{}
69 \renewcommand*{\LWR@frowkeywithin}{}
70 \renewcommand*{\LWR@frowkeycapstyle}{}%
```

Read new key values:

```
71 \LWR@traceinfo{about to setkeys frowkeys}%
72 \setkeys{frowkeys}{#2}%
73 \LWR@traceinfo{finished setkeys frowkeys}%
```

Create a new float with optional [within]:

```

74 \ifthenelse{\equal{\LWR@frowkeywithin}{}}{
75 {
76 \LWR@traceinfo{about to newfloat #1 \LWR@frowkeyplacement\ \LWR@frowkeyfileext}%
77 \newfloat{#1}{\LWR@frowkeyplacement}{\LWR@frowkeyfileext}
78 }%
79 {%
80 \LWR@traceinfo{about to newfloat #1\ \LWR@frowkeyplacement\ \LWR@frowkeyfileext\ \LWR@frowkeywi
81 \newfloat{#1}{\LWR@frowkeyplacement}%
82 {\LWR@frowkeyfileext}[\LWR@frowkeywithin}%
83 \typeout{*** finished newfloat #1}
84 }%

```

Rename the float if a name was given:

```

85 \ifthenelse{\equal{\LWR@frowkeyname}{}}{
86 {}
87 {\floatname{#1}{\LWR@frowkeyname}}}%
88 }

```

Not used:

```

89 \newcommand{\buildFBBBOX}[2]{ }
90 \newcommand*{\CenterFloatBoxes}{ }
91 \newcommand*{\TopFloatBoxes}{ }
92 \newcommand*{\BottomFloatBoxes}{ }
93 \newcommand*{\PlainFloatBoxes}{ }
94
95 \newcommand{\capsubrowsettings}{ }
96
97 \NewDocumentCommand{\RawFloats}{o o}{ }

```

To be used inside a minipage or parbox.

```

98 \newcommand{\RawCaption}[1]{#1}

```

Places additional text inside a float, inside a CSS div of class floatfoot.

```

99 \NewDocumentCommand{\floatfoot}{s +m}{%
100 \begin{BlockClass}{floatfoot}
101 #2
102 \end{BlockClass}
103 }

```

Used to compute \linewidth.

```

104 \newbool{\LWR@insubfloatrow}
105 \boolfalse{\LWR@insubfloatrow}

```

```

106 \newenvironment*{subfloatrow}[1][2]
107 {

```

The row of floats is placed into a `div` of class `floatrow`:

```

108 \BlockClass{floatrow}

```

While inside the `floatrow`, `LWR@insubfloatrow` is set true, which tells `\floatbox` to use `\subfigure` or `\subtable`.

```

109 \begingroup
110 \booltrue{LWR@insubfloatrow}
111 }
112 {
113 \endgroup
114 \endBlockClass
115 \boolfalse{LWR@insubfloatrow}
116 }

```

## Package 16

# lwarp-geometry.sty

## 83 Geometry

Pkg geometry

for **HTML output**: Discard all options for `lwarp-geometry`:

```

1 \LWR@ProvidesPackageDrop{geometry}

```

## Package 17

# lwarp-graphics.sty

## 84 Graphics

Pkg graphics graphics is not used. The user is required to use `graphicx` instead.

for HTML output:

```
1 \PackageError{lwarp-graphics}
2 {The graphics package is not supported by
3 the lwarp package's HTML conversion.
4 Please use graphicx instead.    \protect\usepackage{graphicx}}
5 {Graphicx requires a different syntax but is more flexible,
6 and it is emulated by lwarp.}
```

## Package 18

# lwarp-graphicx.sty

## 85 Graphicx

Pkg `graphicx` `graphicx` is emulated during HTML output, and the `graphicx` package is ignored.

for **HTML output:** `1 \LWR@ProvidesPackageDrop{graphicx}`

## Package 19

# lwarp-hyperref.sty

## 86 Hyperref

Pkg hyperref hyperref is emulated during HTML output, and the hyperref package is ignored.

for HTML output:

```

1 % \LWR@ProvidesPackageDrop{hyperref}
2 \typeout{Using the lwarp html version of package 'hyperref' -- discarding options.}
3 \typeout{   Are not using ProvidesPackage, so that other packages}
4 \typeout{   do not attempt to patch lwarp's version of 'hyperref'.}
5 % \ProvidesPackage{lwarp-#1-#2}
6 \DeclareOption*{}
7 \ProcessOptions\relax

8 \newcommand*{\hypersetup}[1]{}
9 \newcommand*{\hyperbaseurl}[1]{}

```

Insert an image with alt text:

```

10 \NewDocumentCommand{\LWR@hyperimageb}{m +m}{%
11 \LWR@htmltag{img src="#1" alt="#2" class="hyperimage"{} }%
12 \endgroup%
13 \LWR@ensuredoingapar%
14 }
15
16 \newcommand{\hyperimage}{%
17 \LWR@ensuredoingapar%
18 \begingroup\catcode'\_ =12
19 \LWR@hyperimageb%
20 }

```

Creates an HTML anchor to category.name with the given text.

```

21 \NewDocumentCommand{\hyperdef}{m m +m}{%
22 \LWR@ensuredoingapar%
23 \LWR@subsublabel{#1.#2}%
24 #3%
25 }

```

Creates an HTML link to URL#category.name with the given text.

```

26 \NewDocumentCommand{\LWR@hyperrefb}{m m m +m}{%
27 \LWR@htmltag{a href="#1\LWR@hashmark#2.#3"%

```



```

28 #4%
29 \LWR@htmltag{/a}%
30 \endgroup%
31 }

```

Creates text as an HTML link to the L<sup>A</sup>T<sub>E</sub>X label.

```

32 \NewDocumentCommand{\LWR@hyperrefc}{0{label} +m}{%
33 \LWR@startref{#1}%
34 #2%
35 \LWR@htmltag{/a}%
36 \endgroup%
37 }

```

```

38 \newcommand{\hyperref}{%
39 \LWR@ensuredoingapar%
40 \begingroup\catcode'\_ =12
41 \@ifnextchar[\LWR@hyperrefc\LWR@hyperrefb%
42 }

```

Creates an anchor to name with the given text.

```

43 \NewDocumentCommand{\hypertarget}{m +m}{%
44 \label{#1}%
45 #2%
46 }

```

Creates a link to the anchor created by `hypertarget`, with the given link text.

```

47 \NewDocumentCommand{\hyperlink}{m +m}{%
48 \hyperref{#1}{#2}%
49 }

```

For HTML, `\cleverref` is used instead.

```

50 \NewDocumentCommand{\autoref}{s m}{%
51 \IfBooleanTF{#1}{\ref{#2}}{\cref{#2}}%
52 }

```

For HTML, `\cleverref` is used instead.

```

53 \NewDocumentCommand{\autopageref}{s m}{%
54 \IfBooleanTF{#1}{\cpageref{#2}}{\cref{#2}}%
55 }

```

```

56 \newcommand{\pdfstringdef}[2]{}

```

```

57 \newcommand{\pdfbookmark}[3][]{ }

```

```
58 \newcommand{\currentpdfbookmark}[2]{}

```

```
59 \newcommand{\subpdfbookmark}[2]{}

```

```
60 \newcommand{\belowpdfbookmark}[2]{}

```

```
61 \newcommand{\texorpdfstring}[2]{#2}

```

From hyperref.

```
62 \def\hypercalcbp#1{%
63 \strip@pt\dimexpr 0.99626401\dimexpr(#1)\relax\relax
64 }%

```

```
65 \newcommand{\Acrobatmenu}[2]{}

```

```
66 \newcommand*{\TextField}[2][{}]{

```

```
67 \newcommand*{\CheckBox}[2][{}]{

```

```
68 \newcommand{\ChoiceMenu}[3][{}]{

```

```
69 \newcommand*{\PushButton}[2][{}]{

```

```
70 \newcommand*{\Submit}[2][{}]{

```

```
71 \newcommand*{\Reset}[2][{}]{

```

```
72 \newcommand*{\LayoutTextField}[2]{}

```

```
73 \newcommand*{\LayoutChoiceField}[2]{}

```

```
74 \newcommand*{\LayoutCheckField}[2]{}

```

```
75 \newcommand*{\MakeRadioField}[2]{}

```

```
76 \newcommand*{\MakeCheckField}[2]{}

```

```
77 \newcommand*{\MakeTextField}[2]{}

```

```
78 \newcommand*{\MakeChoiceField}[2]{}

```

```
79 \newcommand{\MakeFieldButton}[1]{}

```

## Package 20

# lwarp-keyfloat.sty

## 87 Keyfloat

Pkg keyfloat keyfloat is supported with minor adjustments.

for HTML output: 1 \LWR@ProvidesPackagePass{keyfloat}

After keyfloat has loaded:

```

2 \AtBeginDocument{

3 % \newsavebox{\KFLT@envbox}
4
5 \let\KFLT@boxinner\relax
6 \let\endKFLT@boxinner\relax
7
8 \NewEnviron{KFLT@boxinner}
9 {%
10 \LWR@traceinfo{kflt@boxinner}%
11 \LWR@stoppars%
12 \KFLT@frame{\BODY}%
13 \LWR@startpars%
14 \LWR@traceinfo{ended kflt@boxinner}%
15 }

16 \RenewDocumentEnvironment{KFLT@marginfloat}{0{-1.2ex} m}
17 {% start
18 \LWR@maybeinthisfloat%
19 % \BlockClass{marginblock}
20 \LWR@stoppars%
21 \LWR@htmltag{div class="marginblock" id="autofloat-\arabic{LWR@thisfloat}}
22 \LWR@startpars%
23 \captionsetup{type=#2}%
24 }
25 {
26 % \endBlockClass
27 \LWR@htmldivclassend{div}
28 }

29 }% AtBeginDocument

```

## Package 21

# lwarp-letterspace.sty

## 88 Letterspace

Pkg `letterspace` `letterspace` is a subset of `microtype`, which is pre-loaded by `lwarp`. All user options and macros are ignored and disabled.

for **HTML output**: Discard all options for `lwarp-letterspace`:

```
1 \LWR@ProvidesPackageDrop{letterspace}

2 \newcommand*\lsstyle{}
3 \newcommand\textls[2][]{\textls[2]{}}
4 \def\textls#1#{}
5 \newcommand*\lslig[1]{#1}
```

## Package 22

# lwarp-listings.sty

## 89 Listings

Pkg listings listings is supported with some limitations. Text formatting is not yet supported.

for HTML output:

```
1 \begin{warpHTML}
```

```
2 \LWR@ProvidesPackagePass{listings}
```

Patches to embed listings inside `pre` tags:

```
3 \let\LWR@origlst@Init\lst@Init
```

```
4 \let\LWR@origlst@DeInit\lst@DeInit
```

```
5
```

```
6 \let\LWR@origlsthkEveryPar\lsthk@EveryPar
```

```
7
```

```
8 \renewcommand{\l@lstlisting}[2]{\hypertocfloat{1}{toclstlisting}{#1}{#2}}
```

Done at the start of a listing.

```
9 \renewcommand{\lst@Init}[1]{%
```

First, perform the listings initialization:

```
10 \LWR@traceinfo{\lst@Init}%
```

```
11 \renewcommand*{\@capttype}{lstlisting}%
```

```
12 \LWR@origlst@Init{#1}%
```

```
13 \LWR@traceinfo{finished origlst@Init}%
```

```
14 \lst@ifdisplaystyle%
```

Creating a display.

Disable line numbers, produce the `<pre>`, then reenable line numbers.

```
15 \LWR@traceinfo{About to create verbatim.}%
```

```
16 \let\lsthk@EveryPar\relax%
```

```
17 \LWR@atbeginverbatim{programlisting}%
```

```
18
```

```
19 \let\lsthk@EveryPar\LWR@origlsthkEveryPar%
```

```
20 \else%
```

Inline, so open a `<span>`

```

21 \ifbool{LWR@verbtags}{\LWR@htmltag{span class="inlineprogramlisting"}}{}%
22 \fi%
23 }

24 \renewcommand*{\lst@DeInit}{%
25 \lst@ifdisplaystyle%

```

Creating a display.

Disable line numbers, produce the `</pre>`, then reenable line numbers:

```

26 \let\lsthk@EveryPar\relax%
27
28 \LWR@afterendverbatim%
29 \let\lsthk@EveryPar\LWR@origlsthkEveryPar%
30 \else%

```

Inline, so create the closing `</span>`:

```

31 \ifbool{LWR@verbtags}{\noindent\LWR@htmltag{/span}}{}%
32 \fi%

```

Final listings deinit:

```

33 \LWR@origlst@DeInit%
34 }

```

This is called BOTH at the top and at the bottom of each listing.

Patched for lwarp.

```

35 \def\lst@MakeCaption#1{%
36 \LWR@traceinfo{MAKING CAPTION at #1}%
37 \lst@ifdisplaystyle
38 \LWR@traceinfo{making a listings display caption}%
39 \ifx #1t%
40 \ifx\lst@caption\@empty\expandafter\lst@HRefStepCounter \else
41 \expandafter\refstepcounter
42 \fi {lstlisting}%
43 \LWR@traceinfo{About to assign label: !\lst@label!}%
44 % \ifx\lst@label\@empty\else
45 % \label{\lst@label}\fi
46 \LWR@traceinfo{Finished assigning the label.}%
47 \let\lst@arg\lst@intname \lst@ReplaceIn\lst@arg\lst@filenamerpl
48 \global\let\lst@name\lst@arg \global\let\lstname\lst@name
49 \lst@ifnolol\else
50 \ifx\lst@caption\@empty
51 \ifx\lst@caption\@empty
52 \ifx\lst@intname\@empty \else \def\lst@temp{ }%

```

```
53 \ifx\lst@intname\lst@temp \else
```

This code places a contents entry for a non-float. This would have to be modified for lwarp:

```
54 \LWR@traceinfo{addcontents lst@name: -\lst@name-}%
55 % \addcontentsline{lol}{lstlisting}{\lst@name}
56 \fi\fi
57 \fi
58 \else
```

This would have to be modified for lwarp:

```
59 \LWR@traceinfo{addcontents lst@@caption: -\lst@@caption-}%
60 \addcontentsline{lol}{lstlisting}%
61 {\protect\numberline{\thelstlisting}{\protect\ignorespaces \lst@@caption \
62 \fi
63 \fi
64 \fi
65 \ifx\lst@caption\@empty\else
66 \LWR@traceinfo{lst@caption not empty-}%
67 \lst@ifsubstring #1\lst@captionpos
68 {\begingroup
69 \LWR@traceinfo{at the selected position}%
```

These space and box commands are not needed for HTML output:

```
70 % \let\@vskip\vskip
71 % \def\vskip{\afterassignment\lst@vskip \@tempskipa}%
72 % \def\lst@vskip{\nobreak\@vskip\@tempskipa\nobreak}%
73 % \par\@parboxrestore\normalsize\normalfont % \noindent (AS)
74 % \ifx #1t\allowbreak \fi
75 \ifx\lst@title\@empty
```

New lwarp code to create a caption:

```
76 \lst@makecaption\fnnum\lstlisting{\ignorespaces \lst@caption}
77 \else
```

New lwarp code to create a title:

```
78 % \lst@maketitle\lst@title % (AS)
79 \LWR@traceinfo{Making title: \lst@title}%
80 \begin{BlockClass}{lstlistingtitle}% lwarp
81 \lst@maketitle\lst@title% lwarp
82 \end{BlockClass}% lwarp
83 \fi
84 \LWR@traceinfo{About to assign label: !\lst@label!}%
85 \ifx\lst@label\@empty\else
```

```

86 \leavevmode% gets rid of bad space factor error
87 \GetTitleStringExpand{\lst@caption}%
88 \edef\LWR@lntemp{\GetTitleStringResult}%
89 \edef\@currentlabelname{\detokenize\expandafter{\LWR@lntemp}}%
90 \label{\lst@label}\fi
91 \LWR@traceinfo{Finished assigning the label.}%

```

Not needed for lwarp:

```

92 %           \ifx #1b\allowbreak \fi
93           \endgroup}{}%
94   \fi
95 \LWR@traceinfo{end of making a listings display caption}%
96   \else
97 \LWR@traceinfo{INLINE}%
98   \fi
99 \LWR@traceinfo{DONE WITH CAPTION at #1}%
100 }

```

Patched to keep left line numbers outside of the left margin, and place right line numbers in a field \VerbatimHTMLWidth wide.

```

101 \lst@Key{numbers}{none}{%
102   \let\lst@PlaceNumber\@empty
103   \lstKV@SwitchCases{#1}%
104   {none&\\%
105     left&\def\lst@PlaceNumber{%
106 % \llap{
107 \LWR@orignormalfont%
108 \lst@numberstyle{\thelstnumber}\kern\lst@numbersep%
109 % }
110 }
111 \\%
112   right&\def\lst@PlaceNumber{\rlap{\LWR@orignormalfont
113     \kern\VerbatimHTMLWidth \kern\lst@numbersep
114     \lst@numberstyle{\thelstnumber}}}%
115   }{\PackageError{Listings}{Numbers #1 unknown}\@ehc}}
116
117 \end{warpHTML}

```



## Package 23

# lwarp-longtable.sty

## 90 Longtable

Pkg `longtable` `longtable` is emulated during HTML output, and the `longtable` package is ignored.

for HTML output: `1 \LWR@ProvidesPackageDrop{longtable}`

To emulate `longtable`:



For `longtable` `\endhead`, `\endfoot`, and `\endlastfoot` rows, use

`\warpprintonly{row contents}`

instead of

`\begin{warpprint} ... \end{warpprint}`.

Doing so helps avoid “Misplaced `\noalign`.” when using `\begin{warpprint}`.

Keep the `\endfirsthead` row, which is still relevant to HTML output.

See:

<http://tex.stackexchange.com/questions/43006/why-is-input-not-expandable>

Env `longtable` \* `[<horizontal alignment>] {<colspec>}` Emulates the `longtable` environment.

Per the `caption` package, the starred version steps the counter per caption. The unstarred version steps the counter once at the beginning, but not at each caption.

Options `[c]`, `[l]`, and `[r]` are thrown away.

```

2 \newenvironment{longtable*}[2] [] {%
3 \LWR@floatbegin{table}%
4 \setcaptiontype{\LTcaption}%
5 \caption@setoptions{longtable}%
6 \caption@setoptions{@longtable}%
7 \caption@LT@setup%
8 \booltrue{LWR@starredlongtable}%
9 \let\captionlistentry\LWR@LTcaptionlistentry%
10 \LWR@tabular{#2}
11 }
12 {\endLWR@tabular\LWR@floatend}
13
14 \newenvironment{longtable}[2] [] {%
15 \LWR@floatbegin{table}%
16 \setcaptiontype{\LTcaption}%

```

---

```

17 \caption@setoptions{longtable}%
18 \caption@setoptions{@longtable}%
19 \caption@LT@setup%
20 \refstepcounter{\LTcaptype}%
21 \let\captionlistentry\LWR@LTcaptionlistentry%
22 \LWR@tabular{#2}
23 }
24 {\endLWR@tabular\LWR@floatend}
25

```

Provided for compatibility, but ignored:

```

26 \newcounter{LTchunksize}
27 \def\endhead{\LWR@tabularendofline}% throws away options //[dim] and /**
28 \def\endfirsthead{\LWR@tabularendofline}
29 \def\endfoot{\LWR@tabularendofline}
30 \def\endlastfoot{\LWR@tabularendofline}
31 \newcommand\tabularnewline{\LWR@tabularendofline}
32 \newcommand{\setlongtables}{}% Obsolete command, does nothing.
33 \newlength{\LTleft}
34 \newlength{\LTright}
35 \newlength{\LTpre}
36 \newlength{\LTpost}
37 \newlength{\LTcapwidth}

\kill is ignored, place a \kill line inside
    \begin{warpprint} ... \end{warpprint} or \warpingprintonly.

38 \renewcommand*{\kill}{\LWR@tabularendofline}

```

## Package 24

# lwarp-ltcaption.sty

## 91 Ltcaption

Pkg `ltcaption` `ltcaption` is emulated during HTML output, and the `ltcaption` package is ignored.

for HTML output: `1 \LWR@ProvidesPackageDrop{ltcaption}`

`\LTcaptype` is already defined by `lwarp`.

`longtable*` is already defined by `lwarp-longtable`.

```
2 \newlength{\LTcapskip}
3 \newlength{\LTcapleft}
4 \newlength{\LTcapright}
5 \newcommand*{\LTcapmarginfalse}{}

```

## Package 25


# lwarp-mdframed.sty

## 92 Mdframed

**Pkg** **mdframed** mdframed is loaded with options forced to **framemethod=none**.

**for HTML output:** 1 \LWR@ProvidesPackageDrop{mdframed}

**support** Most basic functionality is supported, including frame background colors and single-border colors and thickness, title and subtitle background colors and borders and thickness, border radius, and shadow. CSS classes are created for **mdframed** environments and frame titles.

 **loading** Only load **mdframed** in a **warpprint** environment. **lwarp** pre-loads **mdframed** in HTML with **framemethod=none**.

**font** For title font, use  
`frametitlefont=\textbf,`  
 instead of  
`frametitlefont=\bfseries,`  
 where `\textbf` must appear just before the comma and will receive the following text as its argument (since the text happens to be between braces in the **mdframed** source). Since **lwarp** does not support `\bfseries` and friends, only one font selection may be made at a time.

**theoremtitlefont** **theoremtitlefont** is not supported, since the following text is not in braces in the **mdframed** source.

**footnotes** Footnotes are currently placed at the bottom of the HTML page.

**ignored options** **userdefinedwidth** and **align** are currently ignored.

**CSS classes** Environments created or encapsulated by **mdframed** are enclosed in a **div** of class **md<environmentname>**, or **mdframed** otherwise.

Frame titles are placed into a **span** of class **mdframedtitle**. Subtitles are in a **span** of class **mdframedsubtitle**, and likewise for subsubtitles.

Pre-existing hooks are used to patch extra functions before and after the frames.

**amsthm** must be loaded before **mdframed**

2 \LWR@origRequirePackage{amsthm}

Do not require *Tikz* or *pstricks*:

```
3 \LWR@origRequirePackage[framemethod=none]{mdframed}
```

To handle CSS and paragraphs, patch code at start and end of environment and contents:

```
4 \mdfsetup{
5 startcode={\LWR@mdframedstart},
6 endcode={\LWR@mdframedend},
7 startinnercode={\LWR@startpars},
8 endinnercode={\LWR@stoppars},
9 }
```

Given the `mdframed` key, print the color.

```
10 \newcommand*{\LWR@mdfprintcolor}[1]{%
11 \convertcolorspec{named}{\csuse{mdf@#1}}{HTML}\LWR@tempcolor%
12 \#\LWR@tempcolor
13 }
```

Given the `mdframed` key, print the length.

```
14 \newcommand*{\LWR@mdfprintlength}[1]{%
15 \rndprintlength{\csuse{mdf@#1@length}}
16 }
```

Actions before an `mdframe` starts.

Encapsulate a frame inside a `div` of the desired `class`.

```
17 \newcommand*{\LWR@mdframedstart}{%
```

Turn off paragraph handling during the generation of the encapsulating tags:

```
18 \LWR@stoppars%
```

Below, print HTML pt units:

```
19 \uselengthunit{PT}%
```

Open a `div` and with custom `class` and custom `style`:

```
20 \LWR@htmltagc{div class="\LWR@mdthisenv" \LWR@orignewline
21 style=" \LWR@orignewline
```

Convert and print the background color:

```
22 background: \LWR@mdfprintcolor{backgroundcolor} ; \LWR@orignewline
```

Convert and print the border color and width:

```
23 border: \LWR@mdfprintlength{linewidth} solid
24 \LWR@mdfprintcolor{linecolor} ; \LWR@orignewline
```

Convert and print the border radius:

```
25 border-radius: \LWR@mdfprintlength{roundcorner} ; \LWR@orignewline
```

Convert and print the shadow:

```
26 \ifbool{mdf@shadow}{%
27 box-shadow:
28 \LWR@mdfprintlength{shadowsize}
29 \LWR@mdfprintlength{shadowsize}
30 \LWR@mdfprintlength{shadowsize}
31 \LWR@mdfprintcolor{shadowcolor} ;
32 }
33 {box-shadow: none ;}
34 \LWR@orignewline

35 "}
36 % \LWR@htmldivclass{\LWR@mdthisenv}
```

`mdframed` environment may not work with the modified `\hspace` and `\rule`, so restore them to their originals while inside `mdframed`:

```
37 \let\hspace\LWR@orighspace%
38 \let\rule\LWR@origrule%
39 }
```

Actions after an `mdframe` ends.

After closing the `div`, globally restore to the default environment type:

```
40 \newcommand*{\LWR@mdframedend}{}
```

Close the custom `div`:

```
41 \LWR@htmldivclassend{\LWR@mdthisenv}
```

Reset future custom class to the default:

```
42 \gdef\LWR@mdthisenv{mdframed}
```

Resume paragraph handling:

```
43 \LWR@startpars%
44 }
```

Encapsulation of the original which places the title inside a `span` of class `mdframedtitle`:

```
45 \let\LWR@origmdfframedtitleenv\mdfframedtitleenv
46
47 \newlength{\LWR@titleroundcorner}
48
49 \renewrobustcmd\mdfframedtitleenv[1]{%
50 \LWR@origmdfframedtitleenv{%
```

Below, print HTML pt lengths:

```
51 \uselengthunit{PT}%
```

Open a `span` with a custom class and custom style:

```
52 \LWR@htmltagc{span class="mdframedtitle" \LWR@orignewline
53 style=" \LWR@orignewline
```

Convert and print the title background color:

```
54 background:
55 \LWR@mdfprintcolor{frametitlebackgroundcolor}
56 ; \LWR@orignewline
```

Convert and print the title rule:

```
57 \ifbool{mdf@frametitlerule}{%
58 border-bottom:
59 \LWR@mdfprintlength{frametitlerulewidth}
60 solid
61 \LWR@mdfprintcolor{frametitlerulecolor}
62 ; \LWR@orignewline
63 }{}%
```

The title's top border radius is adjusted for the line width:

```
64 border-radius:
65 \setlength{\LWR@titleroundcorner}
66 {\maxof{\mdf@roundcorner@length-\mdf@linewidth@length}{0pt}}
67 \rndprintlength{\LWR@titleroundcorner}
68 \rndprintlength{\LWR@titleroundcorner}
69 0pt 0pt
70 \LWR@orignewline
```

Finish the custom style and the opening span tag:

```
71 " \LWR@orignewline
72 }% span
```

Restrict paragraph tags inside a span:

```
73 \begin{LWR@nestspan}%
```

Print the title inside the span:

```
74 #1%
```

Close the span and unnest the paragraph tag restriction:

```
75 \LWR@htmltagc{/span}%
```

```
76 \end{LWR@nestspan}%
```

```
77 }
```

```
78 }
```

Common code for `\LWR@mdfsubtitle` and `\LWR@mdfsubsubtitle`.

Encapsulate the subtitle inside a `span` of class `mdframedsubtitle`:

```
79 \NewDocumentCommand{\LWR@mdfsubtitlecommon}{m o m}
```

```
80 {% the following empty line is required
```

```
81
```

Special handling for `mdframed`: Subtitles have `\pars` around them, so temporarily disable them here.

```
82 \let\par\LWR@origpar%
```

Open a `span` with a custom class and custom style:

```
83 \LWR@htmltagc{span class="mdframed#1title"
```

```
84 style=" \LWR@orignewline
```

Convert and print the background color:

```
85 background:
```

```
86 \LWR@mdfprintcolor{#1titlebackgroundcolor}
```

```
87 ; \LWR@orignewline
```

Convert and print the above line:

```
88 \ifbool{mdf@#1titleaboveline}{%
```

```
89 border-top:
```

```
90 \LWR@mdfprintlength{#1titleabovelinewidth}
```

```
91 solid
```

```
92 \LWR@mdfprintcolor{#1titleabovelinecolor}
```

```
93 ; \LWR@orignewline
```

```
94 }{}%
```



Convert and print the below line:

```

95 \ifbool{mdf@#1titlebelowline}{%
96 border-bottom:
97 \LWR@mdfprintlength{#1titlebelowlinewidth}
98 solid
99 \LWR@mdfprintcolor{#1titlebelowlinecolor}
100 ; \LWR@orignewline
101 }{}%
```

Finish the custom style and the opening span tag:

```

102 "% span
```

Restrict paragraph tags inside a span:

```

103 \begin{LWR@nestspan}%
```

Perform the original subtitle action:

```

104 \IfNoValueTF{#2}
105 {\csuse{LWR@origmdf#1title}{#3}}%
106 {\csuse{LWR@origmdf#1title}[#2]{#3}}%
```

Close the span and unnest the paragraph tag restriction:

```

107 \LWR@htmltagc{/span}% the following empty line is required
108 \end{LWR@nestspan}% must follow the /span or an extra <p> appears
109
110 }
```

```

111 \let\LWR@origmdfsubtitle\mdfsubtitle
112
113 \newcommand*{\LWR@mdfsubtitle}{%
114 \LWR@mdfsubtitlecommon{sub}%
115 }
116 \let\mdfsubtitle\LWR@mdfsubtitle

117 \let\LWR@origmdfsubsubtitle\mdfsubsubtitle
118
119 \newcommand*{\LWR@mdfsubsubtitle}{%
120 \LWR@mdfsubsubtitlecommon{subsub}%
121 }
122 \let\mdfsubsubtitle\LWR@mdfsubsubtitle
```

Stores the environment of the frame about to be created:

```

123 \newcommand*{\LWR@mdthisenv}{mdframed}
```

Modified from the original to remember the environment.

```

124 \renewrobustcmd*\newmdenv[2] [] {%
125 \newenvironment{#2}%
126 {%
127 \mdfsetup{#1}%
128 \renewcommand*\LWR@mdthisenv}{md#2}%
129 \begin{mdframed}%
130 }
131 {\end{mdframed}}%
132 }

```

Modified from the original to remember the environment.

```

133 \renewrobustcmd*\surroundwithmdframed[2] [] {%
134 \BeforeBeginEnvironment{#2}{%
135 \renewcommand*\LWR@mdthisenv}{md#2}%
136 \begin{mdframed}[#1]}%
137 \AfterEndEnvironment{#2}{\end{mdframed}}%
138 }

```

[*numberedlike*] {*caption*} [*within*]

Modified from the original to remember the environment.

```

139 \let\LWR@origmdtheorem\mdtheorem
140
141 \DeclareDocumentCommand{\LWR@mdtheorem}{0}{ m o m o }{%
142 \LWR@origmdtheorem[#1]{#2}[#3]{#4}[#5}%
143 \BeforeBeginEnvironment{#2}{\renewcommand*\LWR@mdthisenv}{md#2}}%
144 }
145
146 \let\mdtheorem\LWR@mdtheorem

```

[*numberedlike*] {*caption*} [*within*]

Modified from the original to remember the environment.

```

147 \DeclareDocumentCommand\newmdtheoremenv{0}{ m o m o }{%
148 \ifboolexpr{ test {\IfNoValueTF {#3}} and test {\IfNoValueTF {#5}} }%
149   {\newtheorem{#2}{#4}}{%
150     \IfValueTF{#3}{\newtheorem{#2}[#3]{#4}}{%
151       \IfValueTF{#5}{\newtheorem{#2}{#4}[#5]}{%
152         }%
153     }%
154   \BeforeBeginEnvironment{#2}{%
155     \renewcommand*\LWR@mdthisenv}{md#2}%
156     \begin{mdframed}[#1]}%
157   \AfterEndEnvironment{#2}{%
158     \end{mdframed}}%
159 }

```

## Package 26

# lwarp-microtype.sty

## 93 Microtype

Pkg microtype microtype is pre-loaded by lwarp. All user options and macros are ignored and disabled.

for HTML output: Discard all options for lwarp-microtype:

```

1 \LWR@ProvidesPackageDrop{microtype}

2 \newcommand*\DeclareMicrotypeSet[3][]{\relax}
3 \newcommand*\UseMicrotypeSet[2][]{\relax}
4 \newcommand*\DeclareMicrotypeSetDefault[2][]{\relax}
5 \newcommand*\SetProtrusion[3][]{\relax}
6 \newcommand*\SetExpansion[3][]{\relax}
7 \newcommand*\SetTracking[3][]{\relax}
8 \newcommand*\SetExtraKerning[3][]{\relax}
9 \newcommand*\SetExtraSpacing[3][]{\relax}
10 \newcommand*\DisableLigatures[2][]{\relax}
11 \newcommand*\DeclareCharacterInheritance[3][]{\relax}
12 \newcommand*\DeclareMicrotypeVariants[1]{\relax}
13 \newcommand*\DeclareMicrotypeAlias[2]{\relax}
14 \newcommand*\LoadMicrotypeFile[1]{\relax}
15 \newcommand*\DeclareMicrotypeBabelHook[2]{\relax}
16 \newcommand*\microtypesetup[1]{\relax}
17 \newcommand*\microtypecontext[1]{\relax}
18 \newcommand*\textmicrotypecontext[2]{\relax}
19 \@ifpackageloaded{letterspace}{\let\MT@textls\relax}{\relax}
20 \newcommand*\lsstyle{\relax}
21 \newcommand*\textls[2][]{\relax}
22 \def\textls#1{\relax}
23 \newcommand*\lslig[1]{\relax}
24 }
25 \def\DeclareMicrotypeSet#1#2{\gobbletwo}
26 \def\DeclareMicrotypeVariants#1#2{\gobble}
27 \@onlypreamble\DeclareMicrotypeSet
28 \@onlypreamble\UseMicrotypeSet
29 \@onlypreamble\DeclareMicrotypeSetDefault
30 \@onlypreamble\DisableLigatures
31 \@onlypreamble\DeclareMicrotypeVariants
32 \@onlypreamble\DeclareMicrotypeBabelHook

```

## Package 27

# lwarp-multicol.sty

## 94 Multicol

Pkg multicol multicol is emulated during HTML output, and the multicol package is ignored.

for HTML output: 1 \LWR@ProvidesPackageDrop{multicol}[2015/09/13]

Multicols are converted into a 1–3 column display, browser-supported.

The optional multicols heading is placed inside a div of class `multicolsheading`.

The content is placed inside a div of class `multicols`.

for HTML output: 2 \begin{warpHTML}

3 \NewDocumentEnvironment{multicols}{s m o}

HTML div class to contain everything:

4 {\BlockClass{multicols}

Optional HTML div class for the heading:

5 \IfValueTF{#3}{\begin{BlockClass}{multicolsheading}#3\end{BlockClass}}{}}

When done with the environment, close the div:

6 {\endBlockClass}

Emulated null functions which are not used in HTML:

```

7 \newcommand*{\columnbreak}{}
8 \newcommand*{\RLmulticolcolumns}{}
9 \newcommand*{\LRmulticolcolumns}{}
10
11 \newlength{\premulticols}
12 \newlength{\postmulticols}
13 \newlength{\multicolsep}
14 \newlength{\multicolbaselineskip}
15 \newlength{\multicoltolerance}
16 \newlength{\multicolpretolerance}
17 \newcommand*{\columnseprulecolor}{\normalcolor}
18 \newcounter{columnbadness}

```

---

```
19 \newcounter{finalcolumnbadness}
20 \newcounter{collectmore}
21 \newcounter{unbalance}
22 \newlength{\multicolovershoot}
23 \newlength{\multicolundershoot}

24 \end{warpHTML}
```

## Package 28

# lwarp-multirow.sty

## 95 Multirow

Pkg `multirow` `multirow` is emulated during HTML output, and the `multirow` package is ignored.

for HTML output: `1 \LWR@ProvidesPackageDrop{multirow}`

## Package 29

# lwarp-needspace.sty

## 96 Needspace

Pkg `needspace` `needspace` is not used during HTML conversion.

for HTML output: Discard all options for `lwarp-needspace`:

```
1 \LWR@ProvidesPackageDrop{needspace}
2
3 \newcommand*{\needspace}[1]{ }
4 \DeclareDocumentCommand{\Needspace}{s m}{ }
```

## Package 30

# lwarp-nowidow.sty

## 97 Nowidow

Pkg `nowidow` `nowidow` is not used during HTML conversion.

for HTML output: Discard all options for `lwarp-nowidow`:

```
1 \LWR@ProvidesPackageDrop{nowidow}
```

```

2 \newcommand*\nowidow[1] [] {}
3 \newcommand*\setnowidow[1] [] {}

4 \newcommand*\noclub[1] [] {}
5 \newcommand*\setnoclub[1] [] {}

```

## Package 31

# lwarp-pagenote.sty

## 98 Pagenote

**Pkg** **pagenote** pagenote is used by lwarp, and its user-interface macros have been nullified.

Both footnotes and pagenotes appear as footnotes in the HTML output.

Any manipulation of **pagenote** in the user's document after the preamble should be encased within a **warpprint** environment.

**for HTML output:** Discard all options for lwarp-pagenote:

```
1 \LWR@ProvidesPackageDrop{pagenote}
```

**\pagenote** is already modified to work with lwarp.

```

2 \renewcommand*\makepagenote{}
3 \RenewDocumentCommand{\printnotes}{s}{}
4 \renewcommand{\printnotes}[2] [] {}
5 \renewcommand{\addtonotes}[1] {}

```

## Package 32

# lwarp-placeins.sty

## 99 Placeins

Pkg placeins placeins is not used during HTML conversion.

for HTML output: Discard all options for lwarp-placeins:

```
1 \LWR@ProvidesPackageDrop{placeins}
```

```
2 \newcommand*{\FloatBarrier}{} 
```



## Package 33

# lwarp-ragged2e.sty

## 100 Ragged2e

Pkg ragged2e ragged2e is not used during HTML conversion.

for HTML output: Discard all options for lwarp-ragged2e:

```

1 \LWR@ProvidesPackageDrop{ragged2e}

2 \newcommand*{\Centering}{\centering}
3 \newcommand*{\RaggedLeft}{\raggedleft}
4 \newcommand*{\RaggedRight}{\raggedright}
5 \newcommand*{\justifying}{\}
6 \newlength{\CenteringLeftskip}
7 \newlength{\RaggedLeftLeftskip}
8 \newlength{\RaggedRightLeftskip}
9 \newlength{\CenteringRightskip}
10 \newlength{\RaggedLeftRightskip}
11 \newlength{\RaggedRightRightskip}
12 \newlength{\CenteringParfillskip}
13 \newlength{\RaggedLeftParfillskip}
14 \newlength{\RaggedRightParfillskip}
15 \newlength{\JustifyingParfillskip}
16 \newlength{\CenteringParindent}
17 \newlength{\RaggedLeftParindent}
18 \newlength{\RaggedRightParindent}
19 \newlength{\JustifyingParindent}
20 \newenvironment*{Center}{\center}{\endcenter}
21 \newenvironment*{FlushLeft}{\flushleft}{\endflushleft}
22 \newenvironment*{FlushRight}{\flushright}{\endflushright}
23 \newenvironment*{justify}{\justifying}{\endjustifying}

```

## Package 34

# lwarp-rotating.sty

## 101 Rotating

Pkg rotating rotating is emulated during HTML output, and the rotating package is ignored.

This code is untested.

for HTML output:

```

1 \LWR@ProvidesPackageDrop{rotating}

2 \let\sidewaystable\table
3 \let\endsidewaystable\endtable
4
5 \let\sidewaysfigure\figure
6 \let\endsidewaysfigure\endfigure
7
8 \newenvironment*{sideways}{}{}
9 \newenvironment*{turn}[1]{}{}
10 \newenvironment*{rotate}[1]{}{}
11 \NewDocumentCommand{\turnbox}{m +m}{#2}
12 \let\rotcaption\caption
13 \let\@makerotcaption\@makecaption

```

## Package 35

# lwarp-setspace.sty

## 102 Setspace

Pkg setspace setspace is not used during HTML conversion.

for HTML output: Discard all options for lwarp-setspace:

```

1 \LWR@ProvidesPackageDrop{setspace}
2
3 \newcommand*{\setstretch}[1]{}
4 \newcommand*{\SetSingleSpace}[1]{}
5 \newcommand*{\singleSpacing}{}
6 \newcommand*{\onehalfSpacing}{}

```

```

7 \newcommand*\doublespacing{}
8
9 \newenvironment*{singlespace}
10 {\BlockClass{singlespace}}
11 {\endBlockClass}
12
13 \newenvironment*{singlespace*}
14 {\BlockClass{singlespace}}
15 {\endBlockClass}
16
17 \newenvironment*{spacing}[1]{
18
19 }{
20
21 }
22
23 \newenvironment*{onehalfspace}
24 {\BlockClass{onehalfspace}}
25 {\endBlockClass}
26
27 \newenvironment*{doublespace}
28 {\BlockClass{doublespace}}
29 {\endBlockClass}

```

## Package 36

# lwarp-textpos.sty

## 103 Textpos

Pkg `textpos` `textpos` is emulated during HTML output, and the `textpos` package is ignored.

```

1 \LWR@ProvidesPackageDrop{textpos}

for HTML output:
2 \NewDocumentEnvironment{textblock}{m r()}{\}{}
3 \NewDocumentEnvironment{textblock*}{m o r()}{\}{}
4 \newcommand*\TPGrid[3][\]{}
5 \NewDocumentCommand{\TPMargin}{s o}{}
6 \newcommand*\textblockcolour[1]{}
7 \newcommand*\textblockrulecolour[1]{}
8 \newcommand*\textblockcolor[1]{}
9 \newcommand*\textblockrulecolor[1]{}
10 \newcommand*\tekstblokkulur[1]{}
11 \newcommand*\tekstblokrulekulur[1]{}

```

---

```
12 \newlength{\TPHorizModule}
13 \newlength{\TPVertModule}
14 \newlength{\TPboxrulesize}
15 \newcommand{\textblocklabel}[1]{\}
16 \newcommand*{\showtextsize}{\}
17 \newcommand{\textblockorigin}[2]{\}
```

## Package 37

# lwarp-threeparttable.sty

## 104 Threeparttable

Pkg `threeparttable` `threeparttable` is emulated during HTML output, and the `threeparttable` package is ignored.

for HTML output: `1 \LWR@ProvidesPackageDrop{threeparttable}`

Prints the table note item header inside a CSS class of `tnoteitemheader`.

```
2 \newcommand{\LWR@printtablenote}[1]{\InlineClass{tnoteitemheader}{#1}}
```

To emulate `threeparttable`:

```
3 \newenvironment*{threeparttable}[1][b]{}{}
```

```
4 \newenvironment*{tablenotes}[1] []
```

```
5 {%
```

```
6 \BlockClass{tnotes}%
```

```
7 \setlist[description]{format=\LWR@printtablenote}%
```

```
8 \description%
```

```
9 }
```

```
10 {%
```

```
11 \enddescription%
```

```
12 \endBlockClass%
```

```
13 }
```

```
14 \newcommand{\tnote}[1]{\textsuperscript{#1}}
```

## Package 38

# lwarp-titles.sty

## 105 Titles

Pkg titles titles is loaded and used by lwarp during HTML output. All user options and macros are ignored and disabled.

for HTML output: Discard all options for lwarp-titles:

```

1 \LWR@ProvidesPackageDrop{titles}

\pagestyle and \thispagestyle are already disabled in the lwarp code.

2 \RenewDocumentCommand{\newpagestyle}{m o m}{}
3 \RenewDocumentCommand{\renewpagestyle}{m o m}{}

4 \RenewDocumentCommand{\sethead}{o o o m m m}{}
5 \RenewDocumentCommand{\setfoot}{o o o m m m}{}

6 \RenewDocumentCommand{\settitledmarks}{s m}{}

7 \renewcommand*{\headrule}{}
8 \renewcommand*{\footrule}{}

9 \renewcommand*{\setheadrule}[1]{}
10 \renewcommand*{\setfootrule}[1]{}

11 \newcommand*{\makeheadrule}{}
12 \newcommand*{\makefootrule}{}

13 \renewcommand{\setmarkboth}[1]{}

14 \RenewDocumentCommand{\widenhead}{s o o m m}{}

15 \renewcommand*{\bottitledmarks}{}
16 \renewcommand*{\toptitledmarks}{}
17 \renewcommand*{\firsttitledmarks}{}
18 \renewcommand*{\nexttoptitledmarks}{}
19 \renewcommand*{\outertitledmarks}{}
20 \renewcommand*{\innertitledmarks}{}

21 \RenewDocumentCommand{\newtitledmark}{s m}{}

```

---

```
22 \RenewDocumentCommand{\pretitlemark}{s m m}{}

23 \renewcommand{\ifsamemark}[4]{}

24 \NewDocumentCommand{\setfloathead}{s o o o m m m m m}{}
25 \NewDocumentCommand{\setfloatfoot}{s o o o m m m m m}{}

26 \NewDocumentCommand{\nextfloathead}{s o o o m m m m m}{}
27 \NewDocumentCommand{\nextfloatfoot}{s o o o m m m m m}{}

28 \newcommand{\newmarkset}[1]{}

29 \NewDocumentCommand{\newextramarkset}{s m m}{}

30 \newcommand{\botextramarks}[1]{}
31 \newcommand{\topextramarks}[1]{}
32 \newcommand{\firstextramarks}[1]{}
33 \newcommand{\nexttoextramarks}[1]{}
34 \newcommand{\outerextramarks}[1]{}
35 \newcommand{\innerextramarks}[1]{}

```

## Package 39

# lwarp-titlesec.sty

## 106 Titlesec

Pkg titlesec titlesec is emulated. All user options and macros are ignored and disabled.  
 for HTML output: Discard all options for lwarp-titlesec:

```

1 \LWR@ProvidesPackageDrop{titlesec}

2 \newcommand*{\titlelabel}[1]{}

3 \newcommand\titleformat{%
4   \@ifstar{\ttl@format@s}%
5     {\ttl@format@i}}
6 \newcommand{\ttl@format@s}[1]{}
7 \NewDocumentCommand{\ttl@format@i}{m o m m m o}{}

8 \@ifundefined{chapapp}{\let\@chapapp\chaptername}{}
9 \newcommand\chaptertitlename{\@chapapp}

10 \NewDocumentCommand{\titlespacing}{s m m m m o}{}

11 \newcommand*{\filright}{}
12 \newcommand*{\filcenter}{}
13 \newcommand*{\filleft}{}
14 \newcommand*{\fillast}{}
15 \newcommand*{\filinner}{}
16 \newcommand*{\filouter}{}

17 \newcommand\wordsep{\fontdimen\tw@\font \@plus
18   \fontdimen\thr@\font \@minus \fontdimen4\font}

19 \NewDocumentCommand{\titleline}{s o m}{}

20 \providecommand*{\titlerule}{\@ifstar{\ttl@row}{\ttl@rule}}
21 \newcommand*{\ttl@rule}[1] [] {}
22 \newcommand*{\ttl@row}[2] [] {}

23 \newcommand{\iftitlemeasuring}[2] {#2}

24 \newcommand{\assignpagestyle}[2] {#2}

25 \NewDocumentCommand{\titleclass}{m o m o}

```



## Package 40

# lwarp-titletoc.sty

## 107 Titletoc

Pkg titletoc titletoc is emulated. All user options and macros are ignored and disabled.  
 for HTML output: Discard all options for lwarp-titletoc:

```

1 \LWR@ProvidesPackageDrop{titletoc}

2 \NewDocumentCommand{\dottedcontents}{m o m m m}{\}

3 \newcommand{\titlecontents}{\@ifstar{\ttl@tcstar}{\ttl@tcnostar}}
4 \NewDocumentCommand{\ttl@tcstar}{m o m m m o o}{\}
5 \NewDocumentCommand{\ttl@tcnostar}{m o m m m m o}{\}

6 \newcommand{\contentsmargin}[2] [] {}

7 \newcommand*{\thecontentslabel}{thecontentslabel}
8 \newcommand*{\thecontentspage}{thecontentspage}

9 \newcommand{\contentslabel}[2] [] {\thecontentslabel}
10 \newcommand{\contentspage}[1] [] {\thecontentspage}

11 \newcommand{\contentspush}[1] {}

12 \newcommand{\contentsuse}[2] {}

13 \newcommand*{\startcontents}[1] [] {}
14 \newcommand*{\stopcontents}[1] [] {}
15 \newcommand*{\resumecontents}[1] [] {}

16 \newcommand{\printcontents}[4] [] {}

17 \newcommand{\startlist}[2] [] {}
18 \newcommand{\stoplist}[2] [] {}
19 \newcommand{\resumelist}[2] [] {}

20 \newcommand{\printlist}[4] [] {}

```

## Package 41

# lwarp-titling.sty

## 108 Titling

**Pkg titling** titling is used by lwarp. The following patches are not needed by lwarp, but are required if the user requests titling.

lwarp uses page notes for footnotes, so the various titling footnote restyling commands have no effect.

**for HTML output:** Pass all options to lwarp-titling:

```
1 \LWR@ProvidesPackagePass{titling}
```

Patch \@bsmttitleempty:

```
2 \let\LWR@orig@bsmttitleempty\@bsmttitleempty
3 \renewcommand*{\@bsmttitleempty}{%
4 \LWR@orig@bsmttitleempty%
5 \global\let\published\relax%
6 \global\let\subtitle\relax%
7 }
```

Patch \keepthetitle:

```
8 \let\LWR@origkeepthetitle\keepthetitle
9 \renewcommand*{\keepthetitle}{%
10 \LWR@orig@keepthetitle%
11 \global\let\@published\@empty%
12 \global\let\@subtitle\@empty%
13 }
```

Patch \killtitle:

```
14 \let\LWR@origkilltitle\killtitle
15 \renewcommand*{\killtitle}{%
16 \LWR@orig@killtitle%
17 \global\let\thepublished\relax%
18 \global\let\thesubtitle\relax%
19 }
```

## Package 42

# lwarp-tocloft.sty

## 109 Tocloft

Pkg `tocloft` `tocloft` is emulated. Most user options and macros are ignored and disabled. `\newlistof` and `\cftchapterprecis` are supported.

for HTML output: Discard all options for `lwarp-tocloft`:

```

1 \LWR@ProvidesPackageDrop{tocloft}

2 \newcommand{\tocloftpagestyle}[1]{}

3 \newcommand*\cftmarktoc{}
4 \newcommand*\cfttoctitlefont{}
5 \newcommand*\cftaftertoctitle{}

6 \newlength{\cftbeforetoctitleskip}
7 \newlength{\cftaftertoctitleskip}

8 \newcommand*\cftmarklof{}
9 \newcommand*\cftloftitlefont{}
10 \newcommand*\cftafterloftitle{}

11 \newlength{\cftbeforeloftitleskip}
12 \newlength{\cftafterloftitleskip}

13 \newcommand*\cftmarklot{}
14 \newcommand*\cftlottitlefont{}
15 \newcommand*\cftafterlottitle{}

16 \newlength{\cftbeforelottitleskip}
17 \newlength{\cftafterlottitleskip}

18 \newcommand*\cftdot}{.}
19 \providecommand*\cftdotsep}{1}
20 \newcommand*\cftnodots}{5000}
21
22 \providecommand{\cftdotfill}[1]{}

23 \newcommand*\cftsetpnumwidth}[1]{}
24 \newcommand*\cftsetrmarg}[1]{}

```

```
25 \newcommand*{\cftpnumalign}[1]{}

26 \newlength{\cftparskip}

27 \newlength{\cftbeforepartskip}
28 \newlength{\cftpartincent}
29 \newlength{\cftpnumwidth}
30 \newcommand*{\cftpfont}{}
31 \newcommand*{\cftpapresnum}{}
32 \newcommand*{\cftpataftersnum}{}
33 \newcommand*{\cftpataftersnumb}{}
34 \newcommand*{\cftpleader}{}
35 \newcommand*{\cftpdotsep}{1}
36 \newcommand*{\cftpfont}{}
37 \newcommand*{\cftpartafterpnum}{}

38 \newlength{\cftbeforechapskip}
39 \newlength{\cftchapindent}
40 \newlength{\cftchapnumwidth}
41 \newcommand*{\cftchapfont}{}
42 \newcommand*{\cftchappresnum}{}
43 \newcommand*{\cftchapaftersnum}{}
44 \newcommand*{\cftchapaftersnumb}{}
45 \newcommand*{\cftchapleader}{}
46 \newcommand*{\cftchapdotsep}{1}
47 \newcommand*{\cftchapfont}{}
48 \newcommand*{\cftchapafterpnum}{}

49 \newlength{\cftbeforesecskip}
50 \newlength{\cftsecindent}
51 \newlength{\cftsecnumwidth}
52 \newcommand*{\cftsecfont}{}
53 \newcommand*{\cftsecpresnum}{}
54 \newcommand*{\cftsecaftersnum}{}
55 \newcommand*{\cftsecaftersnumb}{}
56 \newcommand*{\cftsecleader}{}
57 \newcommand*{\cftsecdotsep}{1}
58 \newcommand*{\cftsecfont}{}
59 \newcommand*{\cftsecafterpnum}{}

60 \newlength{\cftbeforesubsecskip}
61 \newlength{\cftsubsecindent}
62 \newlength{\cftsubsecnumwidth}
63 \newcommand*{\cftsubsecfont}{}
64 \newcommand*{\cftsubsecpresnum}{}
65 \newcommand*{\cftsubsecaftersnum}{}
66 \newcommand*{\cftsubsecaftersnumb}{}
67 \newcommand*{\cftsubsecleader}{}
68 \newcommand*{\cftsubsecdotsep}{1}
```

```
69 \newcommand*{\cftsubsecpagefont}{}
70 \newcommand*{\cftsubsecafterpnum}{}

71 \newlength{\cftbeforesubsubsecskip}
72 \newlength{\cftsubsubsecindent}
73 \newlength{\cftsubsubsecnumwidth}
74 \newcommand*{\cftsubsubsecfont}{}
75 \newcommand*{\cftsubsubsecpresnum}{}
76 \newcommand*{\cftsubsubsecaftersnum}{}
77 \newcommand*{\cftsubsubsecaftersnumb}{}
78 \newcommand*{\cftsubsubsecleader}{}
79 \newcommand*{\cftsubsubsecdotsep}{1}
80 \newcommand*{\cftsubsubsecpagefont}{}
81 \newcommand*{\cftsubsubsecafterpnum}{}

82 \newlength{\cftbeforeparaskip}
83 \newlength{\cftparaindent}
84 \newlength{\cftparanumwidth}
85 \newcommand*{\cftparafont}{}
86 \newcommand*{\cftparapresnum}{}
87 \newcommand*{\cftparaftersnum}{}
88 \newcommand*{\cftparaftersnumb}{}
89 \newcommand*{\cftparaleader}{}
90 \newcommand*{\cftparadotsep}{1}
91 \newcommand*{\cftparapagefont}{}
92 \newcommand*{\cftparaafterpnum}{}

93 \newlength{\cftbeforesubparaskip}
94 \newlength{\cftsubparaindent}
95 \newlength{\cftsubparanumwidth}
96 \newcommand*{\cftsubparafont}{}
97 \newcommand*{\cftsubparapresnum}{}
98 \newcommand*{\cftsubparaftersnum}{}
99 \newcommand*{\cftsubparaftersnumb}{}
100 \newcommand*{\cftsubparaleader}{}
101 \newcommand*{\cftsubparadotsep}{1}
102 \newcommand*{\cftsubparapagefont}{}
103 \newcommand*{\cftsubparaafterpnum}{}

104 \newlength{\cftbeforefigskip}
105 \newlength{\cftfigindent}
106 \newlength{\cftfignumwidth}
107 \newcommand*{\cftfigfont}{}
108 \newcommand*{\cftfigpresnum}{}
109 \newcommand*{\cftfigaftersnum}{}
110 \newcommand*{\cftfigaftersnumb}{}
111 \newcommand*{\cftfigleader}{}
112 \newcommand*{\cftfigdotsep}{1}
113 \newcommand*{\cftfigpagefont}{}

```

```

114 \newcommand*\cftfigafterpnum){}

115 \newlength{\cftbeforesubfigskip}
116 \newlength{\cftsubfigindent}
117 \newlength{\cftsubfignumwidth}
118 \newcommand*\cftsubfigfont){}
119 \newcommand*\cftsubfigpresnum){}
120 \newcommand*\cftsubfigaftersnum){}
121 \newcommand*\cftsubfigaftersnumb){}
122 \newcommand*\cftsubfigleader){}
123 \newcommand*\cftsubfigdotsep}{1}
124 \newcommand*\cftsubfigpagefont){}
125 \newcommand*\cftsubfigafterpnum){}

126 \newlength{\cftbeforetabskip}
127 \newlength{\cfttabindent}
128 \newlength{\cfttabnumwidth}
129 \newcommand*\cfttabfont){}
130 \newcommand*\cfttabpresnum){}
131 \newcommand*\cfttabaftersnum){}
132 \newcommand*\cfttabaftersnumb){}
133 \newcommand*\cfttableader){}
134 \newcommand*\cfttabdotsep}{1}
135 \newcommand*\cfttabpagefont){}
136 \newcommand*\cfttabafterpnum){}

137 \newlength{\cftbeforesubtabskip}
138 \newlength{\cftsubtabindent}
139 \newlength{\cftsubtabnumwidth}
140 \newcommand*\cftsubtabfont){}
141 \newcommand*\cftsubtabpresnum){}
142 \newcommand*\cftsubtabaftersnum){}
143 \newcommand*\cftsubtabaftersnumb){}
144 \newcommand*\cftsubtableader){}
145 \newcommand*\cftsubtabdotsep}{1}
146 \newcommand*\cftsubtabpagefont){}
147 \newcommand*\cftsubtabafterpnum){}

148 \newcommand{\cftsetindents}[3]{}

149 \newcommand{\pagenumbersoff}[1]{}
150 \newcommand{\pagenumberson}[1]{}

```

Emulated through the \newfloat mechanism.

```

151 \NewDocumentCommand{\newlistof}{o m m}
152 {%
153 \IfValueTF{#1}

```

```

154 {\newfloat{#2}{tbp}{#3}[#1]}
155 {\newfloat{#2}{tbp}{#3}}
156 \@namedef{listof#2}{\listof{#2}{#4}}
157 \@namedef{#2depth}{1}
158 \expandafter\newlength\csuse{cftbefore#2skip}
159 \expandafter\newlength\csuse{cft#2indent}
160 \expandafter\newlength\csuse{cft#2numwidth}
161 \@namedef{cft#2font}{}
162 \@namedef{cft#2presnum}{}
163 \@namedef{cft#2aftersnum}{}
164 \@namedef{cft#2aftersnumb}{}
165 \@namedef{cft#2leader}{}
166 \@namedef{cft#2dotsep}{1}
167 \@namedef{cft#2pagefont}{}
168 \@namedef{cft#2afterpnum}{}
169 }

```

\cftchapterprecis from tocloft:

```

170 \newcommand{\cftchapterprecis}[1]{%
171   \cftchapterprecishere{#1}
172   \cftchapterprecistoc{#1}}
173 \newcommand{\cftchapterprecishere}[1]{%
174   \begin{quote}\textit{#1}\end{quote}}
175 \newcommand{\cftchapterprecistoc}[1]{
176   \addtocontents{toc}{%
177     {
178       \protect\begin{quote}#1\protect\end{quote}}
179   }
180 }

```

## Package 43

# lwarp-trivfloat.sty

## 110 Trivfloat

**Pkg** `trivfloat` `trivfloat` is forced to use the built-in `lwarp` emulation for floats.

**for HTML output:** Discard all options for `lwarp-trivfloat`:

```
1 \LWR@ProvidesPackageDrop{trivfloat}
2 \LWR@origRequirePackage{trivfloat}
```

**for HTML & PRINT:** `3 \begin{warpall}`



Package `trivfloat` is supported via the `lwarp` emulation of float. For HTML usage, use only `\usepackage{trivfloat}`, without any options, which tells `trivfloat` to use the emulated float functions instead of using `floatrow` or `memoir`.

To create a new float type and change its name:

---

```
\trivfloat{example}
\renewcommand{\examplename}{Example Name}
\crefname{example}{example}{examples}
\Crefname{example}{Example}{Examples}
```

---

```
4 \end{warpall}
```

**\tfl@chapter@fix** Nullified at the beginning of the document. Is used by `trivfloat` to correct float chapter numbers, but is not needed for `lwarp`.

**for HTML output:** `5 \begin{warpHTML}`

```
6 \AtBeginDocument{\DeclareDocumentCommand{\tfl@chapter@fix}{m m}{}}
```

```
7 \end{warpHTML}
```

### 110.1 Combining `\newfloat`, `\trivfloat`, and `algorithmicx`

**for HTML & PRINT:** `8 \begin{warpall}`



For both print and HTML output:

- ⚠ When using float, trivfloat, or algorithmicx at the same time, be aware of conflicting file usage. algorithmicx uses .loa. trivfloat by default starts with .loa and goes up for additional floats, skipping .lof and .lot.
- ⚠ When using \newfloat, be sure to manually assign higher letters to the \newfloat files to avoid .loa used by algorithmicx, and any files used by trivfloat. Also avoid using .lof and .lot.
- ⚠ When using \trivfloat, you may force it to avoid conflicting with algorithmicx by starting trivfloat's file extensions with .lob:

---

```
\makeatletter
\setcounter{tflfloatcnt}{1} % start trivfloats with .lob
\makeatletter
```

---

```
9 \end{warpall}
```

## Package 44

# lwarp-wallpaper.sty

## 111 Wallpaper

Pkg wallpaper wallpaper is emulated during HTML output, and the wallpaper package is ignored.

```
1 \LWR@ProvidesPackageDrop{wallpaper}
```

for HTML output:

```
2 \newcommand*{\CenterWallPaper}[2]{}
3 \newcommand*{\ThisCenterWallPaper}[2]{}
4 \newcommand*{\TileWallPaper}[3]{}
5 \newcommand*{\ThisTileWallPaper}[3]{}
6 \newcommand*{\TileSquareWallPaper}[2]{}
7 \newcommand*{\ThisTileSquareWallPaper}[2]{}
8 \newcommand*{\ULCornerWallPaper}[2]{}
9 \newcommand*{\ThisULCornerWallPaper}[2]{}
10 \newcommand*{\LLCornerWallPaper}[2]{}
11 \newcommand*{\ThisLLCornerWallPaper}[2]{}
12 \newcommand*{\URCornerWallPaper}[2]{}
13 \newcommand*{\ThisURCornerWallPaper}[2]{}
14 \newcommand*{\LRCornerWallPaper}[2]{}

```

```

15 \newcommand*\ThisLRCornerWallPaper}[2]{}
16 \newcommand*\ClearWallPaper[]{}
17 \newlength{\wpXoffset}
18 \newlength{\wpYoffset}

```

## Package 45

# lwarp-wrapfig.sty

## 112 Wrapfig

Pkg wrapfig wrapfig is emulated during HTML output, and the wrapfig package is ignored.

```
1 \LWR@ProvidesPackageDrop{wrapfig}
```

for HTML output: Computed width of a wrapped object. Used to print the HTML style.

```

2 \newlength{\LWR@wrapwidth}

3
4 \newcommand*\LWR@wrapposition[]{}
5
6 \newcommand*\LWR@subwrapfigure}[2]{%
7 \LWR@maybeinthisfloat%
8 \renewcommand*\LWR@wrapposition[]{}%
9 \ifthenelse{%
10 \equal{#1}{r}\OR\equal{#1}{R}\OR%
11 \equal{#1}{o}\OR\equal{#1}{O}%
12 }{%
13 {\renewcommand*\LWR@wrapposition}{float:right}}%
14 {\renewcommand*\LWR@wrapposition}{float:left}}%
15 \setlength{\LWR@wrapwidth}{#2}%
16 \addtolength{\LWR@wrapwidth}{4em}%
17 \uselengthunit{PT}%
18 % \BlockClass{marginblock}[%
19 % width:\rndprintlength{\LWR@wrapwidth} ; %
20 % \LWR@wrapposition%
21 % ]%
22 \LWR@stoppars%
23 \LWR@htmltag{div class="marginblock" id="autofloat-\arabic{\LWR@thisfloat}"
24 style="width:\rndprintlength{\LWR@wrapwidth} ; %
25 \LWR@wrapposition"%
26 }
27 \LWR@startpars

```

```
28 }
29
30
31 \NewDocumentEnvironment{wrapfigure}{o m o m}
32 {%
33 \LWR@subwrapfigure{#2}{#4}%
34 \captionsetup{type=figure}%
35 }
36 {
37 \LWR@htmldivclassend{div}
38 }
39
40
41 \NewDocumentEnvironment{wraptable}{o m o m}
42 {%
43 \LWR@subwrapfigure{#2}{#4}%
44 \captionsetup{type=table}%
45 }
46 {
47 \LWR@htmldivclassend{div}
48 }
49
50
51 \NewDocumentEnvironment{wrapfloat}{m o m o m}
52 {%
53 \LWR@subwrapfigure{#3}{#5}%
54 \captionsetup{type=#1}%
55 }
56 {
57 \LWR@htmldivclassend{div}
58 }
59
60 \newlength{\wrapoverhang}
```

## Package 46

# lwarp-xcolor.sty

## 113 Xcolor

Pkg `xcolor`

`support` Color definitions, models, and mixing are fully supported without any changes required.

`tables` Colored tables are ignored. Use CSS to style tables.

`colored text and boxes` `\textcolor`, `\colorbox`, and `\fcolorbox` are supported.

`\color` and `\pagecolor` `\color` and `\pagecolor` are ignored. Use CSS or `\textcolor` where possible.

`for HTML output:` `1 \LWR@ProvidesPackagePass{xcolor}`

`2 \newcommand*{\LWR@tempcolor}{}%`

defaulting to black.

`3 \newcommand*{\LWR@currenttextcolor}{black}`

Creates a styled span with a color converted to HTML hex colorspace. Uses `LWR@spandepth` to prevent paragraph tags inside the span. If used for `\textcolor`, with a styletext of `color:`, then the new color is copied into `\LWR@currenttextcolor` for possible re-use in `\rule`.

`4 \NewDocumentCommand{\LWR@colorstyle}{m m m +m}{}%`

Use the `xcolor` package to convert to an HTML color space:

`5 \convertcolorspec{#2}{#3}{HTML}\LWR@tempcolor%`

If is a `\textcolor`, save a copy of this color for use by `\rule`:

`6 \ifthenelse{\equal{#1}{color:}}{}%`

`7 {\renewcommand*{\LWR@currenttextcolor}{\#\LWR@tempcolor}}{}%`

Create the HTML span with the styled color:

`8 \LWR@htmltagc{span style="#1\#\LWR@tempcolor"}{}%`

`9 \begin{\LWR@nestspan}{}%`

Prevent additional paragraph tags inside this span:

Print the contents then close the span:

```
10 #4%
11 \LWR@htmltagc{/span}%
12 \end{LWR@nestspan}%
```

For paragraph-tag handling:

```
13 \LWR@ensuredoingapar%
14 }
```

`\color` appears in the L<sup>A</sup>T<sub>E</sub>X PDF output, but is ignored by `pdftotext` and thus is ignored in the HTML file. Text styling by local group is not yet supported.

Each of the following macros is given a temporary name, and is `\let` to the final name once the HTML conversion starts.

is converted into an HTML hex color span.

```
15 \NewDocumentCommand{\LWR@textcolor}{0{named} m +m}{%
16 \begingroup%
17 \LWR@colorstyle{color:}{#1}{#2}{#3}%
18 \endgroup%
19 }
```

Ignored. Use `\NewCSS` instead.

```
20 \newcommand*{\LWR@pagecolor}[2][named]{}
```

is converted into an HTML hex background color span.

```
21 \NewDocumentCommand{\LWR@colorbox}{0{named} m m}{%
22 \begingroup%
23 \LWR@colorstyle{background:}{#1}{#2}{#3}%
24 \endgroup%
25 }
```

```
[<framemodel>] {<framecolor>} [<boxmodel>] {<boxcolor>} {<text>}
```

is converted into a framed HTML hex background color span.

```
26 \NewDocumentCommand{\LWR@fcolorbox}{0{named} m 0{named} m m}{%
27 \begingroup%
28 \LWR@colorstyle{border:1px solid }{#1}{#2}%
29 {\LWR@colorstyle{background:}{#3}{#4}{#5}}%
```

---

```
30 \endgroup%  
31 }
```

## Package 47

# lwarp-xfrac.sty

## 114 Xfrac

Pkg **xfrac** Supported by adding xfrac instances.

for HTML output: 1 \LWR@ProvidesPackagePass{xfrac}

⚠ font size In the user's document preamble, **lwarp** should be loaded after font-related setup. During HTML conversion, this font is used by **lwarp** to generate its initial PDF output containing HTML tags, later to be converted by **pdftotext** to a plain text file. While the text may be in any font which **pdftotext** can read, the math is directly converted into SVG images using this same user-selected font. **xfrac** below is set for the Latin Modern (lmr) font. If another font is used, it may be desirable to redefine **\xfracHTMLfontsize** with a different em size.

**\sfrac** [*instance*] {*num*} [*sep*] {*denom*}

A text-mode instance for the default font is provided below. The numerator and denominator formats are adjusted to encase everything in HTML tags. **\scalebox** is made null inside the numerator and denominator, since the HTML tags should not be scaled, and we do not want to introduce additional HTML tags for scaling.

In math mode, which will appear inside a **lateximage**, no adjustments are necessary.

for HTML & PRINT: 2 \begin{warpall}

User-redefinable macro which controls the font size of the fraction.

3 \newcommand\*{\xfracHTMLfontsize}{.6em}

4 \end{warpall}

for HTML output: 5 \begin{warpHTML}

font size A span for a small font, used in the numerator and denominator:

```
6 \newcommand*{\LWR@htmlsmallfontstart}{%
7 \LWR@htmltagc{span style="font-size:\xfracHTMLfontsize"}{}}%
8 \LWR@nestspan%
9 %
10 }
```

```

11
12 \newcommand*{\LWR@htmlsmallfontend}{%
13 \LWR@htmltagc{/span}%
14 \endLWR@nestspan%
15 }

```

`\scalebox` A nullified `\scalebox` command, to avoid introducing HTML scaling tags:

```

16 \NewDocumentCommand{\LWR@noscalebox}{m o m}{#3}

```

instances Instances of `xfrac` for various font choices:

Produce HTML tags for a small superscript numerator and a small (non-subscript) denominator.

Scaling is turned off so that `pdftotext` correctly reads the result.

```

17 \DeclareInstance{xfrac}{default}{text}{
18 numerator-format = {%
19 \let\scalebox\LWR@noscalebox%
20 \LWR@htmlsmallfontstart\textsuperscript{#1}\,\LWR@htmlsmallfontend},
21 denominator-format = {%
22 \let\scalebox\LWR@noscalebox%
23 \LWR@htmlsmallfontstart{}\,\#1\LWR@htmlsmallfontend},

```

For `pdftotext`, do not scale the text:

```

24 scaling = false
25 }
26 \DeclareInstance{xfrac}{lmr}{text}{
27 numerator-format = {%
28 \let\scalebox\LWR@noscalebox%
29 \LWR@htmlsmallfontstart\textsuperscript{#1}\,\LWR@htmlsmallfontend},
30 denominator-format = {%
31 \let\scalebox\LWR@noscalebox%
32 \LWR@htmlsmallfontstart{}\,\#1\LWR@htmlsmallfontend},

```

For `pdftotext`, do not scale the text:

```

33 scaling = false
34 }
35 \DeclareInstance{xfrac}{lmss}{text}{
36 numerator-format = {%
37 \let\scalebox\LWR@noscalebox%
38 \LWR@htmlsmallfontstart\textsuperscript{#1}\,\LWR@htmlsmallfontend},
39 denominator-format = {%
40 \let\scalebox\LWR@noscalebox%
41 \LWR@htmlsmallfontstart{}\,\#1\LWR@htmlsmallfontend},

```



For pdftotext, do not scale the text:

```
42 scaling = false
43 }
44 \DeclareInstance{xfrac}{lmtt}{text}{
45 numerator-format = {%
46 \let\scalebox\LWR@noscalebox%
47 \LWR@htmlsmallfontstart\textsuperscript{#1}\,\LWR@htmlsmallfontend},
48 denominator-format = {%
49 \let\scalebox\LWR@noscalebox%
50 \LWR@htmlsmallfontstart{ }\,\#1\LWR@htmlsmallfontend},
```

For pdftotext, do not scale the text:

```
51 scaling = false
52 }
```

```
53 \end{warpHTML}
```

# Change History and Index

## Change History

v0.10		Docs: Commands for a successful HTML conversion. . . . .	49
General: 2016/03/08 Initial version	1		
v0.11		Docs: Commands into a warpprint environment. . . . .	49
General: 2016/03/11 . . . . .	1	Docs: Newclude limitations. . . . .	59
Added section:		Docs: Table: Cross-referencing data structures. . . . .	180
Operating-System portability. . . . .	80	Docs: Table: Float data structures. . . . .	189
Added section: Selecting the operating system. . . . .	47	Docs: Trademarks section. . . . .	258
Test Suite: limages and index in README.txt . . . . .	1	Docs: Troubleshooting cross-references. . . . .	61
Test Suite: MS-Windows in README.txt . . . . .	1	floatrow: Added package . . . . .	313
v0.12		Pagenote: Assigned cleveref name. . . . .	235
\LWR@newhtmlfile: Bugfix: TOC with numbered files. . . . .	111	Test Suite: Assigned cleveref name for Test Float. . . . .	1
General: 2016/03/14 . . . . .	1	Test Suite: Floatrow . . . . .	1
Global: Uses \p@(type) in float captions. . . . .	1		
Test Suite: Sub-figures . . . . .	1	v0.15	
v0.13		\printpublished: No div if empty. . . . .	133
\CaptionSeparator: Fix for newer babel package. . . . .	191	\printsubtitle: No div if empty. . . . .	133
\LWR@LwarpStart: \up and \fup . . . . .	123	General: 2016/04/06 . . . . .	1
\published: Default to empty published. . . . .	129	Added . . . . .	317
\subtitle: Default to empty subtitle. . . . .	130	Ampersand (&): Fixed handling when passed as an argument. . . . .	158
General: 2016/03/24 . . . . .	1	Docs: Added warning icons for items needing special attention. . . . .	66
Removed package: subfig . . . . .	1	Docs: Clarify print/HTML output. . . . .	47
Test Suite: Ordinals, Subcaption . . . . .	1	Docs: Moved the supported functions table to the introduction. . . . .	17
tikzpicture: Fix dollar-redefined bug for newer package. . . . .	237	Files: lwarp_formal.css added. . . . .	1
v0.14		Fix: steps counter . . . . .	316
\LWR@htmlsectionfilename: Fix: Links to home page. . . . .	87	Fixed & handling. . . . .	314
\noteentry: Fix for label inside a footnote. . . . .	106	Test Suite: test_suite_formal.css file added. . . . .	1
General: 2016/03/31 . . . . .	1		

v0.16	
General: 2016/04/11	1
\lwr@titlingpage: Improved	
print-output spacing.	134
Added XeLaTeX, LuaLaTeX	
support.	70
Docs: Font and UTF-8 support.	46
Docs: Moved location of	
\usepackage{lwarp}.	47
Docs: Text not converting.	61
Fix: amsmath options clash	74
Fix: newtxmath compatibility.	74
Lwarp no longer selects fonts.	46, 70
Removed package: suffix	1
Test Suite: Improved	
titlingpage.	134
Test Suite: Lwarp no longer	
selects fonts.	1
Test Suite: Supports XeLaTeX,	
LuaLaTeX.	1
xfrac: Adjusted for the use of	
any font:	368
v0.17	
\lwr@htmlsectionfilename: Fix:	
Links when entire doc is one	
HTML page.	87
General: 2016/04/14	1
mdframed: Added.	332
Test Suite: Fix: Print-version	
front-matter page numbers.	1
Test Suite: Mdfamed	1
v0.18	
\lwr@hspace: \hspace supported	250
\lwr@includegraphicsb: Add:	
svgz file extension.	228
em, ex, %, px dimensions	
preserved.	228
Fix: \linewidth, \textwidth,	
\textheight inside a	
minipage.	228
Improved HTML output	
linebreaks.	228
\lwr@myshorttoc: Reorganize	
\HomeHTMLFilename logic.	195
\lwr@newhtmlfile: Sidetoc after	
title, improving responsive	
design.	111
\lwr@requesttoc: Reorganize	
\HomeHTMLFilename logic.	125
\lwr@subhyperref: Improved	
HTML output linebreaks.	187
\lwr@subhyperrefclass: Improved	
HTML output linebreaks.	187
\lwr@subinlineimage: Surpress	
extra space.	188
General: 2016/05/19	1
File: lwarp.css: Improved TOC	
outline display.	1
Files: lwarp.css and	
lwarp_formal.css: Improved	
responsive design.	1
Microtype disabled during	
HTML generation	71
PDF Unicode input characters.	67
Test Suite: Verse package	1
\lateximage: pdfcrop: --hires	
added.	219
Reorganize \HomeHTMLFilename	
logic.	219
Surpress extra space.	219
verse: Supports verse, memoir	
packages.	143
minipage: Fix: \linewidth,	
\textwidth, \textheight	
inside a minipage.	241
v0.19	
\HTMLFileName: Docs: Escape	
filename underscores.	86
\HomeHTMLFileName: Docs: Escape	
filename underscores.	86
\lwr@LwarpStart: Enabled \\	
equal to \newline.	122
\lwr@doubledollar: MathJax	
support.	205
\lwr@filestart:	
lwarp_mathjax.txt loaded.	120
\lwr@hspace: Fix: \hspace length	
computations	250
\lwr@minipagestartpars:	
Surpresses paragraph tags	
between minipages.	248
\lwr@singledollar: MathJax	
support.	205
\lwr@tabledatamultirowtag:	
Added optional args.	175
\LateximageFontSizeName: Add:	
User-adjustable	
math/lateximage font size.	218

<code>\minipagefullwidth</code> : Added: No width tag for the next minipage in HTML. . . . .	240
<code>\rule</code> : Added . . . . .	253
<code>\warpHTMLonly</code> : Added. . . . .	68
<code>\warpprintonly</code> : Replaces <code>\rowprintedonly</code> . . . . .	68
General: 2016/06/08 . . . . .	1
Added. . . . .	367
Avoids MathJax. . . . .	204
cleveref: Loaded <code>\AtEndPreamble</code> . . . . .	235
CSS for table note item. . . . .	349
Docs: multirow browser bug workaround. . . . .	56, 154
Docs: Math options. . . . .	47
Docs: Table: Cross-referencing data structures, updated. . .	180
File: <code>lwarp_mathjax.txt</code> added. .	1
File: <code>lwarp.css</code> : <code>tnoteitemheader</code> added . . . . .	1
Introduction: MathJax support mentioned. . . . .	16
MathJax support added. . . . .	206, 211, 212
Options: <code>mathsvg</code> and <code>mathjax</code> . .	69
Supports colored <code>\rule</code> . . . . .	364
titles: null <code>\pagestyle</code> and <code>\thispagestyle</code> for HTML. . .	74
<code>tikzpicture</code> : Adapts to <code>tikz</code> version. . . . .	237
<code>equation</code> : MathJax support. . . .	209
v0.20	
<code>\BlockClassSingle</code> : Renamed from "LWR@htmldivclassline". .	95
<code>\HTMLFileName</code> : No longer escape underscores. . . . .	86
<code>\HomeHTMLFileName</code> : No longer escape underscores. . . . .	86
<code>\InlineClass</code> : Renamed from "inlineclass". . . . .	96
<code>\LWR@LwarpStart</code> : Fix: math cross references. . . . .	124
<code>\LWR@closeparagraph</code> : <code>\unskip</code> extra spaces. . . . .	99
No break tags in the start/end of a tabular. . . . .	99
<code>\LWR@endoffline</code> : Fix: <code>\\</code> . . . .	248
<code>\LWR@filestart</code> : Adds meta description. . . . .	120
<code>\LWR@hspace</code> : Add: Supports HTML thin breakable space . . . .	250
<code>\LWR@htmldivclass</code> : Added optional style. . . . .	94
<code>\LWR@htmlelementclass</code> : Added optional style. . . . .	93
<code>\LWR@htmlsectionfilename</code> : <code>HTMLFileName</code> : removed additional trailing '-', and may be empty. . . . .	87
Sections called "Index" or "index" have an underscore prepended to their filenames if no prefix. .	87
<code>\LWR@includegraphicsb</code> : Fix: <code>\linewidth</code> in a floatrow. . .	228
Fix: Expands filename. . . . .	228
<code>\LWR@longtabledatacaptiontag</code> : Fix: Pars in captions. . . . .	172
<code>\LWR@section</code> : Combined higher-level sections together into files. . . . .	114
<code>\LWR@subhtmlclass</code> : Factored code. . . . .	93
<code>\SetHTMLFileNumber</code> : Add: Control file numbers. . . . .	86
<code>\cpagerefFor</code> : User-redefinable word for page references. . . .	235
<code>\dotfill</code> : Inserts an ellipsis. . . .	247
<code>\hfill</code> : Inserts a <code>\quad</code> . . . . .	247
<code>\hrulefill</code> : Inserts a short rule. .	247
<code>\hyperindexref</code> : Print mode provided in case hyperref not used. . . . .	202
<code>\pageref</code> : Added. . . . .	186
<code>\tracinglwarp</code> : Added. . . . .	78
<code>\warpOSwindows</code> : Auto-detects operating system. . . . .	81
General: 2017/02/09 . . . . .	1
<code>afterpage</code> : Added. . . . .	302
<code>alltt</code> : Added. . . . .	304
<code>bookmark</code> : Added. . . . .	305
<code>caption</code> and <code>subcaption</code> supported. . . . .	1
<code>cleveref</code> and referencing patches: Applied <code>\AfterEndPreamble</code> . . . .	235
<code>draftwatermark</code> : Added. . . . .	306
<code>eso-pic</code> : Added. . . . .	308
<code>everypage</code> : Added. . . . .	309
<code>extramarks</code> : Added. . . . .	309
<code>fancyhdr</code> : Added. . . . .	310

hyperref: Additional user macros. . . . .	320	Docs: Examples for generating HTML file names. . . . .	40
keyfloat: Added. . . . .	323	Docs: Improved index. . . . .	1
letterspace: User-interface emulated. . . . .	324	Enhanced titling support. . . . .	131
listings: Added. . . . .	325	File: lwarp.css: Minor fixes for validation. . . . .	1
lrcaption: Added. . . . .	331	File: lwarpmk used to compile print, HTML, indexes, and lateximages. . . . .	1
lwarp-newproject: Added. . . . .	259	Fix:\linewidth in a floatrow. . . . .	315
microtype: User-interface emulated. . . . .	339	Improved float caption type handling. . . . .	311
needspace: Added. . . . .	342	Moved sidebar and example code to test suite. . . . .	1
nowidow: Added. . . . .	342	Page geometry set to 6in wide with large margins. . . . .	71
pagenote: Added. . . . .	343	Parallel versions of aux files for print/HTML. . . . .	1
placeins: Added. . . . .	344	Removed reliance on make, grep, gawk. . . . .	1
ragged2e: Added. . . . .	345	Tabular: \unskip extra spaces. . . . .	155
setspace: Improved support. . . . .	346	Test Suite: HTML meta descriptions. . . . .	1
textpos: Added. . . . .	347	Verbatim: Added. . . . .	146
titles: Added. . . . .	350	verbatim: Added. . . . .	144
titlesec: Added. . . . .	352	BlockClass: Added optional style. . . . .	95
titletoc: Added. . . . .	353	Renamed from "blockclass". . . . .	95
titling: Improved compatibility. . . . .	354	LWR@nestspan: Fix: Minipages inside a span. . . . .	91
tocloft: Added. . . . .	355		
wallpaper: Added. . . . .	361		
wrapfig: Added. . . . .	362		
Added @, <, > columns. . . . .	155		
Added \NewHTMLdescription. . . . .	104		
Added single-expansion data arrays. . . . .	85		
Code factored into independent lwarp_html files. . . . .	302		

## Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

Symbols	
\\$ . . . . .	<i>204</i>
\& . . . . .	<i>85</i> , <u>2055</u>
\( . . . . .	<u>2972</u>
\) . . . . .	<u>2972</u>
\@@setcpageref . . . . .	<u>3691</u>
\@@setcref . . . . .	<u>3687</u>
\@@setcrefrange . . . . .	<u>3688</u>
@author . . . . .	<i>128</i>
\@begintheorem . . . . .	<u>1912</u>
\@capytype . . . . .	<u>2716</u>
\@currentlabel . . . . .	<i>106</i>
\@currentlabelname . . . . .	<u>2506</u>
@date . . . . .	<i>128</i>
\@dlbfloat . . . . .	<u>2699</u>
\@endtheorem . . . . .	<u>1922</u>
\@float . . . . .	<u>2699</u>
\@fnsymbol . . . . .	<u>1620</u>
\@makecaption . . . . .	<u>2718</u>
\@maketitle . . . . .	<u>1631</u>

<code>\@nameedef</code> .....	69	booleans:	
<code>\@opargbegintheorem</code> .....	1917	<code>CombineHigherDepths</code> .....	40, 108
<code>@published</code> .....	128	<code>FileSectionNames</code> .....	40, 87
<code>@subtitle</code> .....	128	<code>LWR@amsmultline</code> .....	211
<code>@title</code> .....	128	<code>LWR@doingapar</code> .....	97
<code>\\</code> .....	248	<code>LWR@doinghline</code> .....	156
<code>\$</code> .....	205	<code>LWR@doingstartpars</code> .....	97
<code>\$\$</code> .....	205	<code>LWR@doingtbrule</code> .....	156
<b>A</b>		<code>LWR@freezethisfloat</code> .....	191
<code>abstract</code> (environment) .....	1661	<code>LWR@infloatrow</code> .....	228
<code>\abstractname</code> .....	41, 1660	<code>LWR@intabularmetadata</code> .....	156
<code>accents</code> .....	112	<code>LWR@minipagefullwidth</code> .....	240
<code>\addcontentsline</code> .....	2767	<code>LWR@minipagethispar</code> .....	241
<code>Adobe</code> (program) .....	23	<code>LWR@skippingmrowcell</code> .....	156
<code>\affiliation</code> .....	1329	<code>LWR@starredlongtable</code> .....	172
<code>afterpage</code> (package) .....	302	<code>LWR@startedrow</code> .....	156
<code>algorithmic</code>		<code>LWR@tableparcell</code> .....	156
<code>with newfloat, trivfloat</code> .....	360	<code>LWR@tikzbabel</code> .....	237
<code>algorithmicx</code> (package) .....	303	<code>LWR@tracinglwarp</code> .....	78
<code>align</code> (environment) .....	3184	<code>LWR@verbtags</code> .....	144
<code>align*</code> (environment) .....	3206	<code>mathjax</code> .....	68
<code>alltt</code> (package) .....	304	<code>usingOSwindows</code> .....	80
<code>\AmS</code> .....	4073	<code>warpingHTML</code> .....	68
<code>amsmath</code> (package) .....	74	<code>warpingprint</code> .....	68
<code>\and</code> .....	128	bugs .....	61
<code>array</code> (package) .....	179	<code>BVerbatim</code> (environment) .....	1898
<code>AsciiDoc</code> (program) .....	22	<b>C</b>	
<code>AsciiDoctor</code> (program) .....	23	<code>calc</code> (package) .....	73
<code>Asciidoctor-LaTeX</code> (program) .....	23	<code>caption</code> (package) .....	73
<code>\attrib</code> .....	52, 142	<code>\caption@begin</code> .....	2748
<code>\attribution</code> .....	1670	<code>\caption@end</code> .....	2748
<code>\author</code> .....	50, 128, 1336	<code>\captionlistentry</code> .....	2750
<code>autosec</code> .....	113	<code>\captionof</code> .....	2779
<b>B</b>		<code>\CaptionSeparator</code> .....	2717
<code>babel</code> (package) .....	59	<code>center</code> (environment) .....	3418
<code>\backmatter</code> .....	1011	<code>\centering</code> .....	123
<code>BaseJobName</code> (option) [lwarp-		<code>\chapter</code> .....	1129
<code>newproject]</code> .....	259	<code>\citetitle</code> .....	1677
<code>\BaseJobname</code> .....	413	<code>cleveref</code> (package) .....	234
<code>BaseJobname</code> (option) [lwarp] .....	69	<code>cmap</code> (package) .....	47
<code>baseline</code>		<code>\color</code> .....	365
<code>tabular</code> .....	163	<code>CombineHigherDepths</code> (boolean) .....	40, 108
<code>\bfseries</code> .....	3888	<code>comment</code> (package) .....	68
<code>\BibTeX</code> .....	4068	<code>Computer Modern</code> .....	46
<code>BlockClass</code> (environment) .....	628	<code>\ConTeXt</code> .....	4065
<code>\BlockClassSingle</code> .....	640	counters:	
<code>bookmark</code> (package) .....	305	<code>FileDepth</code> .....	39, 108
<code>booktabs</code> (package) .....	306	<code>LWR@externalfilecnt</code> .....	204
		<code>LWR@htmlfilenumber</code> .....	87

LWR@latestautopage	192	environ (package)	74
LWR@lateximagedepth	217	environments:	
LWR@lateximagenumber	217	abstract	1661
LWR@midrulecounter	168	align	3184
LWR@minipagedepth	238	align*	3206
LWR@nextautofloat	192	BlockClass	628
LWR@nextautopage	192	BVerbatim	1898
LWR@nextequation	206	center	3418
LWR@nextpage	217	description	1985
LWR@prevFileDepth	114	displaymath	2981
LWR@spandepth	97	enumerate	1954
LWR@tablecolspos	160	equation	3048
LWR@tablecolswidth	160	flalign	3228
LWR@tabletotalcols	160	flalign*	3250
LWR@tabletotalcolsnext	160	flushleft	3424
LWR@thisfloat	191	flushright	3421
SideTOCDepth	39, 198	gather	3140
tocdepth	39	gather*	3162
\cpagerefFor	3690	itemize	1936
\cref@currentlabel	106	lateximage	3324
cross-references		longtable	2
missing or incorrect	61	LVerbatim	1909
CSS		LWR@nestspan	523
file selection	52	LWR@tabular	2464
lwrap.css	52	math	2980
per HTML page	52	minipage	3797
project-specific changes	52	multline	3093
		multline*	3116
<b>D</b>		picture	236, 3708
danger icon	66	quote	1685
\date	51	theindex	2907
debugging	61	tikzpicture	236, 3727
\DeclareGraphicsExtensions	3439	titlepage	50, 1370
\DeclareGraphicsRule	3439	titlingpage	50, 1373, 1386
Deja Vu	46	Verbatim	1766
description		verbatim	1725
HTML meta tag	51, 105	VerbatimClass	1758
description (environment)	1985	verse	1700
displaymath (environment)	2981	warpall	34, 48
\dotfill	3904	warpHTML	42, 42, 48
draftwatermark (package)	306	warpprint	35, 42, 48
		epigraph (package)	307
<b>E</b>		equation (environment)	3048
\emph	3877	error messages	61
\end@dlbfloat	2708	eso-pic (package)	308
\end@float	2708	etoolbox (package)	67
\enlargethispage	4003	everyhook (package)	72
\enskip	248, 3969	everypage (package)	309
enumerate (environment)	1954	expl3 (package)	72
enumitem (package)	60, 73	extramarks (package)	309

<b>F</b>	
fancyhdr (package) . . . . .	310
fancyvrb (package) . . . . .	72
FAQ . . . . .	61
fbox	
around a minipage . . . . .	49
\fbox . . . . .	3867
FileDepth (counter) . . . . .	39, 108
files	
inaccessible . . . . .	39, 40
files:	
glyphtounicode . . . . .	46
lwarp.css . . . . .	51, 262
lwarp_formal.css . . . . .	287
lwarp_html.xdy . . . . .	292
lwarp_mathjax.txt . . . . .	292
lwarp_sagebrush.css . . . . .	282
lwarpmk . . . . .	294
lwarpmk.conf . . . . .	257
project.css . . . . .	51
project.lwarpmkconf . . . . .	261
project_html.tex . . . . .	261
sample_project.css . . . . .	51, 291
FileSectionNames (boolean) . . . . .	40, 87
\flagverse . . . . .	143
flalign (environment) . . . . .	3228
flalign* (environment) . . . . .	3250
Flare (program) . . . . .	23
float (package) . . . . .	311
\floatname . . . . .	11
\floatplacement . . . . .	14
floatrow (package) . . . . .	313
\floatstyle . . . . .	17
flushleft (environment) . . . . .	3424
flushright (environment) . . . . .	3421
font	
bitmapped . . . . .	46
Computer Modern . . . . .	46
Deja Vu . . . . .	46
ligatures . . . . .	46, 47
packages . . . . .	46
selection . . . . .	46
size — lateximage . . . . .	53, 202, 217
size — math, SVG . . . . .	53, 202, 217
size — xfrac . . . . .	367
fontenc (package) . . . . .	46
fontspec (package) . . . . .	46
\footnote . . . . .	845
footnotes . . . . .	105
<b>G</b>	
gather (environment) . . . . .	3140
gather* (environment) . . . . .	3162
GELLMU (program) . . . . .	22
generator	
HTML meta tag . . . . .	121
geometry (package) . . . . .	71, 317
getttitlestring (package) . . . . .	72
glyphtounicode (file) . . . . .	46
graphics (package) . . . . .	223, 317
\graphicspath . . . . .	3437
graphicx (package) . . . . .	223, 319
<b>H</b>	
Hevea (program) . . . . .	22
\hfill . . . . .	3902
\HomeHTMLFileName . . . . .	415
\HomeHTMLFilename . . . . .	39
horizontal space	
between minipages . . . . .	248
\href . . . . .	2651
\hrulefill . . . . .	3903
\hspace . . . . .	248, 3996
HTML	
commands for a successful conver-	
sion . . . . .	49
conversion settings . . . . .	39
filename generation . . . . .	40
headings . . . . .	64
meta tag	
description . . . . .	51, 105
generator . . . . .	121
viewport . . . . .	121
tabular column conversion . . . . .	164
\HTMLentity . . . . .	403
\HTMLFileName . . . . .	39, 414
HTMLleftmargini (length) . . . . .	53, 142, 143
\HTMLunicode . . . . .	404
HTMLvleftskip (length) . . . . .	53, 142
\hyperindexref . . . . .	2924
<b>for</b>	
HTML & PRINT . . . . .	66
HTML output . . . . .	66
PRINT output . . . . .	66
\framebox . . . . .	3864
FrameMaker (program) . . . . .	23
Frequently Asked Questions . . . . .	61
\frontmatter . . . . .	1008
\fup . . . . .	3901



hyperref (package) . . . . .	187, 320	vleftskip . . . . .	52, 142, 143
\hypertoc . . . . .	2866	letterspace (package) . . . . .	324
\hypertocfloat . . . . .	2875	LibreOffice (program) . . . . .	23
<b>I</b>			
icon		ligatures . . . . .	46, 47, 71
warning . . . . .	66	line numbers . . . . .	66
ifplatform (package) . . . . .	67	\linebreak . . . . .	3997
images		\LinkHome . . . . .	463
graphicx package . . . . .	223	list item, empty . . . . .	49
in strange places . . . . .	62	listings (package) . . . . .	325
\includegraphics . . . . .	3619	\listof . . . . .	2833
InDesign (program) . . . . .	23	lmodern (package) . . . . .	46
\InlineClass . . . . .	648	longtable (environment) . . . . .	2
inputenc (package) . . . . .	46	longtable (package) . . . . .	329
item, list, empty . . . . .	49	ltcaption (package) . . . . .	331
itemize (environment) . . . . .	1936	LuaLaTeX	
\itshape . . . . .	3893	detection . . . . .	70
<b>J</b>			
JavaScript		file & section names . . . . .	112
MathJax . . . . .	53, 202	\LuaLaTeX . . . . .	4059
<b>K</b>			
keyfloat (package) . . . . .	323	LuaLaTeX (program) [requirement] . .	25
kvoptions (package) . . . . .	68	\LuaTeX . . . . .	4059
<b>L</b>			
label		LVerbatim (environment) . . . . .	1909
in HTML . . . . .	124	lwarp	
math environment . . . . .	211	loading . . . . .	47
language HTML metadata . . . . .	120	options . . . . .	47
\LaTeX . . . . .	4051	[lwarp]:	
LaTeX2html (program) . . . . .	22	BaseJobname (option) . . . . .	69
\LaTeXe . . . . .	4051	mathjax (option) . . . . .	47, 69
lateximage (environment) . . . . .	3324	mathsvg (option) . . . . .	47, 69
\LateximageFontSizeName . . . . .	3288	warpHTML (option) . . . . .	47, 69
lateximages		warpprint (option) . . . . .	47, 68
font size . . . . .	53, 202, 217	lwarp (package) . . . . .	47
LaTeXML (program) . . . . .	22	[lwarp-newproject]:	
lengths:		BaseJobName (option) . . . . .	259
HTMLleftmargini . . . . .	53, 142, 143	lwarpmk (option) . . . . .	259
HTMLvleftskip . . . . .	53, 142	warpHTML (option) . . . . .	259
LWR@minipageheight . . . . .	238	warpprint (option) . . . . .	259
LWR@minipagewidth . . . . .	238	lwarp-newproject (package) . . . . .	259
LWR@tempheight . . . . .	250	lwarp.css (file) . . . . .	51, 262
LWR@tempraise . . . . .	250	lwarp_formal.css (file) . . . . .	287
LWR@tempwidth . . . . .	250	lwarp_html.xdy (file) . . . . .	292
\VerbatimHTMLwidth . . . . .	144	lwarp_mathjax.txt (file) . . . . .	292
vleftmargini . . . . .	52, 142	lwarp_sagebrush.css (file) . . . . .	282
		lwarpmk (file) . . . . .	294
		lwarpmk (option) [lwarp-newproject]	259
		lwarpmk.conf (file) . . . . .	257
		\LWR@addmathjax . . . . .	3037
		\LWR@afterendverbatim . . . . .	1750
		LWR@amsmultline (boolean) . . . . .	211
		\LWR@atbeginverbatim . . . . .	1742
		\LWR@beginhideamsmath . . . . .	3078

\LWR@botnavigation	471	\LWR@htmlelementclassline	611
\LWR@caption@begin	2730	\LWR@htmlelementend	621
\LWR@caption@end	2739	LWR@htmlfilenumber (counter)	87
\LWR@clearmidrules	2234	\LWR@htmlfileref	2532
\LWR@closeparagraph	715	\LWR@htmlmathlabel	3068
\LWR@closeprevious	491	\LWR@htmlopencomment	554
\LWR@closetabledatacell	2034	\LWR@htmlrefsectionfilename	456
LWR@colafterspec	160	\LWR@HTMLSanitize	3291
LWR@colatspec	160	\LWR@HTMLSanitizeexpand	3309
LWR@colbeforespec	160	\LWR@htmlsectionfilename	425
LWR@commentfooterstyle	217	\LWR@htmlspan	529
\LWR@copyfile	223	\LWR@htmlspanclass	537
\LWR@createautosec	1017	\LWR@htmltag	545
\LWR@currentcss	816	\LWR@htmltagc	513
\LWR@currentHTMLdescription	831	\LWR@includegraphicsb	3530
\LWR@descitem	1973	\LWR@indexitem	2912, 2916, 2920
\LWR@docmidrule	2253	\LWR@indexsection	2896
LWR@doingapar (boolean)	97	LWR@infloatrow (boolean)	228
LWR@doinghline (boolean)	156	LWR@intabularmetadata (boolean)	156
LWR@doingstartpars (boolean)	97	\LWR@itemizeitem	1929
LWR@doingtbrule (boolean)	156	LWR@latestautopage (counter)	192
\LWR@domulticolumn	2316	LWR@lateximagedepth (counter)	217
\LWR@doubledollar	2950	\LWR@lateximagedepthref	2537
\LWR@endhideamsmath	3086	LWR@lateximagenumber (counter)	217
\LWR@endminipage	3759	\LWR@lateximagenumberref	2540
\LWR@endoffline	3947	\LWR@longtabledatacaptiontag	2334
\LWR@ensuredoingapar	3782	\LWR@lookforpackagename	162
\LWR@ensuredoingapar	689	\LWR@LwarpEnd	1310, 4095
LWR@externalfilecnt (counter)	204	\LWR@LwarpStart	1233, 4095
\LWR@filenamenoblanks	879	\LWR@makefnmark	3758
\LWR@filestart	1181	\LWR@makefntext	3751
\LWR@findword	159	\LWR@maketitlesetup	1611
\LWR@floatbegin	2688	\LWR@maybeinthisfloat	2713
\LWR@floatend	2701	\LWR@maybeewtablerow	2187
LWR@freezethisfloat (boolean)	191	LWR@midrulecounter (counter)	168
\LWR@futurenonspacel	2013	LWR@midrules	168
\LWR@getexpparray	399	LWR@minipagedepth (counter)	238
\LWR@getmynexttoken	2020	LWR@minipagefullwidth (boolean)	240
\LWR@hidelatexequation	3023	LWR@minipageheight (length)	238
\LWR@hspace	3977	\LWR@minipagestartpars	3951
\LWR@htmlblockcomment	567	\LWR@minipagestoppars	3957
\LWR@htmlblocktag	575	LWR@minipagethispar (boolean)	241
\LWR@htmlblocktagcomment	580	LWR@minipagewidth (length)	238
\LWR@htmlclosecomment	554	\LWR@multicolother	2277
\LWR@htmlcomment	563	\LWR@multicolpartext	2272
\LWR@htmldivclass	605	\LWR@mynexttoken	2012
\LWR@htmldivclassend	608	\LWR@myshorttoc	2791
\LWR@htmlmlelement	618	\LWR@nameref	2529
\LWR@htmlmlelementclass	594	LWR@nestspan (environment)	523
\LWR@htmlmlelementclassend	599	\LWR@newhtmlfile	957





booktabs	306	pagenote	73, 343
calc	73	placeins	344
caption	73	ragged2e	345
cleveref	234	rotating	346
cmap	47	setspace	346
comment	68	siunitx	59, 223
draftwatermark	306	subcaption	73
enumitem	60, 73	textcomp	47
environ	74	textpos	347
epigraph	307	threeparttable	349
eso-pic	308	tikz	236
etoolbox	67	titles	74, 350
everyhook	72	titlesec	352
everypage	309	titletoc	353
expl3	72	titling	73, 354
extramarks	309	tocloft	355
fancyhdr	310	trivfloat	360
fancyvrb	72	verse	143
float	311	wallpaper	361
floatrow	313	wrapfig	362
fontenc	46	xcolor	364
fontspec	46	xfrac	75, 367
geometry	71, 317	xifthen	72
getttitlestring	72	xparse	72
graphics	223, 317	xstring	72
graphicx	223, 319	zref	74
hyperref	187, 320	\pagebreak	3999
ifplatform	67	pagenote (package)	73, 343
inputenc	46	\pagenotesubhead	844
keyfloat	323	\pagenumbering	151
kvoptions	68	\pageref	2626
letterspace	324	\pagerefPageFor	2625
listings	325	pages	
lmodern	46	inaccessible	39, 40
longtable	329	\pagestyle	147
ltxcaption	331	Pandoc (program)	23
lwrap	47	\paragraph	1164
lwrap-newproject	259	\parbox	3855
makeidx	73	\parsemulticolumnalignment	2304
mdframed	332	\part	1118
memoir	143	pdfcrop (program) [requirement]	25
microtype	47, 71, 339	pdfLaTeX (program) [requirement]	25
multicol	340	pdfseparate (program) [requirement]	25
multirow	175, 342	pdftocairo (program) [requirement]	25
needspace	342	pdftotext (program) [requirement]	25
newclude	59	\phantomsection	4045
newfloat	73	picture (environment)	236, 3708
newtxmath	59	placeins (package)	344
newunicodechar	46	Plastex (program)	22
nowidow	342	\popclose	369

Poppler	25	project_html.tex (file)	261
\postpublished	127	\published	51, 1346
\postsubtitle	127	\pushclose	342
\prepublished	127		
\presubtitle	127, 137		
\printauthor	128, 1479, 1494		
\printdate	128, 1484, 1496		
\printindex	2899		
\printpublished	128, 1445, 1491		
\PrintStack	476		
\prints subtitle	128, 1470, 1493		
\printthanks	127, 131		
\printtitle	128, 1454, 1492		
problems	61		
utility	25		
programs:			
[requirement]:			
LuaLaTeX	25		
MathJax	25		
pdfcrop	25		
pdfLaTeX	25		
pdfseparate	25		
pdftocairo	25		
pdftotext	25		
XeLaTeX	25		
Adobe	23		
AsciiDoc	22		
AsciiDoctor	23		
Asciidoctor-LaTeX	23		
Flare	23		
FrameMaker	23		
GELLMU	22		
Hevea	22		
InDesign	23		
LaTeX2html	22		
LaTeXML	22		
LibreOffice	23		
Madcap	23		
MathJax	54, 203		
MS-Windows	47, 80		
OpenOffice	23		
Pandoc	23		
Plastex	22		
TeX4ht	22		
TtH	22		
Windows	47, 80		
Word	23		
project.css (file)	51		
project.lwarpmkconf (file)	261		
		<b>Q</b>	
		\qqquad	248, 3968
		\quad	248, 3963
		quote (environment)	1685
		<b>R</b>	
		ragged2e (package)	345
		\raggedleft	123
		\raggedright	123
		\raisebox	3872
		\ref	2614
		\reflectbox	3676
		\renewlist	2003
		[requirement]:	
		LuaLaTeX (program)	25
		MathJax (program)	25
		pdfcrop (program)	25
		pdfLaTeX (program)	25
		pdfseparate (program)	25
		pdftocairo (program)	25
		pdftotext (program)	25
		XeLaTeX (program)	25
		\RequirePackage	177
		\resizebox	3679
		\restylefloat	19
		\rmfamily	3889
		\rotatebox	3650
		rotating (package)	346
		\rule	4042
		<b>S</b>	
		sample_project.css (file)	51, 291
		\sb	3897
		\scalebox	3665
		\scshape	3894, 3895
		section	
		depths	64
		missing	39, 40
		\section	1142
		\SetFirstPageTop	41, 794
		\SetHTMLFileNumber	416
		\SetPageBottom	41, 128, 810
		\SetPageTop	41, 802
		setspace (package)	346
		settings	
		CSS project-specific	52

CSS selection	52	\thanks	51
HTML conversion	39	\thanksmarkseries	1652
lwrap package options	47	\theauthor	128
selecting output	47	\thedate	128
title page	50	theindex (environment)	2907
\sffamily	3890	\thepublished	128
\sfrac	367	\thesubtitle	128
SideTOCDepth (counter)	39, 198	\thetitle	128
\sidetocname	2843	\thispagestyle	149
siunitx (package)	59, 223	threeparttable (package)	349
\sp	3896	tikz	
space		catcodes	236
horizontal between minipages	248	dollar redefined	236
stack depths	64	tikz (package)	236
subcaption (package)	73	tikzpicture (environment)	236, 3727
subequations		\title	50
MathJax	54, 203	titlepage (environment)	50, 1370
\ subparagraph	1171	titlesps (package)	74, 350
\ subsection	1149	titlesec (package)	352
\ subsubsection	1156	titletoc (package)	353
\ subtitle	51, 1357	titling (package)	73, 354
SVG		titlingpage (environment)	50, 1373, 1386
math summary	53, 202	tocdepth (counter)	39
mathsvg option	69	tocloft (package)	355
T		\tracinglwrap	63, 241
\tableofcontents	41	trivfloat	
tabular		with newfloat, algorithmic	360
baseline	163	trivfloat (package)	360
HTML columnn conversion	164	troubleshooting	61
tagged equations		\ttfamily	3891
Mathjax	54, 203	TtH (program)	22
\TeX	4048	U	
TeX4ht (program)	22	Unicode	
\textbf	3879	enhanced coverage	46
textcomp (package)	47	file & section names	112
\textgreater	406	input characters	67
\textit	3884	selection	46
\textless	406	\up	3900
\textmd	3878	\upshape	3892
\textnormal	3886	\url	2670
textpos (package)	347	\UseLatexmk	39
\textrm	3880	\UseVerbatim	1910
\textsc	3885	usingOSwindows (boolean)	80
\textsf	3881	UTF-8	
\textsubscript	3899	enhanced coverage	46
\textsuperscript	3898	file & section names	112
\texttt	3882	selection	46
\textup	3883	utility	
\tfl@chapter@fix	360	programs	25

<b>V</b>	
Verbatim (environment) . . . . .	1766
verbatim (environment) . . . . .	1725
VerbatimClass (environment) . . . .	1758
\VerbatimHTMLWidth (length) . . . .	144
verse (environment) . . . . .	1700
verse (package) . . . . .	143
viewport	
HTML meta tag . . . . .	121
vleftmargini (length) . . . . .	52, 142
vleftskip (length) . . . . .	52, 142, 143
<b>W</b>	
wallpaper (package) . . . . .	361
warning icon . . . . .	66
warpall (environment) . . . . .	34, 48
warpHTML (environment) . . . .	42, 42, 48
warpHTML (option) [lwarp-newproject]	259
warpHTML (option) [lwarp] . . . . .	47, 69
\warpHTMLonly . . . . .	33, 42, 48
warpingHTML (boolean) . . . . .	68
warpingprint (boolean) . . . . .	68
\warpOSwindows . . . . .	47, 80, 294
warpprint (environment) . . . .	35, 42, 48
warpprint (option) [lwarp-newproject]	259
warpprint (option) [lwarp] . . . . .	47, 68
\warpprintonly . . . . .	32, 42, 48
Windows (program) . . . . .	47, 80
Word (program) . . . . .	23
wrapfig (package) . . . . .	362
<b>X</b>	
xcolor (package) . . . . .	364
XeLaTeX	
detection . . . . .	70
file & section names . . . . .	112
\XeLaTeX . . . . .	4061
XeLaTeX (program) [requirement] . . .	25
\XeTeX . . . . .	4061
xfrac (package) . . . . .	75, 367
xifthen (package) . . . . .	72
xparse (package) . . . . .	72
xstring (package) . . . . .	72
<b>Z</b>	
zref (package) . . . . .	74