

The latex-lab-firstaid package

Temporary patches to external packages needed for the tagging project

L^AT_EX Project*

v0.86l 2026-10-26

Abstract

1 Introduction

The following contains small temporary changes to external packages to avoid errors with the new tagging code.

Similar to the main firstaid package the goal is to remove the patches once the packages have been updated.

2 Implementation

```
1 <*package>
2 <@@=tag>

3 \ProvidesPackage {latex-lab-testphase-firstaid} [%
4   \l{labfirstaidddate}\space v\l{labfirstaidversion}\space
5   Temporary patches to external packages needed for the tagging project]
```

`\FirstAidNeededT` This is a very simple help to ensure that we only apply first aid to an unmodified package or class. It only works in the case the file has already been loaded and the `cname` `\ver@#1.#2` got defined (holding the current date, version, and short description info). We then compare its content to a frozen string and make the modification `#3` only if both agree. If they differ we assume that the package/class in question got updated by its maintainer.

```
6 \ExplSyntaxOn
7 \providecommand\FirstAidNeededT[3]{
8   \exp_args:Nce\str_if_eq:onF{ver@#1.#2}{#3}
9     { \typeout{=>~ First~ Aid~ for~ #1.#2~ no~ longer~ applied!^^J
10       \@spaces Expected:^^J
11       \@spaces\@spaces #3^^J
12       \@spaces but~ found:^^J
13       \@spaces\@spaces \use:c{ver@#1.#2}^^J
14       \@spaces so~ I'm~ assuming~ it~ got~ fixed.
15     } }
```

*Initial implementation done by Ulrike Fischer

```

16 \exp_args:Nce\str_if_eq:ont{ver@#1.#2}{#3}
17 }

```

(End of definition for `\FirstAidNeededT`.)

2.1 tikz/pgf

`tikz` inputs libraries with the primitive `\input` command. This means that these libraries are not listed in the file list written by `\listfiles` and the new tagging status report created with the `check-tagging-status` key.

We therefore redefine one `pgf` command to use the $\text{\LaTeX}$ `\input` command. Check <https://github.com/pgf-tikz/pgf/issues/1424> for changes.

```

18 \AddToHook{package/pgfrcs/after}
19 {\def\pgfutil@InputIfFileExists#1#2#3{\pgfutil@IfFileExists{#1}{\input{#1}\relax#2}{#3}}}

```

2.2 listings

The `listings` package needs various adaptations to enable tagging support. They are in an external file.

```

20 \AddToHook{package/listings/after}[latex-lab-firstaid]{\RequirePackage{latex-lab-
listings}}

```

2.3 AMS classes

Unfortunately, the AMS classes contain a full inline copy of `amsthm` inside the class (thanks to `dostrip`) and they use `\newcommand`, sigh.

Thus, to make the new code work with these classes we have to hide some definitions, load the class and only afterwards restore our own versions.

So first save some of the problematical definitions under some other names (quite a handful):

```

21 \let \@kernel@newtheorem \newtheorem
22 \let \@kernel@newtheoremstyle \newtheoremstyle
23 \let \@kernel@theoremstyle \theoremstyle
24 \let \@kernel@proof \proof
25 \let \@kernel@endproof \endproof

26 \def\undefine@thm@macros{
27 \let \newtheoremstyle \relax
28 \let \theoremstyle \relax

29 \let \proof \relax
30 \let \endproof \relax

```

These we also have to undefine but there is no need to restore them later (our definitions are identical to those in the classes) we only have to avoid that `\newcommand` generates an error when the class is loaded.

```

31 \let \pushQED \relax
32 \let \popQED \relax
33 \let \qedhere \relax
34 \let \mathqed \relax
35 \let \openbox \relax
36 }

```

Then undefine them just before the class gets loaded:

```

37 \AddToHook{class/amsart/before}[block]{ \undefine@thm@macros }
38 \AddToHook{class/amsbook/before}[block]{ \undefine@thm@macros }
39 \AddToHook{class/amsproc/before}[block]{ \undefine@thm@macros }

```

After the class got loaded we restore the saved versions.

Most of the first aid code for the AMS classes needs to be applied after the class got loaded. Instead of putting all the code in hooks and thus into the format we now write it to files that are loaded after the class has been loaded.

```

40 \AddToHook{class/amsbook/after}[latex-lab-firstaid]{\input{latex-lab-firstaid-amsbook.ltx}}
41 \AddToHook{class/amsart/after}[latex-lab-firstaid]{\input{latex-lab-firstaid-amsart.ltx}}
42 \AddToHook{class/amsproc/after}[latex-lab-firstaid]{\input{latex-lab-firstaid-amsproc.ltx}}

43 \end{package}
44 \end{amscls}
45 \ExplSyntaxOn

```

Let us first restore the thmlike macros while this is fresh on our mind. Note that we don't have to restore all the QED-related commands as ours are identical to those defined by the AMS.

```

46 \let \newtheorem \@kernel@newtheorem
47 \let \newtheoremstyle \@kernel@newtheoremstyle
48 \let \theoremstyle \@kernel@theoremstyle
49 \let \proof \@kernel@proof
50 \let \endproof \@kernel@endproof

```

The amsart, amsbook and amsproc classes do not use \@author to store the author list but a command \authors. To be able to nevertheless use the authors in the xmp-metadata we map \@author to this new command.

The authors are set with a trivlist, this leads to faulty spacing and wrong tagging. This is not yet handled here, but an example how to correct this is in the test-amsart-title test.

```

51 \def\@author{\authors}

```

In these classes the abstract is stored in a \vbox, as it ends with a list an explicit \par is needed.

```

52 \renewenvironment{abstract}{%
53   \ifx\maketitle\relax
54     \ClassWarning{\@classname}{Abstract~ should~ precede~
55       \protect\maketitle\space in~ AMS~ document~ classes;~ reported}%
56   \fi
57   \global\setbox\abstractbox=\vtop \bgroup
58   \normalfont\Small
59   \list{}{\labelwidth\z@
60     \leftmargin3pc \rightmargin\leftmargin
61     \listparindent\normalparindent \itemindent\z@
62     \parsep\z@ \@plus\p@
63     \let\fullwidthdisplay\relax
64   }%
65   \item[\hskip\labelsep\scshape\abstractname.]%
66 }{%
67   \endlist
68   \par % <--- added

```

```

69 \egroup
70 \ifx\@setabstract\relax \@setabstracta \fi
71 }

```

In the ams classes's `\newtheoremstyle` has different defaults compared to the standalone `amsthm` package. Therefore the classes have their own implementation of the command that we need to overwrite as well.

```

72 \tl_set:Nn \newtheoremstyle@vspace@default {
73   .5\baselineskip \@plus .2\baselineskip \@minus .2\baselineskip
74 }

```

The above takes care of newly defined theoremstyles, but the default ones are already defined in the class so we have to adjust their settings directly.

```

75 \EditInstance{block}{thm-plain-1}
76 { begin-vspace = \newtheoremstyle@vspace@default }
77 \DeclareInstanceCopy{block}{thm-plain-2}{thm-plain-1}

78 \EditInstance{block}{thm-remark-1}
79 { begin-vspace = \newtheoremstyle@vspace@default }
80 \DeclareInstanceCopy{block}{thm-remark-2}{thm-remark-1}

81 \EditInstance{block}{thm-definition-1}
82 { begin-vspace = \newtheoremstyle@vspace@default }
83 \DeclareInstanceCopy{block}{thm-definition-2}{thm-definition-1}

```

Beside the spacing the thmstyles also differ from the default (but not for `amsart.cls`):

```

84 <amsbook,amsproc>
85 \EditInstance{thmstyle}{plain}
86 { before-hspace = \parindent , caption-decls = \scshape }
87 \EditInstance{thmstyle}{remark}
88 { before-hspace = \parindent , caption-decls = \scshape }
89 \EditInstance{thmstyle}{definition}
90 { before-hspace = \parindent , caption-decls = \scshape }
91 </amsbook,amsproc>

```

2.3.1 Headings

The classes redefine `\@startsection` and define their own heading commands which are not tagging compatible. This code is therefore disabled and replaced by standard definitions that process the heading arguments and call a suitable template instance. The top-level definitions are identical for all classes (only the heading instances then differ).

```

92 \DeclareDocumentCommand \part {s ={\shorttitle}o m}
93 { \ParseLaTeXeHeading {part} {#1} {#2} {#3} }

```

Chapters only exist in `amsbook` and their top-level definition is also slightly special as we need a different heading instance for numbered and unnumbered chapters.

```

94 <amsbook>
95 \DeclareDocumentCommand \chapter {s ={\shorttitle}O{} m}
96 {
97   \IfBooleanTF{#1}
98   { \ParseLaTeXeHeading {chapter*} {#1} {#2} {#3} }
99   { \ParseLaTeXeHeading {chapter} {#1} {#2} {#3} }
100 }
101 </amsbook>

```

```

102 \DeclareDocumentCommand \section {s ={\shorttitle}o m}
103   { \ParseLaTeXeHeading {section} {#1} {#2} {#3} }

```

The `\specialsection` command is a variation of the `section` layout so has its own heading instance.

```

104 \DeclareDocumentCommand \specialsection {s ={\shorttitle}o m}
105   { \ParseLaTeXeHeading {specialsection} {#1} {#2} {#3} }

106 \DeclareDocumentCommand \subsection {s ={\shorttitle}o m}
107   { \ParseLaTeXeHeading {subsection} {#1} {#2} {#3} }

108 \DeclareDocumentCommand \subsubsection {s ={\shorttitle}o m}
109   { \ParseLaTeXeHeading {subsubsection} {#1} {#2} {#3} }

110 \DeclareDocumentCommand \paragraph {s ={\shorttitle}o m}
111   { \ParseLaTeXeHeading {paragraph} {#1} {#2} {#3} }

112 \DeclareDocumentCommand \subparagraph {s ={\shorttitle}o m}
113   { \ParseLaTeXeHeading {subparagraph} {#1} {#2} {#3} }

```

The heading template instances have some subtle differences in the different AMS classes. We therefore set all of them up by document class, even though some of the code is the same.

```

114 <amsbook>

115 \DeclareInstance{heading}{part}{display}
116   {
117     , name                = part
118     , level               = -1
119     , placement           = page
120     , placement-start-code = \cleardoublepage \thispagestyle{empty}
121                           \null \vfil \markboth{}{}
122     , placement-end-code  = \vfil \vfil \newpage \thispagestyle{empty}
123     , mark-cmd            = \partmark {#1}
124     , heading-decls       = \centering\fontsize{\@xvipt}{22}\bfseries
125     , headformat-instance = part
126     , text                = \partname
127     , number-format       = #1
128     , title-decls         = \fontsize{\@xxpt}{25}\bfseries
129   }

```

Current heading templates aren't flexible enough and do not properly support altering the layout for unnumbered headings compared to their numbered counterparts. For this reason we have two heading instances, one for `\chapter` and one for `\chapter*`.

```

130 \DeclareInstance{heading}{chapter}{display}
131   {
132     , name                = chapter
133     , level               = 0
134     , mark-cmd            = \chaptermark {#1}
135     , para-indent         = true
136     , placement           = top
137     , after-penalty-vspace = 42pt % 32pt plus \topskip
138     , after-vspace        = 22pt
139     , heading-decls       = \normalsize\mdseries\centering

```

```

140     , headformat-instance = chapter
141     , text                = \MakeUppercase{\chaptername}
142     , number-format      = #1
143     , title-decls        = \fontsize{\@xivpt}{18} \bfseries
144     , contents-extra      = \addtocontents{lof}{\protect\addvspace{10\p@}}
145                           \addtocontents{lot}{\protect\addvspace{10\p@}}
146   }

147 \DeclareInstanceCopy{heading}{chapter*}{chapter}
148 \EditInstance{heading}{chapter*}{ after-penalty-vspace = 72pt } % 62pt + \topskip

149 \DeclareInstance{headformat}{chapter}{display}
150   {
151     , heading-indent = 0pt
152     , separator-b    = \par \nobreak \vspace{18pt}
153   }

154 \DeclareInstance{heading}{section}{display}
155   {
156     , name              = section
157     , level             = 1
158     , mark-cmd          = \sectionmark{#1}
159     , para-indent       = true
160     , before-vspace     = 0.7\linespacing plus \linespacing
161     , after-vspace      = 0.5\linespacing
162     , heading-decls     = \normalfont \bfseries \centering
163     , headformat-instance = section
164     , number-format     = #1.
165     , number-decls      = \upshape
166   }
167 \DeclareInstance{headformat}{section}{hang}
168   {
169     separator-b = \hspace{0.5em}
170   }

    A \specialsection is like a \section with two changes in the parameter settings.

171 \DeclareInstanceCopy{heading}{specialsection}{section}
172 \EditInstance{heading}{specialsection}
173   {
174     , before-vspace = 2\linespacing plus \linespacing
175     , heading-decls = \large \scshape \centering
176   }

177 \DeclareInstance{heading}{subsection}{runin}
178   {
179     , name              = subsection
180     , level             = 2
181     , mark-cmd          = \subsectionmark{#1}
182     , before-vspace     = 0.5\linespacing plus 0.7\linespacing
183     , after-hspace      = \fontdimen2\font
184     , heading-decls     = \normalfont \bfseries
185     , headformat-instance = subsection
186     , number-format     = #1.
187     , number-decls      = \upshape
188     , title-format      = #1 \AddPunct{.}

```

```

189 }
190 \DeclareInstance{headformat}{subsection}{runin}
191 {
192   , separator-b      = \hspace{0.5em}
193   , heading-indent   = \normalparindent
194 }

195 \DeclareInstance{heading}{subsubsection}{runin}
196 {
197   , name              = subsubsection
198   , level             = 3
199   , mark-cmd          = \subsubsectionmark{#1}
200   , before-vspace     = 0pt
201   , after-hspace      = \fontdimen2\font
202   , heading-decls     = \normalfont \itshape
203   , headformat-instance = subsubsection
204   , number-format     = #1.
205   , number-decls      = \upshape
206   , title-format      = #1 \AddPunct{.}
207 }
208 \DeclareInstance{headformat}{subsubsection}{runin}
209 {
210   , separator-b      = \hspace{0.5em}
211   , heading-indent   = \normalparindent
212 }

213 \DeclareInstance{heading}{paragraph}{runin}
214 {
215   , name              = paragraph
216   , level             = 4
217   , mark-cmd          = \paragraphmark{#1}
218   , before-vspace     = 0pt
219   , after-hspace      = \fontdimen2 \font
220   , heading-decls     = \normalfont
221   , headformat-instance = paragraph
222   , number-format     = #1.
223   , number-decls      = \upshape
224   , title-format      = #1 \AddPunct{.}
225 }
226 \DeclareInstance{headformat}{paragraph}{runin}
227 {
228   , separator-b      = \hspace{0.5em}
229   , heading-indent   = \normalparindent
230 }

231 \DeclareInstance{heading}{subparagraph}{runin}
232 {
233   , name              = subparagraph
234   , level             = 5
235   , mark-cmd          = \subparagraphmark{#1}
236   , before-vspace     = 0pt
237   , after-hspace      = \fontdimen2\font
238   , heading-decls     = \normalfont
239   , headformat-instance = subparagraph
240   , number-format     = #1.

```

```

241     , number-decls      = \upshape
242     , title-format      = #1 \AddPunct{.}
243   }
244 \DeclareInstance{headformat}{subparagraph}{runin}
245 {
246     , separator-b = \hspace{0.5em}
247 }

```

The running header setup from `amsbook` doesn't correctly function and needs adjustments. For now we provide some simple redefinitions which do not necessarily produce exactly the same results.

```

248 \def\partmark#1{%
249     \markboth{}{}
250 }
251 \def\chaptermark#1{%
252     \markboth {\MakeUppercase{%
253         \ifnum \c@secnumdepth >\m@ne
254             \thechapter. \ %
255         \fi
256         #1}}{}
257 }
258 \def\sectionmark#1{%
259     \markright {\MakeUppercase{%
260         \ifnum \c@secnumdepth >\z@
261             \thesection. \ %
262         \fi
263         #1}}
264 }
265 </amsbook>
266 <*amsart>

```

The use of `\mdseries` in several of the instances is really odd (and not done in `amsbook` or `amsproc`) but it is what the current class generates as a layout, so we keep it that way.

```

267 \DeclareInstance{heading}{part}{display}
268 {
269     , name                = part
270     , level               = -1                % -1 or 0 ?
271     , mark-cmd            = \partmark{#1}
272     , para-indent        = true
273     , before-vspace       = \linespacing plus \linespacing
274     , after-vspace        = 0.5 \linespacing
275     , heading-decls       = \normalfont \bfseries \raggedright
276     , headformat-instance = part
277     , text                = \partname
278     , number-format       = \theheading .
279     , number-decls        = \mdseries \upshape % proc: has just \upshape
280 }
281 \DeclareInstance{headformat}{part}{hang}
282 {
283     , separator-b = \hspace{0.5em}
284 }

```

```

285 \DeclareInstance{heading}{section}{display}
286 {
287   , name                = section
288   , level               = 1
289   , mark-cmd            = \sectionmark{#1}
290   , para-indent         = true
291   , before-vspace       = 0.7\linespacing plus \linespacing
292   , after-vspace        = 0.5\linespacing
293   , heading-decls       = \normalfont \scshape \centering
294   , headformat-instance = section
295   , number-format       = \theheading .
296   , number-decls        = \mdseries \upshape
297 }
298 \DeclareInstance{headformat}{section}{hang}
299 {
300   , separator-b = \hspace{0.5em}
301 }

```

The `\specialsection` command is a variation of the `section` layout.

```

302 \DeclareInstanceCopy{heading}{specialsection}{section}
303 \EditInstance{heading}{specialsection}
304 {
305   , before-vspace = \linespacing plus \linespacing
306   , heading-decls = \normalfont \centering
307 }
308 \DeclareInstance{heading}{subsection}{runin}
309 {
310   , name                = subsection
311   , level               = 2
312   , mark-cmd            = \subsectionmark{#1}
313   , before-vspace       = 0.5\linespacing plus 0.7\linespacing
314   , after-hspace        = \fontdimen2\font
315   , heading-decls       = \normalfont \bfseries
316   , headformat-instance = subsection
317   , number-format       = \theheading .
318   , number-decls        = \mdseries \upshape
319   , title-format        = #1 \AddPunct{.}
320 }
321 \DeclareInstance{headformat}{subsection}{runin}
322 {
323   , separator-b = \hspace{0.5em}
324 }
325 \DeclareInstance{heading}{subsubsection}{runin}
326 {
327   , name                = subsubsection
328   , level               = 3
329   , mark-cmd            = \subsubsectionmark{#1}
330   , before-vspace       = 0.5\linespacing plus 0.7\linespacing
331   , after-hspace        = \fontdimen2\font
332   , heading-decls       = \normalfont \itshape
333   , headformat-instance = subsubsection
334   , number-format       = \theheading .

```

```

335     , number-decls      = \mdseries \upshape
336     , title-format      = #1 \AddPunct{.}
337   }
338 \DeclareInstance{headformat}{subsubsection}{runin}
339 {
340     , separator-b = \hspace{0.5em}
341 }

342 \DeclareInstance{heading}{paragraph}{runin}
343 {
344     , name              = paragraph
345     , level              = 4
346     , mark-cmd           = \paragraphmark{#1}
347     , before-vspace      = 0pt
348     , after-hspace       = \fontdimen2\font
349     , heading-decls      = \normalfont
350     , headformat-instance = paragraph
351     , number-format      = \theheading .
352     , number-decls       = \mdseries \upshape
353     , title-format       = #1 \AddPunct{.}
354 }
355 \DeclareInstance{headformat}{paragraph}{runin}
356 {
357     , separator-b = \hspace{0.5em}
358 }

359 \DeclareInstance{heading}{subparagraph}{runin}
360 {
361     , name              = subparagraph
362     , level              = 5
363     , mark-cmd           = \subparagraphmark{#1}
364     , before-vspace      = 0pt
365     , after-hspace       = \fontdimen2\font
366     , heading-decls      = \normalfont
367     , headformat-instance = subparagraph
368     , number-format      = \theheading .
369     , number-decls       = \mdseries \upshape
370     , title-format       = #1 \AddPunct{.}
371 }
372 \DeclareInstance{headformat}{subparagraph}{runin}
373 {
374     , separator-b = \hspace{0.5em}
375 }

376 \</amsart>
377 \<amsproc>

378 \DeclareInstance{heading}{part}{display}
379 {
380     , name              = part
381     , level              = -1 % -1 or 0 ?
382     , mark-cmd           = \partmark{#1}
383     , para-indent        = true
384     , before-vspace      = \linespacing plus \linespacing

```

```

385     , after-vspace      = 0.5 \linespacing
386     , heading-decls     = \normalfont \bfseries \raggedright
387     , headformat-instance = part
388     , text              = \partname
389     , number-format     = \theheading .
390     , number-decls      = \upshape
391   }
392 \DeclareInstance{headformat}{part}{hang}
393 {
394     , separator-b = \hspace{0.5em}
395 }

396 \DeclareInstance{heading}{section}{display}
397 {
398     , name              = section
399     , level             = 1
400     , mark-cmd          = \sectionmark{#1}
401     , para-indent       = true
402     , before-vspace     = 0.7\linespacing plus \linespacing
403     , after-vspace      = 0.5\linespacing
404     , heading-decls     = \normalfont \bfseries \centering
405     , headformat-instance = section
406     , number-format     = \theheading .
407     , number-decls      = \upshape
408   }
409 \DeclareInstance{headformat}{section}{hang}
410 {
411     , separator-b = \hspace{0.5em}
412 }

```

The `\specialsection` command is a variation of the `section` layout.

```

413 \DeclareInstanceCopy{heading}{specialsection}{section}
414 \EditInstance{heading}{specialsection}
415 {
416     , before-vspace = \linespacing plus \linespacing
417     , heading-decls = \large \scshape \centering
418 }

419 \DeclareInstance{heading}{subsection}{runin}
420 {
421     , name              = subsection
422     , level             = 2
423     , mark-cmd          = \subsectionmark{#1}
424     , before-vspace     = 0.5\linespacing plus 0.7\linespacing
425     , after-hspace      = \fontdimen2\font
426     , heading-decls     = \normalfont \bfseries
427     , headformat-instance = subsection
428     , number-format     = \theheading .
429     , number-decls      = \upshape
430     , title-format      = #1 \AddPunct{.}
431   }
432 \DeclareInstance{headformat}{subsection}{runin}
433 {
434     , separator-b      = \hspace{0.5em}
435     , heading-indent   = \normalparindent

```

```

436 }

437 \DeclareInstance{heading}{subsubsection}{runin}
438 {
439   , name           = subsubsection
440   , level          = 3
441   , mark-cmd       = \subsubsectionmark{#1}
442   , before-vspace  = 0.5\linespacing plus 0.7\linespacing
443   , after-hspace   = \fontdimen2\font
444   , heading-decls  = \normalfont \itshape
445   , headformat-instance = subsubsection
446   , number-format  = \theheading .
447   , number-decls   = \upshape
448   , title-format   = #1 \AddPunct{.}
449 }

450 \DeclareInstance{headformat}{subsubsection}{runin}
451 {
452   , separator-b    = \hspace{0.5em}
453   , heading-indent = \normalparindent
454 }

455 \DeclareInstance{heading}{paragraph}{runin}
456 {
457   , name           = paragraph
458   , level          = 4
459   , mark-cmd       = \paragraphmark{#1}
460   , before-vspace  = 0pt
461   , after-hspace   = \fontdimen2\font
462   , heading-decls  = \normalfont
463   , headformat-instance = paragraph
464   , number-format  = \theheading .
465   , number-decls   = \upshape
466   , title-format   = #1 \AddPunct{.}
467 }

468 \DeclareInstance{headformat}{paragraph}{runin}
469 {
470   , separator-b    = \hspace{0.5em}
471   , heading-indent = \normalparindent
472 }

473 \DeclareInstance{heading}{subparagraph}{runin}
474 {
475   , name           = subparagraph
476   , level          = 5
477   , mark-cmd       = \subparagraphmark{#1}
478   , before-vspace  = 0pt
479   , after-hspace   = \fontdimen2\font
480   , heading-decls  = \normalfont
481   , headformat-instance = subparagraph
482   , number-format  = \theheading .
483   , number-decls   = \upshape
484   , title-format   = #1 \AddPunct{.}
485 }

486 \DeclareInstance{headformat}{subparagraph}{runin}
487 {

```

```

488     , separator-b = \hspace{0.5em}
489   }

490 \end{amsproc}

491 \begin{amsbook}

```

The AMS classes do a lot of low-level overwrites to macros building out the table of contents structures. `\starttoc` being one example which is partially fixed here. This isn't the full fix: various `\l@...` macros also need changes, but that has to wait for another day ... for now we only make sure that the code doesn't break badly, but they layout is not matching what the class normally produces!

```

492 \def\starttoc#1#2{%
493   \begingroup
494   \setTrue{#1}%
495   \ifx\contentsname#2%
496     \chapter[unnumbered,toc]{\contentsname}%
497   \else
498     \chapter[=,unnumbered]{#2}%
499   \fi
500   \typeout{#2}%
501   \makeatletter
502   \@input{\jobname.#1}%
503   \if@filesw
504     \exp\newwrite\csname tf@#1\endcsname
505     \immediate\exp\openout\csname tf@#1\endcsname \jobname.#1\relax
506   \fi
507   \global\@nobreakfalse
508   \endgroup
509   \newpage
510 }

511 \AddToHook{begindocument}{
512   \@namedef{r@tocindent-1}{1em}
513   \@namedef{r@tocindent0}{1em}
514   \@namedef{r@tocindent1}{1em}
515 }

516 \end{amsbook}

517 \ExplSyntaxOff
518 \end{amscls}
519 \end{package}

520 \ExplSyntaxOff

```

2.4 fncychap

First aid for fncychap is kept in a separate file.

```

521 \AddToHook{package/fncychap/after}{\input{latex-lab-firstaid-
522   fncychap.ltx}}

523 \end{package}

524 \end{fncychap}

```

524 `\ExplSyntaxOn`

The `fncychap` package has its own way of offering customizations including arbitrary rewrites for the commands `\DOCH`, `\DOTI`, and `\DOTIS`. In order to support customizations in existing documents (defining chapter layout variations in this fashion) we define a headformat in which the three commands are simply called to typeset the heading, i.e., the template doesn't offer any customizations through keys—everything is done through the interfaces of `fncychap` as before.

For the same reason most of the keys of the chapter instance are nonfunctional, i.e., they exist but aren't used with a few exceptions. After all, the purpose here is to make `fncychap` tagging compatible and not to extend its functionality or offer additional customization possibilities.

The two values in the heading chapter instance that we need to alter are `after-vspace` and set it to zero (because this is handled inside `\DOTI` or `\DOTIS`) and `headformat-instance` as we want to call our own instance suitable to deal with the customization possibilities offered by `fncychap`. Thus we edit the default `heading chapter` instance in the following way.

```
525 \EditInstance{heading}{chapter}{
526   after-vspace      = 0pt
527   , headformat-instance = fncychap
528 }
```

This means we have to declare a new template `fncychap` of type `headformat` (with no keys as explained above):

```
529 \DeclareTemplateInterface{headformat}{fncychap}{6}{}%
```

The code for this template is fairly simple.

```
530 \DeclareTemplateCode{headformat}{fncychap}{6}{}%
531 {
```

Output some debugging info at the start.

```
532   \__head_show_arguments:nnnnnn {#1}{#2}{#3}{#4}{#5}{#6}
```

Everything else happens in a group to keep changes local.

```
533   \group_begin:
```

First, make a link target (with slight difference depending on whether or not the chapter is numbered).

```
534   \bool_if:NTF \l__head_unnumbered_bool
535     { \MakeLinkTarget [ \l__head_name_t1 ] {} }
536     { \MakeLinkTarget { \l__head_name_t1 } }
```

Next follow some default paragraph settings that may or may not be overwritten by declarations for a specific chapter layout from the package.

```
537   \parindent 0pt
538   \raggedright
539   \interlinepenalty 10000
540   \normalfont
```

Then we have a few tagging socket settings for `\parbox` or `minipage` environments, because some heading designs use `\parboxes` and they need the special settings in the context of a heading to make the tagging work.

```
541 \AssignTaggingSocketPlug{parbox/before}{heading}
542 \AssignTaggingSocketPlug{minipage/before}{heading}
543 \AssignTaggingSocketPlug{para/restore}{noop}
```

Finally, we have the bare heading design as used in the package. When the heading is numbered then `\DOCH` is responsible to place the fixed heading text (e.g., Chapter) and the chapter number.

We start by calling `\fncychap@setup@thechapter` which gives `\thechapter` a special definition while the heading is typeset (since we are in a group here this is reverted immediately after).

```
544 \fncychap@setup@thechapter
545 \bool_if:NF \l__head_unnumbered_bool
546 { \DOCH }
```

If we are in “main matter” then `\DOTI` typesets the chapter title, otherwise `\DOTIS` is used to typeset the chapter title.

```
547 \legacy_if:nTF { @mainmatter }
548 { \DOTI{#4} }
549 { \DOTIS{#4} }
```

We finish off with a `\par` to ensure that we are back in vertical mode and then close the group.

```
550 \par
551 \group_end:
552 }
```

We use this template to create the `headformat` instance used in the `heading` instance. Since the template has no keys this is a trivial declaration:

```
553 \DeclareInstance{headformat}{fncychap}{fncychap}{}
```

To put a `<heading-number>` structure around the number without really knowing how the heading layout is done, we hook into `\thechapter`. This redefinition only happens within the `headformat` instance when we typeset the heading. Elsewhere `\thechapter` retains its usual definition, e.g., when used inside the document or in a running header or toc entry.

```
554 \def\fncychap@setup@thechapter{
555 \def\thechapter{\thespecialchapter{\@arabic\c@chapter}}
556 }
```

```
557 \def\appendix{\par
558 \setcounter{chapter}{0}
559 \setcounter{section}{0}
560 \gdef\@chapapp{\appendixname}
561 \gdef\thechapter{\@Alph\c@chapter}}
```

Add the special setup for using in the `headformat` template above.

```
562 \gdef\fncychap@setup@thechapter{
563 \def\thechapter{\thespecialchapter{\@Alph\c@chapter}}
564 }
565 }
```

The `\thespecialchapter` ensures that `hmode` is started and then normally generates a `<heading-number>` structure but only if we aren't already in an `<Lb1>`. It then typesets the current number.

```

566 \protected\def\thespecialchapter#1{
567   \leavevmode
568   \tag_if_in:nTF{Lb1}
569     { #1 }
570     {
571       \UseTaggingSocket{inline/begin}{tag=heading-number}
572       #1
573       \UseTaggingSocket{inline/end}
574     }
575 }

```

`\mghrulefill`

This here is a special rule definition from `fncychap` that we overwrite to add the necessary tagging support.

```

576 \protected\def\mghrulefill#1{\leavevmode
577   \UseTaggingSocket{rule}{}{
578     \leaders\hrule\@height #1\hfill\kern\z@
579   }
580 }

```

(End of definition for `\mghrulefill`. This function is documented on page ??.)

```

581 \ExplSyntaxOff

```

```

582 </fncychap>

```

```

583 <*package>

```

2.5 cleveref

The `cleveref` package redefines `\@makefntext` and this means that the patches in the new footnote code fails. We use a hook instead.

```

584 \AddToHook{package/cleveref/after}
585 {
586   \let\@makefntext\cref@old@makefntext
587   \AddToHook{cmd/\@makefntext/before}{%
588     \cref@constructprefix{footnote}{\cref@result}%
589     \protected@edef\cref@currentlabel{%
590       [footnote] [\arabic{footnote}] [\cref@result]%
591       \p@footnote\@thefnmark}}
592 }

```

2.6 booktabs

In some cases `booktabs` inserts a `\multispan` into the table (through the commands `\@cmidruleb` and `\@cmidrulea` and this then errors with the tagging code. This affects both `tabular` and `longtable` (but `longtable` more as `booktabs` handles lines in `longtable` differently). See also issue <https://github.com/latex3/tagging-project/issues/69>

```

593 \ExplSyntaxOn
594 \AddToHook{package/booktabs/after}
595 {

```

```

596 \def\@cmidrulea{
597   \multispan\@cmidla
598   &\multispan\@cmidlb
599   \unskip\hskip\cmrkern@l
600 {
601   \UseTaggingSocket{tbl/leaders/begin}
602   \CT@arc@\leaders\hrule \@height\@thisrulewidth\hfill\kern\z@}
603   \hskip\cmrkern@r
604   \UseTaggingSocket{tbl/leaders/end}
605   \tbl_gdecr_row_count:
606   \cr}
607
608 \def\@cmidruleb{%
609   \multispan\@cmidlb
610   \unskip\hskip \cmrkern@l%
611 {
612   \UseTaggingSocket{tbl/leaders/begin}
613   \CT@arc@\leaders\hrule \@height\@thisrulewidth\hfill\kern\z@}
614   \hskip\cmrkern@r
615   \UseTaggingSocket{tbl/leaders/end}
616   \tbl_gdecr_row_count:
617   \cr}
618 }
619 \ExplSyntaxOff

```

2.7 fancyvrb

The firstaid adds first partial tagging support to the environments of fancyvrb (inline verbatim is untested). This supports then also packages like minted which internally uses fancyvrb and classes like l3doc (where currently the verbatim environment based on fancyvrb is overwritten by the block code). The environments are surrounded by a **verbatim** structure, every line by a **codeline** structure. Line numbers are tagged as Lbl, inside of the **codeline** structure. The frame lines are marked as artifact.

\FV@LeaveVMode If we are in vmode we have to open a semantic-para structure, if we are in hmode we have to set para mode to flattened before the fancyvrb code issues the **\par**. The closing of the semantic-para structure is handled by the doendpe code in the block code.

```

620 \ExplSyntaxOn
621 \AddToHook{package/fancyvrb/after}
622 {
623   \def\FV@LeaveVMode{%
624     \if@noskipsec
625       \leavevmode
626     \else
627       \if@FV@ResetMargins\if@inlabel\leavevmode\fi\fi
628     \fi
629     \ifvmode
630       \@noparlisttrue
631       \UseTaggingSocket{para/semantic/begin}{\}
632     \else
633       \tagpdfsetup{para/flattened=true}
634       \@noparlistfalse
635       \unskip\par

```

```

636     \fi
637   }

```

(End of definition for `\FV@LeaveVMode`.)

\FV@List At the begin of the list code we have to tag the frame as artifact and start the verbatim structure

```

638   \def\FV@List#1{%
639     \begingroup
640     \FV@UseKeyValues
641     \FV@LeaveVMode
642     \if@inlabel\else\setbox\@labels=\box\voidb@x\fi
643     \FV@ListNesting{#1}%
644     \FV@ListParameterHook
645     \FV@ListVSpace
646     \FV@SetLineWidth
647     \FV@InterLinePenalty
648     \let\FV@ProcessLine\FV@ListProcessLine@i
649     \FV@CatCodes
650     \FV@FormattingPrep
651     \FV@ObeyTabsInit
652     \cs_if_exist:NT \FV@BeginListFrame
653     {
654       \UseTaggingSocket{mc}{artifact}{\FV@BeginListFrame}
655     }
656     \UseTaggingSocket{struct/begin}{tag=\UseStructureName{block/verbatim}}
657   }

```

(End of definition for `\FV@List`.)

\FV@endList At the end of the list code we close the verbatim structure and tag the frame as artifact.

```

658   \def\FV@endList{%
659     \FV@ListProcessLastLine
660     \UseTaggingSocket{struct/end}
661     \cs_if_exist:NT \FV@endListFrame
662     {
663       \UseTaggingSocket{mc}{artifact}{\FV@endListFrame}
664     }
665     \@endparenv
666     \endgroup
667     \@endpetrue
668   }

```

(End of definition for `\FV@endList`.)

\FV@ListProcessLine At last the tagging of the code lines. Here we have to tag also numbers and frame parts if they exist.

```

669   \def\FV@ListProcessLine#1{%
670     \hbox to \hsize{%
671       \kern\leftmargin
672       \hbox to \linewidth{%
673         \UseTaggingSocket{struct/begin}{tag=\UseStructureName{block/verbatim/codeline}}
674         \cs_if_exist:NT \FV@LeftListNumber
675         {
676           \UseTaggingSocket{struct-mc}{tag=\UseStructureName{block/verbatim/linenumber}}

```

```

677         {
678             \FV@LeftListNumber
679         }
680     }
681     \cs_if_exist:NT \FV@LeftListFrame
682     {
683         \UseTaggingSocket{mc}{artifact}{\FV@LeftListFrame}
684     }
685     \UseTaggingSocket{mc}{}{\FancyVerbFormatLine{#1}}%
686     \UseTaggingSocket{struct/end}\hss
687     \cs_if_exist:NT \FV@RightListFrame
688     {
689         \UseTaggingSocket{mc}{artifact}{\FV@RightListFrame}
690     }
691     \cs_if_exist:NT \FV@RightListNumber
692     {
693         \UseTaggingSocket{struct-mc}{tag=\UseStructureName{block/verbatim/linenumber}}
694         {
695             \FV@RightListNumber
696         }
697     }
698 }
699 \hss}}
700 }
701 \ExplSyntaxOff

```

(End of definition for \FV@ListProcessLine.)

2.8 The float package first aid

The float package has a number of incompatibilites:

- Restyling floats with `\restylefloat` breaks the tagging support.
- New float types created with `\newfloat` are not properly tagged (tagging issue 890)
- Non-floating floats using the H placement are not properly tagged.

The following firstaid addresses these problems. A flaw is that if the H-float is inside a paragraph and followed by text there the text is tagged as a new semantic-para structure.

Because of this and other limitations the final solution may look noticeably different including the used tagging sockets, etc. TODO

The firstaid for float is kept in a separate file.

```

702 \AddToHook{package/float/after}[latex-lab-firstaid]{\input{latex-lab-firstaid-float.ltx}}
703 \</package>
704 <float>
705 \ProvidesFile{latex-lab-firstaid-float.ltx}
706 [\ltxlabfirstaiddatetime\space \ltxlabfirstaidversion\space LaTeX-lab-firstaid: Tagging support fo

```

Tagging sockets for the H-floats

```

707 \ExplSyntaxOn
708 \NewTaggingSocket{float/H/begin}{1}
709 \NewTaggingSocketPlug{float/H/begin}{default}
710 {

```

```

711 \mode_if_horizontal:T {\tag_mc_end_push: \tag_struct_end:}
712 \tag_struct_begin:n {tag=\UseStructureName{float/#1}}
713 }
714 \AssignTaggingSocketPlug{float/H/begin}{default}
715
716 \NewTaggingSocket{float/H/end}{0}
717 \NewTaggingSocketPlug{float/H/end}{default}
718 {
719   \tag_struct_end:
720   \mode_if_horizontal:T
721   {
722     \tag_struct_begin:n {tag=\UseStructureName{para/textblock}}
723     \tag_mc_begin_pop:n {}
724   }
725 }
726 \AssignTaggingSocketPlug{float/H/end}{default}

```

and here the redefinitions of the commands of float for the H-floats:

```

727 \def\@float@HH#1[H]{\UseTaggingSocket{float/H/begin}{#1}%
728 \expandafter\let\csname end#1\endcsname\float@endH
729 \let\@currbox\float@box
730 \def\@capttype{#1}\setbox\@floatcapt=\vbox{}%
731 \expandafter\ifx\csname fst#1\endcsname\relax
732 \@flstylefalse\else\@flstyletrue\fi
733 \setbox\@currbox\color@vbox\normalcolor
734 \vbox\bgroup \hsize\columnwidth \@parboxrestore
735 \@floatboxreset \@setnobreak
736 \ignorespaces}
737 \renewcommand\float@endH{\@endfloatbox\UseTaggingSocket{float/H/end}
738 \vskip\intextsep
739 \if@flstyle\setbox\@currbox\float@makebox\columnwidth\fi
740 \box\@currbox\vskip\intextsep\relax}
741 \ExplSyntaxOff

```

Float defines two float styles which need to get tagging support. This here is the plain style which is also used for plaintop and boxed:

```

742 \renewcommand\floatc@plain[2]{%
743 \setbox\@tempboxa\hbox{\measuring@true{\@fs@cfont #1:} #2}%
744 \UseTaggingSocket{caption/begin}{\@current@float@struct}%
745 \ifdim\wd\@tempboxa>\hsize
746 \UseTaggingSocket{caption/label/begin}%
747 {\@fs@cfont #1:}%
748 \UseTaggingSocket{caption/label/end}%

```

The space is intentional!

```

749 \UseTaggingSocket{para/begin} %
750 #2\par
751 \else
752 \hbox to \hsize {\hfil
753 \UseTaggingSocket{caption/label/begin}%
754 {\@fs@cfont #1:}%
755 \UseTaggingSocket{caption/label/end}%

```

The space is intentional!

```

756     \UseTaggingSocket{para/begin} %
757     #2%
758     \UseTaggingSocket{para/end}\hfil}%
759     \par
760     \fi

```

The caption structure is not closed here, but in the `\float@caption` and `\float@ccon` commands.

```

761 }

```

This adds tagging support to the ruled style:

```

762 \renewcommand\floatc@ruled[2]{%
763   \UseTaggingSocket{caption/begin}{\@currentfloat@struct}%
764   \UseTaggingSocket{caption/label/begin}%
765   {\@fs@cfont #1}%
766   \UseTaggingSocket{caption/label/end}%

```

The space is intentional!

```

767   \UseTaggingSocket{para/begin} #2\par %

```

The caption is not closed here but in the `\float@caption` and `\float@ccon` commands.

```

768 }

```

The `float` package adds a final optional argument to `caption` which can be used to add text (this is not documented). We add the `caption/end` socket after this test, so that it is part of the caption. We also force a `\par` to avoid nested mc with `pdflatex`.

```

769 \long\def\float@caption#1[#2]#3{\par\addcontentsline{\@nameuse{ext@#1}}{#1}%
770   {\protect\numberline{\@nameuse{the#1}}{\ignorespaces #2}}
771   \global\setbox\@floatcapt\vbox\bgroup\@parboxrestore
772   \normalsize\@fs@capt{\@nameuse{fnum@#1}}{\ignorespaces #3}%
773   \@ifnextchar[{\float@ccon}{\UseTaggingSocket{caption/end}\egroup}%
774   }
775 \long\def\float@ccon[#1]{#1\par\UseTaggingSocket{caption/end}\egroup}

```

If a new float is declared it should setup the container. We also adapt the tag of this container:

```

776 \renewcommand\newfloat[3]{\@namedef{ext@#1}{#3}
777   \tagpdfsetup{float/new=#1,role/new-tag=#1-floats/Sect}%
778   \AssignStructureRole{float/#1s}{#1-floats}%
779   \let\float@do=\relax
780   \xdef\@tempa{\noexpand\float@exts{\the\float@exts \float@do{#3}}}%
781   \@tempa
782   \floatplacement{#1}{#2}%
783   \@ifundefined{fname@#1}{\floatname{#1}{#1}}{}
784   \expandafter\edef\csname ftype@#1\endcsname{\value{float@type}}%
785   \addtocounter{float@type}{\value{float@type}}
786   \restylefloat{#1}%
787   \expandafter\edef\csname fnum@#1\endcsname%
788     {\expandafter\noexpand\csname fname@#1\endcsname}
789     \expandafter\noexpand\csname the#1\endcsname}
790   \@ifnextchar[{\float@newx{#1}}{}
791     {\@ifundefined{c@#1}{\newcounter{#1}\@namedef{the#1}{\arabic{#1}}}%
792     {}}}
793 \float>

```

2.9 The apacite firstaid

The apacite package redefines `\@lbibitem` and removes the arguments. This means that the patching with a generic hook with arguments fails. So we read the argument.

```
794 \begin{package}
795 \AddToHook{package/apacite/after}
796   {\if@APAC@natbib@apa\def\@lbibitem[#1]#2{\NAT@swattrue\orig@nat@lbibitem[#1]{#2}}\fi}
797 \end{package}
```