The latex-lab-firstaid package
Temporary patches to external packages needed for the tagging project

LaTeX 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

```
⟨*package⟩
⟨@@=tag⟩

\ProvidesPackage {latex-lab-testphase-firstaid} [%
   \ltlabfirstaiddate\space v\ltlabfirstaidversion\space
   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 csname `\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.

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

*Initial implementation done by Ulrike Fischer

```
  \exp_args:Nce\str_if_eq:onT{ver@#1.#2}{#3}
}
```

(*End of definition for* `\FirstAidNeededT`.)

2.1 tikz/pgf

ti*k*z 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 LaTeX `\input` command. Check https://github.com/pgf-tikz/pgf/issues/1424 for changes.

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

2.2 listings

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

```
\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):

```
\let \@kernel@newtheorem      \newtheorem
\let \@kernel@newtheoremstyle \newtheoremstyle
\let \@kernel@theoremstyle    \theoremstyle
\let \@kernel@proof           \proof
\let \@kernel@endproof        \endproof

\def\undefine@thm@macros{
  \let \newtheoremstyle \relax
  \let \theoremstyle    \relax

  \let \proof           \relax
  \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.

```
  \let \pushQED    \relax
  \let \popQED     \relax
  \let \qedhere    \relax
  \let \mathqed    \relax
  \let \openbox    \relax
}
```

Then undefine them just before the class gets loaded:

```
\AddToHook{class/amsart/before} [block]{ \undefine@thm@macros }
\AddToHook{class/amsbook/before}[block]{ \undefine@thm@macros }
\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.

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

```
⟨/package⟩
⟨*amscls⟩
\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.

```
 \let \newtheorem      \@kernel@newtheorem
 \let \newtheoremstyle \@kernel@newtheoremstyle
 \let \theoremstyle    \@kernel@theoremstyle
 \let \proof           \@kernel@proof
 \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.

```
\def\@author{\authors}
```

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

```
\renewenvironment{abstract}{%
    \ifx\maketitle\relax
      \ClassWarning{\@classname}{Abstract~ should~ precede~
        \protect\maketitle\space in~ AMS~ document~ classes;~ reported}%
    \fi
    \global\setbox\abstractbox=\vtop \bgroup
      \normalfont\Small
      \list{}{\labelwidth\z@
        \leftmargin3pc \rightmargin\leftmargin
        \listparindent\normalparindent \itemindent\z@
        \parsep\z@ \@plus\p@
        \let\fullwidthdisplay\relax
      }%
      \item[\hskip\labelsep\scshape\abstractname.]%
 }{%
    \endlist
    \par              % <--- added
```

```
    \egroup
    \ifx\@setabstract\relax \@setabstracta \fi
  }
```

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.

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

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.

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

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

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

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

```
⟨*amsbook, amsproc⟩
\EditInstance{thmstyle}{plain}
  { before-hspace =  \parindent , caption-decls = \scshape }
\EditInstance{thmstyle}{remark}
  { before-hspace =  \parindent , caption-decls = \scshape }
\EditInstance{thmstyle}{definition}
  { before-hspace =  \parindent , caption-decls = \scshape }
⟨/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).

```
\DeclareDocumentCommand \part {s ={shorttitle}o m}
    { \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.

```
⟨*amsbook⟩
\DeclareDocumentCommand \chapter {s ={shorttitle}O{} m}
  {
    \IfBooleanTF{#1}
      { \ParseLaTeXeHeading {chapter*} {#1} {#2} {#3} }
      { \ParseLaTeXeHeading {chapter}  {#1} {#2} {#3} }
  }
⟨/amsbook⟩
```

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

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

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

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

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

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

\DeclareDocumentCommand \subparagraph {s ={shorttitle}o m}
  { \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.

⟨∗amsbook⟩

```
\DeclareInstance{heading}{part}{display}
  {
    , name                  = part
    , level                 = -1
    , placement             = page
    , placement-start-code  =  \cleardoublepage \thispagestyle{empty}
                               \null \vfil \markboth{}{}
    , placement-end-code    = \vfil \vfil \newpage \thispagestyle{empty}
    , mark-cmd              = \partmark {#1}
    , heading-decls         = \centering\fontsize{\@xviipt}{22}\bfseries
    , headformat-instance   = part
    , text                  = \partname
    , number-format         = #1
    , title-decls           = \fontsize{\@xxpt}{25}\bfseries
  }
```

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*`.

```
\DeclareInstance{heading}{chapter}{display}
  {
    , name                = chapter
    , level               = 0
    , mark-cmd            = \chaptermark {#1}
    , para-indent         = true
    , placement           = top
    , after-penalty-vspace = 42pt  % 32pt plus \topskip
    , after-vspace        = 22pt
    , heading-decls       = \normalsize\mdseries\centering
```

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

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

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

\DeclareInstance{heading}{section}{display}
  {
    , name               = section
    , level              = 1
    , mark-cmd           = \sectionmark{#1}
    , para-indent        = true
    , before-vspace      = 0.7\linespacing plus \linespacing
    , after-vspace       = 0.5\linespacing
    , heading-decls      = \normalfont \bfseries \centering
    , headformat-instance = section
    , number-format      = #1.
    , number-decls       = \upshape
  }
\DeclareInstance{headformat}{section}{hang}
  {
    separator-b = \hspace{0.5em}
  }
```

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

```
\DeclareInstanceCopy{heading}{specialsection}{section}
\EditInstance{heading}{specialsection}
   {
     , before-vspace = 2\linespacing plus \linespacing
     , heading-decls = \large \scshape \centering
   }

\DeclareInstance{heading}{subsection}{runin}
  {
    , name               = subsection
    , level              = 2
    , mark-cmd           = \subsectionmark{#1}
    , before-vspace      = 0.5\linespacing plus 0.7\linespacing
    , after-hspace       = \fontdimen2\font
    , heading-decls      = \normalfont \bfseries
    , headformat-instance = subsection
    , number-format      = #1.
    , number-decls       = \upshape
    , title-format       = #1 \AddPunct{.}
```

```
  }
\DeclareInstance{headformat}{subsection}{runin}
  {
    , separator-b    = \hspace{0.5em}
    , heading-indent = \normalparindent
  }

\DeclareInstance{heading}{subsubsection}{runin}
  {
    , name                = subsubsection
    , level               = 3
    , mark-cmd            = \subsubsectionmark{#1}
    , before-vspace       = 0pt
    , after-hspace        = \fontdimen2\font
    , heading-decls       = \normalfont \itshape
    , headformat-instance = subsubsection
    , number-format       = #1.
    , number-decls        = \upshape
    , title-format        = #1 \AddPunct{.}
  }
\DeclareInstance{headformat}{subsubsection}{runin}
  {
    , separator-b    = \hspace{0.5em}
    , heading-indent = \normalparindent
  }

\DeclareInstance{heading}{paragraph}{runin}
  {
    , name                = paragraph
    , level               = 4
    , mark-cmd            = \paragraphmark{#1}
    , before-vspace       = 0pt
    , after-hspace        = \fontdimen2 \font
    , heading-decls       = \normalfont
    , headformat-instance = paragraph
    , number-format       = #1.
    , number-decls        = \upshape
    , title-format        = #1 \AddPunct{.}
  }
\DeclareInstance{headformat}{paragraph}{runin}
  {
    , separator-b    = \hspace{0.5em}
    , heading-indent = \normalparindent
  }

\DeclareInstance{heading}{subparagraph}{runin}
  {
    , name                = subparagraph
    , level               = 5
    , mark-cmd            = \subparagraphmark{#1}
    , before-vspace       = 0pt
    , after-hspace        = \fontdimen2\font
    , heading-decls       = \normalfont
    , headformat-instance = subparagraph
    , number-format       = #1.
```

```
    , number-decls        = \upshape
    , title-format        = #1 \AddPunct{.}
  }
\DeclareInstance{headformat}{subparagraph}{runin}
  {
    , separator-b = \hspace{0.5em}
  }
```

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.

```
\def\partmark#1{%
  \markboth{}{}
}
\def\chaptermark#1{%
  \markboth {\MakeUppercase{%
      \ifnum \c@secnumdepth >\m@ne
        \thechapter. \ %
      \fi
      #1}}{}
}
\def\sectionmark#1{%
      \markright {\MakeUppercase{%
        \ifnum \c@secnumdepth >\z@
          \thesection. \ %
        \fi
        #1}}
}
```

⟨/amsbook⟩

⟨∗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.

```
\DeclareInstance{heading}{part}{display}
  {
    , name                = part
    , level               = -1                  % -1 or 0 ?
    , mark-cmd            = \partmark{#1}
    , para-indent         = true
    , before-vspace       = \linespacing plus \linespacing
    , after-vspace        = 0.5 \linespacing
    , heading-decls       = \normalfont \bfseries \raggedright
    , headformat-instance = part
    , text                = \partname
    , number-format       = \theheading .
    , number-decls        = \mdseries \upshape  % proc: has just \upshape
  }
\DeclareInstance{headformat}{part}{hang}
  {
    , separator-b = \hspace{0.5em}
  }
```

```
\DeclareInstance{heading}{section}{display}
  {
    , name                = section
    , level               = 1
    , mark-cmd            = \sectionmark{#1}
    , para-indent         = true
    , before-vspace       = 0.7\linespacing plus \linespacing
    , after-vspace        = 0.5\linespacing
    , heading-decls       = \normalfont \scshape \centering
    , headformat-instance = section
    , number-format       = \theheading .
    , number-decls        = \mdseries \upshape
  }
\DeclareInstance{headformat}{section}{hang}
  {
    , separator-b = \hspace{0.5em}
  }
```

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

```
\DeclareInstanceCopy{heading}{specialsection}{section}
\EditInstance{heading}{specialsection}
   {
     , before-vspace = \linespacing plus \linespacing
     , heading-decls = \normalfont \centering
   }

\DeclareInstance{heading}{subsection}{runin}
  {
    , name                = subsection
    , level               = 2
    , mark-cmd            = \subsectionmark{#1}
    , before-vspace       = 0.5\linespacing plus 0.7\linespacing
    , after-hspace        = \fontdimen2\font
    , heading-decls       = \normalfont \bfseries
    , headformat-instance = subsection
    , number-format       = \theheading .
    , number-decls        = \mdseries \upshape
    , title-format        = #1 \AddPunct{.}
  }
\DeclareInstance{headformat}{subsection}{runin}
  {
    , separator-b = \hspace{0.5em}
  }

\DeclareInstance{heading}{subsubsection}{runin}
  {
    , name                = subsubsection
    , level               = 3
    , mark-cmd            = \subsubsectionmark{#1}
    , before-vspace       = 0.5\linespacing plus 0.7\linespacing
    , after-hspace        = \fontdimen2\font
    , heading-decls       = \normalfont \itshape
    , headformat-instance = subsubsection
    , number-format       = \theheading .
```

```
    , number-decls        = \mdseries \upshape
    , title-format        = #1 \AddPunct{.}
  }
\DeclareInstance{headformat}{subsubsection}{runin}
  {
    , separator-b = \hspace{0.5em}
  }

\DeclareInstance{heading}{paragraph}{runin}
  {
    , name                = paragraph
    , level               = 4
    , mark-cmd            = \paragraphmark{#1}
    , before-vspace       = 0pt
    , after-hspace        = \fontdimen2\font
    , heading-decls       = \normalfont
    , headformat-instance = paragraph
    , number-format       = \theheading .
    , number-decls        = \mdseries \upshape
    , title-format        = #1 \AddPunct{.}
  }
\DeclareInstance{headformat}{paragraph}{runin}
  {
    , separator-b = \hspace{0.5em}
  }

\DeclareInstance{heading}{subparagraph}{runin}
  {
    , name                = subparagraph
    , level               = 5
    , mark-cmd            = \subparagraphmark{#1}
    , before-vspace       = 0pt
    , after-hspace        = \fontdimen2\font
    , heading-decls       = \normalfont
    , headformat-instance = subparagraph
    , number-format       = \theheading .
    , number-decls        = \mdseries \upshape
    , title-format        = #1 \AddPunct{.}
  }
\DeclareInstance{headformat}{subparagraph}{runin}
  {
    , separator-b = \hspace{0.5em}
  }
```

⟨/amsart⟩

⟨∗amsproc⟩

```
\DeclareInstance{heading}{part}{display}
  {
    , name                = part
    , level               = -1                  % -1 or 0 ?
    , mark-cmd            = \partmark{#1}
    , para-indent         = true
    , before-vspace       = \linespacing plus \linespacing
```

```
    , after-vspace        = 0.5 \linespacing
    , heading-decls       = \normalfont \bfseries \raggedright
    , headformat-instance = part
    , text                = \partname
    , number-format       = \theheading .
    , number-decls        = \upshape
  }
\DeclareInstance{headformat}{part}{hang}
  {
    , separator-b = \hspace{0.5em}
  }

\DeclareInstance{heading}{section}{display}
  {
    , name                = section
    , level               = 1
    , mark-cmd            = \sectionmark{#1}
    , para-indent         = true
    , before-vspace       = 0.7\linespacing plus \linespacing
    , after-vspace        = 0.5\linespacing
    , heading-decls       = \normalfont \bfseries \centering
    , headformat-instance = section
    , number-format       = \theheading .
    , number-decls        = \upshape
  }
\DeclareInstance{headformat}{section}{hang}
  {
    , separator-b = \hspace{0.5em}
  }
```

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

```
\DeclareInstanceCopy{heading}{specialsection}{section}
\EditInstance{heading}{specialsection}
   {
     , before-vspace = \linespacing plus \linespacing
     , heading-decls = \large \scshape \centering
   }

\DeclareInstance{heading}{subsection}{runin}
  {
    , name                = subsection
    , level               = 2
    , mark-cmd            = \subsectionmark{#1}
    , before-vspace       = 0.5\linespacing plus 0.7\linespacing
    , after-hspace        = \fontdimen2\font
    , heading-decls       = \normalfont \bfseries
    , headformat-instance = subsection
    , number-format       = \theheading .
    , number-decls        = \upshape
    , title-format        = #1 \AddPunct{.}
  }
\DeclareInstance{headformat}{subsection}{runin}
  {
    , separator-b    = \hspace{0.5em}
    , heading-indent = \normalparindent
```

```
  }

\DeclareInstance{heading}{subsubsection}{runin}
  {
    , name                = subsubsection
    , level               = 3
    , mark-cmd            = \subsubsectionmark{#1}
    , before-vspace       = 0.5\linespacing plus 0.7\linespacing
    , after-hspace        = \fontdimen2\font
    , heading-decls       = \normalfont \itshape
    , headformat-instance = subsubsection
    , number-format       = \theheading .
    , number-decls        =  \upshape
    , title-format        = #1 \AddPunct{.}
  }
\DeclareInstance{headformat}{subsubsection}{runin}
  {
    , separator-b    = \hspace{0.5em}
    , heading-indent = \normalparindent
  }

\DeclareInstance{heading}{paragraph}{runin}
  {
    , name                = paragraph
    , level               = 4
    , mark-cmd            = \paragraphmark{#1}
    , before-vspace       = 0pt
    , after-hspace        = \fontdimen2\font
    , heading-decls       = \normalfont
    , headformat-instance = paragraph
    , number-format       = \theheading .
    , number-decls        = \upshape
    , title-format        = #1 \AddPunct{.}
  }
\DeclareInstance{headformat}{paragraph}{runin}
  {
    , separator-b = \hspace{0.5em}
    , heading-indent   = \normalparindent
  }

\DeclareInstance{heading}{subparagraph}{runin}
  {
    , name                = subparagraph
    , level               = 5
    , mark-cmd            = \subparagraphmark{#1}
    , before-vspace       = 0pt
    , after-hspace        = \fontdimen2\font
    , heading-decls       = \normalfont
    , headformat-instance = subparagraph
    , number-format       = \theheading .
    , number-decls        = \upshape
    , title-format = #1 \AddPunct{.}
  }
\DeclareInstance{headformat}{subparagraph}{runin}
  {
```

```
    , separator-b = \hspace{0.5em}
  }
```

⟨/amsproc⟩

⟨∗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!

```
\def\@starttoc#1#2{%
  \begingroup
  \setTrue{#1}%
  \ifx\contentsname#2%
    \chapter[unnumbered,toc=]{\contentsname}%
  \else
    \chapter[=,unnumbered]{#2}%
  \fi
  \typeout{#2}%
  \makeatletter
  \@input{\jobname.#1}%
  \if@filesw
    \@xp\newwrite\csname tf@#1\endcsname
    \immediate\@xp\openout\csname tf@#1\endcsname \jobname.#1\relax
  \fi
  \global\@nobreakfalse
  \endgroup
  \newpage
  }
\AddToHook{begindocument}{
  \@namedef{r@tocindent-1}{1em}
  \@namedef{r@tocindent0}{1em}
  \@namedef{r@tocindent1}{1em}
}
```

⟨/amsbook⟩

```
\ExplSyntaxOff
```

⟨/amscls⟩

⟨∗package⟩

```
\ExplSyntaxOff
```

2.4 fncychap

First aid for fncychap is kept in a separate file.

```
\AddToHook{package/fncychap/after}[latex-lab-firstaid]{\input{latex-lab-firstaid-fncychap.ltx}}
```

⟨/package⟩

⟨∗fncychap⟩

```
\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 offerered by fncychap. Thus we edit the default `heading chapter` instance in the following way.

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

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

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

The code for this template is fairly simple.

```
\DeclareTemplateCode{headformat}{fncychap}{6}{}
{
```

Output some debugging info at the start.

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

Everything else happens in a group to keep changes local.

```
  \group_begin:
```

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

```
    \bool_if:NTF \l__head_unnumbered_bool
      { \MakeLinkTarget [ \l__head_name_tl ] {}   }
      { \MakeLinkTarget { \l__head_name_tl }      }
```

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

```
    \parindent 0pt
    \raggedright
    \interlinepenalty 10000
    \normalfont
```

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

```
    \AssignTaggingSocketPlug{parbox/before}{heading}
    \AssignTaggingSocketPlug{minipage/before}{heading}
    \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 immmediately after).

```
    \fncychap@setup@thechapter

    \bool_if:NF \l__head_unnumbered_bool
      { \DOCH }
```

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

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

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

```
    \par
  \group_end:
}
```

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:

```
\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.

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

\def\appendix{\par
  \setcounter{chapter}{0}
  \setcounter{section}{0}
  \gdef\@chapapp{\appendixname}
  \gdef\thechapter{\@Alph\c@chapter}
```

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

```
  \gdef\fncychap@setup@thechapter{
    \def\thechapter{\thespecialchapter{\@Alph\c@chapter}}
  }
}
```

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

```
\protected\def\thespecialchapter#1{
  \leavevmode
  \tag_if_in:nTF{Lbl}
    { #1 }
    {
      \UseTaggingSocket{inline/begin}{tag=heading-number}
      #1
      \UseTaggingSocket{inline/end}
    }
}
```

`\mghrulefill`

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

```
\protected\def\mghrulefill#1{\leavevmode
  \UseTaggingSocket{rule}{}{
    \leaders\hrule\@height #1\hfill\kern\z@
  }
}
```

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

```
\ExplSyntaxOff
```

⟨/fncychap⟩
⟨∗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.

```
\AddToHook{package/cleveref/after}
 {
   \let\@makefntext\cref@old@makefntext
   \AddToHook{cmd/@makefntext/before}{%
    \cref@constructprefix{footnote}{\cref@result}%
    \protected@edef\cref@currentlabel{%
     [footnote][\arabic{footnote}][\cref@result]%
    \p@footnote\@thefnmark}}
 }
```

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

```
\ExplSyntaxOn
\AddToHook{package/booktabs/after}
 {
```

```
  \def\@cmidrulea{
   \multispan\@cmidla
   &\multispan\@cmidlb
   \unskip\hskip\cmrkern@l
  {
   \UseTaggingSocket{tbl/leaders/begin}
   \CT@arc@\leaders\hrule \@height\@thisrulewidth\hfill\kern\z@}
   \hskip\cmrkern@r
   \UseTaggingSocket{tbl/leaders/end}
   \tbl_gdecr_row_count:
   \cr}

  \def\@cmidruleb{%
    \multispan\@cmidlb
    \unskip\hskip \cmrkern@l%
   {
    \UseTaggingSocket{tbl/leaders/begin}
    \CT@arc@\leaders\hrule \@height\@thisrulewidth\hfill\kern\z@}
    \hskip\cmrkern@r
    \UseTaggingSocket{tbl/leaders/end}
    \tbl_gdecr_row_count:
    \cr}
  }
\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.

```
\ExplSyntaxOn
\AddToHook{package/fancyvrb/after}
{
    \def\FV@LeaveVMode{%
      \if@noskipsec
        \leavevmode
      \else
        \if@FV@ResetMargins\if@inlabel\leavevmode\fi\fi
      \fi
      \ifvmode
        \@noparlisttrue
        \UseTaggingSocket{para/semantic/begin}{}
      \else
        \tagpdfsetup{para/flattened=true}
        \@noparlistfalse
        \unskip\par
```

```
    \fi
    }
```

(*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

```
\def\FV@List#1{%
  \begingroup
  \FV@UseKeyValues
  \FV@LeaveVMode
  \if@inlabel\else\setbox\@labels=\box\voidb@x\fi
  \FV@ListNesting{#1}%
  \FV@ListParameterHook
  \FV@ListVSpace
  \FV@SetLineWidth
  \FV@InterLinePenalty
  \let\FV@ProcessLine\FV@ListProcessLine@i
  \FV@CatCodes
  \FV@FormattingPrep
  \FV@ObeyTabsInit
  \cs_if_exist:NT \FV@BeginListFrame
    {
     \UseTaggingSocket{mc}{artifact}{\FV@BeginListFrame}
    }
  \UseTaggingSocket{struct/begin}{tag=\UseStructureName{block/verbatim}}
  }
```

(*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.

```
\def\FV@EndList{%
  \FV@ListProcessLastLine
  \UseTaggingSocket{struct/end}
  \cs_if_exist:NT \FV@EndListFrame
    {
     \UseTaggingSocket{mc}{artifact}{\FV@EndListFrame}
    }
  \@endparenv
  \endgroup
  \@endpetrue
  }
```

(*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.

```
\def\FV@ListProcessLine#1{%
  \hbox to \hsize{%
    \kern\leftmargin
    \hbox to \linewidth{%
     \UseTaggingSocket{struct/begin}{tag=\UseStructureName{block/verbatim/codeline}}
      \cs_if_exist:NT \FV@LeftListNumber
       {
         \UseTaggingSocket{struct-mc}{tag=\UseStructureName{block/verbatim/linenumber}}
```

```
                {
                  \FV@LeftListNumber
                }
            }
           \cs_if_exist:NT \FV@LeftListFrame
            {
              \UseTaggingSocket{mc}{artifact}{\FV@LeftListFrame}
            }
           \UseTaggingSocket{mc}{}{\FancyVerbFormatLine{#1}}%
           \UseTaggingSocket{struct/end}\hss
           \cs_if_exist:NT \FV@RightListFrame
            {
              \UseTaggingSocket{mc}{artifact}{\FV@RightListFrame}
            }
           \cs_if_exist:NT \FV@RightListNumber
            {
              \UseTaggingSocket{struct-mc}{tag=\UseStructureName{block/verbatim/linenumber}}
                {
                  \FV@RightListNumber
                }
            }
           }
           \hss}}
  }
\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 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.

```
\AddToHook{package/float/after}[latex-lab-firstaid]{\input{latex-lab-firstaid-float.ltx}}
⟨/package⟩

⟨*float⟩
\ProvidesFile{latex-lab-firstaid-float.ltx}
 [\ltlabfirstaiddate\space \ltlabfirstaidversion\space LaTeX-lab-firstaid: Tagging support fo
```

Tagging sockets for the H-floats

```
\ExplSyntaxOn
\NewTaggingSocket{float/H/begin}{1}
\NewTaggingSocketPlug{float/H/begin}{default}
 {
```

```
    \mode_if_horizontal:T {\tag_mc_end_push: \tag_struct_end:}
    \tag_struct_begin:n {tag=\UseStructureName{float/#1}}
  }
\AssignTaggingSocketPlug{float/H/begin}{default}

\NewTaggingSocket{float/H/end}{0}
\NewTaggingSocketPlug{float/H/end}{default}
 {
    \tag_struct_end:
    \mode_if_horizontal:T
     {
       \tag_struct_begin:n {tag=\UseStructureName{para/textblock}}
       \tag_mc_begin_pop:n {}
     }
 }
\AssignTaggingSocketPlug{float/H/end}{default}
```

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

```
\def\@float@HH#1[H]{\UseTaggingSocket{float/H/begin}{#1}%
  \expandafter\let\csname end#1\endcsname\float@endH
  \let\@currbox\float@box
  \def\@captype{#1}\setbox\@floatcapt=\vbox{}%
  \expandafter\ifx\csname fst@#1\endcsname\relax
  \@flstylefalse\else\@flstyletrue\fi
  \setbox\@currbox\color@vbox\normalcolor
    \vbox\bgroup \hsize\columnwidth \@parboxrestore
     \@floatboxreset \@setnobreak
  \ignorespaces}
\renewcommand\float@endH{\@endfloatbox\UseTaggingSocket{float/H/end}
  \vskip\intextsep
  \if@flstyle\setbox\@currbox\float@makebox\columnwidth\fi
  \box\@currbox\vskip\intextsep\relax}
\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:

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

The space is intentional!

```
  \UseTaggingSocket{para/begin} %
     #2\par
  \else
   \hbox to \hsize {\hfil
     \UseTaggingSocket{caption/label/begin}%
       {\@fs@cfont #1:}%
     \UseTaggingSocket{caption/label/end}%
```

The space is intentional!

```
      \UseTaggingSocket{para/begin} %
    #2%
    \UseTaggingSocket{para/end}\hfil}%
    \par
  \fi
```

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

```
  }
```

This adds tagging support to the ruled style:

```
\renewcommand\floatc@ruled[2]{%
  \UseTaggingSocket{caption/begin}{\@current@float@struct}%
  \UseTaggingSocket{caption/label/begin}%
   {\@fs@cfont #1}%
  \UseTaggingSocket{caption/label/end}%
```

The space is intentional!

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

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

```
 }
```

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.

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

```
\renewcommand\newfloat[3]{\@namedef{ext@#1}{#3}
  \tagpdfsetup{float/new=#1,role/new-tag=#1-floats/Sect}%
  \AssignStructureRole{float/#1s}{#1-floats}%
  \let\float@do=\relax
  \xdef\@tempa{\noexpand\float@exts{\the\float@exts \float@do{#3}}}%
  \@tempa
  \floatplacement{#1}{#2}%
  \@ifundefined{fname@#1}{\floatname{#1}{#1}}{}
  \expandafter\edef\csname ftype@#1\endcsname{\value{float@type}}%
  \addtocounter{float@type}{\value{float@type}}
  \restylefloat{#1}%
  \expandafter\edef\csname fnum@#1\endcsname%
    {\expandafter\noexpand\csname fname@#1\endcsname{}
       \expandafter\noexpand\csname the#1\endcsname}
  \@ifnextchar[{\float@newx{#1}}%]
    {\@ifundefined{c@#1}{\newcounter{#1}\@namedef{the#1}{\arabic{#1}}}%
      {}}}
```

⟨/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 readd the argument.

```
⟨*package⟩
\AddToHook{package/apacite/after}
  {\if@APAC@natbib@apa\def\@lbibitem[#1]#2{\NAT@swatrue\orig@nat@@lbibitem[#1]{#2}}\fi}
⟨/package⟩
```