Reimplementation of LaTeX 2ε's heading commands using templates

LaTeX Project*

v0.9y 2026-10-27

Abstract

Contents

1 Introduction **3**

2 Changes to the user interface **3**
2.1 New user commands . . . 3
2.2 The optional argument . . . 3
2.3 Class support . . . 4
2.4 Changing heading commands . . . 5
2.5 Converting heading commands defined with `\@startsection` . . . 5
2.5.1 Step 1: Getting the template name and the instance values . . . 5
2.5.2 Step 2: Get the default template values . . . 7
2.5.3 Step 3: Declare the instances . . . 8
2.5.4 Step 4 Redefine the `\section` command . . . 8

3 Setting up LaTeX 2ε heading commands in classes **9**

4 Open points **9**

5 Template types and templates for headings **9**
5.1 Template types . . . 9
5.1.1 The template type 'heading' . . . 10
5.1.2 The template type 'headformat' . . . 10
5.2 Templates . . . 11
5.2.1 Templates of type `heading` . . . 11
5.2.2 The `heading` template '⟨*all*⟩' . . . 11
5.2.3 The `heading` template 'display' . . . 13
5.2.4 The `heading` template 'runin' . . . 15
5.2.5 Templates of type `headformat` . . . 16
5.2.6 The `headformat` template 'display' . . . 16
5.2.7 The `headformat` template 'hang' . . . 17
5.2.8 The `headformat` template 'runin' . . . 18
5.3 Default instances . . . 19

*Initial implementation by Frank Mittelbach.

5.3.1 Instances of `heading` templates . . . 19
5.3.2 Instances of `headformat` templates . . . 20
5.4 Use of templates from the 2026-06-01 release of LaTeX . . . 20

6 Support for legacy classes and packages 21
6.1 Support classes based on legacy LaTeX 2ε interfaces . . . 21

7 Support for links 21

8 Bookmarks 22

9 Tagging support 22

10 Debugging support 22

11 The Implementation 22
11.1 Debugging . . . 23
11.2 Temp variable(s) . . . 24
11.3 variable(s) . . . 24
11.4 New user commands . . . 25
11.5 The LaTeX 2ε parsing interface . . . 25
11.6 General helper commands . . . 30
11.7 Templates . . . 32
11.7.1 Template types . . . 32
11.7.2 heading templates interfaces . . . 32
11.7.3 headformat templates interfaces . . . 35
11.7.4 heading templates code . . . 36
11.7.5 headformat templates code . . . 46
11.7.6 Internal commands used by the template code . . . 51
11.8 Support for legacy classes and packages . . . 54
11.8.1 Instances (sample/default definitions) . . . 55
11.8.2 New `\@startsection` . . . 58
11.8.3 New `\secdef` . . . 61
11.8.4 Core LaTeX . . . 62

12 Support for the titlesec interfaces 64

13 Issues and problems noticed along the way 64
13.1 Local `\DeclareInstance` . . . 64
13.2 O-expansion of `\UseInstance` arguments . . . 64
13.3 Switch `\openright` is not defined by core . . . 65
13.4 Indexheading at least for l3doc is wrong now . . . 65

A Templates from the 2026-06-01 release of LaTeX **65**
A.1 Template documentation for `-v1` templates 65
A.1.1 Templates of type `heading` 65
A.1.2 The `heading` template '⟨*all*⟩' 65
A.1.3 The `heading` template 'display-v1' 66
A.1.4 The `heading` template 'runin-v1' 67
A.1.5 Templates of type `headformat` 68
A.1.6 The `headformat` template 'hang-v1' 68
A.1.7 The `headformat` template 'runin-v1' 69
A.1.8 The `headformat` template 'display-v1' 69
A.2 Implementation of `-v1` templates 70

Index **79**

1 Introduction

This module reimplements tagging aware heading commands with templates. The new implementation is used automatically if `\DocumentMetadata` is used and replaces patches and redefinitions done in the `latex-lab-sec`. In a later step both modules will be merged.

Most of the documentation is of interest only for class and package authors who want to setup their own heading commands but the new implementation changes also the user interface. This is documented first.

2 Changes to the user interface

2.1 New user commands

`\theheading` This command expands inside a heading to the current number.

`\partmark` This replaces the fix `\markboth{}{}` or similar in the standard `\part` definition.

2.2 The optional argument

The standard heading commands (re)defined by this module use the standard 2e syntax, e.g.,

```
\section[toc]{title}
```

The star-form `\section*{Title}` stops the numbering, toc, bookmark and running header as it was the way for the last 30-odd years. New is that the optional argument is handled as a key/val list if an equal sign at the outer level is detected, otherwise the argument is taken to be the value of the key `shorttitle` and passed to the table of contents, the running headers, the bookmarks and used with `\nameref`. Note that this means, that

```
\section*[Title]{Title}
```

will create a toc entry and a bookmark!

The following keys can be used

`toc` This sets the text for the table of contents. It also forces that the entry is created. So

```
\section*[toc=Introduction]{Introduction to this document}
```

will create an unnumbered entry "Introduction" in the table of contents. An empty value (i.e., `toc=`) will suppress the toc entry.

`running` This sets the text for the running header and forces the use of the mark command, so even with a starred section the running header will be updated. An empty value will suppress the call of the mark commands, so a header from a previous section will prevail.

`bookmark` This sets the text for the bookmarks (the PDF outline) if, e.g., hyperref is loaded. As with the previous keys an empty value suppresses the bookmark entry.

`nameref` This allows to set the text used for a cross reference with `\nameref`.

`label` This sets a label, so it is an alternative to using a `\label` command.

`subtitle, quote` These keys will allow to pass a subtitle and a quote to the underlying code, but currently they are not yet used by the implementations of the standard LaTeX heading commands.

`shorttitle` This is the key which is used if the optional argument is not of key/val form. So,

```
\section[short]{long title}
```

is the same as

```
\section[shortitle=short]{long title}
```

The key sets the toc, running, bookmark and nameref text. If `shorttitle` is used together with any of the individual keys (`toc`, `running`, `bookmark`, `nameref`) then they overwrite the default value provided by `shorttitle`.

`numbered, unnumbered` This allow to control the numbering without stopping toc, running head or bookmarks as it would happen with the starred version of the heading command.

template keys Any other key present is assumed to be a key that should be passed to the heading instance to overwrite some of its settings. So, e.g.,

```
\section[placement=top,number-format=\fbox{\theheading}]{Title}
```

will force the section to start a new page and put the number into an `\fbox`. For a list of supported keys, check the following parts of the documentation.

2.3 Class support

The module redefines all heading commands of the three standard classes, article, report and book, to use the new implementation.

Heading commands of other classes that are defined with `\@startsection` are supported (with some restrictions) through a compatibility layer described in the next section.

The heading commands `\part` and `\chapter` are typically defined with `\secdef` and the internal commands `\@part` and `\@chapter`. The module redefines `\secdef` to use the standard instances for these commands—this adds tagging support but *changes the layout* until the class provides its own instances.

Heading commands in other classes defined with `\secdef` will likely error as the code has no real chance to know how to handle such a heading.

Heading commands which use neither `\@startsection` nor `\secdef` (e.g, the `\chapter` command of memoir, or the heading commands of KOMA classes) are not changed by this module and so will not be tagged correctly unless they add explicit tagging support.

For the AMS-classes, `latex-lab-firstaid` contains instances for `\part`, `\chapter`, `\section`, and all other headings used by these classes. The current set of instances is now assumed to reproduce the original layout. Setting them up revealed a number of interesting differences between these classes—not all of them intentional (I think).

2.4 Changing heading commands

It is possible to change heading commands by editing the instance they use. E.g., in the standard classes one could frame every heading number with

```
\EditInstance{heading}{section}
  {number-format=\fbox{\theheading}}
```

The name of the instance (here `section`) is the same as the heading command.

To support other classes which still setup their heading commands with `\@startsection` the code has a legacy compatibility layer: the `\@startsection` command has been redefined to setup instances on-the-fly at the first use of a heading command. These instances have names using the the first argument of `\@startsection` with an attached `-@startsection`. As they are created on-the-fly these internal instances can be only edited *after* the first use of the heading command or by declaring (instead of editing) an instance with this name in the preamble. Also as the names of the instances are built from the first argument of `\@startsection` it is not possible to define two different heading commands of the same level with `\@startsection` as that would require two instances with different names. The next section describes how to lift this restriction by converting such sectioning commands to use the new interfaces.

2.5 Converting heading commands defined with `\@startsection`

To convert a heading command defined with `\@startsection` to the new interface, suitable instances must be declared. The same needs to happen if one wants to alter the layout of a heading that was defined with `\@startsection` in the document preamble. How this can be done is demonstrated here with the `\section` command of the ltugboat class.

2.5.1 Step 1: Getting the template name and the instance values

The sectioning commands use two template *types*, `heading` and `headformat`. Compile the following document

```
\DocumentMetadata{}
\documentclass{ltugboat}

\begin{document}

\section{test} % <--- needed so that the instances get defined
\ShowInstanceValues{heading}{section-@startsection}
\ShowInstanceValues{headformat}{section-@startsection}

\end{document}
```

This will show the instance values in the log-file:

```
The instance 'section-@startsection' of type 'heading' has values:
>  level  =>  1
>  force-unnumbered  =>  false
>  placement  =>  normal
>  column-spanning  =>  false
>  mark-cmd  =>  \sectionmark {##1}
>  para-indent  =>  false
>  before-vspace  =>  8.0pt plus 2.0pt minus 2.0pt
>  penalty  =>  \c_max_int
>  after-penalty-vspace  =>  \maxdimen
>  after-vspace  =>  4.0pt
>  placement-start-code  =>
>  placement-end-code  =>
>  text  =>  \NoValue
>  punct  =>
>  heading-decls  =>  \tubsecfmt
>  text-decls  =>
>  number-decls  =>
>  title-decls  =>
>  subtitle-decls  =>
>  quote-decls  =>
>  heading-format  =>  ##1
>  text-format  =>  ##1
>  number-format  =>  ##1
>  punct-format  =>  ##1
>  title-format  =>  \use:n {##1} \par
>  subtitle-format  =>  ##1
>  quote-format  =>  ##1
>  number-tag  =>  heading-number
>  headformat-instance  =>  section-@startsection
>  contents-extra  =>
>  name  =>  section
>  from template  =>  display.
```

```
The instance 'section-@startsection' of type 'headformat' has values:
>  heading-indent  =>  0pt
>  left-margin  =>  0pt
>  right-margin  =>  0pt
>  order-hang  =>  text,separator-a,?,number,punct,separator-b,?
>  order  =>  title,separator-c,subtitle
>  separator-a  =>  \nobreakspace
>  separator-b  =>  \hspace {1em}
>  separator-c  =>  \
>  separator-d  =>
>  from template  =>  hang.
```

2.5.2 Step 2: Get the default template values

The last line in both lists show after the `from template` the names of the templates of each type. That can be used to get the default values of these templates. Compile the following document:

```
\DocumentMetadata{}
\documentclass{ltugboat}

\begin{document}
\ShowTemplateDefaults{heading}{display}
\ShowTemplateDefaults{headformat}{hang}
\end{document}
```

This gives in the log and terminal:

```
The template 'display' of type 'heading' has default values:
>  level  =>  0
>  force-unnumbered  =>  false
>  placement  =>  normal
>  column-spanning  =>  false
>  mark-cmd  =>
>  para-indent  =>  false
>  before-vspace  =>  0pt
>  penalty  =>  \c_max_int
>  after-penalty-vspace  =>  \maxdimen
>  after-vspace  =>  0pt
>  placement-start-code  =>
>  placement-end-code  =>
>  text  =>  \NoValue
>  punct  =>
>  heading-decls  =>  \normalfont
>  text-decls  =>
>  number-decls  =>
>  title-decls  =>
>  subtitle-decls  =>
>  quote-decls  =>
>  heading-format  =>  ##1
```

```
>  text-format  =>  ##1
>  number-format  =>  ##1
>  punct-format  =>  ##1
>  title-format  =>  ##1\par
>  subtitle-format  =>  ##1
>  quote-format  =>  ##1
>  number-tag  =>  heading-number
>  headformat-instance  =>  std
>  contents-extra  =>
>  from template  =>  display.

The template 'hang' of type 'headformat' has default values:
>  heading-indent  =>  0pt
>  left-margin  =>  0pt
>  right-margin  =>  0pt
>  order-hang  =>  text,separator-a,?,number,punct,separator-b,?
>  order  =>  title,separator-c,subtitle
>  separator-a  =>  \nobreakspace
>  separator-b  =>  \hspace {1em}
>  separator-c  =>  \
>  separator-d  =>
>  from template  =>  hang.
```

2.5.3 Step 3: Declare the instances

By comparing the instance values with the default values one can see which values should be changed in the instance. The names of the new instances can be freely chosen; best practice is to use the name of the heading command (without a backslash).

This leads then to these declarations:

```
\DeclareInstance{heading}{section}{display}
 {
   level               = 1,
   mark-cmd            = \sectionmark{#1},  % remove second hash
   before-vspace       = 8.0pt plus 2.0pt minus 2.0pt,
   after-vspace        = 4.0pt,
   heading-decls       = \tubsecfmt,
   headformat-instance = section,           % new name
 }
```

The value for key `title-format` seems to have changed as well but `\use:n {##1} \par` is really only a fancy way to write `##1 \par` so we can keep the default. Note, however, that the `##1` have to replaced by `#1` when you reenter them.

In case of the `headformat` instance only default values have been used so all that is necessary is:

```
\DeclareInstance{headformat}{section}{hang}{}
```

Of course, if you do not want to rely on default values you can (for clarity) specify all keys and their values in these instances.

2.5.4 Step 4 Redefine the `\section` command

Finally, the heading command should be defined to use the new interface:

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

What `\ParseLaTeXeHeading` does is explained in the next section.

3 Setting up LaTeX 2ε heading commands in classes

Heading commands are managed by defining an instance of a `heading` template which is then used in `\ParseLaTeXeHeading`, e.g.,

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

The name of the `heading` instance is given in the first argument of `\ParseLaTeXeHeading` and it is recommended that classes use the same name as the heading command, to make it easier for users to identify the instance to edit if they want to adjust the layout of the heading. Examples of instances that mimic the behavior of the heading commands in the standard classes can be found below. Section 2.5 shows how to get instances from commands previously defined with `\@startsection`.

Legacy setup using `\@startsection` remains supported albeit not that performant and with some restrictions for the users, see section 2.4 above and in the documentation below.

In contrast, `\secdef` is only rudimentarily supported. It delegates the layout to two commands which can contain arbitrary code (usually hardwired) so that there is no realistic chance to automatically take this apart and figure out what values should be used for what template parameters. We therefore redefine `\secdef` and map the standard commands `\part` and `\chapter` to the standard instance and error if other uses are detected. Classes using `\secdef` should setup suitable instances that mimic their current layout, an example can be found for the ams-classes in `latex-lab-firstaid.dtx`.

4 Open points

- There is no good interface yet to change e.g. the font family of all heading commands in one go.
- Support for titlesec, see section 12.

5 Template types and templates for headings

The template(s) for headings expect(s) a large number of positional arguments containing document user data. The document-level command, e.g., `\section`, may not offer all of them directly, but may do so via a key value interface or not at all. If no interface is provided then the template is passed `\NoNalue`. If it is offered via a key value interface, the template receives whatever is set by the user (and `\NoNalue` otherwise).

In my initial implementation I had only the main data as positional arguments, but I came to the conclusion that this scheme here is better going forward.

5.1 Template types

The template type `heading` has 10 data arguments, which is more than can be specified as positional arguments. For that reason argument #9 holds 2 brace groups. The alternative would be to specify such arguments as keys, but for a number of reasons I think the approach to put seldom used data arguments all in the last positional argument is actually better (besides being faster).

5.1.1 The template type 'heading'

Arg: 1 key/value list to alter the default heading parameters

Arg: 2 unnumbered heading?

Arg: 3 main title of the heading

Arg: 4 toc title

Arg: 5 running title

Arg: 6 bookmark title

Arg: 7 nameref text

Arg: 8 label for the heading in the form `\label{`⟨*string*⟩`}`

Arg: 9 { sub title } { quotation }

Semantics:

Handles the layout and processing aspects of a heading. This includes doing all the work necessary for tagging.

Whether or not the heading is numbered is governed through a boolean, expecting the result of an `s` specification of `\NewDocumentCommand` or equivalent, i.e., expects `\BooleanTrue` or `\BooleanFalse`.

If the title data is also used for bookmarks, toc, running header, and cross references then it has to be given several times which can be arranged for by the parsing interface command that calls the template instance.

If the bookmark, toc, or running argument is set to "empty" then the bookmark, toc or running header should be suppressed by the template. This enables the user on document level to explicitly specify, for example, `[bookmark=]` to suppress the bookmark.

The `nameref` argument is not expected to be empty. Its value should always be used as given if named references are to be generated.

The label argument either holds a `\label` command as provided by the user (or several, see implementation of `\ParseLaTeXeHeading`) or it is empty. I.e., it can be directly executed by a template at the right point where the reference counter (if any) has been set up without the need to check its content.

Argument #9 holds further user data (in brace groups) which are seldom implemented, but if they are the data is available in a positional argument, which then needs to be taken apart using `\@firstoftwo` (for subtitle) or `\@secondoftwo` (for a quotation). If no data is provided then `\NoValue` is used to indicate that.

5.1.2 The template type 'headformat'

Arg: 1 key/value list to alter the default headformat parameters

Arg: 2 fixed text

Arg: 3 formatted heading number

Arg: 4 main title of the heading

Arg: 5 subtitle

Arg: 6 quotation

Semantics:

Handles the layout of just the heading label (text and number), main title, subtitle, and quotation but not the spatial relation to previous and following text.

Whether or not the heading is numbered is governed through a boolean, e.g., result of an `s` specification in `\NewDocumentCommand` or equivalent.

If there is no text, number, subtitle or quotation then this is indicated with `\NoValue`.

The templates are currently implemented to mimic LaTeX 2_ε behavior so if a `heading` template should not number a heading it calls a `headformat` template with both text and number set to `\NoValue`.

The ⟨*subtitle*⟩ and ⟨*quotation*⟩ are always ignored by the currently defined templates but that will probably change soon, either by extending them or by providing additional ones that support these mandatory argument.

The code of templates of this kind should execute `\UseSocket{heading/record}` so that the position of the heading is recorded. Only then special instances can be used to define layouts based on recto/verso pages.

5.2 Templates

5.2.1 Templates of type `heading`

There are quite a number of keys that are expected to be recognized by all heading templates (though they may choose not to make use of them). These are listed below instead of being repeated on the actual templates.

All other keys are either attached to the `heading` or to the `headformat` templates. Those that typically vary from heading instance to the next (e.g., `heading-decls`) are all declared in the `heading` templates even if they are actually only used within `headformat` templates. In other words, `headformat` templates have a number of implicit variables that they expect to be set.[1]

[1] Maybe questionable

5.2.2 The `heading` template '⟨*all*⟩'

Attributes:

`name` (*tokenlist*) Referenceable name of the heading instance. String that is acceptable in csnames for use in building counter names, etc.

`parent-name` (*tokenlist*) Name of the next higher heading instance. If not given, then the internal heading level of the heading instance is set to `0` — not yet implemented/may vanish

`reset-counter` (*tokenlist*) Name of the heading instance that should reset the numbering of this heading level (if any) — not yet implemented/may vanish

`level` (*integer*) Sets the internal heading-level rather than deducing it from `parent-name`. Can be used to specify the top-level heading if not `0`, or all headings in legacy implementations, e.g., through `\@startsection`

`force-unnumbered` (*boolean*) This heading is never numbered, even if explicitly requested in the optional argument to the heading. If `false` then numbering is determined in the normal way (through `secnumdepth`, `*`, or a setting in the optional argument to the heading).
Default: `false`

`placement` (*choice*) Set the heading placement, i.e., the behavior of the heading with respect to page breaks. Allowed values are `page` (heading forms a page if its own), `top` (heading starts a new page), `normal` (heading can appear anywhere on the page), `ragged` (like `normal` but if a page break is taken before the heading then the previous page is set ragged and not stretched). Further possibilities might be `rectopage` and `rectotop` if we implement that.
This key sets the values for the `placement-start-code` and `placement-end-code` keys. However, if `placement-start-code` or `placement-end-code` has been set to non-empty value it will overwrite whatever the `placement` key would put in. Thu providing unsuitable values for one or both of them may produce strange effects.
If a special placement is wanted that is not covered by the choice values for the `placement` key it is normally best to set both `placement-start-code` and `placement-end-code` to a non-empty value, so that the `placement` key is fully ignored. Default: `normal`

`placement-start-code` (*tokenlist*) The value for this key is normally set by the `placement` key. Executed before the heading starts, so can issue, for example, a `\clearpage`. However, it is *not* a user key that can be arbitrarily set! It is only provided as to allow extending the functionality of the `placement` key and if used requires that everything necessary for placement is then handled by the code.
Default: ⟨*empty*⟩

`placement-end-code` (*tokenlist*) The value for this key is normally set by the `placement` key. Executed after the heading is typeset. Can be used to set up code for putting the heading on a page by its own, or arrange for paragraph handling of a following paragraph, etc. This is *not* a user key to place arbitrary code after the heading because this will then replace what was put there by the `placement` key to handle paragraph indentation, etc. after the heading. It is only provided as a means to extend the functionality of the `placement` key and if used requires that everything necessary is handled by the code. Default: ⟨*empty*⟩

`column-spanning` (*boolean*) If true produces a heading that spans columns in `twocolumn` typesetting. Requires `placement = top` (and adjusts if necessary) and only has an effect if the columns are produced via the `twocolumn` class option and not when using `multicol`. This might get extended when we refactor the OR handling.
Default: `false`

`text` (*tokenlist*) A fixed string, such as "Chapter", that can be used together with the number (if any) to form a "heading label". Usage and placement is up to the template, so it could be placed after the number by the template (in which case it isn't a prefix, hence the more neutral name). Default: `\NoValue`

`punct` (*tokenlist*) A fixed string to be added to the end of the heading number, i.e., it shows up in on the heading but not in cross-references. Default: ⟨*empty*⟩

`mark-cmd` (*function(1)*) Function that receives the ⟨*running*⟩ argument and creates a suitable mark insertion Default: \⟨*name*⟩`mark`

Semantics & Comments: The above keys should be implemented by all heading templates.

At the moment I have retained the LaTeX 2_ε interfaces for marks, e.g., one has to set up `\chaptermark`, `\sectionmark`, etc. but I'm not sure this should stay (even though it is certainly simpler from a compatibility perspective).

All templates should set up `\theheading` to correspond to `\the`⟨*name*⟩.

The keys `placement-start-code` and `placement-end-code` have formerly been called `\start-code` and `\final-code` but as they are directly tied to the `placement` key and can't be arbitrarily set, they got renamed. The old names will eventually be removed!

5.2.3 The `heading` template 'display'

Attributes:

`para-indent` (*boolean*) Should the paragraph after the heading be indented?
Default: `false`

`before-vspace` (*skip*) Vertical space before the heading if there is no page or column break. If there is one it vanishes. In particular this means it will not be used in the heading `placement`s `page` or `top` Default: `0pt`

`penalty` (*integer*) Penalty to break before the heading. The default (TeX's largest integer) indicates that no penalty was set in which case `\@secpenalty` is used (to support the LaTeX 2_ε interface). Default: `1073741823`

`after-penalty-vspace` (*skip*) A special vertical space before the heading but after the penalty for the heading. If the penalty results in a page or column break, this space remains at the top of the page. The sentinel default value `\maxdimen` signals that no space at all should be added and the heading will appear at the very top of the page after a break, i.e., the top of the characters touch the top of the page.
If the key is explicitly set to any other value it will be used. In that case the baseline of the heading will be one `\baselineskip` plus the `after-penalty-vspace` away from the top. Thus setting `after-penalty-vspace=0pt` means that the baseline of

the first line of the heading is one `\baselineskip` down from the top of the page, i.e., noticeably lower than when no space is inserted.
Setting it to `0pt` or some other value ensures that the placement of the heading at the top of the page is always in a predictable position and not (as was the case with LaTeX 2_ε) depending on the height of characters in the first line of the heading.
When reimplementing a page heading like `\chapter` using templates it is important to note that a `\vspace*{`⟨*value*⟩`}` means that you have to put ⟨*value*⟩`+\topskip` into the `after-penalty-vspace` key to achieve the same spacing.
Default: `\maxdimen`

`after-vspace` (*skip*) Vertical space after the heading Default: `0pt`

`heading-decls` (*tokenlist*) Declarations (such as font or color settings) applied to all heading elements, i.e., number, title, subtitle, and quotation, if present. Note that color commands (unless luacolor is used) introduce a break point before the heading and therefore one should either surround color commands with `\SaveLastSkip` and `\RestoreLastSkip` or to use the keys `text-decls`, `number-decls`, `title-decls`, etc. instead. Default: `\normalfont`

`text-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the fixed heading text, overwriting the setting of `heading-decls`. Default: ⟨*empty*⟩

`number-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the heading number, overwriting the setting of `heading-decls`. Default: ⟨*empty*⟩

`title-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the heading title, overwriting the setting of `heading-decls`.
Default: ⟨*empty*⟩

`subtitle-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the heading subtitle, overwriting the setting of `heading-decls`. Default: ⟨*empty*⟩

`quote-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the heading quote, overwriting the setting of `heading-decls`.
Default: ⟨*empty*⟩

`heading-format` (*function(1)*) Code that receives the whole heading material (or some parts of it) as an argument. What is passed on in `#1` and precisely how depends on the `headformat` template that is called so the code in this key and the used `headformat` instance need to match. This allows in a limited way to do processing at the start and at the end of the heading material.
Elaborate and complex code in this key may break taging so some care (and checking) is necessary.
Default: `#1`

`text-format` (*function(1)*) Code that receives the text string as an argument.
Default: `#1`

`number-format` (*function(1)*) Code that receives the number string as an argument.
Default: `#1`

`punct-format` (*function(1)*) Code that receives the punctuation string as an argument.
Default: `#1`

`title-format` (*function(1)*) Code that receives the title string as an argument. By default it ends in `\par` so that `\baselineskip` changes in the settings (e.g., in `title-decls`) are applied. Default: `#1 \par`

`subtitle-format` (*function(1)*) Code that receives the subtitle string as an argument. Default: `#1`

`quote-format` (*function(1)*) Code that receives the quote string as an argument. Default: `#1`

`number-tag` (*tokenlist*) Name of the tag to put around the number (empty means no tag). Default: `heading-number`

`headformat-instance` (*instance*) Template instances of type `headformat` Default: `std`

`contents-extra` (*tokenlist*) Code containing `\addcontents` calls to write to files like `.lot` or `.lot` Default: ⟨*empty*⟩

Semantics & Comments: The key `heading-decls` determines the overall `\baselineskip` used in the heading. Font changes in individual declarations do not have that effect, because they are all executed in groups and if if all elements are within the same paragraph then the final `\par` after the heading comes too late. If you want that the font setting for a specific key, e.g., `title`, depends on the font setting in `title-decls` you have to ensure that a `\par` happens within `title-format` key, e.g., `title-format=#1\par` (which is what happens in the default value of `title-format`).

5.2.4 The `heading` template 'runin'

Attributes:

`before-vspace` (*skip*) Vertical space before the heading if there is no page or column break. If there is one it vanishes. Default: `0pt`

`penalty` (*integer*) Penalty to break before the heading. The default (TEX's largest integer) indicates that no penalty was set in which case `\@secpenalty` is used. Default: `1073741823`

`after-penalty-vspace` (*skip*) Special vertical space before the heading but after the penalty for the heading. If the penalty results in a page or column break then this space is added at the top of the next page if it differs from it sentinel default value (`\maxdimen`). See the more exhaustive explanation above in the display template documentation. Default: `\maxdimen`

`after-hspace` (*skip*) Horizontal space after the heading. Default: `0pt`

`heading-decls` (*tokenlist*) Declarations (such as font or color settings) applied to all heading elements, i.e., number, title, subtitle, and quotation, if present. Note that color commands (unless luacolor is used) introduce a break point before the heading and therefore one should either surround color commands with `\SaveLastSkip` and `\RestoreLastSkip` or to use the keys `text-decls`, `number-decls`, `title-decls`, etc. instead. Default: `\normalfont`

`text-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the fixed heading text, overwriting the setting of `heading-decls`. Default: ⟨*empty*⟩

`number-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the heading number, overwriting the setting of `heading-decls`. Default: ⟨*empty*⟩

`title-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the heading title, overwriting the setting of `heading-decls`.
Default: ⟨*empty*⟩

`subtitle-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the heading subtitle, overwriting the setting of `heading-decls`. Default: ⟨*empty*⟩

`quote-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the heading quote, overwriting the setting of `heading-decls`.
Default: ⟨*empty*⟩

`heading-format` (*function(1)*) Code that receives the whole heading material (or some parts of it) as an argument. See comments in the documentation of the `display` template. Default: `#1`

`text-format` (*function(1)*) Code that receives the text string as an argument.
Default: `#1`

`number-format` (*function(1)*) Code that receives the number string as an argument.
Default: `#1`

`punct-format` (*function(1)*) Code that receives the punctuation string as an argument.
Default: `#1`

`title-format` (*function(1)*) Code that receives the title string as an argument. Since this is a run-in heading we should remain in horizontal mode so the default does not include a `\par`. Default: `#1`

`subtitle-format` (*function(1)*) Code that receives the subtitle string as an argument.
Default: `#1`

`quote-format` (*function(1)*) Code that receives the quote string as an argument.
Default: `#1`

`number-tag` (*tokenlist*) Name of the tag to put around the number (empty means no tag). Default: `heading-number`

`headformat-instance` (*instance*) Template instances of type `headformat` Default: `std`

`contents-extra` (*tokenlist*) Code containing `\addcontents` calls to write to files like `.lot` or `.lot` Default: ⟨*empty*⟩

Semantics & Comments: The `runin` template is fairly similar to the `display` one. Main differences are due to the fact that we stay in horizontal mode at the end of the heading so that we have `after-hspace` instead of `after-vspace` and we do not have a `\par` inside `title-format`.

5.2.5 Templates of type `headformat`

The `headformat` templates deal with positioning text, number, punct, title, subtitle and quote. If the heading is unnumbered then text, number and punct are expected to be `\NoValue` (this is arranged in the calling `heading` template).

Tagging is taken care of in the calling template as well, all the the `headformat` templates do is opening and closing MCs as necessary.

The templates make use of the generic order key mechanism.

5.2.6 The `headformat` template 'display'

Attributes:

`heading-indent` (*dimen*) Horizontal space before the first line of the heading (shifting it to the right in a LR typesetting context). To shift all lines use `left-margin` instead or in addition. Default: `0pt`

`left-margin` (*length*) Width of the left margin, i.e., by which all lines of the heading are indented. A negative value allows the heading to protrude into the left margin of the page. Default: `0pt`

`right-margin` (*length*) Width of the right margin, i.e., by space left free on the right. A negative value allows the heading to protrude into the right margin of the page. Default: `0pt`

`order` (*commalist*) Order of elements that form the heading. Some of the separators are dropped if the previous element is `\NoValue`, e.g., `separator-a` and `separator-b`. The keyquote key is currently ignored. Default: `text, separator-a, ?, number, punct, separator-b, ?, title, separator-c, subtitle`

`separator-a` (*tokenlist*) By default the separation between `text` and `number`. Default: `\nobreakspace`

`separator-b` (*tokenlist*) By default the separation between `punct` and `title`. The default starts a new line with extra vertical space. Default: `\par \nobreak \vspace{20pt}`

`separator-c` (*tokenlist*) By default the separation between `title` and `subtitle`. Default: `\`

`separator-d` (*tokenlist*) By default not used. Default: ⟨*empty*⟩

Semantics & Comments: Implements a heading layout where number and title are vertically separated. It is what LaTeX always used by default for `\chapter` and `\part`.

This template can be called from the `heading display` template. It is not suitable for use with the `runin` template.

5.2.7 The `headformat` template 'hang'

Attributes:

`heading-indent` (*dimen*) Horizontal space before the first line of the heading (shifting it to the right in a LR typesetting context). To shift all lines use `left-margin` instead or in addition. Default: `0pt`

`left-margin` (*length*) Width of the left margin, i.e., by which all lines of the heading are indented. A negative value allows the heading to protrude into the left margin of the page. Default: `0pt`

`right-margin` (*length*) Width of the right margin, i.e., by space left free on the right. A negative value allows the heading to protrude into the right margin of the page. Default: `0pt`

`order-hang` (*commalist*) Order of elements of the heading that are used to form the hanging indentation (and are placed in that space). Default: `text, separator-a, ?, number, punct, separator-b, ?`

`order` (*commalist*) Order of remaining headings elements. The keyquote key is currently not used. Default: `title, separator-c, subtitle`

`separator-a` (*tokenlist*) By default the separation between `text` and `number`. Default: `\nobreakspace`

`separator-b` (*tokenlist*) By default the horizontal separation after `punct` still being counted in the hanging indentation Default: `\hspace{1em}`

`separator-c` (*tokenlist*) By default the separation between `title` and `subtitle`. Default: `\`

`separator-d` (*tokenlist*) By default not used. Default: ⟨*empty*⟩

Semantics & Comments: Implements a heading layout where number and title start on the same line and the title is hanging off from that if the title has more than one line. It is what LaTeX always used by default for `\section`, `\subsection`, and `\subsubsection`.

This template can be called from the `heading display` template. It is not suitable for use with the `heading runin` template.

It uses a special parshape to implement the hanging indentation in combination with the margin settings from `left-margin` and `right-margin`.

The code in `heading-format` receives the heading material including the parshape setting, which means that there are some restrictions on what works and what doesn't in this key. For example, `heading-format=\fbox{#1}` simply cancels the hanging indentation and the heading number appears in the margin, since we are now putting everything in LR-mode. On the other hand,

```
heading-format=\colorbox{yellow}{\parbox{\linewidth}{#1}}
```

will produce a yellow background behind the whole title and the hanging indentation will be preserved. Bottom line it needs some experimentation what works and where one would need a different template with more elaborate setting possibilities.

5.2.8 The `headformat` template 'runin'

Attributes:

`heading-indent` (*dimen*) Horizontal space before the heading (shifting it to the right in a LR typesetting context) Default: `0pt`

`order` (*commalist*) Order of elements that form the heading. Some of the separators are dropped if the previous element is `\NoValue`, e.g., `separator-a` and `separator-b`. The keyquote key is currently ignored. Default: `text, separator-a, ?, number, punct, separator-b, ?, title, separator-c, subtitle`

`separator-a` (*tokenlist*) By default the separation between `text` and `number`. Default: `\nobreakspace`

`separator-b` (*tokenlist*) By default the separation between `punct` and `title`. The default starts a new line with extra vertical space. Default: `\hspace{1em}`

`separator-c` (*tokenlist*) By default the separation between `title` and `subtitle`. Default: `\`

`separator-d` (*tokenlist*) By default not used. Default: ⟨*empty*⟩

Semantics & Comments: Implements a heading layout where text after the heading continues on the same line as the heading. It is what LaTeX always used by default for `\paragraph` and `\subparagraph`.

This template is intended to be used with the `heading runin` template (but could perhaps also be used with `display`).

5.3 Default instances

These instances are set up to mimic the design of the standard LaTeX classes. For other classes they could be adjusted by changing the instance values and/or by basing them on a different template.

For each heading command an instance of type `heading` and one of type `headformat` is defined. By convention, the instance names are identical to the heading command without the backslash, e.g. for `\section` they are named `section`, etc.

It is also possible to set up special `heading` template instances that are only used when an unnumbered heading is typeset or when a heading is placed on a recto or verso page in twoside mode. This works as follows: If the `heading` template instance for the heading command is ⟨*name*⟩ then

⟨*name*⟩`:recto` if defined is the `heading` instance name for recto (odd) pages;

⟨*name*⟩`:verso` is the instance name for verso pages;

⟨*name*⟩`:`⟨*recto-or-verso-or-empty*⟩`:unnumbered` is the instance name for unnumbered headings on recto, verso, or all pages.

If some of these instances are defined then the most specific one matching the current context is used, e.g., a `\section*` command on an odd page will try to use the `heading` instance `section:recto:unnumbered` or `section::unnumbered` or `section:recto` or `section` in that order.

This is only done for the instances of type `heading`. The names for instances of type `headformat` are not subject to this mechanism and are used exactly as specified by the `headformat-instance` key in the `heading` instance.

In documents that do not use the `twoside` option, all pages are considered recto pages.

None of these special `heading` instances are set up by default, but classes that wish to produce layouts that differ based on placement or numbering can easily do so by defining such instances as necessary.

Instead of, or in addition to, setting up special `heading` instances it is also possible to use `\IfVersoHeadingTF` inside instance keys. It executes the true or false branch depending on the page the heading is typeset. `T` and `F` variants are also available.

5.3.1 Instances of `heading` templates

`part` (instance of `display` template) Used for the `\part` command.

`chapter` (instance of `display` template) Used for the `\chapter` command.

`section` (instance of `display` template) Used for the `\section` command.

`subsection` (instance of `display` template) Used for the `\subsection` command.

`subsubsection` (instance of `display` template) Used for the `\subsubsection` command.

`paragraph` (instance of `runin` template) Used for the `\paragraph` command.

`subparagraph` (instance of `runin` template) Used for the `\subparagraph` command.

5.3.2 Instances of `headformat` templates

`part` (instance of `display` template) Used in `heading` instance for `\part`.

`chapter` (instance of `display` template) Used in `heading` instance for `\chapter`. By default identical to the one used for `\part`.

`std` (instance of `hang` template) Used as default in the heading template if nothing else is specified.

`section` (instance of `hang` template) Used in `heading` instance for `\section`. By default identical to the `std` instance.

`subsection` (instance of `hang` template) Used in `heading` instance for `\subsection`. By default identical to the `std` instance.

`subsubsection` (instance of `hang` template) Used in `heading` instance for `\subsubsection`. By default identical to the `std` instance.

`paragraph` (instance of `runin` template) Used in `heading` instance for `\paragraph`.

`subparagraph` (instance of `runin` template) Used in `heading` instance for `\subparagraph`.

5.4 Use of templates from the 2026-06-01 release of LaTeX

In the first heading implementation using templates we provided a number of somewhat restricted templates that covered the default layouts of standard LaTeX classes but not much beyond this.

For the 2026-11-01 release we have replaced them with more general templates described above that use the general order key mechanism. As a side effect the new templates have some additional keys and some keys used in the templates of first implementation got dropped.

For the next couple of releases we will keep the original templates available (only renamed to ⟨*name*⟩`-v1`). Eventually, we want to drop the `-v1` versions but for the next couple of releases we provide them alongside the new more flexible templates, so that there is no need to transitioning to the new templates in a rush.

If you have defined instances using the first template version then you can (for now) continue to use them simply by changing the template names in your instance declarations from ⟨*name*⟩ to ⟨*name*⟩`-v1`, e.g., you have to change from

```
\DeclareInstance{heading}   {chapter}{display} { ... }
\DeclareInstance{headformat}{chapter}{display} { ... }

\DeclareInstance{heading}   {section}{display} { ... }
\DeclareInstance{headformat}{section}{hang}    { ... }
...
```

to

```
\DeclareInstance{heading}   {chapter}{display-v1} { ... }
\DeclareInstance{headformat}{chapter}{display-v1} { ... }

\DeclareInstance{heading}   {section}{display-v1} { ... }
\DeclareInstance{headformat}{section}{hang-v1}    { ... }
...
```

No other changes are necessary for now. However, going forward you should switch to the extended templates because the `-v1` templates will eventually be dropped.

Switching mainly means stopping the use of the keys `prefix-number-sep` and `number-title-sep` because they have been replaced with the more general `separator-a`, `separator-b`, … keys.[2]

The template documentation for these deprecated templates is available as part of Appendix A in case you need to consult it in order to update your template instance declarations to use the new templates. Please do not use these templates for new work, as they will eventually get removed.

6 Support for legacy classes and packages

6.1 Support classes based on legacy LaTeX 2ε interfaces

`\@startsection` The `\@startsection` command has the following syntax:

[2] There are some more changes in the keys, but it is less likely that they have been used. If necessary, compare the documentation for the old and the new templates to find out what else needs adjusting.

```
\@startsection{name}{level}{indent}{beforeskip}{afterskip}{style}
```

with a special logic that the sign of ⟨*beforeskip*⟩ determines display or run-in heading and that of ⟨*afterskip*⟩ whether or not the next paragraph is indented.

All arguments can be cleanly mapped to `heading` and `headformat` templates. Thus, if encountered suitable instances are automatically declared unless they already exist.

`\secdef` In contrast, `\secdef` only does the parsing and delegates the layout to two commands which can contain arbitrary code (usually hardwired) so that there is no realistic chance to take this apart and figure out what values should be used for what template parameters.

We therefore do not attempt to model this, but instead just use the standard layout from LaTeX's standard classes for `\chapter` and `\part` if we can identify that the `\secdef` is used to define one of these.

Maybe we can try at some stage to get some static analysis tool going.

7 Support for links

All heading commands (numbered and unnumbered) should contain support for active links directly. hyperref does not patch the new heading commands! As `\refstepcounter` is not used there is no automatic target.

This means that all definitions should contain at an appropriate place a `\MakeLinkTarget` command. Typically numbered heading commands will use `\MakeLinkTarget{`⟨*counter*⟩`}` while unnumbered heading commands use `\MakeLinkTarget[`⟨*counter*⟩`]{}`.

The definition must take care that the target is not separated from the heading by a page break. It also should not affect spacing. The target should be placed so that a jump to the target gives a satisfying user experience, in most cases the best places is at the left margin a bit above the heading.

The targets create also structure destinations that are used by the tagging code. It is therefore important that the targets are in the right place in relation to the tagging commands.

8 Bookmarks

Bookmarks are created by the `\addcontentsline` command, more precisely in the new latex-lab toc code a hook with arguments is inserted at the begin of the command which contains the command that creates the bookmark. The arguments of `\addcontentsline` and so also of the hook are the type of the toc, the level name and the (short-)title of the heading. To pass a differing bookmark text the code uses `\texorpdfstring` in the `\addcontentsline`.

UFi:check if this is still the implementation ...

9 Tagging support

Tagging of heading commands has to do two tasks:

- surround the whole section (heading and text) with a `Sect` structure
- tag the heading with an Hn structure.

This is done with tagging sockets which are documented in the code and in `latex-lab-sec`.

10 Debugging support

\DebugHeadingsOn
\DebugHeadingsOff
\head_debug_on:
\head_debug_off:

These commands enable/disable debugging messages.

11 The Implementation

```
⟨*package⟩
⟨@@=head⟩

\ProvidesExplPackage
 {latex-lab-testphase-sec-template}
 {\ltlabsecIIdate}
 {\ltlabsecIIversion}
 {heading implementation}
```

11.1 Debugging

\g__head_debug_bool

```
\bool_new:N \g__head_debug_bool
```

(*End of definition for* \g__head_debug_bool.)

__head_debug:n
__head_debug_typeout:n

```
\cs_new_eq:NN \__head_debug:n \use_none:n
\cs_new_eq:NN \__head_debug_typeout:n \use_none:n
```

(*End of definition for* __head_debug:n *and* __head_debug_typeout:n.)

\head_debug_on:
\head_debug_off:
__head_debug_gset:

```
\cs_new_protected:Npn \head_debug_on:
  {
    \bool_gset_true:N \g__head_debug_bool
    \__head_debug_gset:
  }
\cs_new_protected:Npn \head_debug_off:
  {
    \bool_gset_false:N \g__head_debug_bool
    \__head_debug_gset:
  }

\cs_new_protected:Npn \__head_debug_gset:
  {
    \cs_gset_protected:Npe \__head_debug:n ##1
      { \bool_if:NT \g__head_debug_bool {##1} }
    \cs_gset_protected:Npe \__head_debug_typeout:n ##1
      { \bool_if:NT \g__head_debug_bool { \typeout{[head]~ ##1} } }
  }
```

(*End of definition for* `\head_debug_on:` , `\head_debug_off:` , *and* `\__head_debug_gset:`. *These functions are documented on page 22.*)

`\DebugHeadingsOn`
`\DebugHeadingsOff`

```
\cs_new_protected:Npn \DebugHeadingsOn  { \head_debug_on:  }
\cs_new_protected:Npn \DebugHeadingsOff { \head_debug_off: }

\DebugHeadingsOff
```

(*End of definition for* `\DebugHeadingsOn` *and* `\DebugHeadingsOff`. *These functions are documented on page 22.*)

`\__head_show_arguments:nnnnnn`

Some debug data ...

```
\cs_new:Npn \__head_show_arguments:nnnnnn #1#2#3#4#5#6 {
  \__head_debug_typeout:n{---------- Headformat~ instance~ arguments:}
  \__head_debug_typeout:n{1:~ keys~ =~ \exp_not:n{#1}}
  \__head_debug_typeout:n{2:~ text~ =~ \exp_not:n{#2}}
  \__head_debug_typeout:n{3:~ number~ =~ \exp_not:n{#3}}
  \__head_debug_typeout:n{4:~ title~ =~ \exp_not:n{#4}}
  \__head_debug_typeout:n{5:~ subtitle~ =~ \exp_not:n{#5}}
  \__head_debug_typeout:n{6:~ quotation~ =~ \exp_not:n{#6}}
}
```

(*End of definition for* `\__head_show_arguments:nnnnnn`.)

`\__head_show_arguments:nnnnnnnnn`

Some debug data ...

```
\cs_new:Npn \__head_show_arguments:nnnnnnnnn #1#2#3#4#5#6#7#8#9 {
  \__head_debug_typeout:n{---------- Heading~ instance~ arguments:}
  \__head_debug_typeout:n{1:~ keys~ =~ \exp_not:n{#1}}
  \__head_debug_typeout:n{2:~ unnumbered~ =~ \exp_not:n{#2}}
  \__head_debug_typeout:n{3:~ title~ =~ \exp_not:n{#3}}
  \__head_debug_typeout:n{4:~ toc~ =~ \exp_not:n{#4}}
  \__head_debug_typeout:n{5:~ running~ =~ \exp_not:n{#5}}
  \__head_debug_typeout:n{6:~ bookmark~ =~ \exp_not:n{#6}}
  \__head_debug_typeout:n{7:~ nameref~ =~ \exp_not:n{#7}}
  \__head_debug_typeout:n{8:~ label~ =~ \exp_not:n{#8}}
  \__head_debug_typeout:n{9:~ subtitle | quote ~ =~ \exp_not:n{#9}}
}
```

(*End of definition for* `\__head_show_arguments:nnnnnnnnn`.)

11.2 Temp variable(s)

`\l__head_tmpa_tl`

```
\tl_new:N\l__head_tmpa_tl
```

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

11.3 variable(s)

These token lists are automatically declared by the key machinery.

```
%\tl_new:N \l__head_bookmark_tl
%\tl_new:N \l__head_running_tl
%\tl_new:N \l__head_toc_tl
%\tl_new:N \l__head_nameref_tl
%\tl_new:N \l__head_subtitle_tl
%\tl_new:N \l__head_quote_tl
%\bool_new:N l__head_unnumbered_bool
```

\l__head_label_tl
\l__head_instance_keys_tl
\l__head_number_tl
\l__head_saved_secnumdepth_tl
\l__head_title_tl
\l__head_placement_tl
\l__head_instance_name_tl
\l__head_page_type_tl

But these token lists needs a declaration:

```
\tl_new:N \l__head_label_tl
\tl_new:N \l__head_instance_keys_tl
\tl_new:N \l__head_number_tl
\tl_new:N \l__head_saved_secnumdepth_tl
\tl_new:N \l__head_title_tl
\tl_new:N \l__head_placement_tl
\tl_new:N \l__head_instance_name_tl
\tl_new:N \l__head_page_type_tl
```

(*End of definition for* `\l__head_label_tl` *and others.*)

11.4 New user commands

\theheading

This command expands to `\thechapter`, `\thesection` etc in every heading command.

```
\tl_new:N\theheading
```

(*End of definition for* `\theheading`. *This function is documented on page 3.*)

\partmark

This replaces the fix `\markboth{}{}` or similar in the standard `\part` definition. As the command is already defined in some classes (like memoir), we provide it through a hook.

```
\AddToHook{class/after}{\providecommand*\partmark[1]{\markboth{}{}}}
```

(*End of definition for* `\partmark`. *This function is documented on page 3.*)

11.5 The LaTeX 2ε parsing interface

LaTeX 2ε provides heading commands with a starred form (unnumbered) and one optional argument (to specify an alternative toc/running head). We augment this slightly by supporting a key value interface in the optional argument. This is handled by defining the commands through `\ParseLaTeXeHeading`. This should happen in the document class.

\ParseLaTeXeHeading

`\ParseLaTeXeHeading` arguments:

1: instance name to use (of type "heading")

2: numbered? boolean

3: shorttitle or key/val for layout adjustments and individual title settings

4: main title

This command handles the document input that may be hidden in the optional argument, i.e., special data for toc, running (header), bookmark, label, and for more specialized headings subtitle and quote. It also handles the legacy case that the optional argument is just an alternate title to be used for toc, running, and bookmark (through the key shorttitle to which it gets converted).

```
\cs_new_protected:Npn \ParseLaTeXeHeading #1 #2 #3 #4 {
```

Before we start parsing a heading we have to make sure that there isn't a run-in heading still waiting to be typeset. Otherwise, the new heading might overwrite some variables that are still needed to typeset the dangling heading.

```
  \if@noskipsec \leavevmode \fi \par

  \__head_debug_typeout:n{==================================}
  \__head_debug_typeout:n{#1:~ \IfBooleanT{#2}{*}
                \IfValueT{#3}{[\exp_not:n{#3}]}
                {\exp_not:n{#4}}}
```

We first check if the title argument contains a `\label` command. If yes, we remove it and add it to `\l__head_label_tl` (and if more than one all of them) and save the rest of the main title in `\l__head_title_tl` for later use. If there was no `\label` command then `\l__head_title_tl` is set to `#4`.

```
  \__head_find_label:w #4 \label\q_no_value \__head_find_label:w
```

By default toc, running, and bookmark use the data provided by the main title. However, if the second argument is true (i.e., a star was given) they are all suppressed (because this is the LaTeX 2_ε logic).

```
  \IfBooleanTF{#2}
    { \tl_clear:N \l__head_toc_tl
      \tl_clear:N \l__head_running_tl
      \tl_clear:N \l__head_bookmark_tl
      \bool_set_true:N \l__head_unnumbered_bool
    }
```

If there was no star then we copy `\l__head_title_tl` to `toc`, `bookmark`, and `running`. However, only when it isn't empty as that would suppress them and that would be right as a default. Instead we pass `\prg_do_nothing:` which also produces no output but then the key isn't empty and thus isn't suppressed by default.

```
    { \tl_if_empty:NTF \l__head_title_tl
        { \tl_set:Nn \l__head_toc_tl { \prg_do_nothing: }
          \tl_set_eq:NN \l__head_running_tl  \l__head_toc_tl
          \tl_set_eq:NN \l__head_bookmark_tl \l__head_toc_tl
        }
        { \tl_set_eq:NN \l__head_toc_tl      \l__head_title_tl
          \tl_set_eq:NN \l__head_running_tl  \l__head_title_tl
          \tl_set_eq:NN \l__head_bookmark_tl \l__head_title_tl
        }
      \bool_set_false:N \l__head_unnumbered_bool
    }
```

The nameref always starts out matching the main title.

```
  \tl_set_eq:NN \l__head_nameref_tl \l__head_title_tl
```

Normally we don't have a subtitle or quote unless they are explicitly given through keys, so we start with `\c_novalue_tl`

```
\tl_set_eq:NN \l__head_subtitle_tl \c_novalue_tl
\tl_set_eq:NN \l__head_quote_tl    \c_novalue_tl
```

If the optional argument is empty we set the `\l__head_instance_keys_tl` to empty, otherwise process the key/val list setting the keys defined in the module "head". This overwrites the defaults specified above if keys like "toc" are in the list. All the key/vals unknown by that module are put into `\l__head_instance_keys_tl` which may or may not make this token list non-empty.
If the user has a `label` key and also a `\label` in the main title argument then the latter is ignored (no error checking for that). We could alternatively set all labels we find, perhaps that is the better approach?

```
\IfNoValueTF { #3 }
  { \tl_clear:N \l__head_instance_keys_tl }
  { \keys_set_known:nnN {heading} { #3 } \l__head_instance_keys_tl }
```

By default the instance name is `#1` so we set `\l__head_instance_name_tl`.

```
\tl_set:Nn \l__head_instance_name_tl {#1}
```

Then we call the socket `heading/instname` which checks if special instances for recto/verso pages or for unnumbered headings exist and if so change `\l__head_instance_name_tl` so that these are called below.

```
\socket_use:n {heading/instname}
```

Finally, we call the heading instance using the collected mandatory arguments. To simplify downstream processing we pass the values not the container tokenlists resulting from the above processing.[3]

```
\__head_debug_typeout:n{use~ 'heading'~ instance:~ \l__head_instance_name_tl }
\use:e {
  \exp_not:N \UseInstance{heading}{ \l__head_instance_name_tl }
             { \exp_not:o \l__head_instance_keys_tl }
             { \bool_if:NTF \l__head_unnumbered_bool \BooleanTrue \BooleanFalse }
             { \exp_not:o \l__head_title_tl }
             { \exp_not:o \l__head_toc_tl }
             { \exp_not:o \l__head_running_tl }
             { \exp_not:o \l__head_bookmark_tl }
             { \exp_not:o \l__head_nameref_tl }
             { \exp_not:o \l__head_label_tl }
             { { \exp_not:o \l__head_subtitle_tl }
               { \exp_not:o \l__head_quote_tl    } }
}
}
```

`heading/instname` (*socket*) This socket checks for special instance names with `:recto` or `:verso` and possibly `:unnumbered` appended to the base name of the `heading` instance. If one is found it adjusts `\l__head_instance_name_tl`.

```
\NewSocket{heading/instname}{0}
```

[3] I think this is a common requirement and perhaps `\UseInstance` should really o-expand all arguments it receives.

`extend` (*plug*)

```
\NewSocketPlug{heading/instname}{extend}{
```

Set `\l__head_page_type_tl` to `recto` or `verso` based on the recorded position of the heading in the previous run (or recto if unknown). Always use `recto` if `@twoside` is `false`.

```
  \bool_set:Nn \l__head_verso_page_bool
    {
      \bool_lazy_and_p:nn
        { \legacy_if_p:n { @twoside } }
        { \int_if_even_p:n
          { \property_ref:een
            { heading*. \int_use:N \g__head_count_int }
            {pagenum} { 1 }
          }
        }
    }
  \tl_set:Ne \l__head_page_type_tl
     { \bool_if:NTF \l__head_verso_page_bool {verso} {recto} }
```

If the heading is unnumbered check for `:unnumbered` as the third component.

```
  \bool_if:NTF \l__head_unnumbered_bool
    { \exp_args:Nne
      \IfInstanceExistsTF {heading}
                          { \l__head_instance_name_tl :
                            \l__head_page_type_tl :
                            unnumbered }
        { \tl_set:Ne \l__head_instance_name_tl
             { \l__head_instance_name_tl :
               \l__head_page_type_tl : unnumbered }
        }
        { \exp_args:Nne
          \IfInstanceExistsTF {heading}
                              { \l__head_instance_name_tl
                                :: unnumbered }
            { \tl_set:Ne \l__head_instance_name_tl
                 { \l__head_instance_name_tl
                    :: unnumbered        }
            }
            { \exp_args:Nne
              \IfInstanceExistsT {heading}
                                 { \l__head_instance_name_tl :
                                   \l__head_page_type_tl }
                {
                  \tl_set:Ne \l__head_instance_name_tl
                     { \l__head_instance_name_tl :
                       \l__head_page_type_tl       }
                }
            }
        }
    }
```

Otherwise there can be at most two components:

```
    { \exp_args:Nne
```

```
      \IfInstanceExistsT {heading}
                         { \l__head_instance_name_tl :
                           \l__head_page_type_tl }
        {
          \tl_set:Ne \l__head_instance_name_tl
             { \l__head_instance_name_tl :
               \l__head_page_type_tl       }
        }
    }
}
\AssignSocketPlug{heading/instname}{extend}
```

`l__head_verso_page_bool`

```
\bool_new:N      \l__head_verso_page_bool
```

`\IfVersoHeadingTF` `\IfVersoHeadingT` `\IfVersoHeadingF`

```
\cs_new_protected:Npn \IfVersoHeadingTF
   { \bool_if:NTF \l__head_verso_page_bool }
\cs_new_protected:Npn \IfVersoHeadingT
   { \bool_if:NT \l__head_verso_page_bool }
\cs_new_protected:Npn \IfVersoHeadingF
   { \bool_if:NF \l__head_verso_page_bool }
```

To get unique labels when recording the property we use a counter. It is stepped in the `heading/record` socket.

`\g__head_count_int`

```
\int_new:N \g__head_count_int
```

`heading/record` (*socket*) This socket is used at the point where the heading typeset. It has to happen late to ensure that we are already on the page/column on which the heading will finally appear, because that is what we want to record. It is therefore used in the `headformat` templates.

```
\NewSocket{heading/record}{0}
```

`page` (*plug*) Record the current page number in a way suitable for numeric comparison and then increment the internal counter so that we get a unique label for the next heading.

```
\NewSocketPlug{heading/record}{page}{
  \property_record:ee { heading*. \int_use:N\g__head_count_int } {pagenum}
  \int_gincr:N \g__head_count_int
}
\AssignSocketPlug{heading/record}{page}
```

Syntax keys that can appear in the optional argument of headings parsed by `\ParseLaTeXeHeading`. All other keys used there are passed to the heading instance to overwrite instance setting.

```
\keys_define:nn {heading} {
  , shorttitle .meta:n    =
      {toc = {#1} , running = {#1} , bookmark = {#1} , nameref= {#1} }
  , bookmark   .tl_set:N  = \l__head_bookmark_tl
  , running    .tl_set:N  = \l__head_running_tl
  , toc        .tl_set:N  = \l__head_toc_tl
  , nameref    .tl_set:N  = \l__head_nameref_tl
%
  , numbered   .bool_set_inverse:N = \l__head_unnumbered_bool
```

```
  , numbered   .default:n = true
  , unnumbered .bool_set:N = \l__head_unnumbered_bool
  , unnumbered .default:n = true
%
  , subtitle   .tl_set:N  = \l__head_subtitle_tl
  , quote      .tl_set:N  = \l__head_quote_tl
```

We collect labels together with their `\label` command in case there is more than one. This way we can later simply execute `\l__head_label_tl` without any status checks.

```
  , label      .code:n    = \tl_put_right:Nn \l__head_label_tl { \label{#1} }
}
```

(*End of definition for* `\ParseLaTeXeHeading` *and others. These functions are documented on page* **??**.)

`\__head_find_label:w` A simple-minded check for an existing `\label` command in the main title (could be done better I guess). We add `\label\q_no_value` at the end of the argument, so that we can be sure to find something.
As currently implemented the spaces on both sides of a `\label` command survive if it is removed. This may be an issue in which case this may need some adjustments.

```
\cs_new:Npn \__head_find_label:w #1 \label #2#3 \__head_find_label:w {
```

If the token after `\label` is `\q_no_value` then the title had no label and we set the two variables accordingly.

```
  \quark_if_no_value:nTF {#2}
     { \__head_debug_typeout:n{----~no~label }
       \tl_clear:N \l__head_label_tl
       \tl_set:Nn\l__head_title_tl {#1#3}
     }
```

Otherwise, there was a real `\label` command and `#2` holds the label string.

```
     { \__head_debug_typeout:n{----~label~found~#2}
       \tl_set:Nn\l__head_label_tl { \label{#2} }
```

To construct the value for `\l__head_title_tl` we have to get rid of the two tokens we added at its end, this is done with `\__head_find_label_aux:w`.

```
       \tl_set:Nn\l__head_title_tl {#1}
       \__head_find_label_aux:w #3\__head_find_label_aux:w
     }
     \__head_debug_typeout:n{----~return:~ '\exp_not:o \l__head_title_tl' }
}
```

There is at least one more `\label` now and the token after it should be our `\q_no_value`. If not then the title argument had at least 2 labels and we collect the newly found one and recurse.

```
\cs_new:Npn \__head_find_label_aux:w #1 \label #2#3 \__head_find_label_aux:w {
```

The `#1` may not be everything we have to pick up but we know for sure that it belongs to the title.

```
  \tl_put_right:Nn \l__head_title_tl { #1 }
```

Now let's see if we are at the end of the argument. If not pick up the newly found `\label` and recurse on the remaining material.

```
  \quark_if_no_value:nF {#2}
    { \__head_debug_typeout:n{---~ extra~ label~ '#2'~ found }
      \tl_put_right:Nn \l__head_label_tl { \label{#2} }
      \__head_find_label_aux:w #3\__head_find_label_aux:w
    }
}
```

(*End of definition for* `\__head_find_label:w`.)

11.6 General helper commands

`\WordSpaceAmount` Produce the amount of a (fractional) word space in the current font in a way that it can be used in a skip or dimen register assignment. The argument specifies the fraction, e.g., `2` gives two word spaces and `.5` would produce half a word space.
This general helper doesn't really belong here, so should be eventually moved.

```
\newcommand\WordSpaceAmount[1]{
  \glueexpr
        #1\fontdimen2\the\font
  plus  #1\fontdimen3\the\font
  minus #1\fontdimen4\the\font
  \relax
}
```

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

Some designs automatically add a punctuation symbol (typically a period) at the end of a title, but that should only happen if the title doesn't already end in a punctuation symbol.

This is achieved by setting the `\spacefactor` for punctuation symbols (slightly) higher than `1000` and then checking the current `\spacefactor` before trying to add the punctuation. If it is `1000` or less then the title didn't end in a punctuation and we can safely add one, otherwise we don't.

If `\nonfrenchspacing` is in force this is automatically the case, but in LaTeX 2_ε `\frenchspacing` did set the `\spacefactor` of all punctuation symbols to `1000` making them indistinguishable from other characters.

Thus, to make this work, we have to change `\frenchspacing` which is a breaking change as it can lead to pagination differences given that the word space after a punctuation gets (very minimally) larger this way. However, this approach was successfully used in the AMS classes for ages and it offers nice design possibilities so we enable it in documents using `\DocumentMetadata` automatically.

`\frenchspacing` `\nonfrenchspacing` We give all punctuation symbols a unique `\sfcode` value so that it is even possible to determine which punctuation was just typeset.

```
\def\frenchspacing{\sfcode`\.1006\sfcode`\?1005\sfcode`\!1004%
  \sfcode`\:1003\sfcode`\;1002\sfcode`\,1001 }
```

The same should be done for `.?!` if we want to be able to distinguish those when `\nonfrenchspacing` is in force.

```
\def\nonfrenchspacing{\sfcode`\.3002\sfcode`\?3001\sfcode`\!3000%
  \sfcode`\:2000\sfcode`\;1500\sfcode`\,1250 }
```

(*End of definition for* `\frenchspacing` *and* `\nonfrenchspacing`*. These functions are documented on page* ??.)

`\nopunct` The AMS classes also offered `\nopunct` that could be added at the end of a title and would prevent the addition of a punctuation symbol.
It should have a value for `\spacefactor` that is not used for `\frenchspacing` so that this special case can be detected.

```
\cs_new_protected:Npn \nopunct {\spacefactor 1007 }
```

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

`\AddPunct` In the AMS classes this was called `\@addpunct`.
Note: this mechanism will fail if the text ending in a punctuation has an uppercase character just before the punctuation, e.g., if somebody writes `SOLUTION:`. In that case `SOLUTION\@.` would be necessary!

```
\cs_new_protected:Npn \AddPunct #1 {
  \relax\ifhmode
    \ifnum\spacefactor>\@m
      \__head_debug_typeout:n {--->~ punctuation~ '#1'~ not~ added}
    \else
      \__head_debug_typeout:n {--->~ punctuation~ '#1'~ added}
      #1
    \fi
  \fi
}
```

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

11.7 Templates

11.7.1 Template types

`heading` (*type*) Templates of type `heading` are used to produce headings. The positional arguments are:

```
1: key/val list
2: unnumbered?
3: title
4: toc
5: running
6: bookmark
7: nameref
8: label(s)
9: { subtitle } { quote }
```

```
\NewTemplateType{heading}{9}
```

`headformat` (*type*) Templates of type `headformat` are used to produce heading layout from the formatted heading number (if any), the heading title, subtitle and quotation. The positional arguments are:

```
1: key/val list for document-level customizations
2: fixed text string or \NoValue
3: formatted heading number or \NoValue
4: title
5: subtitle or \NoValue
6: quote or \NoValue
```

```
\NewTemplateType{headformat}{6}
```

11.7.2 heading templates interfaces

We have two heading templates: `display` and `runin`.

`heading display` (*templ.*) The `display` template produces a display heading, i.e., one that has vertical space before and after it.

```
\DeclareTemplateInterface{heading}{display}{9}
{
  , name                : tokenlist
  , parent-name         : tokenlist
  , reset-counter       : tokenlist
  , level               : integer = 0
```

By default headings can be numbered and the numbering is determined for each heading in the document individually.

```
  , force-unnumbered    : boolean = false
```

The `placement` key sets default values for the keys `placement-start-code` and `placement-end-code`.

```
  , placement           : choice {page , top , normal, ragged } = normal
  , column-spanning     : boolean = false
  , mark-cmd            : function(1) =
  , para-indent         : choice { true , false } = false
```

We use `\c_max_int` as a fake penalty to indicate that no penalty was given in a key.

```
  , before-vspace       : skip    = 0pt
  , penalty             : integer = \c_max_int
  , after-penalty-vspace : skip   = \maxdimen
  , after-vspace        : skip    = 0pt
```

The next two keys are only there to overwrite the `placement` settings with explicit code. Thus, their default value is set up by the `placement` key (which has a default).

```
  , placement-start-code : tokenlist =      % no default values!
  , placement-end-code   : tokenlist =      % no default values!
```

A fixed text string such as `\@chappap` could be specified in the next variable. By default it has no value.

```
  , text                : tokenlist = \NoValue
```

A postfix string to be added to the heading number (if the heading is numbered). By default empty (not `\NoValue`)!

```
  , punct               : tokenlist =
  , heading-decls       : tokenlist = \normalfont
  , text-decls          : tokenlist =
  , number-decls        : tokenlist =
  , title-decls         : tokenlist =
  , subtitle-decls      : tokenlist =
  , quote-decls         : tokenlist =
```

```
  , heading-format       : function{1} = #1
  , text-format          : function{1} = #1
  , number-format        : function{1} = #1
  , punct-format         : function{1} = #1
  , title-format         : function{1} = #1 \par
  , subtitle-format      : function{1} = #1
  , quote-format         : function{1} = #1
```

Not clear where this key should live and if it really makes sense to offer variations on the individual heading levels as in `\UseStructureName{sec/???/number}`.

```
  , number-tag           : tokenlist = heading-number  % \UseStructureName{sec/???}
  , headformat-instance  : tokenlist = std
  , contents-extra       : tokenlist =
}
```

Old names for the above to be dropped soon:

```
\DeclareTemplateInterfaceEquivalents{heading}{display}[2027/11/01]{
  , start-code    : alias = placement-start-code
  , final-code    : alias = placement-end-code
}

\DeclareTemplateInterfaceEquivalents{heading}{display}[2027/11/01]{
  , prefix        : alias = text
  , prefix-decls  : alias = text-decls
  , prefix-format : alias = text-format
}
```

`heading runin` (*templ.*) This template is similar to the `display` template but implements a run-in heading, i.e., one where the paragraph text continues on the same line.
Unfortunately, we can't really use `\DeclareTemplateCopy` to set it up because with `\EditTemplateDefaults` we are not able to alter the setup for the `placement` and `para-indent` keys as that involves changing the implementation code. Thus with `\DeclareTemplateCopy` we can copy the interface setup but the code setup still needs to be done using `\DeclareTemplateCode`.

```
\DeclareTemplateInterface{heading}{runin}{9}
{
  , name                 : tokenlist
  , parent-name          : tokenlist
  , reset-counter        : tokenlist
  , level                : integer = 0
  , force-unnumbered     : boolean = false
  , placement            : choice {page , top , normal, ragged } = normal
  , column-spanning      : boolean = false
  , mark-cmd             : function(1) =
  , para-indent          : choice { true , false } = false
  , before-vspace        : skip      = 0pt
  , penalty              : integer   = \c_max_int
  , after-penalty-vspace : skip      = \maxdimen
  , after-hspace         : skip      = 0pt
  , placement-start-code : tokenlist =      % no default values!
  , placement-end-code   : tokenlist =      % no default values!
```

```
  , text                : tokenlist = \NoValue
  , punct               : tokenlist =
  , heading-decls       : tokenlist = \normalfont
  , text-decls          : tokenlist =
  , number-decls        : tokenlist =
  , title-decls         : tokenlist =
  , subtitle-decls      : tokenlist =
  , quote-decls         : tokenlist =
  , heading-format      : function{1} = #1
  , text-format         : function{1} = #1
  , number-format       : function{1} = #1
  , punct-format        : function{1} = #1
```

We have one difference in the defaults: in run-in headings everything has to stay in horizontal mode so we do not append `\par` in the default value for `title-format`.

```
  , title-format        : function{1} = #1
  , subtitle-format     : function{1} = #1
  , quote-format        : function{1} = #1
  , number-tag          : tokenlist = heading-number  % \UseStructureName{sec/???}
  , headformat-instance : tokenlist = std
  , contents-extra      : tokenlist =
}
```

Old names for the above to be dropped soon:

```
\DeclareTemplateInterfaceEquivalents{heading}{runin}[2027/11/01]{
  , start-code    : alias = placement-start-code
  , final-code    : alias = placement-end-code
}

\DeclareTemplateInterfaceEquivalents{heading}{runin}{
  , prefix        : alias = text
  , prefix-decls  : alias = text-decls
  , prefix-format : alias = text-format
}
```

11.7.3 headformat templates interfaces

We have three template: `display`, `hang` and `runin`.

`headformat display` (*templ.*) The `display` template produces a heading in which the heading elements form a single block, potentially with separators that separates some elements vertically from the other.

```
\DeclareTemplateInterface{headformat}{display}{6}
{
  , heading-indent    : length    = 0pt
  , left-margin       : length    = 0pt
  , right-margin      : length    = 0pt

  , order             : commalist = {text,separator-a,?,number,punct,
                                     separator-b,?,title,
                                     separator-c, subtitle}
  , separator-a       : tokenlist = \nobreakspace
  , separator-b       : tokenlist = \par \nobreak \vspace{20pt}
  , separator-c       : tokenlist = \
  , separator-d       : tokenlist =
}
```

`headformat hang` (*templ.*) The `hang` template produces a heading where some elements are measured to determine a hanging indentation while the remaining ones are then typeset in that constrained space. This is why we have two order keys.

```
\DeclareTemplateInterface{headformat}{hang}{6}
{
  , heading-indent    : length    = 0pt
  , left-margin       : length    = 0pt
  , right-margin      : length    = 0pt
  , order-hang        : commalist = {text,separator-a,?,number,punct,
                                     separator-b,?}
  , order             : commalist = {title, separator-c, subtitle}
  , separator-a       : tokenlist = \nobreakspace
  , separator-b       : tokenlist = \hspace{1em}
  , separator-c       : tokenlist = \
  , separator-d       : tokenlist =
}
```

`headformat runin` (*templ.*) The `runin` template produces a heading which is horizontally oriented and text following the heading will continue on the same line as the heading.

```
\DeclareTemplateInterface{headformat}{runin}{6}
{
  , heading-indent    : length = 0pt
  , order             : commalist = {text,separator-a,?,number,punct,
                                     separator-b,?,title,
                                     separator-c,subtitle}
  , separator-a       : tokenlist = \nobreakspace
  , separator-b       : tokenlist = \hspace{1em}
  , separator-c       : tokenlist = \
  , separator-d       : tokenlist =
}
```

11.7.4 heading templates code

`heading display` (*templ.*)

```
\DeclareTemplateCode{heading}{display}{9}
{
  , name          = \l__head_name_tl
  , level         = \l__head_level_int
% next two not yet used
  , parent-name   = \l__head_pname_tl
  , reset-counter = \l__head_reset_cnt_tl


  , force-unnumbered = \l__head_unnumbered_bool
```

The `placement` key determines whether the heading can appear anywhere (`normal`), automatically starts a new page or column (`top`), or is on a page of its own (`page`.
It is usually implemented by setting `\l__head_placement_start_code_tl` and `\l__head_placement_end_code_tl` (exceptions are `normal` and `ragged`). These settings can be fine-tuned or overwritten with the keys `placement-start-code` and `placement-end-code`, if necessary.

```
  , placement     = {
        ,page   = \__head_debug_typeout:n{ A~ page~ heading }
```

```
                    \tl_set:Nn \l__head_placement_tl { page }
         ,top    = \__head_debug_typeout:n{ A~ top~ heading }
                    \tl_set:Nn \l__head_placement_tl { top }
         ,normal = \__head_debug_typeout:n{ A~ normal~ heading }
                    \tl_set:Nn \l__head_placement_tl { normal }
         ,ragged = \__head_debug_typeout:n{ A~ normal~ heading~ (ragged~ style) }
                    \tl_set:Nn \l__head_placement_tl { ragged }
                      }

   , column-spanning = \l__head_column_spanning_bool

   , mark-cmd      = \__head_mark_cmd:n
```

For now we make use of the legacy coding for indentation of the following paragraph.

```
   , para-indent   = {
                       ,true  = \@afterindenttrue
                       ,false = \@afterindentfalse
                     }
   , before-vspace = \l__head_before_skip
   , penalty       = \l__head_penalty_int
   , after-penalty-vspace  = \l__head_after_penalty_skip
   , after-vspace  = \l__head_after_skip
   , placement-start-code = \l__head_placement_start_code_tl
   , placement-end-code   = \l__head_placement_end_code_tl

   , text            = \l__head_text_tl
   , punct           = \l__head_punct_tl

   , headformat-instance = \l__head_headformat_instance_tl

   , heading-decls   = \l__head_heading_decls_tl
   , text-decls      = \l__head_text_decls_tl
   , number-decls    = \l__head_number_decls_tl
   , title-decls     = \l__head_title_decls_tl
   , subtitle-decls  = \l__head_subtitle_decls_tl
   , quote-decls     = \l__head_quote_decls_tl

   , heading-format  = \__head_heading_format:n
   , text-format     = \__head_text_format:n
   , number-format   = \__head_number_format:n
   , punct-format    = \__head_punct_format:n
   , title-format    = \__head_title_format:n
   , subtitle-format = \__head_subtitle_format:n
   , quote-format    = \__head_quote_format:n
%
   , number-tag      = \l__head_number_tag_tl

   , contents-extra  = \l__head_contents_extra_tl
}

{

  \tl_set_eq:Nc \theheading { the \l__head_name_tl }
```

We store the value of `secnumdepth` so that we can change it locally.

```
  \tl_set:Ne\l__head_saved_secnumdepth_tl{\int_use:N\c@secnumdepth}
```

```
\__head_show_arguments:nnnnnnnnn
    {#1}{#2}{#3}{#4}{#5}{#6}{#7}{#8}{#9}
```

First evaluate any key setting done by the user in the optional first argument.

```
\SetCurrentKnownTemplateKeys{#1}
```

If `column-spanning` is requested but we are not typesetting in two columns we turn it off to avoid using `\@topnewpage` as this would be wrong. Otherwise we check the placement value and adjust it if necessary.

```
\bool_if:NT \l__head_column_spanning_bool
    { \if@twocolumn
        \str_if_eq:VnF \l__head_placement_tl {top}
          {
            \__head_debug_typeout:n {column-spanning~ requires~
              placement=top.~ Adjusted!}
            \tl_set:Nn \l__head_placement_tl {top}
          }
      \else
        \bool_set_false:N \l__head_column_spanning_bool
      \fi
    }
```

Based on the `placement` key we set up `\l__head_placement_start_code_tl` and `\l__head_placement_end_code_tl`. These two variables can also be set with the keys `placement-start-code` and `placement-end-code` in which case we don't alter their definitions.

```
\tl_if_empty:oT \l__head_placement_start_code_tl
 { \str_case:Vn \l__head_placement_tl
    {
      { page }
      { \tl_set:Nn \l__head_placement_start_code_tl
```

Clearly not `\@tempswa` and the page styles should be adjustable.

Straight from the placement definition of `\part`.

```
          { \if@openright
              \cleardoublepage
            \else
              \clearpage
            \fi
            \thispagestyle{plain}%
            \if@twocolumn
              \onecolumn
              \@tempswatrue
            \else
              \@tempswafalse
            \fi
            \null\vfil
          }
      }
      { top }
      { \tl_set:Nn \l__head_placement_start_code_tl
          { \if@openright\cleardoublepage\else\clearpage\fi
            \thispagestyle{plain}%
            \global\@topnum\z@
```

```
            }
        }
```

Ragged headings (i.e., those that leave a previous page ragged bottom if they generate a page break) are produced by replacing `\addpenalty` with a version that also adds `\vfil` and `\vfilneg`. This is handled by altering `\sec_add_penalty_with_possible_fil:n` which after use resets itself to `\addpenalty`.

```
        { ragged }
        { \cs_set_eq:NN \sec_add_penalty_with_possible_fil:n
                        \sec_add_penalty_with_fil:n
        }
      }
```

In all other cases we want `\l__head_placement_start_code_tl` to be empty, which is already the case so no need to have an explicit F branch.

```
%      { \tl_clear:N \l__head_placement_start_code_tl }
   }
%
  \tl_if_empty:oT \l__head_placement_end_code_tl
   { \str_case:VnF \l__head_placement_tl
      {
        { page }
        { \tl_set:Nn \l__head_placement_end_code_tl
            { \vfil\newpage
              \if@twoside
                \if@openright
                  \null
                  \thispagestyle{empty}%
                \newpage
                \fi
              \fi
              \if@tempswa
                \twocolumn
              \fi
            }
        }
      }
```

For all other values of `placement` we execute `\@afterheading`:

```
      { \tl_set:Nn  \l__head_placement_end_code_tl { \@afterheading } }
   }
```

Then we set up the penalty to use (might be given as a key value).

```
  \__head_determine_penalty:

  \__head_determine_number_typesetting:N #2
```

Next comes the vertical spacing and penalty before the heading. This includes running `\l__head_placement_start_code_tl` if it contains any code, e.g., to start a new page.

```
  \__head_vertical_before_spacing:
```

We are in vmode now and here is the point where we (with tagging) can close a previous Sect structure and open the new one.

```
\UseTaggingSocket{sec/end}{\int_use:N\l__head_level_int}
```

```
\UseTaggingSocket{sec/begin}
                 {{\int_use:N\l__head_level_int}
                  {tag=\UseStructureName{sec/\int_use:N\l__head_level_int}}
                 }
```

Up to this point everything is identical for both display and run-in headings. But from now on they have their own code. We have dealt with argument #1 and #2 so now we can unbundle argument #9 so that it is easier to process downstream

```
  \__head_debug_typeout:n{use~ 'headformat'~instance:~
                          \l__head_headformat_instance_tl }
  \UseTaggingSocket{sec/title/begin}{{\int_use:N\l__head_level_int}{#3}}
```

As long as we don't have a decent interface for the OR we have to use `\@topnewpage` for getting spanning headings (which means we are really using a top float box. That in turn means we have to get all the vertical spacing and so inside this box so there is a lot of back and forth based on the value of `\l__head_column_spanning_bool` for now.

```
  \use:e {
    \bool_if:NT \l__head_column_spanning_bool
         { \exp_not:n {
             \@topnewpage [
             \dim_compare:nNnF \l__head_after_penalty_skip = \maxdimen
                 { \vspace* \l__head_after_penalty_skip }
           }
         }
    \UseInstance{headformat} { \l__head_headformat_instance_tl }
                { \exp_not:o \UnusedTemplateKeys }
                { \exp_not:o { \l__head_text_tl } }
                { \exp_not:o { \l__head_number_tl } }
                { \exp_not:n { #3 } }
                { \exp_not:o { \use_i:nn  #9 } }
                { \exp_not:o { \use_ii:nn  #9 } }

    \bool_if:NT \l__head_column_spanning_bool
         {
           \skip_vertical:N \l__head_after_skip
```

Add the final vspace after the heading and close the `\@topnewpage` which has an optional argument.

```
           ]
         }
  }

  \UseTaggingSocket{sec/title/end}
 %
% --- handle marks, toc-entry, bookmark, nameref, and label
%
  \__head_handle_marks_etc:nnnnn {#4}{#5}{#6}{#7}{#8}
%
% --- post-heading handling (vertical)
```

Restore `secnumdepth`

```
  \int_gset:Nn\c@secnumdepth{\l__head_saved_secnumdepth_tl}
  \par
```

If we have a spanning heading then the vertical skip is handled inside `\@topnewpage` and not here.

```
  \bool_if:NF \l__head_column_spanning_bool
       { \nobreak
         \skip_vertical:N \l__head_after_skip
       }
% --- prepare next paragraph (defaults to \cs{@afterheading}):
%
  \l__head_placement_end_code_tl
%
  \ignorespaces
}
```

`\sec_add_penalty_with_fil:n`
`\sec_add_penalty_with_possible_fil:n`

The command `\sec_add_penalty_with_fil:n` adds a penalty surrounded by stretchable glue like the plain TEX `\filbreak` command. After use it sets `\sec_add_penalty_with_possible_fil:n` back to `\addpenalty`.
They have public names so that they can be used in other heading templates outside of this module.

```
\cs_new_protected:Npn \sec_add_penalty_with_fil:n #1 {
  \__head_debug_typeout:n{apply~ ragged~ bottom~ fil}
```

Next code is straight from `\addpenalty` except that the generated `\penalty` is surrounded by `\vfil` and `\vfilneg`.

```
  \addpenaltywithfil { #1 }
```

Then reset so that next time `\addpenalty` is used again.

```
  \cs_set_eq:NN
    \sec_add_penalty_with_possible_fil:n
    \addpenalty
}
```

By default `\sec_add_penalty_with_possible_fil:n` is the same as `\addpenalty`. If you set it to `\sec_add_penalty_with_fil:n` then it uses the `filbreak` mechanism and afterwards resets itself to its default definition.

```
\cs_set_eq:NN
  \sec_add_penalty_with_possible_fil:n
  \addpenalty
```

(*End of definition for* `\sec_add_penalty_with_fil:n` *and* `\sec_add_penalty_with_possible_fil:n`. *These functions are documented on page* **??**.)

`\__kernel_add_penalty:n`
`\addpenalty`
`\addpenaltywithfil`

Next code is for inclusion in the kernel. It is basically the code from LATEX's `\addpenalty` but with a slightly different argument so that you have to supply `\penalty #1\relax` to mimic `\addpenalty` and `\vfil \penalty #1\vfilneg` to produce a penalty generating a filbreak, i.e., a break where the previous page is ragged bottom.

```
\cs_new_protected:Npn \__kernel_add_penalty:n #1 {
```

Don't add anything at the start of a minipage or when `@nobreak` is set (i.e., directly after a heading).

```
  \mode_if_horizontal:T
    { \mode_if_inner:TF { \@LRmoderr }{ \par } }
  \legacy_if:nF { @minipage }
    { \legacy_if:nF { @nobreak }
```

Do everything in a group so that the temp variables are restored afterwards in case they are used at the outside.

```
567         { \group_begin:
```

The rest is also from `\addpenalty` except that we have to give `\penalty` explicitly as part of the argument.
If there wasn't any previous skip or only a zero-sized one simply set the penalty.

```
568           \skip_set_eq:NN \l_tmpa_skip \tex_lastskip:D
569           \dim_compare:nNnTF \l_tmpa_skip = \c_zero_dim
570             { #1 }
```

Otherwise copy the last skip value also into `\l_tmpb_skip` using TEX's fast direct assignment.

```
571             { \skip_set_eq:NN \l_tmpb_skip \l_tmpa_skip
```

Then add to it the current `\prevdepth` unless it is `-1000pt` (which means it should be suppressed) or if it is unusually large, in which case add `\maxdepth` instead. We remember that value because it will be reused below.

```
572               \dim_set:Nn \l_tmpa_dim
573                  { \dim_compare:nNnTF \prevdepth > \maxdepth
574                      \maxdepth
575                      { \dim_compare:nNnTF \prevdepth = { -1000pt }
576                           \c_zero_dim
577                           \prevdepth
578                      }
579                  }
580               \skip_add:Nn \l_tmpb_skip \l_tmpa_dim
```

This is the amount we have to back up in order to position us vertically where the bottom of the page would be if a natural break would happen.

```
581               \vskip -\l_tmpb_skip
```

That is then the point where we have to add the explicit penalty or the filbreak code.

```
582               #1
```

After the penalty (and after a possible `\vfilneg`) we have to re-add what we back up but that isn't as trivial as one might think.
First we have to move forward by whatever amount we backed up due to `\prevdepth` because going forward the `\prevdepth` will be zero.

```
583               \dim_compare:nNnF \l_tmpa_dim = \c_zero_dim
584                 { \vskip \l_tmpa_dim }
```

Finally we have to reinsert what we saved as `\lastskip`, because that is what any following `\addvspace` should see to make its calculations from.

```
585               \vskip \l_tmpa_skip
586             }
587           \group_end:
588         }
589     }
590 }
```

Reimplement `\addpenalty` using the above macro:

```
\cs_set_protected:Npn \addpenalty #1 {
  \__kernel_add_penalty:n { \penalty #1 \scan_stop: }
}
```

And a new command to add a penalty with some `\vfil` around it. They cancel each other if no break is taken.

```
\cs_new_protected:Npn \addpenaltywithfil #1 {
  \__kernel_add_penalty:n { \vfil \penalty #1 \vfilneg }
}
```

(*End of definition for* `\__kernel_add_penalty:n`*,* `\addpenalty`*, and* `\addpenaltywithfil`*. These functions are documented on page* ??.)

`\if@openright`

```
\AddToHook{begindocument}{
  \ifcsname if@openright\endcsname
  \else
```

Need to hide this a little if it is in fact already defined!

```
    \expandafter\newif\csname if@openright\endcsname
  \fi
}
```

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

`heading runin` (*templ.*) The `runin` template is very similar to the `display` one.

```
\DeclareTemplateCode{heading}{runin}{9}
 {
  , name           = \l__head_name_tl
  , level          = \l__head_level_int
% next two not yet used
  , parent-name    = \l__head_pname_tl
  , reset-counter  = \l__head_reset_cnt_tl

  , force-unnumbered = \l__head_unnumbered_bool
```

It wouldn't make sense to have page placement with a run-in heading since there would be nothing to run into.

```
  , placement      = {
        page   = \typeout{ ^^JA~ runin~ page~ placement~ heading~makes~no~sense
                           (top~used)}
                 \tl_set:Nn \l__head_placement_tl { top }
       ,top    = \__head_debug_typeout:n{ A~ top~ heading }
                 \tl_set:Nn \l__head_placement_tl { top }
       ,normal = \__head_debug_typeout:n{ A~ normal~ heading }
                 \tl_set:Nn \l__head_placement_tl { normal }
       ,ragged = \__head_debug_typeout:n{ A~ normal~ heading~ (ragged~ style) }
                 \tl_set:Nn \l__head_placement_tl { ragged }
                   }

  , column-spanning = \l__head_column_spanning_bool

  , mark-cmd       = \__head_mark_cmd:n
```

UFi:this types out messages for all run-in headers! Therefore disabled for now

In a run-in heading you can't set up paragraph indentation of the following paragraph (since that one is "run in". But we accept the key and just spit out a warning.

```
  , para-indent    =  {
                       ,true  = %\typeout{para-indent~ setting~ ignored}
                       ,false = %\typeout{para-indent~ setting~ ignored}
                      }
  , before-vspace  = \l__head_before_skip
  , penalty        = \l__head_penalty_int
  , after-penalty-vspace  = \l__head_after_penalty_skip
  , after-hspace   = \l__head_after_skip
  , placement-start-code = \l__head_placement_start_code_tl
  , placement-end-code   = \l__head_placement_end_code_tl

  , text          = \l__head_text_tl
  , punct         = \l__head_punct_tl

  , headformat-instance = \l__head_headformat_instance_tl

  , heading-decls  = \l__head_heading_decls_tl

  , text-decls     = \l__head_text_decls_tl
  , number-decls   = \l__head_number_decls_tl
  , title-decls    = \l__head_title_decls_tl
  , subtitle-decls = \l__head_subtitle_decls_tl
  , quote-decls    = \l__head_quote_decls_tl

  , heading-format  = \__head_heading_format:n
  , text-format     = \__head_text_format:n
  , number-format   = \__head_number_format:n
  , punct-format    = \__head_punct_format:n
  , title-format    = \__head_title_format:n
  , subtitle-format = \__head_subtitle_format:n
  , quote-format    = \__head_quote_format:n
%
  , number-tag      = \l__head_number_tag_tl

  , contents-extra = \l__head_contents_extra_tl
}
{ \__head_show_arguments:nnnnnnnnn
      {#1}{#2}{#3}{#4}{#5}{#6}{#7}{#8}{#9}
```

store the value of `secnumdepth` so that we can change it locally.

```
  \tl_set:Ne\l__head_saved_secnumdepth_tl{\int_use:N\c@secnumdepth}
```

define `\theheading`

```
  \tl_set_eq:Nc \theheading { the \l__head_name_tl }
  \SetCurrentKnownTemplateKeys{#1}
```

For now we don't support column-spanning in this template.

```
  \bool_if:NT \l__head_column_spanning_bool
    { \__head_debug_typeout:n {column-spanning~ is~
        not~ (yet)~ supported~ for~ run-in~ headings!}
      \bool_set_false:N \l__head_column_spanning_bool
    }
```

```
  \tl_if_empty:oT \l__head_placement_start_code_tl
    { \str_case:Vn \l__head_placement_tl
      {
        { top }   { \tl_set:Nn \l__head_placement_start_code_tl {\clearpage } }

        { ragged }
        { \cs_set_eq:NN \sec_add_penalty_with_possible_fil:n
                        \sec_add_penalty_with_fil:n
        }
      }
```

All other cases want an empty `\l__head_placement_start_code_tl` so nothing to do.

```
%      { \tl_clear:N \l__head_placement_start_code_tl }
    }
%
```

Nothing at all to do (for now) for `\l__head_placement_end_code_tl`, it should by default be empty in all heading placement. But maybe we end up supporting further placements, so …

```
%  \tl_if_empty:oT \l__head_placement_end_code_tl
%   { \str_case:VnF \l__head_placement_tl
%      {
%        { top } { \tl_clear:N \l__head_placement_end_code_tl }
%      }
%      { \tl_clear:N \l__head_placement_end_code_tl }
%   }

  \__head_determine_penalty:
  \__head_determine_number_typesetting:N #2
  \__head_vertical_before_spacing:
```

We are in vmode now and here is the point where we (with tagging) can close a previous Sect structure and open the new one. We also have to initialize the change in the para tagging here as run-in titles typeset the heading in `\everypar`.

```
  \UseTaggingSocket{sec/end}{\int_use:N\l__head_level_int}
  \UseTaggingSocket{sec/begin}
                   {{\int_use:N\l__head_level_int}
                    {tag=\UseStructureName{sec/\int_use:N\l__head_level_int}}
                   }
  \UseTaggingSocket{sec/title/init}{\int_use:N\l__head_level_int}
  \def \@svsechd {
      \__head_debug_typeout:n{use~ 'headformat'~instance:~
                              \l__head_headformat_instance_tl }
      \use:e {
        \UseInstance{headformat} { \l__head_headformat_instance_tl }
                 { \exp_not:o \UnusedTemplateKeys }
                 { \exp_not:o { \l__head_text_tl } }
                 { \exp_not:o { \l__head_number_tl } }
                 { \exp_not:n { #3 } }
                 { \exp_not:o { \use_i:nn  #9 } }
                 { \exp_not:o { \use_ii:nn  #9 } }
      }
      \__head_handle_marks_etc:nnnnn {#4}{#5}{#6}{#7}{#8}
```

Restore `secnumdepth`

```
      \int_gset:Nn\c@secnumdepth{\l__head_saved_secnumdepth_tl}
  }
  \@nobreakfalse
  \global\@noskipsectrue
  \everypar{%
    \if@noskipsec
       \global\@noskipsecfalse
       {\setbox\z@\lastbox}
       \clubpenalty\@M
       \@svsechd
       \unskip
```

This tagging socket starts the "paragraph" after the run-in heading

```
       \UseTaggingSocket{sec/title/split}
       \skip_horizontal:N \l__head_after_skip
    \else
      \clubpenalty \@clubpenalty
      \everypar{}%
    \fi
  }
```

— prepare next paragraph (does nothing by default)

```
  \l__head_placement_end_code_tl
  \ignorespaces
}
```

11.7.5 headformat templates code

`headformat display` (*templ.*) This template creates a headformat where the number is on a line on its own.

```
\DeclareTemplateCode{headformat}{display}{6}
  % args: keys,text,number,title,subtitle,quotation
 {
  , heading-indent    = \l__head_heading_indent_dim
  , order             = \l__head_order_clist
  , separator-a       = name {l__head_separator-a_tl}
  , separator-b       = name {l__head_separator-b_tl}
  , separator-c       = name {l__head_separator-c_tl}
  , separator-d       = name {l__head_separator-d_tl}
  , left-margin       = \l__head_left_margin_dim
  , right-margin      = \l__head_right_margin_dim
 }
 {
  \__head_show_arguments:nnnnnn {#1}{#2}{#3}{#4}{#5}{#6}
```

First evaluate any key setting done by the user (normally supplied from the main heading instance.

```
  \SetTemplateKeys{headformat}{display}{#1}
  \group_begin:
```

\parboxes can be used within the title of a heading (or within its layout). minipage environments currently do not work if used directly in the title because purify can't handle that environment. But it can be used as part of the layout definition which is why we also add a suitable setting for it.

```
    \AssignTaggingSocketPlug{parbox/before}{heading}
    \AssignTaggingSocketPlug{minipage/before}{heading}
    \AssignTaggingSocketPlug{para/restore}{heading}
    \tagpdfparaOff
%
    \tl_set:Nn \l__head_text_tl {#2}
    \tl_set:Nn \l__head_number_tl {#3}
    \tl_set:Nn \l__head_title_tl  {#4}
%
```

To allow positioning the whole heading horizontally we use a block instance with special settings and pass it values for the left and right margin. This unnecessarily sets a vertical space (made 0pt) and a penalty (made 10000) at the beginning, but this is a simple way to achieve our goal. TODO: revisit at some point in the future and perhaps replace with a different approach.

```
    \UseInstance{block}{adjustwidth}
       {   left-margin  = \l__head_left_margin_dim
         , right-margin = \l__head_right_margin_dim }
```

TODO: revisit if \normalcolor is needed, if yes, use as \SaveLastSkip \normalcolor \RestoreLastSkip.

```
    \normalfont
    \interlinepenalty \@M
    \l__head_heading_decls_tl
```

If there is a number and thus no fixed text, the link target should be before the number. We use for now the same place in the unnumbered case. TODO: If we put the target here we do not know the height of the line and the target is perhaps not high enough. And for unnumbered chapter it is perhaps too high. Check!

```
    \bool_if:NTF \l__head_unnumbered_bool
      { \dim_compare:nNnTF  \l__head_heading_indent_dim < \c_zero_skip
          { \skip_horizontal:N \l__head_heading_indent_dim
            \MakeLinkTarget[\l__head_name_tl]{}
          }
          { \MakeLinkTarget[\l__head_name_tl]{}
            \skip_horizontal:N \l__head_heading_indent_dim
          }
      }
      { \dim_compare:nNnTF  \l__head_heading_indent_dim < \c_zero_skip
          { \skip_horizontal:N \l__head_heading_indent_dim
            \MakeLinkTarget{\l__head_name_tl}
          }
          { \MakeLinkTarget{\l__head_name_tl}
            \skip_horizontal:N \l__head_heading_indent_dim
          }
      }
```

At this point we know for sure that we are on the page/column where the heading will finally appear so we can record this position.

```
    \socket_use:n {heading/record}

    \__head_heading_format:n {
      \template_process_order_clist:nnn
         { head }{ order }
         { number, text, punct, separator-a,
           separator-b, separator-c, separator-d, title, subtitle }
    }
    \par
  \group_end:
}
```

A special instance we used above to position the heading horizontally.

```
\DeclareInstance{block}{adjustwidth}{std}{
    para-indent        = \parindent
  , para-vspace        = \parskip
  , begin-vspace       = 0pt
  , end-vspace         = 0pt
  , begin-penalty      = 10000,
  , end-penalty        = 10000,
  , begin-extra-vspace = 0pt,
  , end-extra-vspace   = 0pt
}
```

__head_column_vbox_adjusted:nnn This macro is currently not used. Might vanish (or may get used at a later point in time).

```
\cs_new_protected:Npn \__head_column_vbox_adjusted:nnn #1#2#3 {
  \group_begin:
    \skip_zero:N \lineskip
    \hbox_overlap_right:n {
      \hspace { #1 }
      \vbox:n { \dim_sub:Nn \hsize { #1 + #2 } #3 }
    }
  \group_end:
}
```

(*End of definition for* `\__head_column_vbox_adjusted:nnn`.)

The the plugs we assigned to the tagging sockets in the template need to be defined.

```
\NewTaggingSocketPlug{minipage/before}{heading}
  { \tag_mc_end_push:
    \bool_if:NT \l__tag_para_bool {\tag_struct_end:}
    \tag_struct_begin:n{tag=\UseStructureName{sec/minipage}}
  }

\NewTaggingSocketPlug{parbox/before}{heading}
  { \tag_mc_end_push:
    \bool_if:NT \l__tag_para_bool {\tag_struct_end:}
    \tag_struct_begin:n{tag=\UseStructureName{sec/parbox}}
  }

\NewTaggingSocketPlug{para/restore}{heading}
```

resolve

We have only structure names on heading levels for this, so for now I use `heading-fragment` directly. Should probably become `sec/fragment`.

```
{ \AssignStructureRole{para/textblock}{heading-fragment}
  \bool_set_true:N\l__tag_para_flattened_bool
  \tagpdfparaOn
}
```

`headformat hang` (*templ.*) This template creates a heading with a hanging number. The arguments are user keys, fixed text, number, title, subtitle, quotation.

```
\DeclareTemplateCode{headformat}{hang}{6}
 {
  , heading-indent    = \l__head_heading_indent_dim
  , left-margin       = \l__head_left_margin_dim
  , right-margin      = \l__head_right_margin_dim
  , order-hang        = name {l__head_order-hang_clist}
  , order             = \l__head_order_clist
  , separator-a       = name {l__head_separator-a_tl}
  , separator-b       = name {l__head_separator-b_tl}
  , separator-c       = name {l__head_separator-c_tl}
  , separator-d       = name {l__head_separator-d_tl}
 }
 {
  \__head_show_arguments:nnnnnn {#1}{#2}{#3}{#4}{#5}{#6}
```

First evaluate any key setting done by the user (normally supplied from the main heading instance).

```
  \SetTemplateKeys{headformat}{hang}{#1}

  \group_begin:
%
    \AssignTaggingSocketPlug{parbox/before}{noop}
    \AssignTaggingSocketPlug{parbox/after}{noop}
    \AssignTaggingSocketPlug{para/restore}{noop}
    \tagpdfparaOff
%
    \tl_set:Nn \l__head_text_tl {#2}
    \tl_set:Nn \l__head_number_tl {#3}
    \tl_set:Nn \l__head_title_tl  {#4}
%

    \normalfont
    \interlinepenalty \@M
    \l__head_heading_decls_tl
    \noindent
```

At this point we know for sure that we are on the page/column where the heading will finally appear so we can record this position.

```
    \socket_use:n {heading/record}

    \setbox\@tempboxa\hbox{{
```

The link target should be at the left text margin, or, if the section is moved into the margin, at the left of the number.

```
      \bool_if:NTF \l__head_unnumbered_bool
```

```
        {
          \MakeLinkTarget[\l__head_name_tl]{}
        }
        {
          \MakeLinkTarget{\l__head_name_tl}
          \template_process_order_clist:nnn
              { head }{ order-hang }
              { number, text, punct, separator-a,
                separator-b, separator-c, separator-d, title, subtitle }
        }
    }}
```

As this template sets up a hang indentation we need to set up a special parshape. Using `\hangindent` as was done in LaTeX 2_ε doesn't work if we also want to supply left and right margins on top. Instead we need to combine the appropriate hanging indentation with these margin settings and construct a suitable parshape from that. This is what the next lines of code do. The hanging indentation we want is the width of `\@boxtempa` constructed above.

```
    %    \hangindent \box_wd:N\@tempboxa
    \dim_add:Nn \l__head_left_margin_dim { \box_wd:N \@tempboxa }
    \dim_set:Nn \linewidth { \hsize - \l__head_left_margin_dim
                                    - \l__head_right_margin_dim }

    \__head_heading_format:n {
      \tex_parshape:D 1 ~ \l__head_left_margin_dim \linewidth
```

At this point we are at the very left of the paragraph (indented by whatever has been specified by the parshape). We now shift by `\l__head_heading_indent_dim` and then insert the box into the area to the left of that point. Thus if the `\l__head_heading_indent_dim` is zero all further lines are indented exactly by the width of the box.

```
      \skip_horizontal:N  \l__head_heading_indent_dim
      \hbox_overlap_left:n { \box_use_drop:N \@tempboxa }
```

Then we continue with the rest of the heading material by processing the clist in the `order` key.

```
      \template_process_order_clist:nnn
           { head }{ order }
           { number, text, punct, separator-a,
             separator-b, separator-c, separator-d,
             title, subtitle }
    }
    \par
  \group_end:
}
```

`headformat runin` (*templ.*) This template creates a heading with a run-in title. Arguments are key-value, number, title, subtitle, quotation.

```
\DeclareTemplateCode{headformat}{runin}{6}
    % args: keys,text,number,title,subtitle,quotation
 {
  , heading-indent    = \l__head_heading_indent_dim
  , order             = \l__head_order_clist
```

```
  , separator-a       = name {l__head_separator-a_tl}
  , separator-b       = name {l__head_separator-b_tl}
  , separator-c       = name {l__head_separator-c_tl}
  , separator-d       = name {l__head_separator-d_tl}
}
 {
  \__head_show_arguments:nnnnnn {#1}{#2}{#3}{#4}{#5}{#6}
```

First evaluate any key setting done by the user (normally supplied from the main heading instance.

```
  \SetTemplateKeys{headformat}{runin}{#1}

  \group_begin:
%
    \AssignTaggingSocketPlug{parbox/before}{noop}
    \AssignTaggingSocketPlug{parbox/after}{noop}
    \AssignTaggingSocketPlug{para/restore}{noop}
    \tagpdfparaOff
%
    \tl_set:Nn \l__head_text_tl {#2}
    \tl_set:Nn \l__head_number_tl {#3}
    \tl_set:Nn \l__head_title_tl  {#4}

    \normalfont

    \interlinepenalty \@M
    \l__head_heading_decls_tl
```

Setting `\interlinepenalty` makes little sense I think (but that's the way it was in LaTeX 2_ε)

We must avoid that the sep between number and title is used in the unnumbered case. So we test with the boolean.

```
    \bool_if:NTF \l__head_unnumbered_bool
      { \dim_compare:nNnTF  \l__head_heading_indent_dim < \c_zero_skip
         { \skip_horizontal:N \l__head_heading_indent_dim
           \MakeLinkTarget[\l__head_name_tl]{}
         }
         { \MakeLinkTarget[\l__head_name_tl]{}
           \skip_horizontal:N \l__head_heading_indent_dim
         }
      }
      { \dim_compare:nNnTF  \l__head_heading_indent_dim < \c_zero_skip
         { \skip_horizontal:N \l__head_heading_indent_dim
           \MakeLinkTarget{\l__head_name_tl}
         }
         { \MakeLinkTarget{\l__head_name_tl}
           \skip_horizontal:N \l__head_heading_indent_dim
         }
      }
```

At this point we know for sure that we are on the page/column where the heading will finally appear so we can record this position.

```
    \socket_use:n {heading/record}
```

```
    \__head_heading_format:n {
      \template_process_order_clist:nnn
           { head }{ order }
           { number, text, punct, separator-a,
             separator-b, separator-c, separator-d,
             title, subtitle }
    }
  \group_end:
}
```

11.7.6 Internal commands used by the template code

__head_determine_penalty:

```
\cs_new:Npn \__head_determine_penalty: {
```

If a penalty was specified use it, otherwise use `\@secpenalty`.

```
  \int_compare:nNnT \l__head_penalty_int = \c_max_int
    { \int_set:Nn \l__head_penalty_int \@secpenalty }
}
```

(*End of definition for* `\__head_determine_penalty:`.)

__head_determine_number_typesetting:N

```
\cs_new:Npn \__head_determine_number_typesetting:N #1 {
```

If `force-unnumbered` was set to true (i.e., `\l__head_unnumbered_bool`) we don't do numbering. If that is not the case then using or suppressing a heading number depends on the heading level compared to the document value of `\c@secnumdepth`. If that doesn't suppress the number then an explicit key or a star form might still have suppressed it.

```
  \bool_if:NF \l__head_unnumbered_bool
     { \bool_set:Nn \l__head_unnumbered_bool
        { \bool_lazy_or_p:nn
          { \int_compare_p:nNn \l__head_level_int > \c@secnumdepth }
          { \bool_if_p:N #1 }
        }
     }
```

If we aren't producing a heading with a number we set `\c@secnumdepth` to a low number (it is reset at the end of the template code)

```
  \bool_if:NTF \l__head_unnumbered_bool
    { \int_gset:Nn\c@secnumdepth{-99}
```

and we set `\l__head_number_tl`, `\l__head_text_tl`, and `\l__head_punct_tl` to do nothing.

```
      \tl_set:Nn \l__head_number_tl { \NoValue }
      \tl_set:Nn \l__head_text_tl { \NoValue }
      \tl_set:Nn \l__head_punct_tl  { \NoValue }
    }
```

Otherwise the heading counter is incremented and a formatted version of the number plus any following (or preceding) space is stored in `\l__head_number_tl`. We use the kernel version of `\refstepcounter` as anchors are handled elsewhere.

```
    {
```

```
      \@kernel@refstepcounter{ \l__head_name_tl }
      \tl_set:Nn \l__head_number_tl { \theheading }
    }
}
```

(*End of definition for* `\__head_determine_number_typesetting:N`.)

`\__head_vertical_before_spacing:`

```
\cs_new:Npn \__head_vertical_before_spacing: {

  \dim_compare:nNnF \l__head_after_penalty_skip = \maxdimen
    { \skip_add:Nn \l__head_before_skip        \topskip
      \skip_sub:Nn \l__head_after_penalty_skip \topskip
    }

  \tl_if_blank:VTF \l__head_placement_start_code_tl
     { %  \if@noskipsec \leavevmode \fi \par       % <--- too late
       \if@nobreak
         \everypar{}
       \else
```

Normally the penalty is added with `\addpenalty` but in some cases (`placement=ragged`) we want to use a variant of `\filbreak` instead. So we use `\sec_add_penalty_with_possible_fil:n` which is either equal to `\addpenalty` or to `\sec_add_penalty_with_fil:n` depending on the setup.

```
         \sec_add_penalty_with_possible_fil:n
            \l__head_penalty_int
         \addvspace  \l__head_before_skip
```

The `\vspace*` inserts a rule, we insert therefore the skip only if it is different to zero.

```
         \dim_compare:nNnF \l__head_after_penalty_skip = \maxdimen
             {
               \vspace* \l__head_after_penalty_skip
             }
       \fi
     }
     {
       \l__head_placement_start_code_tl
```

If `\l__head_placement_start_code_tl` holds code, we assume that it handles pagination, e.g., a `\clearpage`, etc. We therefore only add the skip that would follow the penalty.
And we do nothing at all when we have a spanning heading. Then this vspace is done inside `\@topnewpage`.

```
       \bool_if:NF \l__head_column_spanning_bool
         { \dim_compare:nNnF \l__head_after_penalty_skip = \maxdimen
                { \vspace* \l__head_after_penalty_skip }
         }
     }
}
```

(*End of definition for* `\__head_vertical_before_spacing:`.)

\addcontentslinebookmark
\addcontentslinebookmarkOff
\addcontentslinebookmarkOn

We want to be able to control bookmarks independently from the toc entries, and also want to disable a bookmark by setting it to empty. For this we need some command to handle the bookmark command.

```
\newcommand\addcontentslinebookmark[3]{}
\newcommand\addcontentslinebookmarkOff{}
\newcommand\addcontentslinebookmarkReset{}
\providecommand\texorpdfstring[2]{#1}
```

This can go once hyperref is updated (ufi,2026-04-21):

```
\AddToHook{package/hyperref/after}
 {
   \RenewCommandCopy\addcontentslinebookmark
                     \Hy@addcontentsline@addbookmark
   \renewcommand\addcontentslinebookmarkOff
     { \ifHy@bookmarks
       \let\addcontentslinebookmarkReset
           \Hy@bookmarkstrue
       \else
         \let\addcontentslinebookmarkReset\relax
       \fi
       \Hy@bookmarksfalse
     }
 }
```

(*End of definition for* `\addcontentslinebookmark`*,* `\addcontentslinebookmarkOff`*, and* `\addcontentslinebookmarkOn`*. These functions are documented on page* ??.)

__head_handle_marks_etc:nnnnn

Arg 1: toc entry, arg 2: mark, arg 3: bookmark, arg 4: nameref, arg 5: label code

```
\cs_new:Npn \__head_handle_marks_etc:nnnnn #1#2#3#4#5 {
%
  \tl_set:Nn\@currentlabelname{#4}
  \IfBlankF {#2}
     { \__head_mark_cmd:n { #2 } }
  \IfBlankTF {#1}
```

If the toc entry is empty the bookmarks must be set without `\addcontentsline`.

```
    { \IfBlankF{#3}
       {
        \addcontentslinebookmark{toc}{\l__head_name_tl}
          {
            \bool_if:NF \l__head_unnumbered_bool
              {
                \protect\numberline{ \use:c{ the \l__head_name_tl } }
              }
            #3
          }
       }
    }
```

If the bookmark entry is empty we must disable the bookmarks locally before the `\addcontentsline` and reenable afterwards. We do not check if they are already disabled, perhaps later.

```
    { \IfBlankT{#3}{\addcontentslinebookmarkOff}
```

```
      \addcontentsline{toc}{ \l__head_name_tl }
        {
          \bool_if:NF \l__head_unnumbered_bool
            {
              \protect\numberline{ \use:c{ the \l__head_name_tl } }
            }
```

We use `\texorpdfstring` to separate toc and bookmark text.

```
          \texorpdfstring{#1}{#3}
        }
      \addcontentslinebookmarkReset
    }
```

perhaps this should have a hook for packages

Some headings (like `\chapter`) also want to write stuff to other contents files like `.lot` or `.lof`.

```
    \l__head_contents_extra_tl
```

We can always run the label code (it might be empty)

```
    \__head_debug_typeout:n{--->~label(s):~ \exp_not:n{#5}}
    #5
}
```

(*End of definition for* `\__head_handle_marks_etc:nnnnn`.)

11.8 Support for legacy classes and packages

If we define heading commands in the kernel (or in the tagging code) using

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

then this gets overwritten by every class right now.

If we redeclare them after the class was loaded then we overwrite the layout of legacy classes (done, for example, with `\@startsection`). New classes that use the above interface would be fine though, as long as we have declared the instances before the class is loaded.

So to make legacy classes work with their existing layout, we should not (ever) overwrite `\section` but let the class definition call `\@startsection` which would then set up suitable instances and only after that call `\ParseLaTeXeHeading`.

However, that would mean a "new" class like ltx-article would need to add the above definition and declare or edit the corresponding instances, instead of just declaring or adding the instances.

A perhaps better alternative could be to delay the definition of `\section` and friends until after the class got loaded and then take a peak at `\section` as defined by the class and if it contains a `\@startsection` call, leave it alone and otherwise overwrite it. And if the class hasn't defined `\section` (because it is a new class) declare it. Of course that means `\section` would then be available with every class even if it was never meant to contain headings.

But while writing this up, I start to think it is best if a new class defines both the document interface (i.e., the above command) as well as the layout (declare the instances) and the kernel does neither. It has been this way before and it is consistent, so why change.

The situation with headings defined via `\secdef` is worse: we don't have a nice handle as with `\@startsection` so there is no real way other than through static analysis to set up such a heading in the new template style. So all that is possible (I think) is that after the class has been loaded, we look if the usual candidates (`\part` and `\chapter`) have been defined and overwrite them with a standard layout. That would make the class tagging aware but, of course, would likely change the layout.

The current implementation

- provides instance for the heading commands of the standard classes;
- declares the heading commands for the standard classes with the new interfaces through class hooks;
- other classes call the adapted `\@startsection`; other headings command like `\part` or `\chapter` are not handled and must be setup individually.

11.8.1 Instances (sample/default definitions)

`heading part` (*inst.*) An instance suitable for book and report. An instance suitable for article is above in the hook.

```
\DeclareInstance{heading}{part}{display}
{
  , name          = part
  , level         = -1
  , placement     = page
  , text          = \partname
  , number-format = \thepart
  , heading-decls = \centering\bfseries\huge
  , title-decls   = \Huge
  , headformat-instance = part
  , mark-cmd      = \partmark {#1}
}
```

`heading chapter` (*inst.*)

```
\DeclareInstance{heading}{chapter}{display}
{
  , name           = chapter
  , level          = 0
  , placement      = top
  , column-spanning = true
  , after-penalty-vspace = 60pt  % that's 50pt plus \topskip for compatibility
  , after-vspace   = 40pt
  , text           = \@chapapp
  , number-format  = \thechapter
  , heading-decls  = \raggedright \parindent0pt \bfseries \huge
  , title-decls    = \Huge
  , headformat-instance = chapter
  , mark-cmd       = \chaptermark {#1}
  , contents-extra = \addtocontents{lof}{\addvspace{10pt}}
                     \addtocontents{lot}{\addvspace{10pt}}
}
```

`\@chapapp` To improve compatibility with classes that use `\secdef` but don't provide a definition for `\@chapapp` used in the `text` key above, we define this command if necessary. Otherwise such a class would error if `\chapter` is used.

```
\AtBeginDocument{
  \cs_if_exist:NF \@chapapp { \cs_new:Npn \@chapapp {Chapter} } }
```

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

`heading section` (*inst.*)

```
\DeclareInstance{heading}{section}{display}
{
  , name          = section
  , level         = 1
  , mark-cmd      = \sectionmark {#1}
  , before-vspace = 3.5ex plus 1ex minus .2ex
  , after-vspace  = 2.3ex plus .2ex
  , heading-decls = \normalfont\Large\bfseries
  , headformat-instance = section
}
```

`heading subsection` (*inst.*)

```
\DeclareInstance{heading}{subsection}{display}
{
  , name          = subsection
  , parent-name   = section
  , level         = 2
  , mark-cmd      = \subsectionmark {#1}
  , before-vspace = 3.25ex plus 1ex minus .2ex
  , after-vspace  = 1.5ex plus .2ex
  , heading-decls = \normalfont\large\bfseries
  , headformat-instance = subsection
}
```

`heading subsubsection` (*inst.*)

```
\DeclareInstance{heading}{subsubsection}{display}
{
  , name          = subsubsection
  , parent-name   = subsection
  , level         = 3
  , before-vspace = 3.25ex plus 1ex minus .2ex
  , after-vspace  = 1.5ex plus .2ex
  , heading-decls = \normalfont\normalsize\bfseries
  , headformat-instance = subsubsection
}
```

`heading paragraph` (*inst.*)

```
\DeclareInstance{heading}{paragraph}{runin}
{
  , name          = paragraph
  , parent-name   = subsubsection
  , level         = 4
  , before-vspace = 3.25ex \@plus1ex \@minus.2ex
  , after-hspace  = 1em
```

```
  , heading-decls = \normalfont\normalsize\bfseries
  , mark-cmd      =  \paragraphmark {#1}
  , headformat-instance = paragraph

}
```

heading subparagraph (*inst.*)

```
\DeclareInstance{heading}{subparagraph}{runin}
{
  , name          = subparagraph
  , parent-name   = paragraph
  , level         = 5
  , before-vspace = 3.25ex \@plus1ex \@minus .2ex
  , after-hspace  = 1em
  , heading-decls = \normalfont\normalsize\bfseries
  , mark-cmd      =  \subparagraphmark {#1}
  , headformat-instance = subparagraph

}
```

headformat part (*inst.*)
headformat chapter (*inst.*)

The `headformat` instances for part and chapter use the `display` template with identical settings by default, so one is made a copy of the other to speed up loading.

```
\DeclareInstance{headformat}{part}{display}
{
  , heading-indent    = 0pt
  , separator-a       = \nobreakspace                % between text and number
  , separator-b       = \par \nobreak \vspace{20pt}  % between label block and title
}

\DeclareInstance{headformat}{chapter}{part}
```

headformat std (*inst.*)
headformat section (*inst.*)
headformat subsection (*inst.*)
headformat subsubsection (*inst.*)

Similarly, by default the instances for section, subsection, and subsubsection are all using the `hang` template and the same key values as the `std` instance, so they are all made a copy of that one.

```
\DeclareInstance{headformat}{std}{hang}
{
  , heading-indent = 0pt
}

\DeclareInstance{headformat}{section}{std}
\DeclareInstance{headformat}{subsection}{std}
\DeclareInstance{headformat}{subsubsection}{std}
```

headformat paragraph (*inst.*)
headformat subparagraph (*inst.*)

In contrast, paragraph and subparagraph differ even though both use the `runin` template.

```
\DeclareInstance{headformat}{paragraph}{runin}
{
  , heading-indent = 0pt
}

\DeclareInstance{headformat}{subparagraph}{runin}
{
  , heading-indent = \parindent
}
```

adformat section-special (*inst.*)

A special headformat instance for testing. It can be used with `\section[headformat-instance=section-special]{bla}`

```
\DeclareInstance{headformat}{section-special}{hang}
{
  , heading-indent = 4em
%%%%  , title-format   = {#1!}   % no longer
}
```

11.8.2 New \@startsection

\@startsection

To support legacy classes that implement headings through a call to `\@startsection` we redefine this command to generate a heading instance from the arguments of `\@startsection` if it doesn't yet exist and then use this instance to typeset the heading. NOTE: To handle legacy definition like `{\normalfont\Large\bfseries\MakeUppercase}` in the last argument it copies this argument both to `heading-decls` and to `title-format`.

```
\DeclareDocumentCommand \@startsection {mmmmmm s ={shorttitle}o m}{
```

If there already exists an instance `#1-@startsection` then arguments 2–6 are ignored and we simply call that instance. Otherwise we go through the process to set it up. This allows classes, packages and authors to create and overwrite definitions with `\@startsection`. But after a call to a command using the `\@startsection` the instances are created. It is therefore not possible to redefine the command after a first use or to create two commands using the same level, e.g. `\section` and `\specialsection`. Such setups should use the new interfaces to declare heading commands.

```
  \IfInstanceExistsF{heading}{#1-@startsection}{

    \__head_debug_typeout:n{Info:~ setting~ up~ instances~ for~
                        legacy~ \string\@startsection}
```

`\@startsection` supported the use of a macro with one argument as the last token in its ⟨*style*⟩ argument, for example, `\MakeUppercase` to make the whole title uppercase. To support that we need to take a look at `#6` and if that ends in such a macro split it off, so that we can put the first tokens into `heading-decls` and this last token into `title-format`. This is what the next call prepares for:

```
    \__head_prep_decls_and_format_from_tl:nNN {#6}
       \l__head_heading_decls_tl
       \l__head_title_format_tl
```

In the LaTeX 2_ε logic a negative #4 means we do not indent the following paragraph … It is okay to change any `em` or `ex` specifications to real `pt` values here, because this code is executed in the same place where `\@startsection` is normally executed and inside the original definitions such assignments happen as well, before there is any font change happening for #4 and #5.

```
    \@tempskipa #4\relax
    \@afterindenttrue
    \ifdim \@tempskipa <\z@
      \@tempskipa -\@tempskipa
      \@afterindentfalse
    \fi
```

… and a negative #5 means we should produce a run-in heading.

```
    \@tempskipb #5\relax
    \ifdim \@tempskipb>\z@
      \use:e {
        \DeclareInstance{heading}{#1-@startsection}{display}{
          , name          = #1
          , level         = #2
          , mark-cmd      = \exp_not:c {#1mark} {##1}
          , before-vspace = \the\@tempskipa
          , after-vspace  = \the\@tempskipb
          , para-indent   = \if@afterindent true \else false\fi
```

Argument #6 contains the declarations for the whole heading but it is allowed that the last token is a macro with one argument that receives the heading text as its argument. This is why we have split off the last token above, in case it contained a macro with one argument. That token is then used as the `title-format`. We end this key value in `\par` if we are in a display instance.

```
          , heading-decls = \exp_not:o \l__head_heading_decls_tl
          , title-format  = \exp_not:o \l__head_title_format_tl {##1}
                            \exp_not:N \par
          , headformat-instance = #1-@startsection
        }}
        \DeclareInstance{headformat}{#1-@startsection}{hang}{heading-indent = #3}
```

We can expect that the instance `#1-@startsection` is not set up by the user if `#1-@startsection` isn't, so no check.

```
    \else
      \@tempskipb=-\@tempskipb
      \use:e {
       \DeclareInstance{heading}{#1-@startsection}{runin}{
          , name          = #1
          , level         = #2
          , mark-cmd      = \exp_not:c {#1mark} {##1}
          , before-vspace = \the\@tempskipa
          , after-hspace  = \the\@tempskipb
          , heading-decls = \exp_not:o \l__head_heading_decls_tl
```

In a run-in instance `title-format` should not end in `\par`!

```
          , title-format  = \exp_not:o \l__head_title_format_tl {##1}
          , headformat-instance = #1-@startsection
        }}
      \DeclareInstance{headformat}{#1-@startsection}{runin}{heading-indent = #3}
      \fi
    }
    \ParseLaTeXeHeading {#1-@startsection}{#7}{#8}{#9}
}
```

(*End of definition for* `\@startsection`. *This function is documented on page 21.*)

`\__head_prep_decls_and_format_from_tl:nNN`

Examine #1. If the last token is a macro with one argument put it in #3 and the remainder of #1 into #2. Otherwise, put `\use:n` in #3 and all of #1 in #2.

```
\cs_new:Npn \__head_prep_decls_and_format_from_tl:nNN #1#2#3
  { \tl_set:Nn #3 { \use:n }
```

```
    \exp_args:Ne
    \__head_prep_decls_and_auxi:nnNN
        { \tl_reverse:n {#1} } {#1} #2 #3
  }

\cs_new:Npn \__head_prep_decls_and_auxi:nnNN #1#2#3#4
  { \tl_if_head_is_N_type:nTF {#1}
      { \__head_prep_decls_and_auxii:nNwNN {#2} #1 \q_stop #3 #4 }
      { \tl_set:Nn #3 {#2} }
  }

\cs_new:Npn \__head_prep_decls_and_auxii:nNwNN #1#2#3 \q_stop #4 #5
  { \token_if_macro:NTF #2
      { \exp_args:Ne
        \__head_prep_decls_and_auxiii:nnNnNN
          { \cmd_arg_spec:N #2 } {#1} #2 {#3} #4 #5
      }
      { \tl_set:Nn #4 {#1} }
  }

\cs_new:Npn \__head_prep_decls_and_auxiii:nnNnNN #1#2#3#4#5#6 {
  \bool_lazy_any:nTF
    {
```

This tests for macros with one argument and defined by `\def` or similar.

```
      { \str_if_eq_p:ee { \cs_parameter_spec:N #3 } { \c_hash_str 1 } }
```

For those defined with `\NewDocumentCommand` we need to look at the signature instead. We also accept macros with one optional argument first followed by a mandatory one.

```
      { \str_if_eq_p:nn {#1} { m } }
      { \str_if_eq_p:nn {#1} { +m } }
      { \str_if_eq_p:nn {#1} { O{} m } }
      { \str_if_eq_p:nn {#1} { O{} +m } }
```

Another possibility are macros declared with `\DeclareRobustCommand`.

```
      { \bool_lazy_and_p:nn
          { \cs_if_exist_p:c { \cs_to_str:N #3 \c_space_tl } }
          { \str_if_eq_p:ee
              { \cs_parameter_spec:c { \cs_to_str:N #3 \c_space_tl } }
              { \c_hash_str 1 }
          }
      }
      { \bool_lazy_and_p:nn
          { \cs_if_exist_p:c { \token_to_str:N #3 \c_space_tl } }
          { \str_if_eq_p:ee
              { \cs_parameter_spec:c { \token_to_str:N #3 \c_space_tl } }
              { [ \c_hash_str 1 ] \c_hash_str 2 }
          }
      }
    }
    { \tl_set:Ne #5 { \tl_reverse:n {#4} }
      \tl_set:Nn #6 {#3}
    }
    { \tl_set:Nn #5 {#2} }
}
```

```
\cs_generate_variant:Nn \cs_parameter_spec:N { c }
```

(*End of definition for* `\__head_prep_decls_and_format_from_tl:nNN`.)

Some classes redefine `\@startsection` and then our redefinition is lost. So we reinstate it

```
\NewCommandCopy\kernel@startsection\@startsection
\AddToHook{class/after}[head/@startsection]
  {\RenewCommandCopy\@startsection\kernel@startsection}
```

11.8.3 New `\secdef`

The command maps standard `\secdef` definition of `\part` and `\chapter` to the new interface. Unknown heading commands warn (or error if tagging is active) and then call the old commands.

`\secdef`

```
\RenewDocumentCommand\secdef{mmsO{#5}m}
 {
  \str_case:enF {\cs_to_str:N#1}
   {
    {@part}
     { \IfNoValueTF{#4}
        {\ParseLaTeXeHeading{part}{#3} {placement-start-code=\relax} {#5}}
        {
         \__head_process_shorttitle:nN{#4}\l__head_tmpa_tl
         \ExpandArgs{nne}\ParseLaTeXeHeading{part}{#3}
                    {placement-start-code=\relax,\exp_not:o{\l__head_tmpa_tl}} {#5}
        }
      \@latex@warning{\noexpand\secdef~ detected~ in~ \noexpand\part
          command.\MessageBreak
         \noexpand\part~will~be~redefined~to~use~templates.\MessageBreak
         This~may~change~the~layout!}
       \DeclareDocumentCommand \part {s ={shorttitle}o m}
         { \ParseLaTeXeHeading {part} {##1} {##2} {##3} }
     }
    {@chapter}
     { \IfNoValueTF{#4}
        { \ParseLaTeXeHeading{chapter}{#3} {#4} {#5} }
        {
          \__head_process_shorttitle:nN{#4}\l__head_tmpa_tl
          \ExpandArgs{nno}\ParseLaTeXeHeading{chapter}{#3}
                                             {\l__head_tmpa_tl} {#5}
        }
      \@latex@warning{\noexpand\secdef~ detected~ in~
         \noexpand\chapter command.\MessageBreak
         \noexpand\chapter~ will~ be~ redefined~ to~ use~
         templates.\MessageBreak
         This~may~change~the~layout!}
       \DeclareDocumentCommand \chapter {s ={shorttitle}o m}
        { \ParseLaTeXeHeading {chapter} {##1} {##2} {##3} }
     }
   }
   {
```

```
    \tag_if_active:TF\@latex@error\@latex@warning
        {\noexpand\secdef~ with~ unknown~ argument~ \noexpand#1
          found.\MessageBreak
      The~command~can~not~be~adapted~on~the~fly~to~support~tagging.\MessageBreak
      The~heading~command~using~this~\noexpand\secdef~should~be~
                 reimplemented\MessageBreak with~templates}{}
    \IfBooleanTF{#3}{#2{#5}}{#1[#4]{#5}}
  }
}
```

A helper command to reprocess the argument as key-val

```
\NewDocumentCommand\__head_process_shorttitle:nN{={shorttitle}mm}
 { \tl_set:Nn#2{#1} }
```

(*End of definition for* `\secdef`. *This function is documented on page 21.*)

11.8.4 Core LaTeX

We define here as an example the heading commands with the new interfaces for the three standard classes.

```
\AddToHook{class/article/after}[head/example]
 {
   \DeclareInstance{heading}{part}{display}
     {
      , name          = part
      , level         = -1
      , before-vspace = 4ex
      , after-vspace  = 3ex
      , mark-cmd      = \partmark {#1}
      , text          = \partname
      , punct         =
      , heading-decls = \raggedright\bfseries\Large
      , title-decls   = \huge
      , headformat-instance = part
     }
   \DeclareInstance{headformat}{part}{display}
     {
       , heading-indent    = 0pt
       , separator-a       = \nobreakspace
       , separator-b       = \par \nobreak \vspace{20pt}
     }
   \DeclareDocumentCommand \part {s ={shorttitle}o m}
      { \ParseLaTeXeHeading {part} {#1} {#2} {#3} }
   \__head_setup_default_heading:
 }

\AddToHook{class/report/after}[head/example]
 {
    \DeclareDocumentCommand \part {s ={shorttitle}o m}
      { \ParseLaTeXeHeading {part} {#1} {#2} {#3} }
    \DeclareDocumentCommand \chapter {s ={shorttitle}o m}
      {
        \ParseLaTeXeHeading {chapter}{#1} {#2} {#3}
      }
```

```
    \__head_setup_default_heading:
 }
\AddToHook{class/book/after}[head/example]
 {
    \DeclareDocumentCommand \part {s ={shorttitle}o m}
      { \ParseLaTeXeHeading {part} {#1} {#2} {#3} }
    \DeclareDocumentCommand \chapter {s ={shorttitle}o m}
      {
        \if@mainmatter
         \ParseLaTeXeHeading {chapter}{#1} {#2} {#3}
        \else
         \IfBooleanTF{#1}
          {% starred:
           \ParseLaTeXeHeading {chapter}{\BooleanTrue} {#2} {#3}
          }
          {\IfNoValueTF{#2}
            {\ParseLaTeXeHeading {chapter}{\BooleanTrue} {shorttitle={#3}} {#3}}
            {\ParseLaTeXeHeading {chapter}{\BooleanTrue} {#2} {#3}}
          }
        \fi
      }
    \__head_setup_default_heading:
 }
\cs_new_protected:Npn \__head_setup_default_heading:
 {
```

\section

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

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

\subsection

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

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

\subsubsection

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

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

\paragraph

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

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

\subparagraph

```
    \DeclareDocumentCommand \subparagraph {s ={shorttitle}o m}
      { \ParseLaTeXeHeading {subparagraph} {##1} {##2} {##3} }
 } %end of command
```

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

⟨/package⟩

12 Support for the titlesec interfaces

The current implementation incorporates some features of titlesec. The plan is that the titlesec interfaces are either natively available or are emulated and mapped to new interfaces, so that documents using titlesec work seamlessly.

We put the code that emulates titlesec in a separate file to be loaded instead of the original package, but eventually some of the code from there has to move back to the kernel to be always present.

```
\declare@file@substitution{titlesec.sty}{latex-lab-titlesec.sty}
```

13 Issues and problems noticed along the way

13.1 Local \DeclareInstance

`\DeclareInstance` does its declaration locally. That might be the right decision (or not) but we need to make sure that it in that case all of the declaration is local.

One potential problem with that is that it might allocate TEX registers.

The problem showed up in the redefinition of `\@startsection`, because in the current document some headings are local to an environment, so things get redeclared over and over again.

13.2 O-expansion of \UseInstance arguments

Whenever a template code calls a sub-instance and that sub-instance takes arguments, then the values for these arguments are typically inside tokenlists or registers so that for efficiency (and sometimes as a total must) one has to o-expand all arguments. While that can be coded somehow, e.g.,

```
\use:e {
  \UseInstance{headformat} { \l_@@_headformat_instance_tl }
              { \exp_not:o \UnusedTemplateKeys }
              { \exp_not:o { \l_@@_number_tl } }
              { \exp_not:n { #3 } }
              { \exp_not:o { \use_i:nn  #9 } }
              { \exp_not:o { \use_ii:nn  #9 } }
}
```

it would be better to have a version of `\UseInstance` that does this automatically. No suggestion for a name.

13.3 Switch \openright is not defined by core

I think this should change and it might be enough to just define it in the kernel (I think `\newif` no longer complains if it acts on an existing `\if...`

13.4 Indexheading at least for l3doc is wrong now

Needs checking.

A Templates from the 2026-06-01 release of LaTeX

In this appendix we provide the template implementations that I came up with in my first attempt (only renamed to ⟨*name*⟩`-v1`). Eventually, we want to drop the `-v1` versions but for the next couple of releases we provide them alongside the new more flexible templates, so that there is no need to transitioning to the new templates in a rush.

A.1 Template documentation for `-v1` templates

A.1.1 Templates of type `heading`

There are a number of keys that are expected to be recognized by all heading templates (though they may choose not to make use of them). These are listed below instead of being repeated on the actual templates.

All other keys are either attached to the `heading` or to the `headformat` templates. Those that typically vary from heading instance to the next (e.g., `heading-decls`) are all declared in the `heading` templates even if they are actually only used within `headformat` templates. In other words, `headformat` templates have a number of implicit variables that they expect to be set.[4]

A.1.2 The `heading` template '⟨*all*⟩'

Attributes:

`name` (*tokenlist*) Referenceable name of the heading instance. String that is acceptable in csnames for use in building counter names, etc.

`parent-name` (*tokenlist*) Name of the next higher heading instance. If not given, then the internal heading level of the heading instance is set to `0`

`reset-counter` (*tokenlist*) Name of the heading instance that should reset the numbering of this heading level (if any)

`level` (*integer*) Sets the internal heading-level rather than deducing it from `parent-name`. Can be used to specify the top-level heading if not `0`, or all headings in legacy implementations, e.g., through `\@startsection`

`placement` (*choice*) Set the heading placement, i.e., the behavior of the heading with respect to page breaks. Allowed values are `page` (heading forms a page if its own), `top` (heading starts a new page), `normal` (heading can appear anywhere on the page). Further possibilities might be `rectopage` and `rectotop` if we implement that. Default: `normal`

`start-code` (*tokenlist*) Default value is set by the `placement` key. Executed before the heading starts, so can issue, for example, a `\clearpage`

`final-code` (*tokenlist*) Default value is set by the `placement` key. Executed after the heading is typeset. Can set up code for putting the heading on a page by its own, or arrange for paragraph handling of a following paragraph, etc.

[4] Maybe questionable

`prefix` (*tokenlist*) A fixed string, such as "Chapter", that can be used together with the number (if any) to form a "heading label". Usage and placement is up to the template, so it could be placed after the number by the template (in which case it isn't really a prefix). Default: `\NoValue`

`mark-cmd` (*function(1)*) Function that receives the ⟨*running*⟩ argument and creates a suitable mark insertion Default: `\`⟨*name*⟩`mark`

Semantics & Comments: The above keys should be implemented by all heading templates.

At the moment I have retained the LaTeX 2_ε interfaces for marks, e.g., one has to set up `\chaptermark`, `\sectionmark`, etc. but I'm not sure this should stay (even though it is certainly simpler from a compatibility perspective).

All templates should set up `\theheading` to correspond to `\the`⟨*name*⟩.

A.1.3 The `heading` template 'display-v1'

Attributes:

`para-indent` (*boolean*) Should the paragraph after the heading be indented? Default: `false`

`before-vspace` (*skip*) Vertical space before the heading if there is no page or column break. If there is one it vanishes. In particular this means it will not be used in the heading `placements` `page` or `top` Default: `0pt`

`penalty` (*integer*) Penalty to break before the heading. The default (TeX's largest integer) indicates that no penalty was set in which case `\@secpenalty` is used. Default: `1073741823`

`after-penalty-vspace` (*skip*) Vertical space before the heading but after the penalty for the heading. If the penalty results in a page or column break, this space remains at the top of the page Default: `0pt`

`after-vspace` (*skip*) Vertical space after the heading Default: `0pt`

`heading-decls` (*tokenlist*) Declarations (such as font or color settings) applied to all heading elements, i.e., number, title, subtitle, and quotation, if present. Note that color commands (unless luacolor is used) introduce a break point before the heading and therefore one should either surround color commands with `\SaveLastSkip` and `\RestoreLastSkip` or to use the keys `prefix-decls`, `number-decls`, `title-decls`, etc. instead. Default: `\normalfont`

`prefix-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the heading prefix, overwriting the setting of `heading-decls`. Default: ⟨*empty*⟩

`number-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the heading number, overwriting the setting of `heading-decls`. Default: ⟨*empty*⟩

`title-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the heading title, overwriting the setting of `heading-decls`. Default: ⟨*empty*⟩

`subtitle-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the heading subtitle, overwriting the setting of `heading-decls`. Default: ⟨*empty*⟩

`quote-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the heading quote, overwriting the setting of `heading-decls`.
Default: ⟨*empty*⟩

`number-format` (*function(1)*) Code that produces a formatted version of the heading number including any ornamentations. Its argument is the counter name for the head. However, instead of using a specific counter representations, e.g., `\thesection`, it can refer to the counter representation for the current heading counter via `\theheading` and ignore the argument. Default: `\theheading`

`contents-extra` (*tokenlist*) Code containing `\addcontents` calls to write to files like `.lot` or `.lot` Default: ⟨*empty*⟩

`headformat-instance` (*instance*) Template instances of type `headformat` Default: `std`

Semantics & Comments: Several of the key names are simply bad and need revision!

A.1.4 The `heading` template 'runin-v1'

Attributes:

`before-vspace` (*skip*) Vertical space before the heading if there is no page or column break. If there is one it vanishes. Default: `0pt`

`penalty` (*integer*) Penalty to break before the heading. The default (TEX's largest integer) indicates that no penalty was set in which case `\@secpenalty` is used.
Default: `1073741823`

`after-penalty-vspace` (*skip*) Vertical space before the heading but after the penalty for the heading. If the penalty results in a page or column break, this space remains at the top of the page. It is also applied if the heading is a `page` or `top` heading.
Default: `0pt`

`after-vspace` (*skip*) Vertical space after the heading – ignored in run-in headings so should be dropped! Default: `0pt`

`heading-decls` (*tokenlist*) Declarations (such as font or color settings) applied to all heading elements, i.e., number, title, subtitle, and quotation, if present.
Default: `\normalfont`

`prefix-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the heading prefix, overwriting the setting of `heading-decls`. Default: ⟨*empty*⟩

`number-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the heading number, overwriting the setting of `heading-decls`. Default: ⟨*empty*⟩

`title-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the heading title, overwriting the setting of `heading-decls`.
Default: ⟨*empty*⟩

`subtitle-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the heading subtitle, overwriting the setting of `heading-decls`. Default: ⟨*empty*⟩

`quote-decls` (*tokenlist*) Declarations (such as font or color settings) applied to the heading quote, overwriting the setting of `heading-decls`.
Default: ⟨*empty*⟩

`number-format` (*function(1)*) Code that produces a formatted version of the heading number including any ornamentations. Its argument is the counter name for the head. However, instead of using a specific counter representations, e.g., `\thesection`, it can refer to the counter representation for the current heading counter via `\theheading` and ignore the argument. Default: `\theheading`

`headformat-instance` (*instance*) Template instances of type `headformat` Default: `std`

Semantics & Comments: Several of the key names are simply bad and need revision!

A.1.5 Templates of type `headformat`

At the moment all templates of this type assume that placement of prefix, number, title is done in exactly this order. Anything else would require defining further templates.

TODO: consider a more versatile mechanism (like in theorem-like templates) to allow for more flexible placements without the need to define additional templates.

A.1.6 The `headformat` template 'hang-v1'

Attributes:

`heading-indent` (*dimen*) Horizontal space before the heading (shifting it to the right in a LR typesetting context) Default: `0pt`

`title-format` (*tokenlist*) Code executed directly before the heading is typeset and after the vertical spacing is done. The code takes an argument, e.g., `\MakeUppercase`
Default: ⟨`#1`⟩

`prefix-number-sep` (*tokenlist*) Token list that can be used in the assignment to a TEX dimension (can contain `em` or `ex` values) specifying the separation between title prefix (if used) and title number. Evaluated after fonts for title text have been set up, i.e., the `em` will be based on the then current font. Default: `.5em`

`number-title-sep` (*tokenlist*) Token list that can be used in the assignment to a TEX dimension (can contain `em` or `ex` values) specifying the separation between title number and title text. Evaluated after fonts for title text have been set up, i.e., the `em` will be based on the then current font. Default: `1em`

Semantics & Comments: Implements a heading layout where number and title start on the same line and the title is hanging off from that if the title has more than one line. It is what LATEX always used by default for `\section`, `\subsection`, and `\subsubsection`.

Several of the key names are simply bad and need a revision!

A.1.7 The `headformat` template 'runin-v1'

Attributes:

`heading-indent` (*dimen*) Horizontal space before the heading (shifting it to the right in a LR typesetting context) Default: `0pt`

`title-format` (*tokenlist*) Code executed directly before the heading is typeset and after the vertical spacing is done. The code can take an argument, e.g., `\MakeUppercase` Default: ⟨*`empty`*⟩

`prefix-number-sep` (*tokenlist*) Token list that can be used in the assignment to a TEX dimension (can contain `em` or `ex` values) specifying the separation between title prefix (if used) and title number. Evaluated after fonts for title text have been set up, i.e., the `em` will be based on the then current font. Default: `.5em`

`number-title-sep` (*tokenlist*) Token list that can be used in the assignment to a TEX dimension (can contain `em` or `ex` values) specifying the separation between title number and title text. Evaluated after fonts for title text have been set up, i.e., the `em` will be based on the then current font.

In the hang template it is a horizontal dimension! Default: `1em`

Semantics & Comments: Implements a heading layout where number and title start on the same line but then continue with the following paragraph on the same line (or the next if the title overflows, i.e., the title is run-in. It is what LATEX always used by default for `\paragraph` and `\subparagraph`.

Several of the key names are simply bad and need a revision!

A.1.8 The `headformat` template 'display-v1'

Attributes:

`heading-indent` (*dimen*) Horizontal space before the heading (shifting it to the right in a LR typesetting context) Default: `0pt`

`title-format` (*tokenlist*) Code executed directly before the heading is typeset and after the vertical spacing is done. The code can take an argument, e.g., `\MakeUppercase` Default: ⟨*`#1`*⟩

`prefix-number-sep` (*tokenlist*) Token list that can be used in the assignment to a TEX dimension (can contain `em` or `ex` values) specifying the separation between title prefix (if used) and title number. Evaluated after fonts for title text have been set up, i.e., the `em` will be based on the then current font. Default: `.5em`

`number-title-sep` (*tokenlist*) Token list that can be used in the assignment to a TEX dimension (can contain `em` or `ex` values) specifying the separation between title number and title text. Evaluated after fonts for title text have been set up, i.e., the `em` will be based on the then current font.

In the display template it is a vertical dimension! Default: `20 pt`

Semantics & Comments: Implements a heading layout where number and title are vertically separated. It is what LaTeX always used by default for `\chapter` and `\part`.

Several of the key names are simply bad and need a revision!

A.2 Implementation of -v1 templates

`heading display-v1` (*templ.*)

```
\DeclareTemplateInterface{heading}{display-v1}{9}
{
  , name          : tokenlist
  , parent-name   : tokenlist
  , reset-counter : tokenlist
  , level         : integer = 0
  , placement     : choice {page , top , normal } = normal
  , mark-cmd      : function(1) =
  , para-indent   : choice { true , false } = false
  , before-vspace : skip = 0pt
  , penalty       : integer = \c_max_int
  , after-penalty-vspace : skip = 0pt
  , after-vspace  : skip = 0pt
  , start-code    : tokenlist =      % no default values!
  , final-code    : tokenlist =      % no default values!
  , prefix         : tokenlist = \NoValue
  , heading-decls  : tokenlist = \normalfont
  , prefix-decls   : tokenlist =
  , number-decls   : tokenlist =
  , title-decls    : tokenlist =
  , subtitle-decls : tokenlist =
  , quote-decls    : tokenlist =
  , headformat-instance : tokenlist = std
  , number-format  : function(1) = \theheading
  , contents-extra : tokenlist =
}
\DeclareTemplateCode{heading}{display-v1}{9}
{
  , name          = \l__head_name_tl
  , level         = \l__head_level_int
  , parent-name   = \l__head_pname_tl
  , reset-counter = \l__head_reset_cnt_tl
  , placement     = {
       ,page   = \__head_debug_typeout:n{ A~ page~ heading }
                 \tl_set:Nn \l__head_placement_tl { page }
       ,top    = \__head_debug_typeout:n{ A~ top~ heading }
                 \tl_set:Nn \l__head_placement_tl { top }
       ,normal = \__head_debug_typeout:n{ A~ normal~ heading }
                 \tl_set:Nn \l__head_placement_tl { normal }
                   }
  , mark-cmd      = \__head_mark_cmd:n
  , para-indent   = {
                      ,true  = \@afterindenttrue
                      ,false = \@afterindentfalse
                    }
```

```
  , before-vspace = \l__head_before_skip
  , penalty       = \l__head_penalty_int
  , after-penalty-vspace  = \l__head_after_penalty_skip
  , after-vspace  = \l__head_after_skip
  , start-code    = \l__head_start_code_tl
  , final-code    = \l__head_final_code_tl
  , prefix        = \l__head_typeset_prefix_tl
  , headformat-instance = \l__head_headformat_instance_tl
  , heading-decls  = \l__head_heading_decls_tl
  , prefix-decls   = \l__head_prefix_decls_tl
  , number-decls   = \l__head_number_decls_tl
  , title-decls    = \l__head_title_decls_tl
  , subtitle-decls = \l__head_subtitle_decls_tl
  , quote-decls    = \l__head_quote_decls_tl
  , number-format  = \__head_number_format:n
  , contents-extra = \l__head_contents_extra_tl
}
{
  \tl_set_eq:Nc \theheading { the \l__head_name_tl }
  \tl_set:Ne\l__head_saved_secnumdepth_tl{\int_use:N\c@secnumdepth}
  \__head_show_arguments:nnnnnnnnn
       {#1}{#2}{#3}{#4}{#5}{#6}{#7}{#8}{#9}
  \tl_if_empty:oF {#1} { \SetTemplateKeys{heading}{display}{#1} }
  \tl_if_empty:oT \l__head_start_code_tl
   { \str_case:VnF \l__head_placement_tl
      {
        { page }
        { \tl_set:Nn \l__head_start_code_tl
            {
             \if@openright
               \cleardoublepage
             \else
               \clearpage
             \fi
             \thispagestyle{plain}
             \if@twocolumn
               \onecolumn
               \@tempswatrue
             \else
               \@tempswafalse
             \fi
             \null\vfil
            }
        }
        { top }
        { \tl_set:Nn \l__head_start_code_tl
            {
              \if@openright\cleardoublepage\else\clearpage\fi
              \thispagestyle{plain}
              \global\@topnum\z@
            }
        }
      }
      { \tl_clear:N \l__head_start_code_tl }
```

```
 }
\tl_if_empty:oT \l__head_final_code_tl
 { \str_case:VnF \l__head_placement_tl
    {
      { page }
      { \tl_set:Nn \l__head_final_code_tl
          {
            \vfil\newpage
            \if@twoside
              \if@openright
                \null
                \thispagestyle{empty}
              \newpage
              \fi
            \fi
            \if@tempswa
              \twocolumn
            \fi
          }
      }
    }
    { \tl_set:Nn  \l__head_final_code_tl { \@afterheading } }
 }
\__head_determine_penalty:
```

Next lines are an inline version of `\__head_determine_number_typesetting:N #2` This macro has changed so here we inline the original code;

```
\bool_set:Nn \l__head_unnumbered_bool
   { \bool_lazy_or_p:nn
     { \int_compare_p:nNn \l__head_level_int > \c@secnumdepth }
     { \bool_if_p:N #2 }
   }
\bool_if:NTF \l__head_unnumbered_bool
  {
    \int_gset:Nn\c@secnumdepth{-99}
    \tl_clear:N \l__head_typeset_number_tl
  }
  {
    \@kernel@refstepcounter{ \l__head_name_tl }
    \protected@edef \l__head_typeset_number_tl
      {
        \__head_number_format:n { \l__head_name_tl }
      }
  }
```

End of inlined code.

```
\__head_vertical_before_spacing:
\UseTaggingSocket{sec/end}{\int_use:N\l__head_level_int}
\UseTaggingSocket{sec/begin}
                 {{\int_use:N\l__head_level_int}
                  {tag=\UseStructureName{sec/\int_use:N\l__head_level_int}}}
\__head_debug_typeout:n{use~ 'headformat'~instance:~
                        \l__head_headformat_instance_tl }
\use:e {
```

```
    \UseInstance{headformat} { \l__head_headformat_instance_tl }
                { \exp_not:o \UnusedTemplateKeys }
                { \exp_not:o { \l__head_typeset_prefix_tl } }
                { \exp_not:o { \l__head_typeset_number_tl } }
                { \exp_not:n { #3 } }
                { \exp_not:o { \use_i:nn  #9 } }
                { \exp_not:o { \use_ii:nn  #9 } }
  }
  \__head_handle_marks_etc:nnnnn {#4}{#5}{#6}{#7}{#8}
  \int_gset:Nn\c@secnumdepth{\l__head_saved_secnumdepth_tl}
  \par \nobreak
  \skip_vertical:N \l__head_after_skip
  \l__head_final_code_tl
  \ignorespaces
}
```

heading runin-v1 (*templ.*)

```
\DeclareTemplateCopy{heading}{runin-v1}{display-v1}
\DeclareTemplateCode{heading}{runin-v1}{9}
 {
  , name           = \l__head_name_tl
  , level          = \l__head_level_int
  , parent-name    = \l__head_pname_tl
  , reset-counter  = \l__head_reset_cnt_tl
  , placement      = {
        page   = \typeout{ ^^JA~ runin~ page~ placement~ heading~makes~no~sense
                           (top~used)}
                 \tl_set:Nn \l__head_placement_tl { top }
       ,top    = \__head_debug_typeout:n{ A~ top~ heading }
                 \tl_set:Nn \l__head_placement_tl { top }
       ,normal = \__head_debug_typeout:n{ A~ normal~ heading }
                 \tl_set:Nn \l__head_placement_tl { normal }
                  }
  , mark-cmd       = \__head_mark_cmd:n
  , para-indent    =  {
                       ,true  = %\typeout{para-indent~ setting~ ignored}
                       ,false = %\typeout{para-indent~ setting~ ignored}
                     }
  , before-vspace  = \l__head_before_skip
  , penalty        = \l__head_penalty_int
  , after-penalty-vspace  = \l__head_after_penalty_skip
  , after-vspace   = \l__head_after_skip
  , start-code     = \l__head_start_code_tl
  , final-code     = \l__head_final_code_tl
  , prefix        = \l__head_typeset_prefix_tl
  , headformat-instance = \l__head_headformat_instance_tl
  , heading-decls  = \l__head_heading_decls_tl
  , prefix-decls   = \l__head_prefix_decls_tl
  , number-decls   = \l__head_number_decls_tl
  , title-decls    = \l__head_title_decls_tl
  , subtitle-decls = \l__head_subtitle_decls_tl
  , quote-decls    = \l__head_quote_decls_tl
  , number-format  = \__head_number_format:n
  , contents-extra = \l__head_contents_extra_tl
```

```
}
{
  \__head_show_arguments:nnnnnnnnn
       {#1}{#2}{#3}{#4}{#5}{#6}{#7}{#8}{#9}
  \tl_set:Ne\l__head_saved_secnumdepth_tl{\int_use:N\c@secnumdepth}
  \tl_set_eq:Nc \theheading { the \l__head_name_tl }
  \tl_if_empty:oF {#1} { \SetTemplateKeys{heading}{runin}{#1} }
  \tl_if_empty:oT \l__head_start_code_tl
   {
    \str_case:Vn \l__head_placement_tl
      {
        { top }   { \tl_set:Nn \l__head_start_code_tl {\clearpage } }
      }
   }
  \__head_determine_penalty:
```

Next lines are an inline version of `\__head_determine_number_typesetting:N #2` This macro has changed so here we inline the original code;

```
  \bool_set:Nn \l__head_unnumbered_bool
     { \bool_lazy_or_p:nn
       { \int_compare_p:nNn \l__head_level_int > \c@secnumdepth }
       { \bool_if_p:N #2 }
     }
  \bool_if:NTF \l__head_unnumbered_bool
    {
      \int_gset:Nn\c@secnumdepth{-99}
      \tl_clear:N \l__head_typeset_number_tl
    }
    {
      \@kernel@refstepcounter{ \l__head_name_tl }
      \protected@edef \l__head_typeset_number_tl
        {
          \__head_number_format:n { \l__head_name_tl }
        }
    }
```

End of inlined code.

```
  \__head_vertical_before_spacing:
  \UseTaggingSocket{sec/end}{\int_use:N\l__head_level_int}
  \UseTaggingSocket{sec/begin}
    {{\int_use:N\l__head_level_int}{tag=\UseStructureName{sec/\int_use:N\l__head_level_int}}
  \UseTaggingSocket{sec/title/init}{\int_use:N\l__head_level_int}
  \def \@svsechd {
      \__head_debug_typeout:n{use~ 'headformat'~instance:~
                              \l__head_headformat_instance_tl }
      \use:e {
        \UseInstance{headformat} { \l__head_headformat_instance_tl }
                 { \exp_not:o \UnusedTemplateKeys }
                 { \exp_not:o { \l__head_typeset_prefix_tl } }
                 { \exp_not:o { \l__head_typeset_number_tl } }
                 { \exp_not:n { #3 } }
                 { \exp_not:o { \use_i:nn  #9 } }
                 { \exp_not:o { \use_ii:nn  #9 } }
      }
```

```
      \__head_handle_marks_etc:nnnnn {#4}{#5}{#6}{#7}{#8}
      \int_gset:Nn\c@secnumdepth{\l__head_saved_secnumdepth_tl}
  }
  \@nobreakfalse
  \global\@noskipsectrue
  \everypar{%
    \if@noskipsec
       \global\@noskipsecfalse
       {\setbox\z@\lastbox}
       \clubpenalty\@M
       \@svsechd
       \unskip
       \UseTaggingSocket{sec/title/split}
       \skip_horizontal:N \l__head_after_skip
    \else
      \clubpenalty \@clubpenalty
      \everypar{}%
    \fi
  }
  \l__head_final_code_tl
  \ignorespaces
}
```

`headformat display-v1` (*templ.*) The `display` template produces

```
\DeclareTemplateInterface{headformat}{display-v1}{6}
{
  , heading-indent    : length = 0pt
  , title-format      : function(1) = #1
  , prefix-number-sep : tokenlist = \WordSpaceAmount{1}
  , number-title-sep  : tokenlist = 20pt
}
```

`headformat hang-v1` (*templ.*)

```
\DeclareTemplateInterface{headformat}{hang-v1}{6}
{
  , heading-indent    : length = 0pt
  , title-format      : function(1) = #1
  , prefix-number-sep : tokenlist = \WordSpaceAmount{1}
  , number-title-sep  : tokenlist = 1em
}
```

`headformat runin-v1` (*templ.*)

```
\DeclareTemplateInterface{headformat}{runin-v1}{6}
{
  , heading-indent    : length = 0pt
  , title-format      : function(1) = #1
  , prefix-number-sep : tokenlist = \WordSpaceAmount{1}
  , number-title-sep  : tokenlist = 1em
}
```

`headformat display-v1` (*templ.*)

```
\DeclareTemplateCode{headformat}{display-v1}{6}
       % args: keys,prefix,number,title,subtitle,quotation
```

```
{
 , heading-indent    = \l__head_heading_indent_dim
 , title-format      = \__head_title_format:n
 , prefix-number-sep = \l__head_prefix_number_sep_tl
 , number-title-sep  = \l__head_number_title_sep_tl
}
{
 \__head_show_arguments:nnnnnn {#1}{#2}{#3}{#4}{#5}{#6}
 \tl_if_empty:oF {#1} { \SetTemplateKeys{headformat}{display-v1}{#1} }
 \group_begin:
   \UseTaggingSocket{sec/title/begin}{{\int_use:N\l__head_level_int}{#4}}
   \normalfont
   \interlinepenalty \@M
   \l__head_heading_decls_tl{}
   \bool_if:NTF \l__head_unnumbered_bool
     {
       \dim_compare:nNnTF  \l__head_heading_indent_dim < \c_zero_skip
         {
           \skip_horizontal:N \l__head_heading_indent_dim
           \MakeLinkTarget[\l__head_name_tl]{}
         }
         {
           \MakeLinkTarget[\l__head_name_tl]{}
           \skip_horizontal:N \l__head_heading_indent_dim
         }
     }
     {
       \dim_compare:nNnTF  \l__head_heading_indent_dim < \c_zero_skip
         {
           \skip_horizontal:N \l__head_heading_indent_dim
           \MakeLinkTarget{\l__head_name_tl}
         }
         {
           \MakeLinkTarget{\l__head_name_tl}
           \skip_horizontal:N \l__head_heading_indent_dim
         }
       \IfNoValueF{#2}
         {
           { \l__head_prefix_decls_tl #2 }
           \nobreak
           \skip_horizontal:n { \l__head_prefix_number_sep_tl }
         }
       \l__head_number_decls_tl
       #3
       \par\nobreak
       \skip_vertical:n { \l__head_number_title_sep_tl }
     }
   \l__head_title_decls_tl
   \__head_title_format:n {#4}
   \par
   \UseTaggingSocket{sec/title/end}
 \group_end:
}
```

`headformat hang-v1` (*templ.*)

```
\DeclareTemplateCode{headformat}{hang-v1}{6}
 {
  , heading-indent    = \l__head_heading_indent_dim
  , title-format      = \__head_title_format:n
  , prefix-number-sep = \l__head_prefix_number_sep_tl
  , number-title-sep  = \l__head_number_title_sep_tl
 }
 {
  \__head_show_arguments:nnnnnn {#1}{#2}{#3}{#4}{#5}{#6}
  \tl_if_empty:oF {#1} { \SetTemplateKeys{headformat}{hang-v1}{#1} }
  \group_begin:
    \UseTaggingSocket{sec/title/init}{\int_use:N\l__head_level_int}
    \normalfont
    \interlinepenalty \@M
    \l__head_heading_decls_tl{}
    \bool_if:NTF \l__head_unnumbered_bool
      {
        \tl_set:Nn\l__head_tmpa_tl
         {
          \dim_compare:nNnTF  \l__head_heading_indent_dim < \c_zero_skip
            {
              \skip_horizontal:N \l__head_heading_indent_dim
              \MakeLinkTarget[\l__head_name_tl]{}
            }
            {
              \MakeLinkTarget[\l__head_name_tl]{}
              \skip_horizontal:N \l__head_heading_indent_dim
            }
         }
      }
      {
       \tl_set:Nn \l__head_tmpa_tl
        {
          \dim_compare:nNnTF  \l__head_heading_indent_dim < \c_zero_skip
              {
                \skip_horizontal:N \l__head_heading_indent_dim
                \MakeLinkTarget{\l__head_name_tl}
              }
              {
                \MakeLinkTarget{\l__head_name_tl}
                \skip_horizontal:N \l__head_heading_indent_dim
              }
          \IfNoValueF{#2}
              {
                { \l__head_prefix_decls_tl #2 }
                \nobreak
                \skip_horizontal:n { \l__head_prefix_number_sep_tl }
              }
          { \l__head_number_decls_tl #3 }
          \skip_horizontal:n { \l__head_number_title_sep_tl }
        }
       }
     \UseTaggingSocket{sec/title/hang}
```

```
        {{\int_use:N\l__head_level_int}\l__head_unnumbered_bool{\l__head_tmpa_tl}{#4}}
        {
         \@hangfrom
           {
            \l__head_tmpa_tl
           }
        }
    \l__head_title_decls_tl
    \__head_title_format:n {#4}
    \par
  \group_end:
}
```

`headformat runin-v1` (*templ.*)

```
\DeclareTemplateCode{headformat}{runin-v1}{6}
 {
  , heading-indent    = \l__head_heading_indent_dim
  , title-format      = \__head_title_format:n
  , prefix-number-sep = \l__head_prefix_number_sep_tl
  , number-title-sep  = \l__head_number_title_sep_tl
 }
 {
  \__head_show_arguments:nnnnnn {#1}{#2}{#3}{#4}{#5}{#6}
  \tl_if_empty:oF {#1} { \SetTemplateKeys{headformat}{runin-v1}{#1} }
  \group_begin:
    \normalfont
    \interlinepenalty \@M
    \l__head_heading_decls_tl{}
    \bool_if:NTF \l__head_unnumbered_bool
      {
        \dim_compare:nNnTF  \l__head_heading_indent_dim < \c_zero_skip
         {
           \skip_horizontal:N \l__head_heading_indent_dim
           \MakeLinkTarget[\l__head_name_tl]{}
         }
         {
           \MakeLinkTarget[\l__head_name_tl]{}
           \skip_horizontal:N \l__head_heading_indent_dim
         }
      }
      {
         \dim_compare:nNnTF  \l__head_heading_indent_dim < \c_zero_skip
         {
           \skip_horizontal:N \l__head_heading_indent_dim
           \MakeLinkTarget{\l__head_name_tl}
         }
         {
           \MakeLinkTarget{\l__head_name_tl}
           \skip_horizontal:N \l__head_heading_indent_dim
         }
         {
           \UseTaggingSocket{sec/title/number}{\int_use:N\l__head_level_int}
             {
               \IfNoValueF{#2}
```

```
                {
                  { \l__head_prefix_decls_tl #2 }
                  \nobreak
                  \skip_horizontal:n { \l__head_prefix_number_sep_tl }
                }
                \l__head_number_decls_tl
                #3
              }
          }
          \skip_horizontal:n { \l__head_number_title_sep_tl }
        }
      \l__head_title_decls_tl
      \__head_title_format:n {#4}
  \group_end:
}
```

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

Symbols

\! 233, 235
\, ... 234, 236
\. 233, 235
\: 234, 236
\; 234, 236
\? 233, 235
\@startsection *21*
\␣ 350, 363, 374

\␣ *17, 18*

A

\addcontents *14*, *16*, *67*
\addcontentsline *22*, *53*, *54*, 1020
\addcontentslinebookmark 983, 1009
\addcontentslinebookmarkOff . . 983, 1019
\addcontentslinebookmarkOn 983
\addcontentslinebookmarkReset 985, 993, 996, 1028
\addpenalty . . *38*, *40–42*, *52*, 557, 561, 562
\addpenaltywithfil 554, 562
\AddPunct 238
\addtocontents 1060, 1061
\AddToHook 69, 597, 987, 1243, 1293, 1318, 1328
\addvspace *42*, 968, 1060, 1061
\AssignSocketPlug 173, 187
\AssignStructureRole 814
\AssignTaggingSocketPlug 742, 743, 744, 835, 836, 837, 895, 896, 897
\AtBeginDocument 1063

B

\baselineskip *13–15*
\bfseries 1041, 1056, 1072, 1083, 1093, 1103, 1115, 1304
bool commands:
  \bool_gset_false:N 18
  \bool_gset_true:N 13
  \bool_if:NTF 24, 26, 106, 131, 132, 176, 178, 180, 433, 513, 527, 542, 659, 757, 805, 810, 850, 906, 938, 945, 977, 1011, 1022, 1492, 1584, 1672, 1726, 1790
  \bool_if_p:N 942, 1490, 1582
  \bool_lazy_and_p:nn . . 121, 1221, 1228
  \bool_lazy_any:nTF 1214
  \bool_lazy_or_p:nn . . 940, 1488, 1580
  \bool_new:N 8, 59, 174
  \bool_set:Nn 119, 939, 1487, 1579
  \bool_set_false:N 92, 442, 662
  \bool_set_true:N 81, 815
\BooleanFalse *10*, 106
\BooleanTrue . . . *10*, 106, 1339, 1342, 1343
box commands:
  \box_use_drop:N 869
  \box_wd:N 862, 863

C

\centering 1041
\chapter . *4*, *5*, *9*, *13*, *17*, *19–21*, *54–56*, *61*, *70*, 1273, 1274, 1277, 1322, 1332
\chaptermark *13*, *66*, 1059
\cleardoublepage 451, 467, 1439, 1456
\clearpage *12*, *53*, *66*, 453, 467, 667, 1441, 1456, 1575
\clubpenalty 713, 719, 1622, 1628
cmd commands:
\cmd_arg_spec:N 1209
\cs 546
cs commands:
\cs_generate_variant:Nn 1241
\cs_gset_protected:Npe 23, 25
\cs_if_exist:NTF 1064
\cs_if_exist_p:N 1222, 1229
\cs_new:Npn 31, 40, 205, 218, 933, 937, 956, 1001, 1064, 1194, 1200, 1205, 1213
\cs_new_eq:NN 9, 10
\cs_new_protected:Npn 11, 16, 21, 28, 29, 70, 175, 177, 179, 237, 238, 552, 562, 594, 794, 1349
\cs_parameter_spec:N 1216, 1224, 1231, 1241
\cs_set_eq:NN 473, 555, 559, 669
\cs_set_protected:Npn 591
\cs_to_str:N 1222, 1224, 1247
\csname 600

D

\DebugHeadingsOff *22*, 28
\DebugHeadingsOn *22*, 28
\DeclareDocumentCommand 1147, 1261, 1277, 1314, 1320, 1322, 1330, 1332, 1351, 1353, 1355, 1357, 1359
\DeclareInstance *64*, 784, 1034, 1046, 1065, 1075, 1086, 1096, 1108, 1120, 1127, 1134, 1138, 1142, 1163, 1175, 1179, 1189, 1295, 1308
\DeclareInstanceCopy 1126, 1131, 1132, 1133
\DeclareRobustCommand *60*
\DeclareTemplateCode *34*, 377, 603, 726, 818, 880, 1390, 1528, 1656, 1711, 1776
\DeclareTemplateCopy *34*, 1527
\DeclareTemplateInterface . 250, 295, 340, 353, 366, 1364, 1635, 1642, 1649
\DeclareTemplateInterfaceEquivalents 286, 290, 331, 335
\def *60*, 233, 235, 692, 1601
dim commands:
\dim_add:Nn 863
\dim_compare:nNnTF 516, 569, 573, 575, 583, 758, 766, 907, 915, 957, 969, 978, 1674, 1685, 1730, 1744, 1792, 1803
\dim_set:Nn 572, 864
\dim_sub:Nn 799
\l_tmpa_dim 572, 580, 583, 584
\c_zero_dim 569, 576, 583
\DocumentMetadata *3*, *31*

E

\EditTemplateDefaults *34*
\else 242, 441, 452, 459, 467, 599, 718, 965, 995, 1169, 1176, 1336, 1440, 1447, 1456, 1627
\endcsname 598, 600
\everypar . . . *45*, 709, 720, 964, 1618, 1629
exp commands:
\exp_args:Ne 1196, 1207
\exp_args:Nne 133, 142, 150, 162
\exp_not:N 104, 1166, 1172, 1182
\exp_not:n 33, 34, 35, 36, 37, 38, 42, 43, 44, 45, 46, 47, 48, 49, 50, 74, 75, 105, 107, 108, 109, 110, 111, 112, 113, 114, 216, 514, 521, 522, 523, 524, 525, 526, 697, 698, 699, 700, 701, 702, 1031, 1170, 1171, 1185, 1186, 1255, 1513, 1514, 1515, 1516, 1517, 1518, 1606, 1607, 1608, 1609, 1610, 1611
\expandafter 600
\ExpandArgs 1254, 1269
extend (plug) 118

F

\fbox *4*
\fi . . . 71, 245, 246, 443, 454, 461, 467, 491, 492, 495, 601, 721, 962, 973, 997, 1159, 1169, 1190, 1345, 1442, 1449, 1456, 1476, 1477, 1480, 1630
\filbreak *40*, *52*
\final-code *13*
\font 228, 229, 230
\fontdimen 228, 229, 230
\frenchspacing *31*, 233

G

\global . . . 469, 708, 711, 1458, 1617, 1620
\glueexpr 227
group commands:
\group_begin: 567, 741, 795, 833, 893, 1667, 1721, 1786
\group_end: 587, 782, 801, 877, 931, 1709, 1774, 1829

H

\hangindent *49*, 862

\hbox 849
hbox commands:
\hbox_overlap_left:n 869
\hbox_overlap_right:n 797
head commands:
\head_debug_off: *22*, 11, 16, 29
\head_debug_on: *22*, 11, 11, 28
head internal commands:
\l__head_after_penalty_skip 403, 516, 517, 630, 957, 959, 969, 971, 978, 979, 1411, 1550
\l__head_after_skip 404, 529, 544, 631, 717, 1412, 1523, 1551, 1626
\l__head_before_skip 401, 628, 958, 968, 1409, 1548
\l__head_bookmark_tl 53, 80, 86, 90, 110, 191
\l__head_column_spanning_bool *39*, 395, 433, 442, 513, 527, 542, 622, 659, 662, 977
__head_column_vbox_adjusted:nnn 794, 794
\l__head_contents_extra_tl 425, 652, 1030, 1424, 1563
\g__head_count_int 125, 181, 184, 185
__head_debug:n 9, 9, 23
\g__head_debug_bool . . 8, 13, 18, 24, 26
__head_debug_gset: 11, 14, 19, 21
__head_debug_typeout:n 9, 10, 25, 32, 33, 34, 35, 36, 37, 38, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 72, 73, 102, 207, 211, 216, 221, 241, 243, 386, 388, 390, 392, 437, 509, 553, 615, 617, 619, 660, 693, 1031, 1149, 1397, 1399, 1401, 1509, 1538, 1540, 1602
__head_determine_number_typesetting:N *72*, *74*, 502, 684, 937, 937
__head_determine_penalty: 501, 683, 933, 933, 1486, 1578
\l__head_final_code_tl 1414, 1464, 1468, 1484, 1524, 1553, 1632
__head_find_label:w 76, 205, 205
__head_find_label_aux:w *30*, 214, 218, 223
__head_handle_marks_etc:nnnnn 537, 704, 1001, 1001, 1520, 1613
\l__head_headformat_instance_tl 409, 510, 520, 636, 694, 696, 1416, 1510, 1512, 1555, 1603, 1605
\l__head_heading_decls_tl 410, 637, 756, 846, 905, 1152, 1170, 1185, 1417, 1556, 1671, 1725, 1789
__head_heading_format:n 416, 643, 775, 866, 924
\l__head_heading_indent_dim *49*, 729, 758, 759, 763, 766, 767, 771, 820, 868, 883, 907, 908, 912, 915, 916, 920, 1659, 1674, 1676, 1681, 1685, 1687, 1692, 1713, 1730, 1732, 1737, 1744, 1746, 1751, 1778, 1792, 1794, 1799, 1803, 1805, 1810
\l__head_instance_keys_tl *26*, 60, 98, 99, 105
\l__head_instance_name_tl *26*, *27*, 60, 100, 102, 104, 135, 138, 139, 144, 146, 147, 152, 155, 156, 164, 167, 168
\l__head_label_tl *25*, *29*, 60, 112, 203, 208, 212, 222
\l__head_left_margin_dim 735, 752, 821, 863, 864, 867
\l__head_level_int 380, 504, 506, 507, 511, 606, 686, 688, 689, 691, 941, 1393, 1489, 1505, 1507, 1508, 1531, 1581, 1597, 1599, 1600, 1668, 1722, 1764, 1813
__head_mark_cmd:n 396, 623, 1005, 1404, 1543
\l__head_name_tl 379, 428, 605, 657, 760, 762, 768, 770, 852, 855, 909, 911, 917, 919, 952, 1009, 1013, 1020, 1024, 1392, 1427, 1498, 1501, 1530, 1569, 1590, 1593, 1677, 1680, 1688, 1691, 1733, 1736, 1747, 1750, 1795, 1798, 1806, 1809
\l__head_nameref_tl . . 56, 94, 111, 194
\l__head_number_decls_tl 412, 639, 1419, 1558, 1700, 1759, 1821
__head_number_format:n 418, 645, 1423, 1501, 1562, 1593
\l__head_number_tag_tl 424, 651
\l__head_number_title_sep_tl 1662, 1703, 1716, 1760, 1781, 1825
\l__head_number_tl *52*, 60, 523, 699, 748, 841, 901, 947, 953
\l__head_order_clist . . . 730, 824, 884
\l__head_page_type_tl *27*, 60, 130, 136, 140, 153, 157, 165, 169
\l__head_penalty_int 402, 629, 934, 935, 967, 1410, 1549
\l__head_placement_end_code_tl *36*, *37*, *44*, 406, 480, 484, 499, 548, 633, 676, 679, 681, 723
\l__head_placement_start_code_tl *36–39*, *44*, *53*, 405, 445, 449, 466, 477, 632, 664, 667, 673, 961, 976

\l__head_placement_tl 60, 387, 389, 391, 393, 435, 439, 446, 481, 614, 616, 618, 620, 665, 677, 1398, 1400, 1402, 1433, 1465, 1537, 1539, 1541, 1573
\l__head_pname_tl 382, 608, 1394, 1532
\l__head_prefix_decls_tl 1418, 1557, 1696, 1755, 1817
\l__head_prefix_number_sep_tl 1661, 1698, 1715, 1757, 1780, 1819
__head_prep_decls_and_auxi:nnNN 1197, 1200
__head_prep_decls_and_auxii:nNwNN 1202, 1205
__head_prep_decls_and_auxiii:nnNnNN 1208, 1213
__head_prep_decls_and_format_-from_tl:nNN 1151, 1194, 1194
__head_process_shorttitle:nN 1253, 1268, 1291
__head_punct_format:n 419, 646
\l__head_punct_tl . . *52*, 408, 635, 949
\l__head_quote_decls_tl 415, 642, 1422, 1561
__head_quote_format:n 422, 649
\l__head_quote_tl . . . 58, 96, 114, 202
\l__head_reset_cnt_tl 383, 609, 1395, 1533
\l__head_right_margin_dim 736, 753, 822, 865
\l__head_running_tl 54, 79, 85, 89, 109, 192
\l__head_saved_secnumdepth_tl 60, 429, 540, 656, 705, 1428, 1521, 1568, 1614
__head_setup_default_heading: 1316, 1326, 1347, 1349
__head_show_arguments:nnnnnn 31, 31, 739, 831, 891, 1665, 1719, 1784
__head_show_arguments:nnnnnnnnn 40, 40, 430, 654, 1429, 1566
\l__head_start_code_tl 1413, 1432, 1436, 1454, 1462, 1552, 1571, 1575
\l__head_subtitle_decls_tl 414, 641, 1421, 1560
__head_subtitle_format:n . 421, 648
\l__head_subtitle_tl . 57, 95, 113, 201
\l__head_text_decls_tl 411, 638
__head_text_format:n 417, 644
\l__head_text_tl *52*, 407, 522, 634, 698, 747, 840, 900, 948
\l__head_title_decls_tl 413, 640, 1420, 1559, 1705, 1771, 1827
__head_title_format:n 420, 647, 1660, 1706, 1714, 1772, 1779, 1828
\l__head_title_format_tl 1153, 1171, 1186
\l__head_title_tl *25*, *26*, *30*, 60, 83, 88, 89, 90, 94, 107, 209, 213, 216, 219, 749, 842, 902
\l__head_tmpa_tl . . 52, 1253, 1255, 1268, 1270, 1728, 1742, 1764, 1768
\l__head_toc_tl 55, 78, 84, 85, 86, 88, 108, 193
\l__head_typeset_number_tl 1495, 1499, 1515, 1587, 1591, 1608
\l__head_typeset_prefix_tl 1415, 1514, 1554, 1607
\l__head_unnumbered_bool *51*, 81, 92, 106, 132, 196, 198, 384, 610, 757, 850, 906, 938, 939, 945, 1011, 1022, 1487, 1492, 1579, 1584, 1672, 1726, 1764, 1790
\l__head_verso_page_bool 119, 131, 174, 176, 178, 180
l__head_verso_page_bool 174
__head_vertical_before_spacing: 503, 685, 956, 956, 1504, 1596
headformat (type) 249
headformat chapter (instance) 1120
headformat display (template) . . 340, 726
headformat display-v1 (template) 1635, 1656
headformat hang (template) 353, 818
headformat hang-v1 (template) . 1642, 1711
headformat paragraph (instance) . . . 1134
headformat part (instance) 1120
headformat runin (template) . . . 366, 880
headformat runin-v1 (template) 1649, 1776
headformat section (instance) 1127
headformat section-special (instance) 1142
headformat std (instance) 1127
headformat subparagraph (instance) . 1134
headformat subsection (instance) . . . 1127
headformat subsubsection (instance) 1127
heading (type) 248
heading chapter (instance) 1046
heading display (template) 250, 377
heading display-v1 (template) 1364
heading paragraph (instance) 1096
heading part (instance) 1034
heading runin (template) 295, 603
heading runin-v1 (template) 1527
heading section (instance) 1065
heading subparagraph (instance) . . . 1108
heading subsection (instance) 1075

heading subsubsection (instance) . . . 1086
heading/instname (socket) 117
heading/record (socket) 182
\hsize 799, 864
\hspace *18*, 362, 373, 798
\Huge 1042, 1057
\huge 1041, 1056, 1305

I

\if... . . . . *65*
\IfBlankF 1004, 1007
\IfBlankT 1019
\IfBlankTF 1006
\IfBooleanT 73
\IfBooleanTF 77, 1288, 1337
\ifcsname 598
\ifdim 1156, 1161
\ifhmode 239
\IfInstanceExistsF 1148
\IfInstanceExistsT 151, 163
\IfInstanceExistsTF 134, 143
\IfNoValueF 1694, 1753, 1815
\IfNoValueTF 97, 1250, 1265, 1341
\ifnum 240
\IfValueT 74
\IfVersoHeadingF 175
\IfVersoHeadingT 175
\IfVersoHeadingTF *19*, 175
\ignorespaces 550, 724, 1525, 1633
instances:
headformat chapter 1120
headformat paragraph 1134
headformat part 1120
headformat section 1127
headformat section-special . . . 1142
headformat std 1127
headformat subparagraph 1134
headformat subsection 1127
headformat subsubsection 1127
heading chapter 1046
heading paragraph 1096
heading part 1034
heading section 1065
heading subparagraph 1108
heading subsection 1075
heading subsubsection 1086
int commands:
\int_compare:nNnTF 934
\int_compare_p:nNn . . 941, 1489, 1581
\int_gincr:N 185
\int_gset:Nn 540, 705, 946, 1494, 1521, 1586, 1614
\int_if_even_p:n 123
\int_new:N 181
\int_set:Nn 935
\int_use:N 125, 184, 429, 504, 506, 507, 511, 656, 686, 688, 689, 691, 1428, 1505, 1507, 1508, 1568, 1597, 1599, 1600, 1668, 1722, 1764, 1813
\c_max_int *33*, 262, 307, 934, 1374
\interlinepenalty *50*, 755, 845, 904, 1670, 1724, 1788

K

kernel internal commands:
__kernel_add_penalty:n 562, 562, 592, 595
keys commands:
\keys_define:nn 188
\keys_set_known:nnN 99

L

\label *4*, *10*, *25*, *26*, *29*, *30*, 76, 203, 205, 212, 218, 222
\Large 1072, 1304
\large 1083
\lastbox 712, 1621
\lastskip *42*
\leavevmode 71, 962
legacy commands:
\legacy_if:nTF 565, 566
\legacy_if_p:n 122
\let 993, 996
\lineskip 796
\linewidth 864, 867
\ltlabsecIIdate 5
\ltlabsecIIversion 6

M

\MakeLinkTarget *21*, 760, 762, 768, 770, 852, 855, 909, 911, 917, 919, 1677, 1680, 1688, 1691, 1733, 1736, 1747, 1750, 1795, 1798, 1806, 1809
\MakeUppercase *58*, *68–70*
\markboth 69
\maxdepth *41*, 573, 574
\maxdimen *13*, *15*, 263, 308, 516, 957, 969, 978
\MessageBreak 1258, 1259, 1273, 1275, 1284, 1285, 1287
mode commands:
\mode_if_horizontal:TF 563
\mode_if_inner:TF 564

N

\nameref *3*, *4*
\newcommand 226, 983, 984, 985
\NewCommandCopy 1242
\NewDocumentCommand *10*, *11*, *60*, 1291
\newif *65*, 600

\newpage 485, 490, 1470, 1475
\NewSocket 117, 182
\NewSocketPlug 118, 183
\NewTaggingSocketPlug 803, 808, 813
\NewTemplateType 248, 249
\nobreak *17*, 349, 543, 1124, 1312, 1522, 1697, 1702, 1756, 1818
\nobreakspace *17*, *18*, 348, 361, 372, 1123, 1311
\noexpand 1257, 1259, 1272, 1273, 1274, 1283, 1286
\noindent 847
\NoNalue *9*
\nonfrenchspacing *31*, 233
\nopunct *31*, 237
\normalcolor *46*
\normalfont . . *13*, *15*, *67*, *68*, 269, 314, 754, 844, 903, 1072, 1083, 1093, 1103, 1115, 1380, 1669, 1723, 1787
\normalsize 1093, 1103, 1115
\NoValue *10–12*, *16–18*, *33*, *66*, 267, 312, 947, 948, 949, 1379
\null 462, 488, 1450, 1473
\numberline 1013, 1024

O

\onecolumn 457, 1445
\openright *65*

P

page (plug) 183
\par *14–17*, *34*, *59*, 71, 279, 349, 541, 564, 781, 876, 962, 1124, 1172, 1312, 1522, 1702, 1707, 1773
\paragraph *19*, *20*, *69*, 1357
\paragraphmark 1104
\parbox *46*
\parindent 785, 1056, 1140
\ParseLaTeXeHeading *8–10*, *25*, *29*, *54*, 70, 1192, 1251, 1254, 1262, 1266, 1269, 1278, 1315, 1321, 1324, 1331, 1335, 1339, 1342, 1343, 1352, 1354, 1356, 1358, 1360
\parskip 786
\part . *3–5*, *9*, *17*, *19–21*, *25*, *38*, *55*, *61*, *70*, 1257, 1259, 1261, 1314, 1320, 1330
\partmark *3*
\partmark 69, 1044, 1301
\partname 1039, 1302
\penalty *40*, *41*, 592, 595
Plugs:
  extend 118
  page 183
\prevdepth *41*, *42*, 573, 575, 577
prg commands:
  \prg_do_nothing: *26*, 84
property commands:
  \property_record:nn 184
  \property_ref:nnn 124
\protect 1013, 1024
\providecommand 69, 986
\ProvidesExplPackage 3

Q

quark commands:
  \q_no_value *29*, *30*, 76
  \quark_if_no_value:nTF 206, 220
  \q_stop 1202, 1205

R

\raggedright 1056, 1304
\refstepcounter *21*, *52*
\relax 231, 239, 996, 1154, 1160, 1251, 1255
\renewcommand 991
\RenewCommandCopy 989, 1244
\RenewDocumentCommand 1245
\RestoreLastSkip *13*, *15*, *46*, *67*

S

\SaveLastSkip *13*, *15*, *46*, *67*
scan commands:
  \scan_stop: 592
sec commands:
  \sec_add_penalty_with_fil:n *40*, *41*, *52*, 474, 552, 552, 670
  \sec_add_penalty_with_possible_fil:n *38*, *40*, *41*, *52*, 473, 552, 556, 560, 669, 966
\secdef *21*
\secdef *4*, *5*, *9*, *21*, *55*, *56*, *61*, 1245
\section *5*, *8*, *9*, *18–20*, *54*, *55*, *58*, *69*, 1351
\section* *19*
\sectionmark *13*, *66*, 1069
\setbox 712, 849, 1621
\SetCurrentKnownTemplateKeys . 432, 658
\SetTemplateKeys 740, 832, 892, 1431, 1570, 1666, 1720, 1785
\sfcode *31*, 233, 234, 235, 236
skip commands:
  \skip_add:Nn 580, 958
  \skip_horizontal:N 717, 759, 763, 767, 771, 868, 908, 912, 916, 920, 1626, 1676, 1681, 1687, 1692, 1732, 1737, 1746, 1751, 1794, 1799, 1805, 1810
  \skip_horizontal:n 1698, 1757, 1760, 1819, 1825
  \skip_set_eq:NN 568, 571

\skip_sub:Nn 959
\skip_vertical:N 529, 544, 1523
\skip_vertical:n 1703
\skip_zero:N 796
\l_tmpa_skip 568, 569, 571, 585
\l_tmpb_skip *41*, 571, 580, 581
\c_zero_skip 758, 766, 907, 915, 1674, 1685, 1730, 1744, 1792, 1803
socket commands:
\socket_use:n 101, 774, 848, 923
Sockets:
heading/instname 117
heading/record 182
\spacefactor *30*, *31*, 237, 240
\specialsection *58*
\start-code *13*
str commands:
\c_hash_str 1216, 1225, 1232
\str_case:nn 446, 665, 1573
\str_case:nnTF 481, 677, 1247, 1433, 1465
\str_if_eq:nnTF 435
\str_if_eq_p:nn 1216, 1217, 1218, 1219, 1220, 1223, 1230
\string 1150
\subparagraph *19*, *20*, *69*, 1359
\subparagraphmark 1116
\subsection *18–20*, *69*, 1353
\subsectionmark 1080
\subsubsection *18*, *20*, *69*, 1355

T

tag commands:
\tag_if_active:TF 1282
\tag_mc_end_push: 804, 809
\tag_struct_begin:n 806, 811
\tag_struct_end: 805, 810
tag internal commands:
\l__tag_para_bool 805, 810
\l__tag_para_flattened_bool 815
\tagpdfparaOff 745, 838, 898
\tagpdfparaOn 816
template commands:
\template_process_order_clist:nnn 776, 856, 870, 925
template types:
headformat 249
heading 248
templates:
headformat display 340, 726
headformat display-v1 . . . . 1635, 1656
headformat hang 353, 818
headformat hang-v1 . . . . . 1642, 1711
headformat runin 366, 880
headformat runin-v1 . . . . . 1649, 1776
heading display 250, 377
heading display-v1 1364
heading runin 295, 603
heading runin-v1 1527
TEX and LATEX 2ε commands:
\@ *31*
\@LRmoderr 564
\@M 713, 755, 845, 904, 1622, 1670, 1724, 1788
\@addpunct *31*
\@afterheading *39*, 499, 1484
\@afterindentfalse . . 399, 1158, 1407
\@afterindenttrue . . . 398, 1155, 1406
\@boxtempa *49*
\@chapapp *56*, 1054, 1063
\@chappap *33*
\@chapter *4*
\@clubpenalty 719, 1628
\@currentlabelname 1003
\@firstoftwo *10*
\@hangfrom 1766
\@kernel@refstepcounter 952, 1498, 1590
\@latex@error 1282
\@latex@warning 1257, 1272, 1282
\@m 240
\@minus 1101, 1113
\@nobreakfalse 707, 1616
\@noskipsecfalse 711, 1620
\@noskipsectrue 708, 1617
\@part *4*
\@plus 1101, 1113
\@secondoftwo *10*
\@secpenalty . . . *13*, *15*, *51*, *66*, *67*, 935
\@startsection . *4*, *5*, *9*, *11*, *21*, *54*, *55*, *58*, *59*, *61*, *64*, *66*, 1147, 1242, 1244
\@svsechd 692, 714, 1601, 1623
\@tempboxa 849, 862, 863, 869
\@tempskipa 1154, 1156, 1157, 1167, 1183
\@tempskipb 1160, 1161, 1168, 1177, 1184
\@tempswa *38*
\@tempswafalse 460, 1448
\@tempswatrue 458, 1446
\@topnewpage *37*, *39*, *40*, *53*, 515
\@topnum 469, 1458
\c@secnumdepth *51*, *52*, 429, 540, 656, 705, 941, 946, 1428, 1489, 1494, 1521, 1568, 1581, 1586, 1614
\declare@file@substitution . . . 1363
\Hy@addcontentsline@addbookmark 990
\Hy@bookmarksfalse 998
\Hy@bookmarkstrue 994
\if@afterindent 1169

\if@mainmatter 1334
\if@nobreak 963
\if@noskipsec 71, 710, 962, 1619
\if@openright 450, 467, 487, 597, 1438, 1456, 1472
\if@tempswa 493, 1478
\if@twocolumn 434, 456, 1444
\if@twoside 486, 1471
\ifHy@bookmarks 992
\kernel@startsection 1242, 1244
\protected@edef 1499, 1591
\z@ 469, 712, 1156, 1161, 1458, 1621
tex commands:
\tex_lastskip:D 568
\tex_parshape:D 867
\texorpdfstring *22*, *54*, 986, 1026
\the . 228, 229, 230, 1167, 1168, 1183, 1184
\the⟨*name*⟩ *13*, *66*
\thechapter *25*, 1055
\theheading *3*
\theheading *13*, *44*, *66–68*, 68, 428, 657, 953, 1387, 1427, 1569
\thepart 1040
\thesection *25*, *67*, *68*
\thispagestyle 455, 468, 489, 1443, 1457, 1474
tl commands:
\c_novalue_tl *26*, 95, 96
\c_space_tl 1222, 1224, 1229, 1231
\tl_clear:N 78, 79, 80, 98, 208, 477, 673, 679, 681, 1462, 1495, 1587
\tl_if_blank:nTF 961
\tl_if_empty:NTF 83
\tl_if_empty:nTF 445, 480, 664, 676, 1431, 1432, 1464, 1570, 1571, 1666, 1720, 1785
\tl_if_head_is_N_type:nTF 1201
\tl_new:N 52, 53, 54, 55, 56, 57, 58, 60, 61, 62, 63, 64, 65, 66, 67, 68
\tl_put_right:Nn 203, 219, 222
\tl_reverse:n 1198, 1236
\tl_set:Nn 84, 100, 130, 138, 146, 155, 167, 209, 212, 213, 387, 389, 391, 393, 429, 439, 449, 466, 484, 499, 614, 616, 618, 620, 656, 667, 747, 748, 749, 840, 841, 842, 900, 901, 902, 947, 948, 949, 953, 1003, 1195, 1203, 1211, 1236, 1237, 1239, 1292, 1398, 1400, 1402, 1428, 1436, 1454, 1468, 1484, 1537, 1539, 1541, 1568, 1575, 1728, 1742
\tl_set_eq:NN 85, 86, 88, 89, 90, 94, 95, 96, 428, 657, 1427, 1569
token commands:
\token_if_macro:NTF 1206
\token_to_str:N 1229, 1231
\topskip *13*, 958, 959, 1052
\twocolumn 494, 1479
\typeout 26, 612, 625, 626, 1535, 1545, 1546

U

\unskip 715, 1624
\UnusedTemplateKeys . 521, 697, 1513, 1606
use commands:
\use:N 1013, 1024
\use:n *60*, 103, 512, 695, 1162, 1178, 1195, 1511, 1604
\use_i:nn 525, 701, 1517, 1610
\use_ii:nn 526, 702, 1518, 1611
\use_none:n 9, 10
\UseInstance *27*, *64*, *65*, 104, 520, 696, 751, 1512, 1605
\UseSocket *11*
\UseStructureName 282, 327, 507, 689, 806, 811, 1508, 1599
\UseTaggingSocket 504, 505, 511, 533, 686, 687, 691, 716, 1505, 1506, 1597, 1598, 1600, 1625, 1668, 1708, 1722, 1763, 1813

V

vbox commands:
\vbox:n 799
\vfil . *38*, *40*, *42*, 462, 485, 595, 1450, 1470
\vfilneg *38*, *40*, *42*, 595
\vskip 581, 584, 585
\vspace . *17*, 349, 517, 971, 979, 1124, 1312
\vspace* *13*, *52*

W

\WordSpaceAmount . . . 226, 1639, 1646, 1653