

Reimplementation of $\text{\LaTeX}$ 2 ϵ 's heading commands using templates

$\text{\LaTeX}$ Project*

v0.9w 2026-10-17

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 <code>\@startsection</code>	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 <code>\section</code> command	8
3	Setting up $\text{\LaTeX}$ 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 <code>heading</code>	11
5.2.2	The <code>heading</code> template ‘ <code>\all</code> ’	11
5.2.3	The <code>heading</code> template ‘display’	13
5.2.4	The <code>heading</code> template ‘runin’	15
5.2.5	Templates of type <code>headformat</code>	16
5.2.6	The <code>headformat</code> template ‘display’	16
5.2.7	The <code>headformat</code> template ‘hang’	17
5.2.8	The <code>headformat</code> template ‘runin’	18
5.3	Default instances	19

*Initial implementation by Frank Mittelbach.

5.3.1	Instances of <code>heading</code> templates	19
5.3.2	Instances of <code>headformat</code> templates	20
5.4	Use of templates from the 2026-06-01 release of <code>L^AT_EX</code>	20
6	Support for legacy classes and packages	21
6.1	Support classes based on legacy <code>L^AT_EX 2_ε</code> 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 <code>L^AT_EX 2_ε</code> 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 <code>\@startsection</code>	58
11.8.3	New <code>\secdef</code>	61
11.8.4	Core <code>L^AT_EX</code>	62
12	Support for the <code>titlesec</code> interfaces	64
13	Issues and problems noticed along the way	64
13.1	Local <code>\DeclareInstance</code>	64
13.2	O-expansion of <code>\UseInstance</code> arguments	64
13.3	Switch <code>\openright</code> is not defined by core	65
13.4	Indexheading at least for <code>l3doc</code> is wrong now	65

A	Templates from the 2026-06-01 release of L^AT_EX	65
A.1	Template documentation for -v1 templates	65
A.1.1	Templates of type <code>heading</code>	65
A.1.2	The <code>heading</code> template ‘ <code><all></code> ’	65
A.1.3	The <code>heading</code> template ‘ <code>display-v1</code> ’	66
A.1.4	The <code>heading</code> template ‘ <code>runin-v1</code> ’	67
A.1.5	Templates of type <code>headformat</code>	68
A.1.6	The <code>headformat</code> template ‘ <code>hang-v1</code> ’	68
A.1.7	The <code>headformat</code> template ‘ <code>runin-v1</code> ’	69
A.1.8	The <code>headformat</code> template ‘ <code>display-v1</code> ’	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

<code>\theheading</code>	This command expands inside a heading to the current number.
<code>\partmark</code>	This replaces the fix <code>\markboth{}{}</code> or similar in the standard <code>\part</code> 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 L^AT_EX 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[shorttitle=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 re-defined 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 => Opt
> left-margin => Opt
> right-margin => Opt
> 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 be 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 L^AT_EX 2_ε 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 `\NoValue`. If it is offered via a key value interface, the template receives whatever is set by the user (and `\NoValue` 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 L^AT_EX 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

¹Maybe 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. Thus 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 $\text{\LaTeX 2}_{\epsilon}$ 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 placements `page` or `top`
Default: `0pt`
- penalty** (*integer*) Penalty to break before the heading. The default ($\text{\TeX}$ ’s largest integer) indicates that no penalty was set in which case `\@secpenalty` is used (to support the $\text{\LaTeX 2}_{\epsilon}$ 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 L^AT_EX 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 tagging 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 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 its 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 $\text{\LaTeX}$ 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 L^AT_EX 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 $\text{\LaTeX}$ 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 $\text{\LaTeX}$ 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 L^AT_EX

In the first heading implementation using templates we provided a number of somewhat restricted templates that covered the default layouts of standard L^AT_EX 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 L^AT_EX 2_ε interfaces

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

²There 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 L^AT_EX'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

```

1 <*package>
2 <@@=head>

3 \ProvidesExplPackage
4 {latex-lab-testphase-sec-template}
5 {\ltlabsecIIdate}
6 {\ltlabsecIIversion}
7 {heading implementation}

```

11.1 Debugging

```

\g__head_debug_bool

8 \bool_new:N \g__head_debug_bool
(End of definition for \g__head_debug_bool.)

\__head_debug:n
\__head_debug_typeout:n
9 \cs_new_eq:NN \__head_debug:n \use_none:n
10 \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:
11 \cs_new_protected:Npn \head_debug_on:
12 {
13   \bool_gset_true:N \g__head_debug_bool
14   \__head_debug_gset:
15 }

16 \cs_new_protected:Npn \head_debug_off:
17 {
18   \bool_gset_false:N \g__head_debug_bool
19   \__head_debug_gset:
20 }

21 \cs_new_protected:Npn \__head_debug_gset:
22 {
23   \cs_gset_protected:Npe \__head_debug:n ##1
24   { \bool_if:NT \g__head_debug_bool {##1} }
25   \cs_gset_protected:Npe \__head_debug_typeout:n ##1
26   { \bool_if:NT \g__head_debug_bool { \typeout{[head]~ ##1} } }
27 }

```

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

`\DebugHeadingsOn`
`\DebugHeadingsOff`

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

30 \DebugHeadingsOff
```

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

`\__head_show_arguments:nnnnnn`

Some debug data ...

```
31 \cs_new:Npn \__head_show_arguments:nnnnnn #1#2#3#4#5#6 {
32   \__head_debug_typeout:n{-----~ Headformat~ instance~ arguments:}
33   \__head_debug_typeout:n{1:~ keys~ =~ \exp_not:n{#1}}
34   \__head_debug_typeout:n{2:~ text~ =~ \exp_not:n{#2}}
35   \__head_debug_typeout:n{3:~ number~ =~ \exp_not:n{#3}}
36   \__head_debug_typeout:n{4:~ title~ =~ \exp_not:n{#4}}
37   \__head_debug_typeout:n{5:~ subtitle~ =~ \exp_not:n{#5}}
38   \__head_debug_typeout:n{6:~ quotation~ =~ \exp_not:n{#6}}
39 }
```

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

`\__head_show_arguments:nnnnnnnn`

Some debug data ...

```
40 \cs_new:Npn \__head_show_arguments:nnnnnnnn #1#2#3#4#5#6#7#8#9 {
41   \__head_debug_typeout:n{-----~ Heading~ instance~ arguments:}
42   \__head_debug_typeout:n{1:~ keys~ =~ \exp_not:n{#1}}
43   \__head_debug_typeout:n{2:~ unnumbered~ =~ \exp_not:n{#2}}
44   \__head_debug_typeout:n{3:~ title~ =~ \exp_not:n{#3}}
45   \__head_debug_typeout:n{4:~ toc~ =~ \exp_not:n{#4}}
46   \__head_debug_typeout:n{5:~ running~ =~ \exp_not:n{#5}}
47   \__head_debug_typeout:n{6:~ bookmark~ =~ \exp_not:n{#6}}
48   \__head_debug_typeout:n{7:~ nameref~ =~ \exp_not:n{#7}}
49   \__head_debug_typeout:n{8:~ label~ =~ \exp_not:n{#8}}
50   \__head_debug_typeout:n{9:~ subtitle | quote ~ =~ \exp_not:n{#9}}
51 }
```

(End of definition for `\__head_show_arguments:nnnnnnnn`.)

11.2 Temp variable(s)

`\l__head_tmpa_tl`

```
52 \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.

```

53 %\tl_new:N \l__head_bookmark_tl
54 %\tl_new:N \l__head_running_tl
55 %\tl_new:N \l__head_toc_tl
56 %\tl_new:N \l__head_nameref_tl
57 %\tl_new:N \l__head_subtitle_tl
58 %\tl_new:N \l__head_quote_tl
59 %\bool_new:N l__head_unnumbered_bool

```

But these token lists needs a declaration:

```

\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

60 \tl_new:N \l__head_label_tl
61 \tl_new:N \l__head_instance_keys_tl
62 \tl_new:N \l__head_number_tl
63 \tl_new:N \l__head_saved_secnumdepth_tl
64 \tl_new:N \l__head_title_tl
65 \tl_new:N \l__head_placement_tl
66 \tl_new:N \l__head_instance_name_tl
67 \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.

```

68 \tl_new:N\theheading

```

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

\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.

```

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

```

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

11.5 The L^AT_EX 2_ε parsing interface

L^AT_EX 2_ε 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).

```
70 \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.

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

72 \__head_debug_typeout:n{=====}
73 \__head_debug_typeout:n{#1:~ \IfBooleanT{#2}{*}
74 \IfValueT{#3}{[\exp_not:n{#3}]}
75 {\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`.

```
76 \__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 L^AT_EX 2_ε logic).

```
77 \IfBooleanTF{#2}
78 { \tl_clear:N \l__head_toc_tl
79 \tl_clear:N \l__head_running_tl
80 \tl_clear:N \l__head_bookmark_tl
81 \bool_set_true:N \l__head_unnumbered_bool
82 }
83 { \tl_set_eq:NN \l__head_toc_tl \l__head_title_tl
84 \tl_set_eq:NN \l__head_running_tl \l__head_title_tl
85 \tl_set_eq:NN \l__head_bookmark_tl \l__head_title_tl
86 \bool_set_false:N \l__head_unnumbered_bool
87 }
```

The `\nameref` always starts out matching the main title.

```
88 \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`

```
89 \tl_set_eq:NN \l__head_subtitle_tl \c_novalue_tl
90 \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?

```

91 \IfNoValueTF { #3 }
92   { \tl_clear:N \l__head_instance_keys_tl }
93   { \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`.

```

94 \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.

```

95 \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

```

96 \__head_debug_typeout:n{use~ 'heading'~ instance::~ \l__head_instance_name_tl }
97 \use:e {
98   \exp_not:N \UseInstance{heading}{ \l__head_instance_name_tl }
99   { \exp_not:o \l__head_instance_keys_tl }
100   { \bool_if:NTF \l__head_unnumbered_bool \BooleanTrue \BooleanFalse }
101   { \exp_not:o \l__head_title_tl }
102   { \exp_not:o \l__head_toc_tl }
103   { \exp_not:o \l__head_running_tl }
104   { \exp_not:o \l__head_bookmark_tl }
105   { \exp_not:o \l__head_nameref_tl }
106   { \exp_not:o \l__head_label_tl }
107   { { \exp_not:o \l__head_subtitle_tl }
108     { \exp_not:o \l__head_quote_tl } } }
109 }
110 }

```

`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`.

```

111 \NewSocket{heading/instname}{0}

```

`extend (plug)`

```

112 \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`.

```

113 \bool_set:Nn \l__head_verso_page_bool
114   {
115     \bool_lazy_and_p:nn
116       { \legacy_if_p:n { @twoside } }
117       { \int_if_even_p:n

```

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

```

118         { \property_ref:een
119           { heading*. \int_use:N \g__head_count_int }
120           {pagenum} { 1 }
121         }
122       }
123     }
124     \tl_set:Nc \l__head_page_type_tl
125       { \bool_if:NTF \l__head_verso_page_bool {verso} {recto} }

```

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

```

126     \bool_if:NTF \l__head_unnumbered_bool
127     { \exp_args:Nne
128       \IfInstanceExistsTF {heading}
129         { \l__head_instance_name_tl :
130           \l__head_page_type_tl :
131           unnumbered }
132     { \tl_set:Nc \l__head_instance_name_tl
133       { \l__head_instance_name_tl :
134         \l__head_page_type_tl : unnumbered }
135     }
136     { \exp_args:Nne
137       \IfInstanceExistsTF {heading}
138         { \l__head_instance_name_tl
139           :: unnumbered }
140     { \tl_set:Nc \l__head_instance_name_tl
141       { \l__head_instance_name_tl
142         :: unnumbered }
143     }
144     { \exp_args:Nne
145       \IfInstanceExistsT {heading}
146         { \l__head_instance_name_tl :
147           \l__head_page_type_tl }
148     {
149       \tl_set:Nc \l__head_instance_name_tl
150       { \l__head_instance_name_tl :
151         \l__head_page_type_tl }
152     }
153   }
154 }
155 }

```

Otherwise there can be at most two components:

```

156     { \exp_args:Nne
157       \IfInstanceExistsT {heading}
158         { \l__head_instance_name_tl :
159           \l__head_page_type_tl }
160     {
161       \tl_set:Nc \l__head_instance_name_tl
162       { \l__head_instance_name_tl :
163         \l__head_page_type_tl }
164     }
165   }
166 }
167 \AssignSocketPlug{heading/instname}{extend}

```

```

l__head_verso_page_bool 168 \bool_new:N      \l__head_verso_page_bool

\IfVersoHeadingTF
\IfVersoHeadingT 169 \cs_new_protected:Npn \IfVersoHeadingTF
\IfVersoHeadingF 170 { \bool_if:NTF \l__head_verso_page_bool }
171 \cs_new_protected:Npn \IfVersoHeadingT
172 { \bool_if:NT \l__head_verso_page_bool }
173 \cs_new_protected:Npn \IfVersoHeadingF
174 { \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 175 \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.

```

176 \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.

```

177 \NewSocketPlug{heading/record}{page}{
178   \property_record:ee { heading*. \int_use:N\g__head_count_int } {pagenum}
179   \int_gincr:N \g__head_count_int
180 }
181 \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.

```

182 \keys_define:nn {heading} {
183   , shorttitle .meta:n =
184     {toc = {#1} , running = {#1} , bookmark = {#1} , nameref= {#1} }
185   , bookmark .tl_set:N = \l__head_bookmark_tl
186   , running .tl_set:N = \l__head_running_tl
187   , toc .tl_set:N = \l__head_toc_tl
188   , nameref .tl_set:N = \l__head_nameref_tl
189   %
190   , numbered .bool_set_inverse:N = \l__head_unnumbered_bool
191   , numbered .default:n = true
192   , unnumbered .bool_set:N = \l__head_unnumbered_bool
193   , unnumbered .default:n = true
194   %
195   , subtitle .tl_set:N = \l__head_subtitle_tl
196   , 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.

```

197   , label .code:n = \tl_put_right:Nn \l__head_label_tl { \label{#1} }
198 }

```

(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.

```
199 \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.

```
200   \quark_if_no_value:nTF {#2}
201     { \__head_debug_typeout:n{---no-label }
202       \tl_clear:N \l__head_label_tl
203       \tl_set:Nn\l__head_title_tl {#1#3}
204     }
```

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

```
205     { \__head_debug_typeout:n{---label~found~#2}
206       \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`.

```
207       \tl_set:Nn\l__head_title_tl {#1}
208       \__head_find_label_aux:w #3\__head_find_label_aux:w
209     }
210     \__head_debug_typeout:n{---return:~ '\exp_not:o \l__head_title_tl' }
211 }
```

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.

```
212 \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.

```
213   \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.

```
214   \quark_if_no_value:nF {#2}
215     { \__head_debug_typeout:n{--- extra~ label~ '#2'~ found }
216       \tl_put_right:Nn \l__head_label_tl { \label{#2} }
217       \__head_find_label_aux:w #3\__head_find_label_aux:w
218     }
219 }
```

(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.

```

220 \newcommand\WordSpaceAmount[1]{
221   \glueexpr
222     #1\fontdimen2\the\font
223   plus #1\fontdimen3\the\font
224   minus #1\fontdimen4\the\font
225   \relax
226 }
```

(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 $\text{\LaTeX} 2_{\epsilon}$ `\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 We give all punctuation symbols a unique `\sfcode` value so that it is even possible to determine which punctuation was just typeset.

```

227 \def\frenchspacing{\sfcode`\.1006\sfcode`\'?1005\sfcode`\'!1004%
228   \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.

```

229 \def\nonfrenchspacing{\sfcode`\.3002\sfcode`\'?3001\sfcode`\'!3000%
230   \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.

```

231 \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!

```

232 \cs_new_protected:Npn \AddPunct #1 {
233   \relax\ifhmode
234     \ifnum\spacefactor>\@m
235       \__head_debug_typeout:n {--->~ punctuation~ '#1'~ not~ added}
236     \else
237       \__head_debug_typeout:n {--->~ punctuation~ '#1'~ added}
238     #1
239   \fi
240 \fi
241 }
```

(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 }

```

242 \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`

```

243 \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.

```

244 \DeclareTemplateInterface{heading}{display}{9}
245 {
246   , name                : tokenlist
247   , parent-name         : tokenlist
248   , reset-counter       : tokenlist
249   , level               : integer = 0

```

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

```

250   , force-unnumbered    : boolean = false

```

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

```

251   , placement           : choice {page , top , normal, ragged } = normal
252   , column-spanning     : boolean = false
253   , mark-cmd            : function(1) =
254   , 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.

```

255   , before-vspace       : skip    = Opt
256   , penalty             : integer = \c_max_int
257   , after-penalty-vspace : skip    = \maxdimen
258   , after-vspace        : skip    = Opt

```

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).

```

259   , placement-start-code : tokenlist =      % no default values!
260   , 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.

```

261   , text                : tokenlist = \NoValue

```

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

```

262   , punct               : tokenlist =
263   , heading-decls       : tokenlist = \normalfont
264   , text-decls          : tokenlist =
265   , number-decls        : tokenlist =
266   , title-decls         : tokenlist =
267   , subtitle-decls      : tokenlist =
268   , quote-decls         : tokenlist =
269   , heading-format      : function{1} = #1
270   , text-format         : function{1} = #1
271   , number-format       : function{1} = #1
272   , punct-format        : function{1} = #1
273   , title-format        : function{1} = #1 \par
274   , subtitle-format     : function{1} = #1
275   , 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}`.

```

276 , number-tag           : tokenlist = heading-number % \UseStructureName{sec/???}
277 , headformat-instance  : tokenlist = std
278 , contents-extra       : tokenlist =
279 }

```

Old names for the above to be dropped soon:

```

280 \DeclareTemplateInterfaceEquivalents{heading}{display}[2027/11/01]{
281   , start-code      : alias = placement-start-code
282   , final-code      : alias = placement-end-code
283 }

284 \DeclareTemplateInterfaceEquivalents{heading}{display}{
285   , prefix          : alias = text
286   , prefix-decls    : alias = text-decls
287   , prefix-format   : alias = text-format
288 }

```

`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`.

```

289 \DeclareTemplateInterface{heading}{runin}{9}
290 {
291   , name                : tokenlist
292   , parent-name         : tokenlist
293   , reset-counter       : tokenlist
294   , level               : integer = 0
295   , force-unnumbered    : boolean = false
296   , placement           : choice {page , top , normal , ragged } = normal
297   , column-spanning     : boolean = false
298   , mark-cmd            : function(1) =
299   , para-indent         : choice { true , false } = false
300   , before-vspace       : skip      = Opt
301   , penalty             : integer   = \c_max_int
302   , after-penalty-vspace : skip      = \maxdimen
303   , after-hspace        : skip      = Opt
304   , placement-start-code : tokenlist =      % no default values!
305   , placement-end-code   : tokenlist =      % no default values!

306   , text                : tokenlist = \NoValue
307   , punct               : tokenlist =
308   , heading-decls       : tokenlist = \normalfont
309   , text-decls          : tokenlist =
310   , number-decls        : tokenlist =
311   , title-decls         : tokenlist =
312   , subtitle-decls      : tokenlist =
313   , quote-decls         : tokenlist =
314   , heading-format      : function{1} = #1

```

```

315 , text-format          : function{1} = #1
316 , number-format       : function{1} = #1
317 , 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`.

```

318 , title-format         : function{1} = #1
319 , subtitle-format      : function{1} = #1
320 , quote-format         : function{1} = #1
321 , number-tag           : tokenlist = heading-number % \UseStructureName{sec/???}
322 , headformat-instance  : tokenlist = std
323 , contents-extra       : tokenlist =
324 }

```

Old names for the above to be dropped soon:

```

325 \DeclareTemplateInterfaceEquivalents{heading}{runin}[2027/11/01]{
326   , start-code      : alias = placement-start-code
327   , final-code      : alias = placement-end-code
328 }

329 \DeclareTemplateInterfaceEquivalents{heading}{runin}{
330   , prefix          : alias = text
331   , prefix-decls    : alias = text-decls
332   , prefix-format   : alias = text-format
333 }

```

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.

```

334 \DeclareTemplateInterface{headformat}{display}{6}
335 {
336   , heading-indent    : length    = 0pt
337   , left-margin       : length    = 0pt
338   , right-margin      : length    = 0pt
339   , order              : commalist = {text,separator-a,?,number,punct,
340                                     separator-b,?,title,
341                                     separator-c, subtitle}
342   , separator-a        : tokenlist = \nobreakspace
343   , separator-b        : tokenlist = \par \nobreak \vspace{20pt}
344   , separator-c        : tokenlist = \
345   , separator-d        : tokenlist =
346 }

```

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.

```

347 \DeclareTemplateInterface{headformat}{hang}{6}
348 {
349   , heading-indent    : length    = 0pt
350   , left-margin       : length    = 0pt

```

```

351 , right-margin      : length      = 0pt
352 , order-hang        : commalist = {text,separator-a,?,number,punct,
353                               separator-b,?}
354 , order              : commalist = {title, separator-c, subtitle}
355 , separator-a        : tokenlist = \nobreakspace
356 , separator-b        : tokenlist = \hspace{1em}
357 , separator-c        : tokenlist = \
358 , separator-d        : tokenlist =
359 }

```

`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.

```

360 \DeclareTemplateInterface{headformat}{runin}{6}
361 {
362   , heading-indent    : length = 0pt
363   , order              : commalist = {text,separator-a,?,number,punct,
364                                   separator-b,?,title,
365                                   separator-c,subtitle}
366   , separator-a        : tokenlist = \nobreakspace
367   , separator-b        : tokenlist = \hspace{1em}
368   , separator-c        : tokenlist = \
369   , separator-d        : tokenlist =
370 }

```

11.7.4 heading templates code

`heading display (templ.)`

```

371 \DeclareTemplateCode{heading}{display}{9}
372 {
373   , name              = \l__head_name_tl
374   , level              = \l__head_level_int
375   % next two not yet used
376   , parent-name       = \l__head_pname_tl
377   , reset-counter     = \l__head_reset_cnt_tl
378
379   , 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.

```

379   , placement         = {
380     ,page              = \__head_debug_typeout:n{ A~ page~ heading }
381                       \tl_set:Nn \l__head_placement_tl { page }
382     ,top               = \__head_debug_typeout:n{ A~ top~ heading }
383                       \tl_set:Nn \l__head_placement_tl { top }
384     ,normal            = \__head_debug_typeout:n{ A~ normal~ heading }
385                       \tl_set:Nn \l__head_placement_tl { normal }
386     ,ragged            = \__head_debug_typeout:n{ A~ normal~ heading~ (ragged~ style) }
387                       \tl_set:Nn \l__head_placement_tl { ragged }
388   }

```

```

389 , column-spanning = \l__head_column_spanning_bool
390 , mark-cmd        = \__head_mark_cmd:n

```

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

```

391 , para-indent      = {
392                     ,true  = \@afterindenttrue
393                     ,false = \@afterindentfalse
394                     }
395 , before-vspace    = \l__head_before_skip
396 , penalty          = \l__head_penalty_int
397 , after-penalty-vspace = \l__head_after_penalty_skip
398 , after-vspace     = \l__head_after_skip
399 , placement-start-code = \l__head_placement_start_code_tl
400 , placement-end-code  = \l__head_placement_end_code_tl

401 , text             = \l__head_text_tl
402 , punct            = \l__head_punct_tl

403 , headformat-instance = \l__head_headformat_instance_tl

404 , heading-decls     = \l__head_heading_decls_tl
405 , text-decls        = \l__head_text_decls_tl
406 , number-decls      = \l__head_number_decls_tl
407 , title-decls       = \l__head_title_decls_tl
408 , subtitle-decls    = \l__head_subtitle_decls_tl
409 , quote-decls       = \l__head_quote_decls_tl

410 , heading-format     = \__head_heading_format:n
411 , text-format        = \__head_text_format:n
412 , number-format      = \__head_number_format:n
413 , punct-format       = \__head_punct_format:n
414 , title-format       = \__head_title_format:n
415 , subtitle-format    = \__head_subtitle_format:n
416 , quote-format      = \__head_quote_format:n
417 %
418 , number-tag         = \l__head_number_tag_tl

419 , contents-extra     = \l__head_contents_extra_tl
420 }

421 {

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

```

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

```

423   \tl_set:Nc\l__head_saved_secnumdepth_tl{\int_use:N\c@secnumdepth}

424   \__head_show_arguments:nnnnnnnnn
425     {#1}{#2}{#3}{#4}{#5}{#6}{#7}{#8}{#9}

```

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

```

426   \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.

```

427 \bool_if:NT \l__head_column_spanning_bool
428 { \if@twocolumn
429   \str_if_eq:VnF \l__head_placement_tl {top}
430   {
431     \__head_debug_typeout:n {column-spanning~ requires~
432       placement=top.~ Adjusted!}
433     \tl_set:Nn \l__head_placement_tl {top}
434   }
435   \else
436     \bool_set_false:N \l__head_column_spanning_bool
437   \fi
438 }

```

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.

```

439 \tl_if_empty:oT \l__head_placement_start_code_tl
440 { \str_case:Vn \l__head_placement_tl
441   {
442     { page }
443     { \tl_set:Nn \l__head_placement_start_code_tl

```

Straight from the placement definition of `\part`.

```

444     { \if@openright
445       \cleardoublepage
446       \else
447         \clearpage
448       \fi
449       \thispagestyle{plain}%
450       \if@twocolumn
451         \onecolumn
452         \@tempwatrue
453       \else
454         \@tempwafalse
455       \fi
456       \null\vfil
457     }
458   }
459   { top }
460   { \tl_set:Nn \l__head_placement_start_code_tl
461     { \if@openright\cleardoublepage\else\clearpage\fi
462       \thispagestyle{plain}%
463       \global\@topnum\z@
464     }
465   }

```

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`

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

and `\vfilneg`. This is handled by altering `\sec_add_penalty_with_possible_fil:n` which after use resets itself to `\addpenalty`.

```

466         { ragged }
467         { \cs_set_eq:NN \sec_add_penalty_with_possible_fil:n
468           \sec_add_penalty_with_fil:n
469         }
470     }

```

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.

```

471 %     { \tl_clear:N \l__head_placement_start_code_tl }
472 }
473 %
474 \tl_if_empty:oT \l__head_placement_end_code_tl
475 { \str_case:VnF \l__head_placement_tl
476   {
477     { page }
478     { \tl_set:Nn \l__head_placement_end_code_tl
479       { \vfil\newpage
480         \if@twoside
481           \if@openright
482             \null
483             \thispagestyle{empty}%
484           \newpage
485         \fi
486       \fi
487       \if@tempwa
488         \twocolumn
489       \fi
490     }
491   }
492 }

```

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

```

493     { \tl_set:Nn \l__head_placement_end_code_tl { \@afterheading } }
494 }

```

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

```

495 \__head_determine_penalty:
496 \__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.

```

497 \__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.

```

498 \UseTaggingSocket{sec/end}{\int_use:N\l__head_level_int}
499 \UseTaggingSocket{sec/begin}
500     {\int_use:N\l__head_level_int}
501     {tag=\UseStructureName{sec/\int_use:N\l__head_level_int}}
502 }

```

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

```

503 \_head_debug_typeout:n{use~ 'headformat'~instance:~
504 \l__head_headformat_instance_tl }
505 \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.

```

506 \use:e {
507   \bool_if:NT \l__head_column_spanning_bool
508   { \exp_not:n {
509     \@topnewpage [
510       \dim_compare:nNnF \l__head_after_penalty_skip = \maxdimen
511       { \vspace* \l__head_after_penalty_skip }
512     }
513   }
514   \UseInstance{headformat} { \l__head_headformat_instance_tl }
515   { \exp_not:o \UnusedTemplateKeys }
516   { \exp_not:o { \l__head_text_tl } }
517   { \exp_not:o { \l__head_number_tl } }
518   { \exp_not:n { #3 } }
519   { \exp_not:o { \use_i:nn #9 } }
520   { \exp_not:o { \use_ii:nn #9 } }

521   \bool_if:NT \l__head_column_spanning_bool
522   {
523     \skip_vertical:N \l__head_after_skip

```

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

```

524   ]
525 }
526 }

527 \UseTaggingSocket{sec/title/end}
528 %
529 % --- handle marks, toc-entry, bookmark, nameref, and label
530 %
531 \_head_handle_marks_etc:nnnnn {#4}{#5}{#6}{#7}{#8}
532 %
533 % --- post-heading handling (vertical)

```

Restore secnumdepth

```

534 \int_gset:Nn\c@secnumdepth{\l__head_saved_secnumdepth_tl}
535 \par

```

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

```

536 \bool_if:NF \l__head_column_spanning_bool
537 { \nobreak

```

```

538         \skip_vertical:N \l__head_after_skip
539     }
540 % --- prepare next paragraph (defaults to \cs{@afterheading}):
541 %
542 \l__head_placement_end_code_tl
543 %
544 \ignorespaces
545 }

```

`\sec_add_penalty_with_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`.
`\sec_add_penalty_with_possible_fil:n` They have public names so that they can be used in other heading templates outside of this module.

```

546 \cs_new_protected:Npn \sec_add_penalty_with_fil:n #1 {
547     \__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`.

```

548     \addpenaltywithfil { #1 }

```

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

```

549     \cs_set_eq:NN
550     \sec_add_penalty_with_possible_fil:n
551     \addpenalty
552 }

```

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.

```

553 \cs_set_eq:NN
554 \sec_add_penalty_with_possible_fil:n
555 \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 L^AT_EX'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.

```

556 \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).

```

557     \mode_if_horizontal:T
558     { \mode_if_inner:TF { \@LRmoderr }{ \par } }
559     \legacy_if:nF { @minipage }
560     { \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.

```

561         { \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.

```

562         \skip_set_eq:NN \l_tmpa_skip \tex_lastskip:D
563         \dim_compare:nNnTF \l_tmpa_skip = \c_zero_dim
564         { #1 }

```

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

```

565         { \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.

```

566         \dim_set:Nn \l_tmpa_dim
567         { \dim_compare:nNnTF \prevdepth > \maxdepth
568           \maxdepth
569           { \dim_compare:nNnTF \prevdepth = { -1000pt }
570             \c_zero_dim
571             \prevdepth
572           }
573         }
574         \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.

```

575         \vskip -\l_tmpb_skip

```

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

```

576         #1

```

After the penalty (and after a possible `\vfилneg`) 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.

```

577         \dim_compare:nNnF \l_tmpa_dim = \c_zero_dim
578         { \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.

```

579         \vskip \l_tmpa_skip
580     }
581     \group_end:
582 }
583 }
584 }

```

Reimplement `\addpenalty` using the above macro:

```

585 \cs_set_protected:Npn \addpenalty #1 {
586   \__kernel_add_penalty:n { \penalty #1 \scan_stop: }
587 }

```

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

```
588 \cs_new_protected:Npn \addpenaltywithfil #1 {
589   \__kernel_add_penalty:n { \vfil \penalty #1 \vfilneg }
590 }
```

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

`\if@openright`

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

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

```
594   \expandafter\newif\csname if@openright\endcsname
595   \fi
596 }
```

(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.

```
597 \DeclareTemplateCode{heading}{runin}{9}
598 {
599   , name           = \l__head_name_tl
600   , level          = \l__head_level_int
601   % next two not yet used
602   , parent-name    = \l__head_pname_tl
603   , reset-counter  = \l__head_reset_cnt_tl
604
605   , 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.

```
605   , placement      = {
606     page            = \typeout{ ^^JA~ runin~ page~ placement~ heading~makes-no~sense
607                       (top~used)}
608     \tl_set:Nn \l__head_placement_tl { top }
609     ,top            = \__head_debug_typeout:n{ A~ top~ heading }
610     \tl_set:Nn \l__head_placement_tl { top }
611     ,normal         = \__head_debug_typeout:n{ A~ normal~ heading }
612     \tl_set:Nn \l__head_placement_tl { normal }
613     ,ragged         = \__head_debug_typeout:n{ A~ normal~ heading~ (ragged~ style) }
614     \tl_set:Nn \l__head_placement_tl { ragged }
615   }
616
617   , column-spanning = \l__head_column_spanning_bool
618
619   , 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.

```

618 , para-indent      = {
619                     ,true  = %\typeout{para-indent~ setting~ ignored}
620                     ,false = %\typeout{para-indent~ setting~ ignored}
621                     }
622 , before-vspace     = \l__head_before_skip
623 , penalty           = \l__head_penalty_int
624 , after-penalty-vspace = \l__head_after_penalty_skip
625 , after-hspace      = \l__head_after_skip
626 , placement-start-code = \l__head_placement_start_code_tl
627 , placement-end-code  = \l__head_placement_end_code_tl

628 , text              = \l__head_text_tl
629 , punct              = \l__head_punct_tl

630 , headformat-instance = \l__head_headformat_instance_tl

631 , heading-decls      = \l__head_heading_decls_tl

632 , text-decls         = \l__head_text_decls_tl
633 , number-decls       = \l__head_number_decls_tl
634 , title-decls        = \l__head_title_decls_tl
635 , subtitle-decls     = \l__head_subtitle_decls_tl
636 , quote-decls        = \l__head_quote_decls_tl

637 , heading-format     = \__head_heading_format:n
638 , text-format        = \__head_text_format:n
639 , number-format      = \__head_number_format:n
640 , punct-format       = \__head_punct_format:n
641 , title-format       = \__head_title_format:n
642 , subtitle-format    = \__head_subtitle_format:n
643 , quote-format       = \__head_quote_format:n
644 %
645 , number-tag         = \l__head_number_tag_tl

646 , contents-extra     = \l__head_contents_extra_tl
647 }
648 { \__head_show_arguments:nnnnnnnnn
649   {#1}{#2}{#3}{#4}{#5}{#6}{#7}{#8}{#9}

```

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

```

650 \tl_set:Nc\l__head_saved_secnumdepth_tl{\int_use:N\c@secnumdepth}

define \theheading

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

```

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

```

653 \bool_if:NT \l__head_column_spanning_bool
654 { \__head_debug_typeout:n {column-spanning~ is~
655   not~ (yet)~ supported~ for~ run-in~ headings!}
656   \bool_set_false:N \l__head_column_spanning_bool
657 }

```

```

658 \tl_if_empty:oT \l__head_placement_start_code_tl
659 { \str_case:Vn \l__head_placement_tl
660   {
661     { top } { \tl_set:Nn \l__head_placement_start_code_tl {\clearpage } }

662     { ragged }
663     { \cs_set_eq:NN \sec_add_penalty_with_possible_fil:n
664       \sec_add_penalty_with_fil:n
665     }
666   }

```

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

```

667 % { \tl_clear:N \l__head_placement_start_code_tl }
668 }
669 %

```

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 ...

```

670 % \tl_if_empty:oT \l__head_placement_end_code_tl
671 % { \str_case:VnF \l__head_placement_tl
672 %   {
673 %     { top } { \tl_clear:N \l__head_placement_end_code_tl }
674 %   }
675 %   { \tl_clear:N \l__head_placement_end_code_tl }
676 % }

677 \__head_determine_penalty:
678 \__head_determine_number_typesetting:N #2
679 \__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`.

```

680 \UseTaggingSocket{sec/end}{\int_use:N\l__head_level_int}
681 \UseTaggingSocket{sec/begin}
682   {\int_use:N\l__head_level_int}
683   {tag=\UseStructureName{sec/\int_use:N\l__head_level_int}}
684 }
685 \UseTaggingSocket{sec/title/init}{\int_use:N\l__head_level_int}
686 \def \@svsechd {
687   \__head_debug_typeout:n{use~ 'headformat'~instance:~
688     \l__head_headformat_instance_tl }
689   \use:e {
690     \UseInstance{headformat} { \l__head_headformat_instance_tl }
691     { \exp_not:o \UnusedTemplateKeys }
692     { \exp_not:o { \l__head_text_tl } }
693     { \exp_not:o { \l__head_number_tl } }
694     { \exp_not:n { #3 } }
695     { \exp_not:o { \use_i:nn #9 } }
696     { \exp_not:o { \use_ii:nn #9 } }
697   }
698   \__head_handle_marks_etc:nnnnn {#4}{#5}{#6}{#7}{#8}

```

Restore `secnumdepth`

```

699     \int_gset:Nn\c@secnumdepth{\l__head_saved_secnumdepth_tl}
700   }
701   \@nobreakfalse
702   \global\@noskipsectrue
703   \everypar{%
704     \if@noskipsec
705       \global\@noskipsecfalse
706       {\setbox\z@\lastbox}
707       \clubpenalty\@M
708       \@svsechd
709       \unskip

```

This tagging socket starts the “paragraph” after the run-in heading

```

710     \UseTaggingSocket{sec/title/split}
711     \skip_horizontal:N \l__head_after_skip
712   \else
713     \clubpenalty \@clubpenalty
714     \everypar{}%
715   \fi
716 }

```

— prepare next paragraph (does nothing by default)

```

717   \l__head_placement_end_code_tl
718   \ignorespaces
719 }

```

11.7.5 headformat templates code

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

```

720 \DeclareTemplateCode{headformat}{display}{6}
721   % args: keys,text,number,title,subtitle,quotation
722   {
723     , heading-indent      = \l__head_heading_indent_dim
724     , order               = \l__head_order_clist
725     , separator-a         = name {l__head_separator-a_tl}
726     , separator-b         = name {l__head_separator-b_tl}
727     , separator-c         = name {l__head_separator-c_tl}
728     , separator-d         = name {l__head_separator-d_tl}
729
730     , left-margin        = \l__head_left_margin_dim
731     , right-margin       = \l__head_right_margin_dim
732   }
733   {
734     \__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.

```

734   \SetTemplateKeys{headformat}{display}{#1}
735   \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.

```

736 \AssignTaggingSocketPlug{parbox/before}{heading}
737 \AssignTaggingSocketPlug{minipage/before}{heading}
738 \AssignTaggingSocketPlug{para/restore}{heading}
739 \tagpdfparaOff
740 %
741 \tl_set:Nn \l__head_text_tl {#2}
742 \tl_set:Nn \l__head_number_tl {#3}
743 \tl_set:Nn \l__head_title_tl {#4}
744 %

```

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.

```

745 \UseInstance{block}{adjustwidth}
746 { left-margin = \l__head_left_margin_dim
747   , right-margin = \l__head_right_margin_dim }

```

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

```

748 \normalfont
749 \interlinepenalty \@M
750 \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!

```

751 \bool_if:NTF \l__head_unnumbered_bool
752 { \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
753   { \skip_horizontal:N \l__head_heading_indent_dim
754     \MakeLinkTarget[\l__head_name_tl]}
755   }
756   { \MakeLinkTarget[\l__head_name_tl]}
757   \skip_horizontal:N \l__head_heading_indent_dim
758   }
759 }
760 { \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
761   { \skip_horizontal:N \l__head_heading_indent_dim
762     \MakeLinkTarget{\l__head_name_tl}
763   }
764   { \MakeLinkTarget{\l__head_name_tl}
765     \skip_horizontal:N \l__head_heading_indent_dim
766   }
767 }

```

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.

```

768 \socket_use:n {heading/record}

769 \__head_heading_format:n {
770   \template_process_order_clist:nnn
771   { head }{ order }
772   { number, text, punct, separator-a,
773     separator-b, separator-c, separator-d, title, subtitle }
774 }
775 \par
776 \group_end:
777 }

```

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

```

778 \DeclareInstance{block}{adjustwidth}{std}{
779   para-indent      = \parindent
780   , para-vspace    = \parskip
781   , begin-vspace   = 0pt
782   , end-vspace     = 0pt
783   , begin-penalty  = 10000,
784   , end-penalty    = 10000,
785   , begin-extra-vspace = 0pt,
786   , end-extra-vspace  = 0pt
787 }

```

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

```

788 \cs_new_protected:Npn \__head_column_vbox_adjusted:nnn #1#2#3 {
789   \group_begin:
790     \skip_zero:N \lineskip
791     \hbox_overlap_right:n {
792       \hspace { #1 }
793       \vbox:n { \dim_sub:Nn \hsize { #1 + #2 } #3 }
794     }
795   \group_end:
796 }

```

(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.

```

797 \NewTaggingSocketPlug{minipage/before}{heading}
798 { \tag_mc_end_push:
799   \bool_if:NT \l__tag_para_bool {\tag_struct_end:}
800   \tag_struct_begin:n{tag=\UseStructureName{sec/minipage}}
801 }

802 \NewTaggingSocketPlug{parbox/before}{heading}
803 { \tag_mc_end_push:
804   \bool_if:NT \l__tag_para_bool {\tag_struct_end:}
805   \tag_struct_begin:n{tag=\UseStructureName{sec/parbox}}
806 }

807 \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`.

```
808 { \AssignStructureRole{para/textblock}{heading-fragment}
809   \bool_set_true:N\l__tag_para_flattened_bool
810   \tagpdfparaOn
811 }
```

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

```
812 \DeclareTemplateCode{headformat}{hang}{6}
813 {
814   , heading-indent      = \l__head_heading_indent_dim
815   , left-margin         = \l__head_left_margin_dim
816   , right-margin        = \l__head_right_margin_dim
817   , order-hang          = name {l__head_order-hang_clist}
818   , order               = \l__head_order_clist
819   , separator-a         = name {l__head_separator-a_tl}
820   , separator-b         = name {l__head_separator-b_tl}
821   , separator-c         = name {l__head_separator-c_tl}
822   , separator-d         = name {l__head_separator-d_tl}
823 }
824 {
825   \__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).

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

827   \group_begin:
828   %
829   \AssignTaggingSocketPlug{parbox/before}{noop}
830   \AssignTaggingSocketPlug{parbox/after}{noop}
831   \AssignTaggingSocketPlug{para/restore}{noop}
832   \tagpdfparaOff
833   %
834   \tl_set:Nn \l__head_text_tl {#2}
835   \tl_set:Nn \l__head_number_tl {#3}
836   \tl_set:Nn \l__head_title_tl {#4}
837   %

838   \normalfont
839   \interlinepenalty \OM
840   \l__head_heading_decls_tl
841   \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.

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

843   \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.

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

```

845     {
846       \MakeLinkTarget[\l__head_name_tl]{ }
847     }
848     {
849       \MakeLinkTarget{\l__head_name_tl}
850       \template_process_order_clist:nnn
851         { head }{ order-hang }
852         { number, text, punct, separator-a,
853           separator-b, separator-c, separator-d, title, subtitle }
854     }
855   }}

```

As this template sets up a hang indentation we need to set up a special parshape. Using `\hangindent` as was done in L^AT_EX 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.

```

856   % \hangindent \box_wd:N\@tempboxa
857   \dim_add:Nn \l__head_left_margin_dim { \box_wd:N \@tempboxa }
858   \dim_set:Nn \linewidth { \hspace - \l__head_left_margin_dim
859                             - \l__head_right_margin_dim }
860   \__head_heading_format:n {
861     \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.

```

862     \skip_horizontal:N \l__head_heading_indent_dim
863     \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.

```

864     \template_process_order_clist:nnn
865     { head }{ order }
866     { number, text, punct, separator-a,
867       separator-b, separator-c, separator-d,
868       title, subtitle }
869   }
870   \par
871 \group_end:
872 }
873

```

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

```

874 \DeclareTemplateCode{headformat}{runin}{6}
875   % args: keys,text,number,title,subtitle,quotation
876   {
877     , heading-indent      = \l__head_heading_indent_dim
878     , order               = \l__head_order_clist

```

```

879 , separator-a      = name {l__head_separator-a_tl}
880 , separator-b      = name {l__head_separator-b_tl}
881 , separator-c      = name {l__head_separator-c_tl}
882 , separator-d      = name {l__head_separator-d_tl}
883 }
884 {
885   \__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).

```

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

887   \group_begin:
888   %
889   \AssignTaggingSocketPlug{parbox/before}{noop}
890   \AssignTaggingSocketPlug{parbox/after}{noop}
891   \AssignTaggingSocketPlug{para/restore}{noop}
892   \tagpdfparaOff
893   %
894   \tl_set:Nn \l__head_text_tl {#2}
895   \tl_set:Nn \l__head_number_tl {#3}
896   \tl_set:Nn \l__head_title_tl {#4}

897   \normalfont

898   \interlinepenalty \@M
899   \l__head_heading_decl_tl

```

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

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

```

900   \bool_if:NTF \l__head_unnumbered_bool
901   { \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
902     { \skip_horizontal:N \l__head_heading_indent_dim
903       \MakeLinkTarget[\l__head_name_tl]{}
904     }
905     { \MakeLinkTarget[\l__head_name_tl]{}
906       \skip_horizontal:N \l__head_heading_indent_dim
907     }
908   }
909   { \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
910     { \skip_horizontal:N \l__head_heading_indent_dim
911       \MakeLinkTarget{\l__head_name_tl}
912     }
913     { \MakeLinkTarget{\l__head_name_tl}
914       \skip_horizontal:N \l__head_heading_indent_dim
915     }
916   }

```

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.

```

917   \socket_use:n {heading/record}

```

```

918 \__head_heading_format:n {
919 \template_process_order_clist:nnn
920 { head }{ order }
921 { number, text, punct, separator-a,
922 separator-b, separator-c, separator-d,
923 title, subtitle }
924 }
925 \group_end:
926 }

```

11.7.6 Internal commands used by the template code

__head_determine_penalty:

```

927 \cs_new:Npn \__head_determine_penalty: {

```

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

```

928 \int_compare:nNt \l__head_penalty_int = \c_max_int
929 { \int_set:Nn \l__head_penalty_int \@secpenalty }
930 }

```

(End of definition for __head_determine_penalty:.)

__head_determine_number_typesetting:N

```

931 \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.

```

932 \bool_if:NF \l__head_unnumbered_bool
933 { \bool_set:Nn \l__head_unnumbered_bool
934 { \bool_lazy_or_p:nn
935 { \int_compare_p:nNn \l__head_level_int > \c@secnumdepth }
936 { \bool_if_p:N #1 }
937 }
938 }

```

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)

```

939 \bool_if:NTF \l__head_unnumbered_bool
940 { \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.

```

941 \tl_set:Nn \l__head_number_tl { \NoValue }
942 \tl_set:Nn \l__head_text_tl { \NoValue }
943 \tl_set:Nn \l__head_punct_tl { \NoValue }
944 }

```

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.

```

945 {

```

```

946     \@kernel@refstepcounter{ \l__head_name_tl }
947     \tl_set:Nn \l__head_number_tl { \theheading }
948   }
949 }

```

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

`\__head_vertical_before_spacing:`

```

950 \cs_new:Npn \__head_vertical_before_spacing: {
951   \dim_compare:nNnF \l__head_after_penalty_skip = \maxdimen
952   { \skip_add:Nn \l__head_before_skip \topskip
953     \skip_sub:Nn \l__head_after_penalty_skip \topskip
954   }
955   \tl_if_blank:VTF \l__head_placement_start_code_tl
956   { % \if@noskipsec \leavevmode \fi \par % <--- too late
957     \if@nobreak
958       \everypar{}
959     \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.

```

960     \sec_add_penalty_with_possible_fil:n
961     \l__head_penalty_int
962     \addvspace \l__head_before_skip

```

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

```

963     \dim_compare:nNnF \l__head_after_penalty_skip = \maxdimen
964     {
965       \vspace* \l__head_after_penalty_skip
966     }
967   \fi
968 }
969 {
970   \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`.

```

971   \bool_if:NF \l__head_column_spanning_bool
972   { \dim_compare:nNnF \l__head_after_penalty_skip = \maxdimen
973     { \vspace* \l__head_after_penalty_skip }
974   }
975 }
976 }

```

(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.

```

977 \newcommand\addcontentslinebookmark[3]{}
978 \newcommand\addcontentslinebookmarkOff{}
979 \newcommand\addcontentslinebookmarkReset{}
980 \providecommand\texorpdfstring[2]{#1}

```

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

```

981 \AddToHook{package/hyperref/after}
982 {
983   \RenewCommandCopy\addcontentslinebookmark
984     \Hy@addcontentsline@addbookmark
985   \renewcommand\addcontentslinebookmarkOff
986     { \ifHy@bookmarks
987       \let\addcontentslinebookmarkReset
988         \Hy@bookmarkstrue
989       \else
990         \let\addcontentslinebookmarkReset\relax
991       \fi
992       \Hy@bookmarksfalse
993     }
994 }

```

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

```

\_head_handle_marks_etc:nmmnn

```

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

```

995 \cs_new:Npn \_head_handle_marks_etc:nmmnn #1#2#3#4#5 {
996   %
997   \tl_set:Nn\@currentlabelname{#4}
998   \IfBlankF {#2}
999     { \_head_mark_cmd:n { #2 } }
1000   \IfBlankTF {#1}

```

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

```

1001   { \IfBlankF{#3}
1002     {
1003       \addcontentslinebookmark{toc}{\l__head_name_tl}
1004       {
1005         \bool_if:NF \l__head_unnumbered_bool
1006         {
1007           \protect\numberline{ \use:c{ the \l__head_name_tl } }
1008         }
1009         #3
1010       }
1011     }
1012   }

```

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

```

1013   { \IfBlankT{#3}{\addcontentslinebookmarkOff}

```

```

1014     \addcontentsline{toc}{ \l__head_name_tl }
1015     {
1016         \bool_if:NF \l__head_unnumbered_bool
1017         {
1018             \protect\numberline{ \use:c{ the \l__head_name_tl } }
1019         }

```

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

```

1020     \texorpdfstring{#1}{#3}
1021 }
1022 \addcontentslinebookmarkReset
1023 }

```

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

```

1024     \l__head_contents_extra_tl

```

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

```

1025     \__head_debug_typeout:n{--->~label(s):~ \exp_not:n{#5}}
1026     #5
1027 }

```

(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.

perhaps this should have a hook for packages

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.

```

1028 \DeclareInstance{heading}{part}{display}
1029 {
1030   , name           = part
1031   , level          = -1
1032   , placement      = page
1033   , text           = \partname
1034   , number-format = \thepart
1035   , heading-decls = \centering\bfseries\huge
1036   , title-decls   = \Huge
1037   , headformat-instance = part
1038   , mark-cmd      = \partmark {#1}
1039 }
```

`heading chapter (inst.)`

```

1040 \DeclareInstance{heading}{chapter}{display}
1041 {
1042   , name           = chapter
1043   , level          = 0
1044   , placement      = top
1045   , column-spanning = true
1046   , after-penalty-vspace = 60pt % that's 50pt plus \topskip for compatibility
1047   , after-vspace   = 40pt
1048   , text           = \@chapapp
1049   , number-format = \thechapter
1050   , heading-decls = \raggedright \parindent0pt \bfseries \huge
1051   , title-decls   = \Huge
1052   , headformat-instance = chapter
1053   , mark-cmd      = \chaptermark {#1}
1054   , contents-extra = \addtocontents{lof}{\addvspace{10pt}}
1055                   \addtocontents{lot}{\addvspace{10pt}}
1056 }
```

`\@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.

```
1057 \AtBeginDocument{
1058   \cs_if_exist:NF \@chapapp { \cs_new:Npn \@chapapp {Chapter} } }
```

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

`heading section (inst.)`

```
1059 \DeclareInstance{heading}{section}{display}
1060 {
1061   , name           = section
1062   , level          = 1
1063   , mark-cmd       = \sectionmark {#1}
1064   , before-vspace  = 3.5ex plus 1ex minus .2ex
1065   , after-vspace   = 2.3ex plus .2ex
1066   , heading-decls  = \normalfont\Large\bfseries
1067   , headformat-instance = section
1068 }
```

`heading subsection (inst.)`

```
1069 \DeclareInstance{heading}{subsection}{display}
1070 {
1071   , name           = subsection
1072   , parent-name    = section
1073   , level          = 2
1074   , mark-cmd       = \subsectionmark {#1}
1075   , before-vspace  = 3.25ex plus 1ex minus .2ex
1076   , after-vspace   = 1.5ex plus .2ex
1077   , heading-decls  = \normalfont\large\bfseries
1078   , headformat-instance = subsection
1079 }
```

`heading subsubsection (inst.)`

```
1080 \DeclareInstance{heading}{subsubsection}{display}
1081 {
1082   , name           = subsubsection
1083   , parent-name    = subsection
1084   , level          = 3
1085   , before-vspace  = 3.25ex plus 1ex minus .2ex
1086   , after-vspace   = 1.5ex plus .2ex
1087   , heading-decls  = \normalfont\normalsize\bfseries
1088   , headformat-instance = subsubsection
1089 }
```

`heading paragraph (inst.)`

```
1090 \DeclareInstance{heading}{paragraph}{runin}
1091 {
1092   , name           = paragraph
1093   , parent-name    = subsubsection
1094   , level          = 4
1095   , before-vspace  = 3.25ex \@plus1ex \@minus.2ex
1096   , after-hspace   = 1em

```

```

1097 , heading-decls = \normalfont\normalsize\bfseries
1098 , mark-cmd      = \paragraphmark {#1}
1099 , headformat-instance = paragraph
1100
1101 }

```

heading subparagraph (*inst.*)

```

1102 \DeclareInstance{heading}{subparagraph}{runin}
1103 {
1104   , name          = subparagraph
1105   , parent-name   = paragraph
1106   , level         = 5
1107   , before-vspace = 3.25ex \@plus1ex \@minus .2ex
1108   , after-hspace  = 1em
1109   , heading-decls = \normalfont\normalsize\bfseries
1110   , mark-cmd      = \subparagraphmark {#1}
1111   , headformat-instance = subparagraph
1112
1113 }

```

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

```

1114 \DeclareInstance{headformat}{part}{display}
1115 {
1116   , heading-indent    = 0pt
1117   , separator-a       = \nobreakspace           % between text and number
1118   , separator-b       = \par \nobreak \vspace{20pt} % between label block and title
1119 }
1120 \DeclareInstanceCopy{headformat}{chapter}{part}

```

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

```

1121 \DeclareInstance{headformat}{std}{hang}
1122 {
1123   , heading-indent = 0pt
1124 }
1125 \DeclareInstanceCopy{headformat}{section}{std}
1126 \DeclareInstanceCopy{headformat}{subsection}{std}
1127 \DeclareInstanceCopy{headformat}{subsubsection}{std}

```

headformat paragraph (*inst.*) In contrast, paragraph and subparagraph differ even though both use the runin template.
headformat subparagraph (*inst.*)

```

1128 \DeclareInstance{headformat}{paragraph}{runin}
1129 {
1130   , heading-indent = 0pt
1131 }
1132 \DeclareInstance{headformat}{subparagraph}{runin}
1133 {
1134   , heading-indent = \parindent
1135 }

```

adformat section-special (*inst.*)

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

```

1136 \DeclareInstance{headformat}{section-special}{hang}
1137 {
1138   , heading-indent = 4em
1139   %%% , title-format = {#1!} % no longer
1140 }

```

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`.

```

1141 \DeclareDocumentCommand \@startsection {mmmmm 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.

```

1142   \IfInstanceExistsF{heading}{#1-\@startsection}{
1143     \__head_debug_typeout:n{Info:~ setting~ up~ instances~ for~
1144       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:

```

1145   \__head_prep_decls_and_format_from_tl:nNN {#6}
1146   \l__head_heading_decls_tl
1147   \l__head_title_format_tl

```

In the $\text{\LaTeX} 2_{\epsilon}$ 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.

```

1148   \@tempskipa #4\relax
1149   \@afterindenttrue
1150   \ifdim \@tempskipa <\z@
1151     \@tempskipa -\@tempskipa
1152   \@afterindentfalse
1153   \fi

```

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

```

1154 \@tempskipb #5\relax
1155 \ifdim \@tempskipb>\z@
1156 \use:e {
1157 \DeclareInstance{heading}{#1-@startsection}{display}{
1158 , name = #1
1159 , level = #2
1160 , mark-cmd = \exp_not:c {#1mark} {##1}
1161 , before-vspace = \the\@tempskipa
1162 , after-vspace = \the\@tempskipb
1163 , 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.

```

1164 , heading-decls = \exp_not:o \l__head_heading_decls_tl
1165 , title-format = \exp_not:o \l__head_title_format_tl {##1}
1166 \exp_not:N \par
1167 , headformat-instance = #1-@startsection
1168 }}
1169 \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.

```

1170 \else
1171 \@tempskipb=-\@tempskipb
1172 \use:e {
1173 \DeclareInstance{heading}{#1-@startsection}{runin}{
1174 , name = #1
1175 , level = #2
1176 , mark-cmd = \exp_not:c {#1mark} {##1}
1177 , before-vspace = \the\@tempskipa
1178 , after-hspace = \the\@tempskipb
1179 , heading-decls = \exp_not:o \l__head_heading_decls_tl

```

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

```

1180 , title-format = \exp_not:o \l__head_title_format_tl {##1}
1181 , headformat-instance = #1-@startsection
1182 }}
1183 \DeclareInstance{headformat}{#1-@startsection}{runin}{heading-indent = #3}
1184 \fi
1185 }
1186 \ParseLaTeXeHeading {#1-@startsection}{#7}{#8}{#9}
1187 }

```

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

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.

```

1188 \cs_new:Npn \__head_prep_decls_and_format_from_tl:nNN #1#2#3
1189 { \tl_set:Nn #3 { \use:n }

```

```

1190     \exp_args:Ne
1191     \__head_prep_decls_and_auxi:nnNN
1192         { \tl_reverse:n {#1} } {#1} #2 #3
1193 }
1194 \cs_new:Npn \__head_prep_decls_and_auxi:nnNN #1#2#3#4
1195 { \tl_if_head_is_N_type:nTF {#1}
1196   { \__head_prep_decls_and_auxii:nNwNN {#2} #1 \q_stop #3 #4 }
1197   { \tl_set:Nn #3 {#2} }
1198 }
1199 \cs_new:Npn \__head_prep_decls_and_auxii:nNwNN #1#2#3 \q_stop #4 #5
1200 { \token_if_macro:NTF #2
1201   { \exp_args:Ne
1202     \__head_prep_decls_and_auxiii:nnNnNN
1203     { \cmd_arg_spec:N #2 } {#1} #2 {#3} #4 #5
1204   }
1205   { \tl_set:Nn #4 {#1} }
1206 }
1207 \cs_new:Npn \__head_prep_decls_and_auxiii:nnNnNN #1#2#3#4#5#6 {
1208   \bool_lazy_any:nTF
1209   {

```

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

```

1210     { \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.

```

1211     { \str_if_eq_p:nn {#1} { m } }
1212     { \str_if_eq_p:nn {#1} { +m } }
1213     { \str_if_eq_p:nn {#1} { 0{ } m } }
1214     { \str_if_eq_p:nn {#1} { 0{ } +m } }

```

Another possibility are macros declared with `\DeclareRobustCommand`.

```

1215     { \bool_lazy_and_p:nn
1216       { \cs_if_exist_p:c { \cs_to_str:N #3 \c_space_tl } }
1217       { \str_if_eq_p:ee
1218         { \cs_parameter_spec:c { \cs_to_str:N #3 \c_space_tl } }
1219         { \c_hash_str 1 }
1220       }
1221     }
1222     { \bool_lazy_and_p:nn
1223       { \cs_if_exist_p:c { \token_to_str:N #3 \c_space_tl } }
1224       { \str_if_eq_p:ee
1225         { \cs_parameter_spec:c { \token_to_str:N #3 \c_space_tl } }
1226         { [ \c_hash_str 1 ] \c_hash_str 2 }
1227       }
1228     }
1229 }
1230 { \tl_set:Ne #5 { \tl_reverse:n {#4} }
1231   \tl_set:Nn #6 {#3}
1232 }
1233 { \tl_set:Nn #5 {#2} }
1234 }

```

```
1235 \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

```
1236 \NewCommandCopy\kernel@startsection\@startsection
1237 \AddToHook{class/after}[head/@startsection]
1238 {\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`

```
1239 \RenewDocumentCommand\secdef{mmsO{#5}m}
1240 {
1241   \str_case:enF {\cs_to_str:N#1}
1242   {
1243     {@part}
1244     { \IfNoValueTF{#4}
1245       {\ParseLaTeXeHeading{part}{#3} {placement-start-code=\relax} {#5}}
1246       {
1247         \__head_process_shorttitle:nN{#4}\l__head_tmpa_tl
1248         \ExpandArgs{nne}\ParseLaTeXeHeading{part}{#3}
1249         {placement-start-code=\relax,\exp_not:o{\l__head_tmpa_tl}} {#5}
1250       }
1251       \@latex@warning{\noexpand\secdef~ detected~ in~ \noexpand\part
1252         command.\MessageBreak
1253         \noexpand\part~will~be~redefined~to~use~templates.\MessageBreak
1254         This~may~change~the~layout!}
1255       \DeclareDocumentCommand \part {s ={shorttitle}o m}
1256       { \ParseLaTeXeHeading {part} {##1} {##2} {##3} }
1257     }
1258     {@chapter}
1259     { \IfNoValueTF{#4}
1260       { \ParseLaTeXeHeading{chapter}{#3} {#4} {#5} }
1261       {
1262         \__head_process_shorttitle:nN{#4}\l__head_tmpa_tl
1263         \ExpandArgs{nno}\ParseLaTeXeHeading{chapter}{#3}
1264         {\l__head_tmpa_tl} {#5}
1265       }
1266       \@latex@warning{\noexpand\secdef~ detected~ in~
1267         \noexpand\chapter command.\MessageBreak
1268         \noexpand\chapter~ will~ be~ redefined~ to~ use~
1269         templates.\MessageBreak
1270         This~may~change~the~layout!}
1271       \DeclareDocumentCommand \chapter {s ={shorttitle}o m}
1272       { \ParseLaTeXeHeading {chapter} {##1} {##2} {##3} }
1273     }
1274   }
1275 }
```

```

1276 \tag_if_active:TF\@latex@error\@latex@warning
1277 {\noexpand\secdef~ with~ unknown~ argument~ \noexpand#1
1278 found.\MessageBreak
1279 The~command~can~not~be~adapted~on~the~fly~to~support~tagging.\MessageBreak
1280 The~heading~command~using~this~\noexpand\secdef~should~be~
1281 reimplemented\MessageBreak with~templates}{~}
1282 \IfBooleanTF{#3}{#2{#5}}{#1[#4]{#5}}
1283 }
1284 }

```

A helper command to reprocess the argument as key-val

```

1285 \NewDocumentCommand\__head_process_shorttitle:nN{={shorttitle}mm}
1286 { \tl_set:Nn#2{#1} }

```

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

11.8.4 Core L^AT_EX

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

```

1287 \AddToHook{class/article/after}[head/example]
1288 {
1289   \DeclareInstance{heading}{part}{display}
1290   {
1291     , name           = part
1292     , level          = -1
1293     , before-vspace  = 4ex
1294     , after-vspace   = 3ex
1295     , mark-cmd       = \partmark {#1}
1296     , text           = \partname
1297     , punct          =
1298     , heading-decls  = \raggedright\bfseries\Large
1299     , title-decls    = \huge
1300     , headformat-instance = part
1301   }
1302   \DeclareInstance{headformat}{part}{display}
1303   {
1304     , heading-indent = 0pt
1305     , separator-a    = \nobreakspace
1306     , separator-b    = \par \nobreak \vspace{20pt}
1307   }
1308   \DeclareDocumentCommand \part {s = {shorttitle}o m}
1309   { \ParseLaTeXeHeading {part} {#1} {#2} {#3} }
1310   \__head_setup_default_heading:
1311 }

1312 \AddToHook{class/report/after}[head/example]
1313 {
1314   \DeclareDocumentCommand \part {s = {shorttitle}o m}
1315   { \ParseLaTeXeHeading {part} {#1} {#2} {#3} }
1316   \DeclareDocumentCommand \chapter {s = {shorttitle}o m}
1317   {
1318     \ParseLaTeXeHeading {chapter}{#1} {#2} {#3}
1319   }

```

```

1320     \__head_setup_default_heading:
1321   }
1322   \AddToHook{class/book/after}[head/example]
1323   {
1324     \DeclareDocumentCommand \part {s ={\shorttitle}o m}
1325     { \ParseLaTeXeHeading {part} {#1} {#2} {#3} }
1326     \DeclareDocumentCommand \chapter {s ={\shorttitle}o m}
1327     {
1328       \if@mainmatter
1329         \ParseLaTeXeHeading {chapter}{#1} {#2} {#3}
1330       \else
1331         \IfBooleanTF{#1}
1332         {% starred:
1333          \ParseLaTeXeHeading {chapter}{\BooleanTrue} {#2} {#3}
1334         }
1335         {\IfNoValueTF{#2}
1336          {\ParseLaTeXeHeading {chapter}{\BooleanTrue} {\shorttitle={#3}} {#3}}
1337          {\ParseLaTeXeHeading {chapter}{\BooleanTrue} {#2} {#3}}
1338         }
1339       \fi
1340     }
1341     \__head_setup_default_heading:
1342   }

1343   \cs_new_protected:Npn \__head_setup_default_heading:
1344   {

\section
1345     \DeclareDocumentCommand \section {s ={\shorttitle}o m}
1346     { \ParseLaTeXeHeading {section} {##1} {##2} {##3} }
(End of definition for \section. This function is documented on page ??.)

\subsection
1347     \DeclareDocumentCommand \subsection {s ={\shorttitle}o m}
1348     { \ParseLaTeXeHeading {subsection} {##1} {##2} {##3} }
(End of definition for \subsection. This function is documented on page ??.)

\subsubsection
1349     \DeclareDocumentCommand \subsubsection {s ={\shorttitle}o m}
1350     { \ParseLaTeXeHeading {subsubsection} {##1} {##2} {##3} }
(End of definition for \subsubsection. This function is documented on page ??.)

\paragraph
1351     \DeclareDocumentCommand \paragraph {s ={\shorttitle}o m}
1352     { \ParseLaTeXeHeading {paragraph} {##1} {##2} {##3} }
(End of definition for \paragraph. This function is documented on page ??.)

\subparagraph
1353     \DeclareDocumentCommand \subparagraph {s ={\shorttitle}o m}
1354     { \ParseLaTeXeHeading {subparagraph} {##1} {##2} {##3} }
1355   } %end of command
(End of definition for \subparagraph. This function is documented on page ??.)

1356 \end{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.

```
1357 \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 L^AT_EX

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.

⁴Maybe 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 $\text{\LaTeX} 2_{\epsilon}$ 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 ($\text{\TeX}$ ’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: `Opt`

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 L^AT_EX 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.*)

```

1358 \DeclareTemplateInterface{heading}{display-v1}{9}
1359 {
1360   , name           : tokenlist
1361   , parent-name    : tokenlist
1362   , reset-counter  : tokenlist
1363   , level          : integer = 0
1364   , placement      : choice {page , top , normal } = normal
1365   , mark-cmd       : function(1) =
1366   , para-indent    : choice { true , false } = false
1367   , before-vspace  : skip = Opt
1368   , penalty        : integer = \c_max_int
1369   , after-penalty-vspace : skip = Opt
1370   , after-vspace   : skip = Opt
1371   , start-code     : tokenlist =      % no default values!
1372   , final-code     : tokenlist =      % no default values!
1373   , prefix         : tokenlist = \NoValue
1374   , heading-decls  : tokenlist = \normalfont
1375   , prefix-decls   : tokenlist =
1376   , number-decls   : tokenlist =
1377   , title-decls    : tokenlist =
1378   , subtitle-decls : tokenlist =
1379   , quote-decls    : tokenlist =
1380   , headformat-instance : tokenlist = std
1381   , number-format  : function(1) = \theheading
1382   , contents-extra : tokenlist =
1383 }
1384 \DeclareTemplateCode{heading}{display-v1}{9}
1385 {
1386   , name           = \l__head_name_tl
1387   , level          = \l__head_level_int
1388   , parent-name    = \l__head_pname_tl
1389   , reset-counter  = \l__head_reset_cnt_tl
1390   , placement      = {
1391     ,page = \__head_debug_typeout:n{ A~ page~ heading }
1392             \tl_set:Nn \l__head_placement_tl { page }
1393     ,top  = \__head_debug_typeout:n{ A~ top~ heading }
1394             \tl_set:Nn \l__head_placement_tl { top }
1395     ,normal = \__head_debug_typeout:n{ A~ normal~ heading }
1396             \tl_set:Nn \l__head_placement_tl { normal }
1397   }
1398   , mark-cmd       = \__head_mark_cmd:n
1399   , para-indent    = {
1400     ,true  = \@afterindenttrue
1401     ,false = \@afterindentfalse
1402   }

```

```

1403 , before-vspace = \l__head_before_skip
1404 , penalty       = \l__head_penalty_int
1405 , after-penalty-vspace = \l__head_after_penalty_skip
1406 , after-vspace  = \l__head_after_skip
1407 , start-code    = \l__head_start_code_tl
1408 , final-code    = \l__head_final_code_tl
1409 , prefix        = \l__head_typeset_prefix_tl
1410 , headformat-instance = \l__head_headformat_instance_tl
1411 , heading-decls = \l__head_heading_decls_tl
1412 , prefix-decls  = \l__head_prefix_decls_tl
1413 , number-decls  = \l__head_number_decls_tl
1414 , title-decls   = \l__head_title_decls_tl
1415 , subtitle-decls = \l__head_subtitle_decls_tl
1416 , quote-decls   = \l__head_quote_decls_tl
1417 , number-format = \__head_number_format:n
1418 , contents-extra = \l__head_contents_extra_tl
1419 }
1420 {
1421   \tl_set_eq:Nc \theheading { the \l__head_name_tl }
1422   \tl_set:Nc\l__head_saved_secnumdepth_tl{\int_use:N\c@secnumdepth}
1423   \__head_show_arguments:nnnnnnnnn
1424     {#1}{#2}{#3}{#4}{#5}{#6}{#7}{#8}{#9}
1425   \tl_if_empty:oF {#1} { \SetTemplateKeys{heading}{display}{#1} }
1426   \tl_if_empty:oT \l__head_start_code_tl
1427   { \str_case:VnF \l__head_placement_tl
1428     {
1429       { page }
1430       { \tl_set:Nn \l__head_start_code_tl
1431         {
1432           \if@openright
1433             \cleardoublepage
1434           \else
1435             \clearpage
1436           \fi
1437           \thispagestyle{plain}
1438           \if@twocolumn
1439             \onecolumn
1440             \@tempswattrue
1441           \else
1442             \@tempswafalse
1443           \fi
1444           \null\vfil
1445         }
1446       }
1447       { top }
1448       { \tl_set:Nn \l__head_start_code_tl
1449         {
1450           \if@openright\cleardoublepage\else\clearpage\fi
1451           \thispagestyle{plain}
1452           \global\@topnum\z@
1453         }
1454       }
1455     }
1456   { \tl_clear:N \l__head_start_code_tl }

```

```

1457 }
1458 \tl_if_empty:oT \l__head_final_code_tl
1459 { \str_case:VnF \l__head_placement_tl
1460   {
1461     { page }
1462     { \tl_set:Nn \l__head_final_code_tl
1463       {
1464         \vfil\newpage
1465         \if@twoside
1466           \if@openright
1467             \null
1468             \thispagestyle{empty}
1469           \newpage
1470         \fi
1471       \fi
1472       \if@tempwa
1473         \twocolumn
1474       \fi
1475     }
1476   }
1477 }
1478 { \tl_set:Nn \l__head_final_code_tl { \@afterheading } }
1479 }
1480 \__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;

```

1481 \bool_set:Nn \l__head_unnumbered_bool
1482 { \bool_lazy_or:p:nn
1483   { \int_compare_p:nNn \l__head_level_int > \c@secnumdepth }
1484   { \bool_if_p:N #2 }
1485 }
1486 \bool_if:NTF \l__head_unnumbered_bool
1487 {
1488   \int_gset:Nn\c@secnumdepth{-99}
1489   \tl_clear:N \l__head_typeset_number_tl
1490 }
1491 {
1492   \@kernel@refstepcounter{ \l__head_name_tl }
1493   \protected@edef \l__head_typeset_number_tl
1494   {
1495     \__head_number_format:n { \l__head_name_tl }
1496   }
1497 }

```

End of inlined code.

```

1498 \__head_vertical_before_spacing:
1499 \UseTaggingSocket{sec/end}{\int_use:N\l__head_level_int}
1500 \UseTaggingSocket{sec/begin}
1501   {{\int_use:N\l__head_level_int}
1502     {tag=\UseStructureName{sec/\int_use:N\l__head_level_int}}}
1503 \__head_debug_typeout:n{use~ 'headformat'~instance:~
1504   \l__head_headformat_instance_tl }
1505 \use:e {

```

```

1506     \UseInstance{headformat} { \l__head_headformat_instance_tl }
1507         { \exp_not:o \UnusedTemplateKeys }
1508         { \exp_not:o { \l__head_typeset_prefix_tl } }
1509         { \exp_not:o { \l__head_typeset_number_tl } }
1510         { \exp_not:n { #3 } }
1511         { \exp_not:o { \use_i:nn #9 } }
1512         { \exp_not:o { \use_ii:nn #9 } }
1513     }
1514     \__head_handle_marks_etc:nnnnn {#4}{#5}{#6}{#7}{#8}
1515     \int_gset:Nn\c@secnumdepth{\l__head_saved_secnumdepth_tl}
1516     \par \nobreak
1517     \skip_vertical:N \l__head_after_skip
1518     \l__head_final_code_tl
1519     \ignorespaces
1520 }

```

heading runin-v1 (*templ.*)

```

1521 \DeclareTemplateCopy{heading}{runin-v1}{display-v1}
1522 \DeclareTemplateCode{heading}{runin-v1}{9}
1523 {
1524     , name           = \l__head_name_tl
1525     , level          = \l__head_level_int
1526     , parent-name    = \l__head_pname_tl
1527     , reset-counter  = \l__head_reset_cnt_tl
1528     , placement      = {
1529         page         = \typeout{ ^^JA~ runin~ page~ placement~ heading-makes-no-sense
1530                         (top-used)}
1531         \tl_set:Nn \l__head_placement_tl { top }
1532         ,top         = \__head_debug_typeout:n{ A~ top~ heading }
1533         \tl_set:Nn \l__head_placement_tl { top }
1534         ,normal      = \__head_debug_typeout:n{ A~ normal~ heading }
1535         \tl_set:Nn \l__head_placement_tl { normal }
1536     }
1537     , mark-cmd       = \__head_mark_cmd:n
1538     , para-indent    = {
1539         ,true         = %\typeout{para-indent~ setting~ ignored}
1540         ,false        = %\typeout{para-indent~ setting~ ignored}
1541     }
1542     , before-vspace  = \l__head_before_skip
1543     , penalty         = \l__head_penalty_int
1544     , after-penalty-vspace = \l__head_after_penalty_skip
1545     , after-vspace   = \l__head_after_skip
1546     , start-code      = \l__head_start_code_tl
1547     , final-code      = \l__head_final_code_tl
1548     , prefix          = \l__head_typeset_prefix_tl
1549     , headformat-instance = \l__head_headformat_instance_tl
1550     , heading-decls   = \l__head_heading_decls_tl
1551     , prefix-decls    = \l__head_prefix_decls_tl
1552     , number-decls    = \l__head_number_decls_tl
1553     , title-decls     = \l__head_title_decls_tl
1554     , subtitle-decls  = \l__head_subtitle_decls_tl
1555     , quote-decls     = \l__head_quote_decls_tl
1556     , number-format   = \__head_number_format:n
1557     , contents-extra  = \l__head_contents_extra_tl

```

```

1558 }
1559 {
1560   \__head_show_arguments:nnnnnnnnn
1561     {#1}{#2}{#3}{#4}{#5}{#6}{#7}{#8}{#9}
1562   \tl_set:N\l__head_saved_secnumdepth_tl{\int_use:N\c@secnumdepth}
1563   \tl_set_eq:Nc \theheading { the \l__head_name_tl }
1564   \tl_if_empty:oF {#1} { \SetTemplateKeys{heading}{runin}{#1} }
1565   \tl_if_empty:oT \l__head_start_code_tl
1566   {
1567     \str_case:Vn \l__head_placement_tl
1568     {
1569       { top }    { \tl_set:Nn \l__head_start_code_tl {\clearpage} } }
1570   }
1571 }
1572 \__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;

```

1573 \bool_set:Nn \l__head_unnumbered_bool
1574   { \bool_lazy_or_p:nn
1575     { \int_compare_p:nNn \l__head_level_int > \c@secnumdepth }
1576     { \bool_if_p:N #2 }
1577   }
1578 \bool_if:NTF \l__head_unnumbered_bool
1579   {
1580     \int_gset:Nn\c@secnumdepth{-99}
1581     \tl_clear:N \l__head_typeset_number_tl
1582   }
1583   {
1584     \@kernel@refstepcounter{ \l__head_name_tl }
1585     \protected@edef \l__head_typeset_number_tl
1586       {
1587         \__head_number_format:n { \l__head_name_tl }
1588       }
1589   }

```

End of inlined code.

```

1590 \__head_vertical_before_spacing:
1591 \UseTaggingSocket{sec/end}{\int_use:N\l__head_level_int}
1592 \UseTaggingSocket{sec/begin}
1593   {{\int_use:N\l__head_level_int}{tag=\UseStructureName{sec/\int_use:N\l__head_level_int}}}
1594 \UseTaggingSocket{sec/title/init}{\int_use:N\l__head_level_int}
1595 \def \@svsechd {
1596   \__head_debug_typeout:n{use~ 'headformat'~instance:~
1597     \l__head_headformat_instance_tl }
1598   \use:e {
1599     \UseInstance{headformat} { \l__head_headformat_instance_tl }
1600     { \exp_not:o \UnusedTemplateKeys }
1601     { \exp_not:o { \l__head_typeset_prefix_tl } }
1602     { \exp_not:o { \l__head_typeset_number_tl } }
1603     { \exp_not:n { #3 } }
1604     { \exp_not:o { \use_i:nn #9 } }
1605     { \exp_not:o { \use_ii:nn #9 } }
1606   }

```

```

1607     \__head_handle_marks_etc:nnnnn {#4}{#5}{#6}{#7}{#8}
1608     \int_gset:Nn\c@secnumdepth{\l__head_saved_secnumdepth_tl}
1609   }
1610   \@nobreakfalse
1611   \global\@noskipsectrue
1612   \everypar{%
1613     \if@noskipsec
1614       \global\@noskipsecfalse
1615       {\setbox\z@\lastbox}
1616       \clubpenalty\@M
1617       \@svsechd
1618       \unskip
1619       \UseTaggingSocket{sec/title/split}
1620       \skip_horizontal:N \l__head_after_skip
1621     \else
1622       \clubpenalty \@clubpenalty
1623       \everypar{}}%
1624   \fi
1625 }
1626 \l__head_final_code_tl
1627 \ignorespaces
1628 }

```

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

```

1629 \DeclareTemplateInterface{headformat}{display-v1}{6}
1630 {
1631   , heading-indent      : length = Opt
1632   , title-format        : function(1) = #1
1633   , prefix-number-sep   : tokenlist = \WordSpaceAmount{1}
1634   , number-title-sep    : tokenlist = 20pt
1635 }

```

headformat hang-v1 (*templ.*)

```

1636 \DeclareTemplateInterface{headformat}{hang-v1}{6}
1637 {
1638   , heading-indent      : length = Opt
1639   , title-format        : function(1) = #1
1640   , prefix-number-sep   : tokenlist = \WordSpaceAmount{1}
1641   , number-title-sep    : tokenlist = 1em
1642 }

```

headformat runin-v1 (*templ.*)

```

1643 \DeclareTemplateInterface{headformat}{runin-v1}{6}
1644 {
1645   , heading-indent      : length = Opt
1646   , title-format        : function(1) = #1
1647   , prefix-number-sep   : tokenlist = \WordSpaceAmount{1}
1648   , number-title-sep    : tokenlist = 1em
1649 }

```

headformat display-v1 (*templ.*)

```

1650 \DeclareTemplateCode{headformat}{display-v1}{6}
1651   % args: keys,prefix,number,title,subtitle,quotation

```

```

1652 {
1653   , heading-indent      = \l__head_heading_indent_dim
1654   , title-format        = \__head_title_format:n
1655   , prefix-number-sep   = \l__head_prefix_number_sep_tl
1656   , number-title-sep    = \l__head_number_title_sep_tl
1657 }
1658 {
1659   \__head_show_arguments:nnnnnn {#1}{#2}{#3}{#4}{#5}{#6}
1660   \tl_if_empty:oF {#1} { \SetTemplateKeys{headformat}{display-v1}{#1} }
1661   \group_begin:
1662     \UseTaggingSocket{sec/title/begin}{\int_use:N\l__head_level_int}{#4}}
1663     \normalfont
1664     \interlinepenalty \@M
1665     \l__head_heading_decls_tl{}
1666     \bool_if:NTF \l__head_unnumbered_bool
1667     {
1668       \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
1669       {
1670         \skip_horizontal:N \l__head_heading_indent_dim
1671         \MakeLinkTarget[\l__head_name_tl]{}
1672       }
1673       {
1674         \MakeLinkTarget[\l__head_name_tl]{}
1675         \skip_horizontal:N \l__head_heading_indent_dim
1676       }
1677     }
1678     {
1679       \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
1680       {
1681         \skip_horizontal:N \l__head_heading_indent_dim
1682         \MakeLinkTarget{\l__head_name_tl}
1683       }
1684       {
1685         \MakeLinkTarget{\l__head_name_tl}
1686         \skip_horizontal:N \l__head_heading_indent_dim
1687       }
1688       \IfNoValueF{#2}
1689       {
1690         { \l__head_prefix_decls_tl #2 }
1691         \nobreak
1692         \skip_horizontal:n { \l__head_prefix_number_sep_tl }
1693       }
1694       \l__head_number_decls_tl
1695       #3
1696       \par\nobreak
1697       \skip_vertical:n { \l__head_number_title_sep_tl }
1698     }
1699     \l__head_title_decls_tl
1700     \__head_title_format:n {#4}
1701     \par
1702     \UseTaggingSocket{sec/title/end}
1703   \group_end:
1704 }

```

headformat hang-v1 (*templ.*)

```

1705 \DeclareTemplateCode{headformat}{hang-v1}{6}
1706 {
1707   , heading-indent      = \l__head_heading_indent_dim
1708   , title-format        = \__head_title_format:n
1709   , prefix-number-sep   = \l__head_prefix_number_sep_tl
1710   , number-title-sep    = \l__head_number_title_sep_tl
1711 }
1712 {
1713   \__head_show_arguments:nnnnnn {#1}{#2}{#3}{#4}{#5}{#6}
1714   \tl_if_empty:oF {#1} { \SetTemplateKeys{headformat}{hang-v1}{#1} }
1715   \group_begin:
1716     \UseTaggingSocket{sec/title/init}{\int_use:N\l__head_level_int}
1717     \normalfont
1718     \interlinepenalty \@M
1719     \l__head_heading_decls_tl{}
1720     \bool_if:NTF \l__head_unnumbered_bool
1721     {
1722       \tl_set:Nn\l__head_tmpa_tl
1723       {
1724         \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
1725         {
1726           \skip_horizontal:N \l__head_heading_indent_dim
1727           \MakeLinkTarget[\l__head_name_tl]{}
1728         }
1729         {
1730           \MakeLinkTarget[\l__head_name_tl]{}
1731           \skip_horizontal:N \l__head_heading_indent_dim
1732         }
1733       }
1734     }
1735     {
1736       \tl_set:Nn \l__head_tmpa_tl
1737       {
1738         \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
1739         {
1740           \skip_horizontal:N \l__head_heading_indent_dim
1741           \MakeLinkTarget{\l__head_name_tl}
1742         }
1743         {
1744           \MakeLinkTarget{\l__head_name_tl}
1745           \skip_horizontal:N \l__head_heading_indent_dim
1746         }
1747         \IfNoValueF{#2}
1748         {
1749           { \l__head_prefix_decls_tl #2 }
1750           \nobreak
1751           \skip_horizontal:n { \l__head_prefix_number_sep_tl }
1752         }
1753         { \l__head_number_decls_tl #3 }
1754         \skip_horizontal:n { \l__head_number_title_sep_tl }
1755       }
1756     }
1757     \UseTaggingSocket{sec/title/hang}

```

```

1758     {{\int_use:N\l__head_level_int}\l__head_unnumbered_bool{\l__head_tmpa_tl}{#4}}
1759     {
1760         \@hangfrom
1761         {
1762             \l__head_tmpa_tl
1763         }
1764     }
1765     \l__head_title_decls_tl
1766     \__head_title_format:n {#4}
1767     \par
1768     \group_end:
1769 }

```

headformat runin-v1 (*templ.*)

```

1770 \DeclareTemplateCode{headformat}{runin-v1}{6}
1771 {
1772     , heading-indent      = \l__head_heading_indent_dim
1773     , title-format       = \__head_title_format:n
1774     , prefix-number-sep = \l__head_prefix_number_sep_tl
1775     , number-title-sep  = \l__head_number_title_sep_tl
1776 }
1777 {
1778     \__head_show_arguments:nnnnnn {#1}{#2}{#3}{#4}{#5}{#6}
1779     \tl_if_empty:oF {#1} { \SetTemplateKeys{headformat}{runin-v1}{#1} }
1780     \group_begin:
1781         \normalfont
1782         \interlinepenalty \@M
1783         \l__head_heading_decls_tl{}
1784         \bool_if:NTF \l__head_unnumbered_bool
1785         {
1786             \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
1787             {
1788                 \skip_horizontal:N \l__head_heading_indent_dim
1789                 \MakeLinkTarget[\l__head_name_tl]{}
1790             }
1791             {
1792                 \MakeLinkTarget[\l__head_name_tl]{}
1793                 \skip_horizontal:N \l__head_heading_indent_dim
1794             }
1795         }
1796         {
1797             \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
1798             {
1799                 \skip_horizontal:N \l__head_heading_indent_dim
1800                 \MakeLinkTarget{\l__head_name_tl}
1801             }
1802             {
1803                 \MakeLinkTarget{\l__head_name_tl}
1804                 \skip_horizontal:N \l__head_heading_indent_dim
1805             }
1806         }
1807         \UseTaggingSocket{sec/title/number}{\int_use:N\l__head_level_int}
1808         {
1809             \IfNoValueF{#2}

```

```

1810         {
1811             { \l__head_prefix_decls_tl #2 }
1812             \nobreak
1813             \skip_horizontal:n { \l__head_prefix_number_sep_tl }
1814         }
1815         \l__head_number_decls_tl
1816         #3
1817     }
1818 }
1819 \skip_horizontal:n { \l__head_number_title_sep_tl }
1820 }
1821 \l__head_title_decls_tl
1822 \__head_title_format:n {#4}
1823 \group_end:
1824 }

```

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		\AssignTaggingSocketPlug 736,	
\!	227, 229	737, 738, 829, 830, 831, 889, 890, 891	
\,	228, 230	\AtBeginDocument 1057	
\.	227, 229		
\:	228, 230		
\;	228, 230		
\?	227, 229		
\@startsection	21		
_	344, 357, 368		
_	17, 18		
A		B	
\addcontents	14, 16, 67	\baselineskip 13–15	
\addcontentsline	22, 53, 54, 1014	\bfseries 1035, 1050,	
\addcontentslinebookmark	977, 1003	1066, 1077, 1087, 1097, 1109, 1298	
\addcontentslinebookmarkOff	977, 1013	bool commands:	
\addcontentslinebookmarkOn	977	\bool_gset_false:N 18	
\addcontentslinebookmarkReset		\bool_gset_true:N 13	
	979, 987, 990, 1022	\bool_if:N 24,	
\addpenalty	38, 40–42, 52, 551, 555, 556	26, 100, 125, 126, 170, 172, 174,	
\addpenaltywithfil	548, 556	427, 507, 521, 536, 653, 751, 799,	
\AddPunct	232	804, 844, 900, 932, 939, 971, 1005,	
\addtocontents	1054, 1055	1016, 1486, 1578, 1666, 1720, 1784	
\AddToHook		\bool_if_p:N 936, 1484, 1576	
	69, 591, 981, 1237, 1287, 1312, 1322	\bool_lazy_and_p:nn . . 115, 1215, 1222	
\addvspace	42, 962, 1054, 1055	\bool_lazy_any:nTF 1208	
\AssignSocketPlug	167, 181	\bool_lazy_or_p:nn . . 934, 1482, 1574	
\AssignStructureRole	808	\bool_new:N 8, 59, 168	
		\bool_set:Nn 113, 933, 1481, 1573	
		\bool_set_false:N 86, 436, 656	
		\bool_set_true:N 81, 809	
		\BooleanFalse 10, 100	
		\BooleanTrue 10, 100, 1333, 1336, 1337	
		box commands:	
		\box_use_drop:N 863	
		\box_wd:N 856, 857	

C	
\centering	1035
\chapter	4, 5, 9, 13, 17, 19–21, 54–56, 61, 70, 1267, 1268, 1271, 1316, 1326
\chaptermark	13, 66, 1053
\cleardoublepage	445, 461, 1433, 1450
\clearpage	12, 53, 66, 447, 461, 661, 1435, 1450, 1569
\clubpenalty	707, 713, 1616, 1622
cmd commands:	
\cmd_arg_spec:N	1203
\cs	540
cs commands:	
\cs_generate_variant:Nn	1235
\cs_gset_protected:Npe	23, 25
\cs_if_exist:NTF	1058
\cs_if_exist_p:N	1216, 1223
\cs_new:Npn	31, 40, 199, 212, 927, 931, 950, 995, 1058, 1188, 1194, 1199, 1207
\cs_new_eq:NN	9, 10
\cs_new_protected:Npn	11, 16, 21, 28, 29, 70, 169, 171, 173, 231, 232, 546, 556, 588, 788, 1343
\cs_parameter_spec:N	1210, 1218, 1225, 1235
\cs_set_eq:NN	467, 549, 553, 663
\cs_set_protected:Npn	585
\cs_to_str:N	1216, 1218, 1241
\csname	594
D	
\DebugHeadingsOff	22, 28
\DebugHeadingsOn	22, 28
\DeclareDocumentCommand	1141, 1255, 1271, 1308, 1314, 1316, 1324, 1326, 1345, 1347, 1349, 1351, 1353
\DeclareInstance	64, 778, 1028, 1040, 1059, 1069, 1080, 1090, 1102, 1114, 1121, 1128, 1132, 1136, 1157, 1169, 1173, 1183, 1289, 1302
\DeclareInstanceCopy	1120, 1125, 1126, 1127
\DeclareRobustCommand	60
\DeclareTemplateCode	34, 371, 597, 720, 812, 874, 1384, 1522, 1650, 1705, 1770
\DeclareTemplateCopy	34, 1521
\DeclareTemplateInterface	244, 289, 334, 347, 360, 1358, 1629, 1636, 1643
\DeclareTemplateInterfaceEquivalents	280, 284, 325, 329
\def	60, 227, 229, 686, 1595
dim commands:	
\dim_add:Nn	857
\dim_compare:nNnTF	510, 563, 567, 569, 577, 752, 760, 901, 909, 951, 963, 972, 1668, 1679, 1724, 1738, 1786, 1797
\dim_set:Nn	566, 858
\dim_sub:Nn	793
\l_tmpa_dim	566, 574, 577, 578
\c_zero_dim	563, 570, 577
\DocumentMetadata	3, 31
E	
\EditTemplateDefaults	34
\else	236, 435, 446, 453, 461, 593, 712, 959, 989, 1163, 1170, 1330, 1434, 1441, 1450, 1621
\endcsname	592, 594
\everypar	45, 703, 714, 958, 1612, 1623
exp commands:	
\exp_args:Ne	1190, 1201
\exp_args:Nne	127, 136, 144, 156
\exp_not:N	98, 1160, 1166, 1176
\exp_not:n	33, 34, 35, 36, 37, 38, 42, 43, 44, 45, 46, 47, 48, 49, 50, 74, 75, 99, 101, 102, 103, 104, 105, 106, 107, 108, 210, 508, 515, 516, 517, 518, 519, 520, 691, 692, 693, 694, 695, 696, 1025, 1164, 1165, 1179, 1180, 1249, 1507, 1508, 1509, 1510, 1511, 1512, 1600, 1601, 1602, 1603, 1604, 1605
\expandafter	594
\ExpandArgs	1248, 1263
extend (plug)	112
F	
\fbox	4
\fi	71, 239, 240, 437, 448, 455, 461, 485, 486, 489, 595, 715, 956, 967, 991, 1153, 1163, 1184, 1339, 1436, 1443, 1450, 1470, 1471, 1474, 1624
\filbreak	40, 52
\final-code	13
\font	222, 223, 224
\fontdimen	222, 223, 224
\frenchspacing	31, 227
G	
\global	463, 702, 705, 1452, 1611, 1614
\glueexpr	221
group commands:	
\group_begin:	561, 735, 789, 827, 887, 1661, 1715, 1780
\group_end:	581, 776, 795, 871, 925, 1703, 1768, 1823
H	
\hangindent	49, 856

<code>\hbox</code>	843	<code>__head_heading_format:n</code>	410, 637, 769, 860, 918
hbox commands:			
<code>\hbox_overlap_left:n</code>	863	<code>\l__head_heading_indent_dim</code>	49, 723, 752, 753, 757, 760, 761,
<code>\hbox_overlap_right:n</code>	791		765, 814, 862, 877, 901, 902, 906,
head commands:			
<code>\head_debug_off:</code>	22, 11, 16, 29		909, 910, 914, 1653, 1668, 1670,
<code>\head_debug_on:</code>	22, 11, 11, 28		1675, 1679, 1681, 1686, 1707, 1724,
head internal commands:			
<code>\l__head_after_penalty_skip</code>	397, 510, 511, 624,	<code>\l__head_instance_keys_tl</code>	26, 60, 92, 93, 99
	951, 953, 963, 965, 972, 973, 1405, 1544	<code>\l__head_instance_name_tl</code> 26, 27,	60, 94, 96, 98, 129, 132, 133, 138,
<code>\l__head_after_skip</code>	398, 523,		140, 141, 146, 149, 150, 158, 161, 162
<code>\l__head_before_skip</code>	395, 622, 952, 962, 1403, 1542	<code>\l__head_label_tl</code>	25, 29, 60, 106, 197, 202, 206, 216
<code>\l__head_bookmark_tl</code>	53, 80, 85, 104, 185	<code>\l__head_left_margin_dim</code>	729, 746, 815, 857, 858, 861
<code>\l__head_column_spanning_bool</code> ..	39, 389, 427,	<code>\l__head_level_int</code>	374, 498, 500, 501, 505, 600,
	436, 507, 521, 536, 616, 653, 656, 971		680, 682, 683, 685, 935, 1387, 1483,
<code>__head_column_vbox_adjusted:nnn</code>	788, 788		1499, 1501, 1502, 1525, 1575, 1591,
<code>\l__head_contents_extra_tl</code>	419, 646, 1024, 1418, 1557	<code>__head_mark_cmd:n</code>	390, 617, 999, 1398, 1537
<code>\g__head_count_int</code> 119, 175, 178, 179		<code>\l__head_name_tl</code>	373,
<code>__head_debug:n</code>	9, 9, 23		422, 599, 651, 754, 756, 762, 764,
<code>\g__head_debug_bool</code> ..	8, 13, 18, 24, 26		846, 849, 903, 905, 911, 913, 946,
<code>__head_debug_gset:</code>	11, 14, 19, 21		1003, 1007, 1014, 1018, 1386, 1421,
<code>__head_debug_typeof:n</code>	9, 10, 25, 32, 33,		1492, 1495, 1524, 1563, 1584, 1587,
	34, 35, 36, 37, 38, 41, 42, 43, 44, 45,		1671, 1674, 1682, 1685, 1727, 1730,
	46, 47, 48, 49, 50, 72, 73, 96, 201,		1741, 1744, 1789, 1792, 1800, 1803
	205, 210, 215, 235, 237, 380, 382,	<code>\l__head_nameref_tl</code> ..	56, 88, 105, 188
	384, 386, 431, 503, 547, 609, 611,	<code>\l__head_number_decls_tl</code>	406, 633, 1413, 1552, 1694, 1753, 1815
	613, 654, 687, 1025, 1143, 1391,	<code>__head_number_format:n</code>	412, 639, 1417, 1495, 1556, 1587
	1393, 1395, 1503, 1532, 1534, 1596	<code>\l__head_number_tag_tl</code>	418, 645
<code>__head_determine_number_-</code>		<code>\l__head_number_title_sep_tl</code> ...	1656, 1697, 1710, 1754, 1775, 1819
<code>typesetting:N</code>	72, 74, 496, 678, 931, 931	<code>\l__head_number_tl</code>	52,
<code>__head_determine_penalty:</code>	495, 677, 927, 927, 1480, 1572		60, 517, 693, 742, 835, 895, 941, 947
<code>\l__head_final_code_tl</code>	1408,	<code>\l__head_order_clist</code> ...	724, 818, 878
	1458, 1462, 1478, 1518, 1547, 1626	<code>\l__head_page_type_tl</code>	27,
<code>__head_find_label:w</code>	76, 199, 199		60, 124, 130, 134, 147, 151, 159, 163
<code>__head_find_label_aux:w</code>	30, 208, 212, 217	<code>\l__head_penalty_int</code>	396, 623, 928, 929, 961, 1404, 1543
<code>__head_handle_marks_etc:nnnnn</code> .	531, 698, 995, 995, 1514, 1607	<code>\l__head_placement_end_code_tl</code> .	36, 37, 44, 400, 474,
<code>\l__head_headformat_instance_tl</code>	403, 504, 514, 630, 688,		478, 493, 542, 627, 670, 673, 675, 717
	690, 1410, 1504, 1506, 1549, 1597, 1599	<code>\l__head_placement_start_code_tl</code>	36-39, 44, 53, 399, 439, 443,
<code>\l__head_heading_decls_tl</code>	404, 631, 750, 840, 899, 1146, 1164,		460, 471, 626, 658, 661, 667, 955, 970
	1179, 1411, 1550, 1665, 1719, 1783		

\l__head_placement_tl	\l__head_title_format_tl
..... 60 , 381 , 383 , 385 , 387 ,	 1147 , 1165 , 1180
429 , 433 , 440 , 475 , 608 , 610 , 612 ,	\l__head_title_tl
614 , 659 , 671 , 1392 , 1394 , 1396 ,	 25 , 30 , 60 , 83 , 84 , 85 , 88 ,
1427 , 1459 , 1531 , 1533 , 1535 , 1567	101 , 203 , 207 , 210 , 213 , 743 , 836 , 896
\l__head_pname_tl 376 , 602 , 1388 , 1526	\l__head_tmpa_tl .. 52 , 1247 , 1249 ,
\l__head_prefix_decls_tl	1262 , 1264 , 1722 , 1736 , 1758 , 1762
..... 1412 , 1551 , 1690 , 1749 , 1811	\l__head_toc_tl .. 55 , 78 , 83 , 102 , 187
\l__head_prefix_number_sep_tl ..	\l__head_typeset_number_tl
.. 1655 , 1692 , 1709 , 1751 , 1774 , 1813	.. 1489 , 1493 , 1509 , 1581 , 1585 , 1602
__head_prep_decls_and_auxi:nnNN	\l__head_typeset_prefix_tl
..... 1191 , 1194	 1409 , 1508 , 1548 , 1601
__head_prep_decls_and_auxii:nNwNN	\l__head_unnumbered_bool
..... 1196 , 1199	 51 , 81 , 86 , 100 , 126 , 190 ,
__head_prep_decls_and_auxiii:nnNnNN	192 , 378 , 604 , 751 , 844 , 900 , 932 ,
..... 1202 , 1207	933 , 939 , 1005 , 1016 , 1481 , 1486 ,
__head_prep_decls_and_format_-	1573 , 1578 , 1666 , 1720 , 1758 , 1784
from_tl:nNN	\l__head_verso_page_bool
1145 , 1188 , 1188	 113 , 125 , 168 , 170 , 172 , 174
__head_process_shorttitle:nN ..	\l__head_verso_page_bool
..... 1247 , 1262 , 1285	168
__head_punct_format:n	__head_vertical_before_spacing:
413 , 640	 497 , 679 , 950 , 950 , 1498 , 1590
\l__head_punct_tl .. 52 , 402 , 629 , 943	headformat (type)
\l__head_quote_decls_tl	243
..... 409 , 636 , 1416 , 1555	headformat chapter (instance)
__head_quote_format:n	1114
416 , 643	headformat display (template) ..
\l__head_quote_tl ... 58 , 90 , 108 , 196	334 , 720
\l__head_reset_cnt_tl	headformat display-v1 (template) ...
..... 377 , 603 , 1389 , 1527	 1629 , 1650
\l__head_right_margin_dim	headformat hang (template)
..... 730 , 747 , 816 , 859	347 , 812
\l__head_running_tl 54 , 79 , 84 , 103 , 186	headformat hang-v1 (template) .
\l__head_saved_secnumdepth_tl ..	1636 , 1705
..... 60 , 423 ,	headformat paragraph (instance) ...
534 , 650 , 699 , 1422 , 1515 , 1562 , 1608	1128
__head_setup_default_heading: .	headformat part (instance)
..... 1310 , 1320 , 1341 , 1343	1114
__head_show_arguments:nnnnnn 31 ,	headformat runin (template) ...
31 , 733 , 825 , 885 , 1659 , 1713 , 1778	360 , 874
__head_show_arguments:nnnnnnnnnn	headformat runin-v1 (template) 1643 , 1770
..... 40 , 40 , 424 , 648 , 1423 , 1560	headformat section (instance)
\l__head_start_code_tl 1407 , 1426 ,	1121
1430 , 1448 , 1456 , 1546 , 1565 , 1569	headformat section-special (instance)
\l__head_subtitle_decls_tl	 1136
..... 408 , 635 , 1415 , 1554	headformat std (instance)
__head_subtitle_format:n .	1121
415 , 642	headformat subparagraph (instance) .
\l__head_subtitle_tl .	1128
57 , 89 , 107 , 195	headformat subsection (instance) ...
\l__head_text_decls_tl	1121
405 , 632	headformat subsection (instance) 1121
__head_text_format:n	heading (type)
411 , 638	242
\l__head_text_tl	heading chapter (instance)
52 ,	1040
401 , 516 , 628 , 692 , 741 , 834 , 894 , 942	heading display (template)
\l__head_title_decls_tl	244 , 371
..... 407 , 634 , 1414 , 1553 , 1699 , 1765 , 1821	heading display-v1 (template)
__head_title_format:n	1358
414 ,	heading paragraph (instance)
641 , 1654 , 1700 , 1708 , 1766 , 1773 , 1822	1090
	heading part (instance)
	1028
	heading runin (template)
	289 , 597
	heading runin-v1 (template)
	1521
	heading section (instance)
	1059
	heading subparagraph (instance) ...
	1102
	heading subsection (instance)
	1069
	heading subsubsection (instance) ...
	1080
	heading/instname (socket)
	111
	heading/record (socket)
	176

<code>\hspace</code>	18, 356, 367, 792	<code>\hspace</code>	793, 858
<code>\Huge</code>	1036, 1051	<code>\c_max_int</code>	33, 256, 301, 928, 1368
<code>\huge</code>	1035, 1050, 1299	<code>\interlinepenalty</code>	50, 749, 839, 898, 1664, 1718, 1782
I			
<code>\if...</code>	65	K	
<code>\IfBlankF</code>	998, 1001	kernel internal commands:	
<code>\IfBlankT</code>	1013	<code>_kernel_add_penalty:n</code>	556, 556, 586, 589
<code>\IfBlankTF</code>	1000	keys commands:	
<code>\IfBooleanT</code>	73	<code>\keys_define:nn</code>	182
<code>\IfBooleanTF</code>	77, 1282, 1331	<code>\keys_set_known:nnN</code>	93
<code>\ifcsname</code>	592	L	
<code>\ifdim</code>	1150, 1155	<code>\label</code>	4, 10, 25, 26, 29, 30, 76, 197, 199, 206, 212, 216
<code>\ifhmode</code>	233	<code>\Large</code>	1066, 1298
<code>\IfInstanceExistsF</code>	1142	<code>\large</code>	1077
<code>\IfInstanceExistsT</code>	145, 157	<code>\lastbox</code>	706, 1615
<code>\IfInstanceExistsTF</code>	128, 137	<code>\lastskip</code>	42
<code>\IfNoValueF</code>	1688, 1747, 1809	<code>\leavevmode</code>	71, 956
<code>\IfNoValueTF</code>	91, 1244, 1259, 1335	legacy commands:	
<code>\ifnum</code>	234	<code>\legacy_if:nTF</code>	559, 560
<code>\IfValueT</code>	74	<code>\legacy_if_p:n</code>	116
<code>\IfVersoHeadingF</code>	169	<code>\let</code>	987, 990
<code>\IfVersoHeadingT</code>	169	<code>\lineskip</code>	790
<code>\IfVersoHeadingTF</code>	19, 169	<code>\linewidth</code>	858, 861
<code>\ignorespaces</code>	544, 718, 1519, 1627	<code>\ltxlabsecIIdate</code>	5
instances:		<code>\ltxlabsecIIversion</code>	6
<code>headformat chapter</code>	1114	M	
<code>headformat paragraph</code>	1128	<code>\MakeLinkTarget</code>	21, 754, 756, 762, 764, 846, 849, 903, 905, 911, 913, 1671, 1674, 1682, 1685, 1727, 1730, 1741, 1744, 1789, 1792, 1800, 1803
<code>headformat part</code>	1114	<code>\MakeUppercase</code>	58, 68–70
<code>headformat section</code>	1121	<code>\markboth</code>	69
<code>headformat section-special</code>	1136	<code>\maxdepth</code>	41, 567, 568
<code>headformat std</code>	1121	<code>\maxdimen</code> 13, 15, 257, 302, 510, 951, 963, 972	
<code>headformat subparagraph</code>	1128	<code>\MessageBreak</code>	1252, 1253, 1267, 1269, 1278, 1279, 1281
<code>headformat subsection</code>	1121	mode commands:	
<code>headformat subsubsection</code>	1121	<code>\mode_if_horizontal:TF</code>	557
<code>heading chapter</code>	1040	<code>\mode_if_inner:TF</code>	558
<code>heading paragraph</code>	1090	N	
<code>heading part</code>	1028	<code>\nameref</code>	3, 4
<code>heading section</code>	1059	<code>\newcommand</code>	220, 977, 978, 979
<code>heading subparagraph</code>	1102	<code>\NewCommandCopy</code>	1236
<code>heading subsection</code>	1069	<code>\NewDocumentCommand</code>	10, 11, 60, 1285
<code>heading subsubsection</code>	1080	<code>\newif</code>	65, 594
int commands:		<code>\newpage</code>	479, 484, 1464, 1469
<code>\int_compare:nNnTF</code>	928	<code>\NewSocket</code>	111, 176
<code>\int_compare_p:nNn</code> ..	935, 1483, 1575	<code>\NewSocketPlug</code>	112, 177
<code>\int_gincr:N</code>	179		
<code>\int_gset:Nn</code>	534, 699, 940, 1488, 1515, 1580, 1608		
<code>\int_if_even_p:n</code>	117		
<code>\int_new:N</code>	175		
<code>\int_set:Nn</code>	929		
<code>\int_use:N</code> 119, 178, 423, 498, 500, 501, 505, 650, 680, 682, 683, 685,			

<code>\NewTaggingSocketPlug</code>	797, 802, 807	<code>\protect</code>	1007, 1018
<code>\NewTemplateType</code>	242, 243	<code>\providecommand</code>	69, 980
<code>\nobreak</code>	17, 343, 537, 1118, 1306, 1516, 1691, 1696, 1750, 1812	<code>\ProvidesExplPackage</code>	3
<code>\nobreakspace</code>		Q	
.	17, 18, 342, 355, 366, 1117, 1305	quark commands:	
<code>\noexpand</code>	1251, 1253, 1266, 1267, 1268, 1277, 1280	<code>\q_no_value</code>	29, 30, 76
<code>\noindent</code>	841	<code>\quark_if_no_value:nTF</code>	200, 214
<code>\NoNalue</code>	9	<code>\q_stop</code>	1196, 1199
<code>\nonfrenchspacing</code>	31, 227	R	
<code>\nopunct</code>	31, 231	<code>\raggedright</code>	1050, 1298
<code>\normalcolor</code>	46	<code>\refstepcounter</code>	21, 52
<code>\normalfont</code>	13, 15, 67, 68, 263, 308, 748, 838, 897, 1066, 1077, 1087, 1097, 1109, 1374, 1663, 1717, 1781	<code>\relax</code>	225, 233, 990, 1148, 1154, 1245, 1249
<code>\normalsize</code>	1087, 1097, 1109	<code>\renewcommand</code>	985
<code>\NoValue</code>	10–12, 16–18, 33, 66, 261, 306, 941, 942, 943, 1373	<code>\RenewCommandCopy</code>	983, 1238
<code>\null</code>	456, 482, 1444, 1467	<code>\RenewDocumentCommand</code>	1239
<code>\numberline</code>	1007, 1018	<code>\RestoreLastSkip</code>	13, 15, 46, 67
O			
<code>\onecolumn</code>	451, 1439	S	
<code>\openright</code>	65	<code>\SaveLastSkip</code>	13, 15, 46, 67
P			
<code>page (plug)</code>	177	scan commands:	
<code>\par</code>	14–17, 34, 59, 71, 273, 343, 535, 558, 775, 870, 956, 1118, 1166, 1306, 1516, 1696, 1701, 1767	<code>\scan_stop:</code>	586
<code>\paragraph</code>	19, 20, 69, 1351	sec commands:	
<code>\paragraphmark</code>	1098	<code>\sec_add_penalty_with_fil:n</code>	
<code>\parbox</code>	46		40, 41, 52, 468, 546, 546, 664
<code>\parindent</code>	779, 1050, 1134	<code>\sec_add_penalty_with_possible_-</code> <code>fil:n</code>	38, 40, 41, 52, 467, 546, 550, 554, 663, 960
<code>\ParseLaTeXeHeading</code>		<code>\secdef</code>	21
.	8–10, 25, 29, 54, 70, 1186, 1245, 1248, 1256, 1260, 1263, 1272, 1309, 1315, 1318, 1325, 1329, 1333, 1336, 1337, 1346, 1348, 1350, 1352, 1354	<code>\secdef</code>	4, 5, 9, 21, 55, 56, 61, 1239
<code>\parskip</code>	780	<code>\section</code>	5, 8, 9, 18–20, 54, 55, 58, 69, 1345
<code>\part</code>	3–5, 9, 17, 19–21, 25, 38, 55, 61, 70, 1251, 1253, 1255, 1308, 1314, 1324	<code>\section*</code>	19
<code>\partmark</code>	3	<code>\sectionmark</code>	13, 66, 1063
<code>\partmark</code>	69, 1038, 1295	<code>\setbox</code>	706, 843, 1615
<code>\partname</code>	1033, 1296	<code>\SetCurrentKnownTemplateKeys</code>	426, 652
<code>\penalty</code>	40, 41, 586, 589	<code>\SetTemplateKeys</code>	734, 826, 886, 1425, 1564, 1660, 1714, 1779
Plugs:			
<code>extend</code>	112	<code>\sfcode</code>	31, 227, 228, 229, 230
<code>page</code>	177	skip commands:	
<code>\prevdepth</code>	41, 42, 567, 569, 571	<code>\skip_add:Nn</code>	574, 952
property commands:			
<code>\property_record:nn</code>	178	<code>\skip_horizontal:N</code>	
<code>\property_ref:nnn</code>	118		711, 753, 757, 761, 765, 862, 902, 906, 910, 914, 1620, 1670, 1675, 1681, 1686, 1726, 1731, 1740, 1745, 1788, 1793, 1799, 1804
		<code>\skip_horizontal:n</code>	
			1692, 1751, 1754, 1813, 1819
		<code>\skip_set:eq:NN</code>	562, 565
		<code>\skip_sub:Nn</code>	953
		<code>\skip_vertical:N</code>	523, 538, 1517
		<code>\skip_vertical:n</code>	1697
		<code>\skip_zero:N</code>	790
		<code>\l_tmpa_skip</code>	562, 563, 565, 579

`\l_tmpb_skip` 41, 565, 574, 575
`\c_zero_skip` 752, 760, 901,
 909, 1668, 1679, 1724, 1738, 1786, 1797
 socket commands:
`\socket_use:n` 95, 768, 842, 917
 Sockets:
`heading/instname` 111
`heading/record` 176
`\spacefactor` 30, 31, 231, 234
`\specialsection` 58
`\start-code` 13
 str commands:
`\c_hash_str` 1210, 1219, 1226
`\str_case:nn` 440, 659, 1567
`\str_case:nnTF`
 475, 671, 1241, 1427, 1459
`\str_if_eq:nnTF` 429
`\str_if_eq_p:nn` 1210,
 1211, 1212, 1213, 1214, 1217, 1224
`\string` 1144
`\subparagraph` 19, 20, 69, 1353
`\subparagraphmark` 1110
`\subsection` 18–20, 69, 1347
`\subsectionmark` 1074
`\subsubsection` 18, 20, 69, 1349

T

tag commands:
`\tag_if_active:TF` 1276
`\tag_mc_end_push:` 798, 803
`\tag_struct_begin:n` 800, 805
`\tag_struct_end:` 799, 804
 tag internal commands:
`\l_tag_para_bool` 799, 804
`\l_tag_para_flattened_bool` 809
`\tagpdfpara0ff` 739, 832, 892
`\tagpdfpara0n` 810
 template commands:
`\template_process_order_clist:nnn`
 770, 850, 864, 919
 template types:
`headformat` 243
`heading` 242
 templates:
`headformat display` 334, 720
`headformat display-v1` 1629, 1650
`headformat hang` 347, 812
`headformat hang-v1` 1636, 1705
`headformat runin` 360, 874
`headformat runin-v1` 1643, 1770
`heading display` 244, 371
`heading display-v1` 1358
`heading runin` 289, 597
`heading runin-v1` 1521

TeX and L^AT_EX 2_ε commands:

`\@` 31
`\@LRmoderr` 558
`\@M` 707,
 749, 839, 898, 1616, 1664, 1718, 1782
`\@addpunct` 31
`\@afterheading` 39, 493, 1478
`\@afterindentfalse` .. 393, 1152, 1401
`\@afterindenttrue` ... 392, 1149, 1400
`\@boxtempa` 49
`\@chapapp` 56, 1048, 1057
`\@chappap` 33
`\@chapter` 4
`\@clubpenalty` 713, 1622
`\@currentlabelname` 997
`\@firstoftwo` 10
`\@hangfrom` 1760
`\@kernel@refstepcounter`
 946, 1492, 1584
`\@latex@error` 1276
`\@latex@warning` 1251, 1266, 1276
`\@m` 234
`\@minus` 1095, 1107
`\@nobreakfalse` 701, 1610
`\@noskipsecfalse` 705, 1614
`\@noskipsectrue` 702, 1611
`\@part` 4
`\@plus` 1095, 1107
`\@secondoftwo` 10
`\@secpenalty` ... 13, 15, 51, 66, 67, 929
`\@startsection` . 4, 5, 9, 11, 21, 54,
 55, 58, 59, 61, 64, 66, 1141, 1236, 1238
`\@svsechd` 686, 708, 1595, 1617
`\@tempboxa` 843, 856, 857, 863
`\@tempskipa` 1148, 1150, 1151, 1161, 1177
`\@tempskipb` 1154, 1155, 1162, 1171, 1178
`\@tempswa` 38
`\@tempswafalse` 454, 1442
`\@tempswatrue` 452, 1440
`\@topnewpage` 37, 39, 40, 53, 509
`\@topnum` 463, 1452
`\c@secnumdepth` 51, 52, 423,
 534, 650, 699, 935, 940, 1422, 1483,
 1488, 1515, 1562, 1575, 1580, 1608
`\declare@file@substitution` ... 1357
`\Hy@addcontentsline@addbookmark` 984
`\Hy@bookmarksfalse` 992
`\Hy@bookmarkstrue` 988
`\if@afterindent` 1163
`\if@mainmatter` 1328
`\if@nobreak` 957
`\if@noskipsec` 71, 704, 956, 1613
`\if@openright`
 . 444, 461, 481, 591, 1432, 1450, 1466

<code>\if@tempswa</code>	487, 1472	1231, 1233, 1286, 1392, 1394, 1396,
<code>\if@twocolumn</code>	428, 450, 1438	1422, 1430, 1448, 1462, 1478, 1531,
<code>\if@twoside</code>	480, 1465	1533, 1535, 1562, 1569, 1722, 1736
<code>\ifHy@bookmarks</code>	986	<code>\tl_set_eq:NN</code> 83,
<code>\kernel@startsection</code>	1236, 1238	84, 85, 88, 89, 90, 422, 651, 1421, 1563
<code>\protected@edef</code>	1493, 1585	token commands:
<code>\z@</code>	463, 706, 1150, 1155, 1452, 1615	<code>\token_if_macro:N</code> 1200
tex commands:		<code>\token_to_str:N</code> 1223, 1225
<code>\tex_lastskip:D</code>	562	<code>\topskip</code> 13, 952, 953, 1046
<code>\tex_parshape:D</code>	861	<code>\twocolumn</code> 488, 1473
<code>\texorpdfstring</code>	22, 54, 980, 1020	<code>\typeout</code> 26, 606, 619, 620, 1529, 1539, 1540
<code>\the</code>	222, 223, 224, 1161, 1162, 1177, 1178	
<code>\the<name></code>	13, 66	U
<code>\thechapter</code>	25, 1049	<code>\unskip</code> 709, 1618
<code>\theheading</code>	3	<code>\UnusedTemplateKeys</code> 515, 691, 1507, 1600
<code>\theheading</code>	13, 44, 66–68,	use commands:
68, 422, 651, 947, 1381, 1421, 1563		<code>\use:N</code> 1007, 1018
<code>\thepart</code>	1034	<code>\use:n</code> 60, 97,
<code>\thesection</code>	25, 67, 68	506, 689, 1156, 1172, 1189, 1505, 1598
<code>\thispagestyle</code>		<code>\use_i:nn</code> 519, 695, 1511, 1604
449, 462, 483, 1437, 1451, 1468		<code>\use_ii:nn</code> 520, 696, 1512, 1605
tl commands:		<code>\use_none:n</code> 9, 10
<code>\c_novalue_tl</code>	26, 89, 90	<code>\UseInstance</code> 26,
<code>\c_space_tl</code>	1216, 1218, 1223, 1225	64, 65, 98, 514, 690, 745, 1506, 1599
<code>\tl_clear:N</code>	78, 79, 80, 92, 202,	<code>\UseSocket</code> 11
471, 667, 673, 675, 1456, 1489, 1581		<code>\UseStructureName</code> 276,
<code>\tl_if_blank:nTF</code>	955	321, 501, 683, 800, 805, 1502, 1593
<code>\tl_if_empty:nTF</code>		<code>\UseTaggingSocket</code>
439, 474, 658, 670, 1425, 1426,		498, 499, 505, 527, 680, 681, 685,
1458, 1564, 1565, 1660, 1714, 1779		710, 1499, 1500, 1591, 1592, 1594,
<code>\tl_if_head_is_N_type:nTF</code>	1195	1619, 1662, 1702, 1716, 1757, 1807
<code>\tl_new:N</code>	52, 53, 54, 55, 56,	V
57, 58, 60, 61, 62, 63, 64, 65, 66, 67, 68		vbox commands:
<code>\tl_put_right:Nn</code>	197, 213, 216	<code>\vbox:n</code> 793
<code>\tl_reverse:n</code>	1192, 1230	<code>\vfil</code> 38, 40, 42, 456, 479, 589, 1444, 1464
<code>\tl_set:Nn</code>	94, 124,	<code>\vfilneg</code> 38, 40, 42, 589
132, 140, 149, 161, 203, 206, 207,		<code>\vskip</code> 575, 578, 579
381, 383, 385, 387, 423, 433, 443,		<code>\vspace</code> 17, 343, 511, 965, 973, 1118, 1306
460, 478, 493, 608, 610, 612, 614,		<code>\vspace*</code> 13, 52
650, 661, 741, 742, 743, 834, 835,		W
836, 894, 895, 896, 941, 942, 943,		<code>\WordSpaceAmount</code> 220, 1633, 1640, 1647
947, 997, 1189, 1197, 1205, 1230,		