

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

$\text{\LaTeX}$ Project*

v0.9q 2026-09-22

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	8
4	Open points	9
5	Template types and templates for headings	9
5.1	Template types	9
5.1.1	The template type ‘heading’	9
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’	14
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’	16
5.2.8	The <code>headformat</code> template ‘runin’	17
5.3	Default instances	18

*Initial implementation by Frank Mittelbach.

5.3.1	Instances of <code>heading</code> templates	18
5.3.2	Instances of <code>headformat</code> templates	18
5.4	Use of templates from the 2026-06-01 release of <code>L^AT_EX</code>	18
6	Support for legacy classes and packages	19
6.1	Support classes based on legacy <code>L^AT_EX 2_ε</code> interfaces	19
6.2	Support for the <code>titlesec</code> package interfaces	20
7	Support for links	20
8	Bookmarks	20
9	Tagging support	20
10	Debugging support	21
11	The Implementation	21
11.1	Debugging	21
11.2	Temp variable(s)	22
11.3	variable(s)	23
11.4	New user commands	23
11.5	The <code>L^AT_EX 2_ε</code> parsing interface	23
11.6	General helper commands	27
11.7	Templates	28
11.7.1	Template types	28
11.7.2	<code>heading</code> templates interfaces	28
11.7.3	<code>headformat</code> templates interfaces	31
11.7.4	<code>heading</code> templates code	32
11.7.5	<code>headformat</code> templates code	42
11.7.6	Internal commands used by the template code	47
11.8	Support for legacy classes and packages	50
11.8.1	Instances (sample/default definitions)	51
11.8.2	New <code>\@startsection</code>	53
11.8.3	New <code>\secdef</code>	57
11.8.4	Core <code>L^AT_EX</code>	58
12	Issues and problems noticed along the way	60
12.1	Local <code>\DeclareInstance</code>	60
12.2	<code>\SetCurrentTemplateKeys</code>	60
12.3	O-expansion of <code>\UseInstance</code> arguments	60
12.4	Switch <code>\openright</code> is not defined by core	60
12.5	Indexheading at least for <code>l3doc</code> is wrong now	60

A	Templates from the 2026-06-01 release of L^AT_EX	60
A.1	Template documentation for -v1 templates	61
A.1.1	Templates of type <code>heading</code>	61
A.1.2	The <code>heading</code> template ‘ <code><all></code> ’	61
A.1.3	The <code>heading</code> template ‘ <code>display-v1</code> ’	62
A.1.4	The <code>heading</code> template ‘ <code>runin-v1</code> ’	63
A.1.5	Templates of type <code>headformat</code>	64
A.1.6	The <code>headformat</code> template ‘ <code>hang-v1</code> ’	64
A.1.7	The <code>headformat</code> template ‘ <code>runin-v1</code> ’	65
A.1.8	The <code>headformat</code> template ‘ <code>display-v1</code> ’	65
A.2	Implementation of -v1 templates	66
	Index	75

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

2.5 Converting heading commands defined with `\@startsection`

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

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

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

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

\begin{document}

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

\end{document}
```

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

The instance 'section-@startsection' of type 'heading' has values:

```
> level => 1
> force-unnumbered => false
> placement => normal
> column-spanning => false
> mark-cmd => \sectionmark {##1}
> para-indent => false
> before-vspace => 8.0pt plus 2.0pt minus 2.0pt
> penalty => \c_max_int
> after-penalty-vspace => 0pt
> after-vspace => 4.0pt
> placement-start-code =>
> placement-end-code =>
> prefix => \NoValue
> punct =>
> heading-decls => \tubsecfmt
> prefix-decls =>
> number-decls =>
> title-decls =>
> subtitle-decls =>
> quote-decls =>
> heading-format => ##1
> prefix-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 => Opt
> order-hang => prefix,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 => Opt
> penalty => \c_max_int
> after-penalty-vspace => Opt
> after-vspace => Opt
> placement-start-code =>
> placement-end-code =>
> prefix => \NoValue
> punct =>
> heading-decls => \normalfont
> prefix-decls =>
> number-decls =>
> title-decls =>
> subtitle-decls =>
> quote-decls =>
> heading-format => ##1
> prefix-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 => .

```

The template 'hang' of type 'headformat' has default values:

```

> heading-indent => 0pt
> order-hang => prefix,separator-a,?,number,punct,separator-b,?
> order => title,separator-c,subtitle
> separator-a => \nobreakspace
> separator-b => \hspace {1em}
> separator-c => \
> separator-d => .

```

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 6.2.

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 prefix 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 (prefix 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 prefix, 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 prefix 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.

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

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**

¹Maybe questionable

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**

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`

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

All templates should set up `\theheading` to correspond to `\the<name>`.

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

5.2.3 The heading template ‘display’

Attributes:

para-indent (*boolean*) Should the paragraph after the heading be indented?

Default: `false`

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

penalty (*integer*) Penalty to break before the heading. The default (T_EX's largest integer) indicates that no penalty was set in which case `\@secpenalty` is used (to support the L^AT_EX 2_ε interface). 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>`

heading-format (*function(1)*) Code that receives the whole heading as an argument.
Default: #1

prefix-format (*function(1)*) Code that receives the prefix 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*) 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-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 `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`

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

prefix-format (*function(1)*) Code that receives the prefix 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 runin 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 prefix, number, punct, title, subtitle and quote. If the heading is unnumbered then prefix, 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 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: `prefix, separator-a, ?, number, punct, separator-b, ?, title, separator-c, subtitle`

separator-a (*tokenlist*) By default the separation between `prefix` 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 heading (shifting it to the right in a LR typesetting context) 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: `prefix, 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 `prefix` 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 `runin` template.

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: `prefix, separator-a, ?, number, punct, separator-b, ?, title, separator-c, subtitle`

separator-a (*tokenlist*) By default the separation between `prefix` 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 L^AT_EX 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.

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 $\text{\LaTeX}$

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

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

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

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

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

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

to

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

\DeclareInstance{heading}    {section}{hang-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:

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

with a special logic that the sign of `<beforeskip>` determines display or runin 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.

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.

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.

6.2 Support for the `titlesec` package interfaces

TODO

`\titleformat` The `\titleformat` declaration has the following syntax:

```
\titleformat{cmd}[shape]{format}{label}{sep}{before-code}[after-code]
```

`\titlespacing` The `\titlespacing` declaration has the following syntax:

```
\titlespacing*{cmd}{left-sep}{before-vspace}{after-vspace}[right-sep]
```

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`.

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`.

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

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 21.)

`\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 21.)

`\__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:~ prefix~ =~ \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

```

```

\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

```

But these token lists needs a declaration:

```

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

```

(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.

```

66 \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.

```

67 \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).

```
68 \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 runin heading still waiting to be typeset. Otherwise, the new heading might overwrite some variables that are still needed to typeset the dangling heading.

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

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

```
74 \__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).

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

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

```
86 \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`

```
87 \tl_set_eq:NN \l__head_subtitle_tl \c_novalue_tl
88 \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?

```

89 \IfNoValueTF { #3 }
90 { \tl_clear:N \l__head_instance_keys_tl }
91 { \keys_set_known:nnN {heading} { #3 } \l__head_instance_keys_tl }

```

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

```

92 \__head_debug_typeout:n{use~ 'heading'~ instance:~ #1 }
93 \use:e {
94   \exp_not:N \UseInstance{heading}{#1}
95   { \exp_not:o \l__head_instance_keys_tl }
96   { \bool_if:NTF \l__head_unnumbered_bool \BooleanTrue \BooleanFalse }
97   { \exp_not:o \l__head_title_tl }
98   { \exp_not:o \l__head_toc_tl }
99   { \exp_not:o \l__head_running_tl }
100  { \exp_not:o \l__head_bookmark_tl }
101  { \exp_not:o \l__head_nameref_tl }
102  { \exp_not:o \l__head_label_tl }
103  { { \exp_not:o \l__head_subtitle_tl }
104    { \exp_not:o \l__head_quote_tl } }
105 }
106 }

```

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.

```

107 \keys_define:nn {heading} {
108   , shorttitle .meta:n =
109     {toc = {#1} , running = {#1} , bookmark = {#1} , nameref= {#1} }
110   , bookmark .tl_set:N = \l__head_bookmark_tl
111   , running .tl_set:N = \l__head_running_tl
112   , toc .tl_set:N = \l__head_toc_tl
113   , nameref .tl_set:N = \l__head_nameref_tl
114   %
115   , numbered .bool_set_inverse:N = \l__head_unnumbered_bool
116   , numbered .default:n = true
117   , unnumbered .bool_set:N = \l__head_unnumbered_bool
118   , unnumbered .default:n = true
119   %
120   , subtitle .tl_set:N = \l__head_subtitle_tl
121   , 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.

```

122   , label .code:n = \tl_put_right:Nn \l__head_label_tl { \label{#1} }
123 }

```

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

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

`\__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.

```
124 \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.

```
125   \quark_if_no_value:nTF {#2}
126   { \__head_debug_typeout:n{---no-label }
127     \tl_clear:N \l__head_label_tl
128     \tl_set:Nn\l__head_title_tl {#1#3}
129   }
```

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

```
130   { \__head_debug_typeout:n{---label~found~#2}
131     \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`.

```
132     \tl_set:Nn\l__head_title_tl {#1}
133     \__head_find_label_aux:w #3\__head_find_label_aux:w
134   }
135   \__head_debug_typeout:n{---return:~ '\exp_not:o \l__head_title_tl' }
136 }
```

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.

```
137 \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.

```
138   \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.

```
139   \quark_if_no_value:nF {#2}
140   { \__head_debug_typeout:n{--- extra~ label~ '#2'~ found }
141     \tl_put_right:Nn \l__head_label_tl { \label{#2} }
142     \__head_find_label_aux:w #3\__head_find_label_aux:w
143   }
144 }
```

(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.

```

145 \newcommand\WordSpaceAmount[1]{
146   \glueexpr
147     #1\fontdimen2\the\font
148   plus #1\fontdimen3\the\font
149   minus #1\fontdimen4\the\font
150   \relax
151 }
```

(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.

```

152 \def\frenchspacing{\sfcode`.1006\sfcode`?\?1005\sfcode`\!1004%
153   \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.

```

154 \def\nonfrenchspacing{\sfcode`\.3002\sfcode`?\?3001\sfcode`\!3000%
155   \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.

```

156 \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!

```

157 \cs_new_protected:Npn \AddPunct #1 {
158   \relax\ifhmode
159     \ifnum\spacefactor>\@m
160       \__head_debug_typeout:n {--->~ punctuation~ '#1'~ not~ added}
161     \else
162       \__head_debug_typeout:n {--->~ punctuation~ '#1'~ added}
163     #1
164   \fi
165 \fi
166 }
```

(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 }

```

167 \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: prefix string
- 3: formatted heading number
- 4: title
- 5: subtitle
- 6: quote

```

168 \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.

```

169 \DeclareTemplateInterface{heading}{display}{9}
170 {
171   , name                : tokenlist
172   , parent-name         : tokenlist
173   , reset-counter       : tokenlist
174   , level                : integer = 0

```

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

```

175   , force-unnumbered    : boolean = false

```

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

```

176   , placement           : choice {page , top , normal, ragged } = normal
177   , column-spanning     : boolean = false
178   , mark-cmd            : function(1) =
179   , 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.

```

180   , before-vspace       : skip = 0pt
181   , penalty              : integer = \c_max_int
182   , after-penalty-vspace : skip = 0pt
183   , after-vspace        : skip = 0pt

```

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

```

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

```

Old names for the above to be dropped soon:

```

186   , start-code          : tokenlist =      % no default values!
187   , final-code          : tokenlist =      % no default values!

```

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

```

188   , prefix              : tokenlist = \NoValue

```

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

```

189   , punct               : tokenlist =
190   , heading-decls       : tokenlist = \normalfont
191   , prefix-decls        : tokenlist =
192   , number-decls        : tokenlist =
193   , title-decls         : tokenlist =
194   , subtitle-decls      : tokenlist =
195   , quote-decls         : tokenlist =

```

```

196 , heading-format      : function{1} = #1
197 , prefix-format       : function{1} = #1
198 , number-format       : function{1} = #1
199 , punct-format        : function{1} = #1
200 , title-format        : function{1} = #1 \par
201 , subtitle-format     : function{1} = #1
202 , 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}`.

```

203 , number-tag          : tokenlist = heading-number % \UseStructureName{sec/???}

204 , headformat-instance : tokenlist = std
205 , contents-extra      : tokenlist =
206 }

```

`heading runin (templ.)` This template is similar to the `display` template but implements a `runin` 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`.

```

207 \DeclareTemplateInterface{heading}{runin}{9}
208 {
209 , name           : tokenlist
210 , parent-name    : tokenlist
211 , reset-counter  : tokenlist
212 , level          : integer = 0
213 , force-unnumbered : boolean = false
214 , placement      : choice {page , top , normal, ragged } = normal
215 , column-spanning : boolean = false
216 , mark-cmd       : function(1) =
217 , para-indent    : choice { true , false } = false
218 , before-vspace  : skip = 0pt
219 , penalty        : integer = \c_max_int
220 , after-penalty-vspace : skip = 0pt
221 , after-hspace   : skip = 0pt
222 , placement-start-code : tokenlist = % no default values!
223 , placement-end-code  : tokenlist = % no default values!

```

Old names for the above to be dropped soon:

```

224 , start-code      : tokenlist = % no default values!
225 , final-code      : tokenlist = % no default values!

226 , prefix          : tokenlist = \NoValue
227 , punct           : tokenlist =
228 , heading-decls    : tokenlist = \normalfont
229 , prefix-decls     : tokenlist =
230 , number-decls     : tokenlist =
231 , title-decls      : tokenlist =
232 , subtitle-decls   : tokenlist =
233 , quote-decls      : tokenlist =
234 , heading-format   : function{1} = #1

```

```

235 , prefix-format      : function{1} = #1
236 , number-format      : function{1} = #1
237 , punct-format       : function{1} = #1

```

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

```

238 , title-format        : function{1} = #1
239 , subtitle-format     : function{1} = #1
240 , quote-format        : function{1} = #1
241 , number-tag          : tokenlist = heading-number % \UseStructureName{sec/???}
242 , headformat-instance : tokenlist = std
243 , contents-extra      : tokenlist =
244 }

```

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.

```

245 \DeclareTemplateInterface{headformat}{display}{6}
246 {
247   , heading-indent    : length = 0pt
248   %
249   , order              : commalist = {prefix,separator-a,?,number,punct,
250                                     separator-b,?,title,
251                                     separator-c, subtitle}
252   , separator-a        : tokenlist = \nobreakspace
253   , separator-b        : tokenlist = \par \nobreak \vspace{20pt}
254   , separator-c        : tokenlist = \
255   , separator-d        : tokenlist =
256 }

```

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.

```

257 \DeclareTemplateInterface{headformat}{hang}{6}
258 {
259   , heading-indent    : length = 0pt
260   , order-hang        : commalist = {prefix,separator-a,?,number,punct,
261                                     separator-b,?}
262   , order              : commalist = {title, separator-c, subtitle}
263   , separator-a        : tokenlist = \nobreakspace
264   , separator-b        : tokenlist = \hspace{1em}
265   , separator-c        : tokenlist = \
266   , separator-d        : tokenlist =
267 }

```

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.

```

268 \DeclareTemplateInterface{headformat}{runin}{6}
269 {
270   , heading-indent    : length = 0pt

```

```

271 , order          : commalist = {prefix,separator-a,?,number,punct,
272                               separator-b,?,title,
273                               separator-c,subtitle}
274 , separator-a     : tokenlist = \nobreakspace
275 , separator-b     : tokenlist = \hspace{1em}
276 , separator-c     : tokenlist = \
277 , separator-d     : tokenlist =
278 }

```

11.7.4 heading templates code

heading display (*templ.*)

```

279 \DeclareTemplateCode{heading}{display}{9}
280 {
281   , name          = \l__head_name_tl
282   , level         = \l__head_level_int
283   % next two not yet used
284   , parent-name   = \l__head_pname_tl
285   , reset-counter = \l__head_reset_cnt_tl
286
287   , 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.

```

287   , placement     = {
288     ,page         = \__head_debug_typeout:n{ A~ page~ heading }
289                   \tl_set:Nn \l__head_placement_tl { page }
290     ,top          = \__head_debug_typeout:n{ A~ top~ heading }
291                   \tl_set:Nn \l__head_placement_tl { top }
292     ,normal       = \__head_debug_typeout:n{ A~ normal~ heading }
293                   \tl_set:Nn \l__head_placement_tl { normal }
294     ,ragged       = \__head_debug_typeout:n{ A~ normal~ heading~ (ragged~ style) }
295                   \tl_set:Nn \l__head_placement_tl { ragged }
296   }
297
298   , column-spanning = \l__head_column_spanning_bool
299
300   , mark-cmd      = \__head_mark_cmd:n

```

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

```

299   , para-indent   = {
300     ,true         = \@afterindenttrue
301     ,false        = \@afterindentfalse
302   }
303   , before-vspace = \l__head_before_skip
304   , penalty       = \l__head_penalty_int
305   , after-penalty-vspace = \l__head_after_penalty_skip
306   , after-vspace  = \l__head_after_skip
307   , placement-start-code = \l__head_placement_start_code_tl
308   , placement-end-code   = \l__head_placement_end_code_tl

```

Old names for the above, to be dropped

```

309 , start-code   = \l__head_deprecated_start_code_tl
310 , final-code   = \l__head_deprecated_end_code_tl

311 , prefix       = \l__head_prefix_tl
312 , punct        = \l__head_punct_tl

313 , headformat-instance = \l__head_headformat_instance_tl

314 , heading-decls = \l__head_heading_decls_tl
315 , prefix-decls  = \l__head_prefix_decls_tl
316 , number-decls  = \l__head_number_decls_tl
317 , title-decls   = \l__head_title_decls_tl
318 , subtitle-decls = \l__head_subtitle_decls_tl
319 , quote-decls   = \l__head_quote_decls_tl

320 , heading-format = \__head_heading_format:n
321 , prefix-format  = \__head_prefix_format:n
322 , number-format  = \__head_number_format:n
323 , punct-format   = \__head_punct_format:n
324 , title-format   = \__head_title_format:n
325 , subtitle-format = \__head_subtitle_format:n
326 , quote-format   = \__head_quote_format:n
327 %
328 , number-tag      = \l__head_number_tag_tl

329 , contents-extra = \l__head_contents_extra_tl
330 }

331 {
332   \tl_set_eq:Nc \theheading { the \l__head_name_tl }

```

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

```

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

334   \__head_show_arguments:nnnnnnnnn
335     {#1}{#2}{#3}{#4}{#5}{#6}{#7}{#8}{#9}

```

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

```

336   \SetKnownTemplateKeys{heading}{display}{#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.

```

337   \bool_if:NT \l__head_column_spanning_bool
338     { \if@twocolumn
339       \str_if_eq:VnF \l__head_placement_tl {top}
340       {
341         \__head_debug_typeout:n {column-spanning~ requires~
342           placement=top.~ Adjusted!}
343         \tl_set:Nn \l__head_placement_tl {top}
344       }
345     }
346     \bool_set_false:N \l__head_column_spanning_bool
347   \fi
348 }

```

It would be nice if there is a variant of `\SetTemplateKey` in which the template type and name are implicit, e.g., `\SetCurrentTemplateKeys` because this is usually what I need.

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.

Support the old key names (will vanish the moment we have support for this in `ltemplate`).

```

349 \tl_if_empty:oF \l__head_deprecated_start_code_tl
350 {
351     \tl_set_eq:NN \l__head_placement_start_code_tl
352         \l__head_deprecated_start_code_tl
353 }
354 \tl_if_empty:oF \l__head_deprecated_end_code_tl
355 {
356     \tl_set_eq:NN \l__head_placement_end_code_tl
357         \l__head_deprecated_end_code_tl
358 }
359 \tl_if_empty:oT \l__head_placement_start_code_tl
360 { \str_case:Vn \l__head_placement_tl
361   {
362     { page }
363     { \tl_set:Nn \l__head_placement_start_code_tl

```

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

Straight from the placement definition of `\part`.

```

364     { \if@openright
365       \cleardoublepage
366     \else
367       \clearpage
368     \fi
369     \thispagestyle{plain}%
370     \if@twocolumn
371       \onecolumn
372       \@tempwattrue
373     \else
374       \@tempwafalse
375     \fi
376     \null\vfil
377   }
378 }
379 { top }
380 { \tl_set:Nn \l__head_placement_start_code_tl
381   { \if@openright\cleardoublepage\else\clearpage\fi
382     \thispagestyle{plain}%
383     \global\@topnum\z@
384   }
385 }

```

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

```

386 { ragged }
387 { \cs_set_eq:NN \sec_add_penalty_with_possible_fil:n

```

```

388             \sec_add_penalty_with_fil:n
389         }
390     }

```

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.

```

391 %      { \tl_clear:N \l__head_placement_start_code_tl }
392 }
393 %
394 \tl_if_empty:oT \l__head_placement_end_code_tl
395 { \str_case:VnF \l__head_placement_tl
396   {
397     { page }
398     { \tl_set:Nn \l__head_placement_end_code_tl
399       { \vfil\newpage
400         \if@twoside
401           \if@openright
402             \null
403             \thispagestyle{empty}%
404             \newpage
405             \fi
406             \fi
407             \if@tempswa
408               \twocolumn
409             \fi
410           }
411         }
412       }

```

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

```

413     { \tl_set:Nn \l__head_placement_end_code_tl { \@afterheading } }
414 }

```

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

```

415 \__head_determine_penalty:
416 \__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.

```

417 \__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.

```

418 \UseTaggingSocket{sec/end}{\int_use:N\l__head_level_int}
419 \UseTaggingSocket{sec/begin}
420   {\int_use:N\l__head_level_int}
421   {tag=\UseStructureName{sec/\int_use:N\l__head_level_int}}
422 }

```

Up to this point everything is identical for both display and runin 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

```

423 \__head_debug_typeout:n{use~ 'headformat'~instance:~

```

```

424         \l__head_headformat_instance_tl }
425 \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.

```

426 \use:e {
427   \bool_if:NT \l__head_column_spanning_bool
428   { \exp_not:n {
429     \@topnewpage [
430       \dim_compare:nNnF{\l__head_after_penalty_skip}={Opt}
431       { \vspace*      \l__head_after_penalty_skip }
432     }
433   }
434   \UseInstance{headformat} { \l__head_headformat_instance_tl }
435     { \exp_not:o \UnusedTemplateKeys }
436     { \exp_not:o { \l__head_prefix_tl } }
437     { \exp_not:o { \l__head_number_tl } }
438     { \exp_not:n { #3 } }
439     { \exp_not:o { \use_i:nn #9 } }
440     { \exp_not:o { \use_ii:nn #9 } }
441
442   \bool_if:NT \l__head_column_spanning_bool
443   {
444     \skip_vertical:N \l__head_after_skip

```

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

```

444   ]
445   }
446 }
447 \UseTaggingSocket{sec/title/end}
448 %
449 % --- handle marks, toc-entry, bookmark, nameref, and label
450 %
451 % \__head_handle_marks_etc:nnnnn {#4}{#5}{#6}{#7}{#8}
452 %
453 % --- post-heading handling (vertical)

```

Restore secnumdepth

```

454 \int_gset:Nn\c@secnumdepth{\l__head_saved_secnumdepth_tl}
455 \par

```

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

```

456 \bool_if:NF \l__head_column_spanning_bool
457 { \nobreak
458   \skip_vertical:N \l__head_after_skip
459 }
460 % --- prepare next paragraph (defaults to \cs{@afterheading}):
461 %
462 \l__head_placement_end_code_tl

```

```

463 %
464 \ignorespaces
465 }

```

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

```

466 \cs_new_protected:Npn \sec_add_penalty_with_fil:n #1 {
467   \__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`.

```

468   \addpenaltywithfil { #1 }

```

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

```

469   \cs_set_eq:NN
470     \sec_add_penalty_with_possible_fil:n
471     \addpenalty
472 }

```

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.

```

473 \cs_set_eq:NN
474   \sec_add_penalty_with_possible_fil:n
475   \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.

```

476 \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).

```

477   \mode_if_horizontal:T
478     { \mode_if_inner:TF { \@LRmoderr }{ \par } }
479   \legacy_if:nF { @minipage }
480     { \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.

```

481     { \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.

```

482         \skip_set_eq:NN \l_tmpa_skip \tex_lastskip:D
483         \dim_compare:nNnTF \l_tmpa_skip = \c_zero_dim
484         { #1 }

```

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

```

485         { \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.

```

486         \dim_set:Nn \l_tmpa_dim
487         { \dim_compare:nNnTF \prevdepth > \maxdepth
488           \maxdepth
489           { \dim_compare:nNnTF \prevdepth = { -1000pt }
490             \c_zero_dim
491             \prevdepth
492           }
493         }
494         \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.

```

495         \vskip -\l_tmpb_skip

```

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

```

496         #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.

```

497         \dim_compare:nNnF \l_tmpa_dim = \c_zero_dim
498         { \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.

```

499         \vskip \l_tmpa_skip
500     }
501     \group_end:
502 }
503 }
504 }

```

Reimplement `\addpenalty` using the above macro:

```

505 \cs_set_protected:Npn \addpenalty #1 {
506   \__kernel_add_penalty:n { \penalty #1 \scan_stop: }
507 }

```

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

```
508 \cs_new_protected:Npn \addpenaltywithfil #1 {
509   \__kernel_add_penalty:n { \vfil \penalty #1 \vfilneg }
510 }
```

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

`\if@openright`

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

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

```
514   \expandafter\newif\csname if@openright\endcsname
515   \fi
516 }
```

(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.

```
517 \DeclareTemplateCode{heading}{runin}{9}
518 {
519   , name           = \l__head_name_tl
520   , level          = \l__head_level_int
521   % next two not yet used
522   , parent-name    = \l__head_pname_tl
523   , reset-counter  = \l__head_reset_cnt_tl
524   , force-unnumbered = \l__head_unnumbered_bool
```

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

```
525   , placement      = {
526     page           = \typeout{ ^^JA~ runin~ page~ placement~ heading~makes-no~sense
527                       (top~used)}
528     \tl_set:Nn \l__head_placement_tl { top }
529     ,top           = \__head_debug_typeout:n{ A~ top~ heading }
530     \tl_set:Nn \l__head_placement_tl { top }
531     ,normal        = \__head_debug_typeout:n{ A~ normal~ heading }
532     \tl_set:Nn \l__head_placement_tl { normal }
533     ,ragged         = \__head_debug_typeout:n{ A~ normal~ heading~ (ragged~ style) }
534     \tl_set:Nn \l__head_placement_tl { ragged }
535   }
536   , column-spanning = \l__head_column_spanning_bool
537   , mark-cmd        = \__head_mark_cmd:n
```

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

In a runin 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.

```

538 , para-indent      = {
539                     ,true  = %\typeout{para-indent~ setting~ ignored}
540                     ,false = %\typeout{para-indent~ setting~ ignored}
541                     }
542 , before-vspace     = \l__head_before_skip
543 , penalty           = \l__head_penalty_int
544 , after-penalty-vspace = \l__head_after_penalty_skip
545 , after-hspace      = \l__head_after_skip
546 , placement-start-code = \l__head_placement_start_code_tl
547 , placement-end-code  = \l__head_placement_end_code_tl

```

Old names for the above, to be dropped

```

548 , start-code      = \l__head_deprecated_start_code_tl
549 , final-code      = \l__head_deprecated_end_code_tl

550 , prefix          = \l__head_prefix_tl
551 , punct           = \l__head_punct_tl

552 , headformat-instance = \l__head_headformat_instance_tl

553 , heading-decls   = \l__head_heading_decls_tl

554 , prefix-decls    = \l__head_prefix_decls_tl
555 , number-decls    = \l__head_number_decls_tl
556 , title-decls     = \l__head_title_decls_tl
557 , subtitle-decls  = \l__head_subtitle_decls_tl
558 , quote-decls     = \l__head_quote_decls_tl

559 , heading-format   = \__head_heading_format:n
560 , prefix-format    = \__head_prefix_format:n
561 , number-format    = \__head_number_format:n
562 , punct-format     = \__head_punct_format:n
563 , title-format     = \__head_title_format:n
564 , subtitle-format  = \__head_subtitle_format:n
565 , quote-format     = \__head_quote_format:n
566 %
567 , number-tag       = \l__head_number_tag_tl

568 , contents-extra   = \l__head_contents_extra_tl
569 }
570 { \__head_show_arguments:nnnnnnnnn
571     {#1}{#2}{#3}{#4}{#5}{#6}{#7}{#8}{#9}

```

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

```

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

```

define \theheading

```

573 \tl_set_eq:Nc \theheading { the \l__head_name_tl }
574 \SetKnownTemplateKeys{heading}{runin}{#1}

```

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

```

575 \bool_if:NT \l__head_column_spanning_bool
576 { \__head_debug_typeout:n {column-spanning~ is~

```

```

577         not~ (yet)~ supported~ for~ runin~ headings!}
578     \bool_set_false:N \l__head_column_spanning_bool
579 }

```

Support the old key names (will vanish the moment we have support for this in litemplate).

```

580 \tl_if_empty:oF \l__head_deprecated_start_code_tl
581 {
582     \tl_set_eq:NN \l__head_placement_start_code_tl
583         \l__head_deprecated_start_code_tl
584 }
585 \tl_if_empty:oF \l__head_deprecated_end_code_tl
586 {
587     \tl_set_eq:NN \l__head_placement_end_code_tl
588         \l__head_deprecated_end_code_tl
589 }
590 \tl_if_empty:oT \l__head_placement_start_code_tl
591 { \str_case:Vn \l__head_placement_tl
592 {
593     { top }    { \tl_set:Nn \l__head_placement_start_code_tl {\clearpage } }
594
595     { ragged }
596     { \cs_set_eq:NN \sec_add_penalty_with_possible_fil:n
597         \sec_add_penalty_with_fil:n
598     }
599 }

```

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

```

599 %     { \tl_clear:N \l__head_placement_start_code_tl }
600 }
601 %

```

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 ...

```

602 % \tl_if_empty:oT \l__head_placement_end_code_tl
603 % { \str_case:VnF \l__head_placement_tl
604 % {
605 %     { top } { \tl_clear:N \l__head_placement_end_code_tl }
606 % }
607 % { \tl_clear:N \l__head_placement_end_code_tl }
608 % }
609 \__head_determine_penalty:
610 \__head_determine_number_typesetting:N #2
611 \__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`.

```

612 \UseTaggingSocket{sec/end}{\int_use:N\l__head_level_int}
613 \UseTaggingSocket{sec/begin}
614     {\int_use:N\l__head_level_int}

```

```

615             {tag=\UseStructureName{sec/\int_use:N\l__head_level_int}}
616         }
617 \UseTaggingSocket{sec/title/init}{\int_use:N\l__head_level_int}
618 \def \@svsechd {
619     \__head_debug_typeout:n{use~ 'headformat'~instance:~
620         \l__head_headformat_instance_tl }
621     \use:e {
622         \UseInstance{headformat} { \l__head_headformat_instance_tl }
623         { \exp_not:o \UnusedTemplateKeys }
624         { \exp_not:o { \l__head_prefix_tl } }
625         { \exp_not:o { \l__head_number_tl } }
626         { \exp_not:n { #3 } }
627         { \exp_not:o { \use_i:nn #9 } }
628         { \exp_not:o { \use_ii:nn #9 } }
629     }
630     \__head_handle_marks_etc:nnnnn {#4}{#5}{#6}{#7}{#8}

```

Restore secnumdepth

```

631     \int_gset:Nn\c@secnumdepth{\l__head_saved_secnumdepth_tl}
632 }
633 \@nobeckfalse
634 \global\@noskipsectrue
635 \everypar{%
636     \if@noskipsec
637         \global\@noskipsecfalse
638         {\setbox\z@\lastbox}
639         \clubpenalty\@M
640         \@svsechd
641         \unskip

```

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

```

642     \UseTaggingSocket{sec/title/split}
643     \skip_horizontal:N \l__head_after_skip
644 \else
645     \clubpenalty \@clubpenalty
646     \everypar{}%
647 \fi
648 }

```

— prepare next paragraph (does nothing by default)

```

649 \l__head_placement_end_code_tl
650 \ignorespaces
651 }

```

11.7.5 headformat templates code

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

```

652 \DeclareTemplateCode{headformat}{display}{6}
653 % args: keys,prefix,number,title,subtitle,quotation
654 {
655     , heading-indent      = \l__head_heading_indent_dim
656     , order               = \l__head_order_clist
657     , separator-a         = name {l__head_separator-a_tl}
658     , separator-b         = name {l__head_separator-b_tl}

```

```

659 , separator-c      = name {l__head_separator-c_tl}
660 , separator-d      = name {l__head_separator-d_tl}
661 }
662 {
663   \__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).

```

664   \SetTemplateKeys{headformat}{display}{#1}

665   \group_begin:

```

\parboxes can be used within the title of a heading (or withing it 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.

```

666   \AssignTaggingSocketPlug{parbox/before}{heading}
667   \AssignTaggingSocketPlug{minipage/before}{heading}
668   \AssignTaggingSocketPlug{para/restore}{heading}
669   \tagpdfparaOff
670 %
671   \tl_set:Nn \l__head_prefix_tl {#2}
672   \tl_set:Nn \l__head_number_tl {#3}
673   \tl_set:Nn \l__head_title_tl {#4}
674 %

```

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

```

675   \normalfont
676   \interlinepenalty \@M
677   \l__head_heading_decls_tl

```

If there is a number and so a prefix, 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!

```

678   \bool_if:NTF \l__head_unnumbered_bool
679   { \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
680     { \skip_horizontal:N \l__head_heading_indent_dim
681       \MakeLinkTarget[\l__head_name_tl]{}
682     }
683     { \MakeLinkTarget[\l__head_name_tl]{}
684       \skip_horizontal:N \l__head_heading_indent_dim
685     }
686   }
687   { \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
688     { \skip_horizontal:N \l__head_heading_indent_dim
689       \MakeLinkTarget{\l__head_name_tl}
690     }
691     { \MakeLinkTarget{\l__head_name_tl}
692       \skip_horizontal:N \l__head_heading_indent_dim
693     }
694   }
695 %

```

```

696 \__head_heading_format:n {
697   \template_process_order_clist:nnn
698   { head }{ order }
699   { number, prefix, punct, separator-a,
700     separator-b, separator-c, separator-d, title, subtitle }
701 }
702 \par
703 \group_end:
704 }

```

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

```

705 \NewTaggingSocketPlug{minipage/before}{heading}
706 { \tag_mc_end_push:
707   \bool_if:NT \l__tag_para_bool {\tag_struct_end:}
708   \tag_struct_begin:n{tag=\UseStructureName{sec/minipage}}
709 }
710 \NewTaggingSocketPlug{parbox/before}{heading}
711 { \tag_mc_end_push:
712   \bool_if:NT \l__tag_para_bool {\tag_struct_end:}
713   \tag_struct_begin:n{tag=\UseStructureName{sec/parbox}}
714 }
715 \NewTaggingSocketPlug{para/restore}{heading}

```

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

```

716 { \AssignStructureRole{para/textblock}{heading-fragment}
717   \bool_set_true:N\l__tag_para_flattened_bool
718   \tagpdfpara0n
719 }

```

headformat hang (*templ.*) This template creates a heading with a hanging number.

```

720 \DeclareTemplateCode{headformat}{hang}{6}
721 % args: keys,prefix,number,title,subtitle,quotation
722 {
723   , heading-indent    = \l__head_heading_indent_dim
724   , order-hang        = name {l__head_order-hang_clist}
725   , order             = \l__head_order_clist
726   , separator-a       = name {l__head_separator-a_tl}
727   , separator-b       = name {l__head_separator-b_tl}
728   , separator-c       = name {l__head_separator-c_tl}
729   , separator-d       = name {l__head_separator-d_tl}
730 }
731 {
732   \__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.

```

733 \SetTemplateKeys{headformat}{hang}{#1}
734 \group_begin:
735 %
736 \AssignTaggingSocketPlug{parbox/before}{noop}

```

```

737 \AssignTaggingSocketPlug{parbox/after}{noop}
738 \AssignTaggingSocketPlug{para/restore}{noop}
739 \tagpdfparaOff
740 %
741 \tl_set:Nn \l__head_prefix_tl {#2}
742 \tl_set:Nn \l__head_number_tl {#3}
743 \tl_set:Nn \l__head_title_tl {#4}
744 %
745 \normalfont
746 \interlinepenalty \@M
747 \l__head_heading_declsl_tl
748 \noindent
749 \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.

```

750 \bool_if:NTF \l__head_unnumbered_bool
751 { \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
752   { \skip_horizontal:N \l__head_heading_indent_dim
753     \MakeLinkTarget[\l__head_name_tl]{}
754   }
755   { \MakeLinkTarget[\l__head_name_tl]{}
756     \skip_horizontal:N \l__head_heading_indent_dim
757   }
758 }
759 { \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
760   { \skip_horizontal:N \l__head_heading_indent_dim
761     \MakeLinkTarget{\l__head_name_tl}
762   }
763   { \MakeLinkTarget{\l__head_name_tl}
764     \skip_horizontal:N \l__head_heading_indent_dim
765   }
766   \template_process_order_clist:nnn
767   { head }{ order-hang }
768   { number, prefix, punct, separator-a,
769     separator-b, separator-c, separator-d, title, subtitle }
770 }
771 }}
772 \hangindent \wd\@tempboxa
773 \box\@tempboxa
774 \template_process_order_clist:nnn
775 { head }{ order }
776 { number, prefix, punct, separator-a,
777   separator-b, separator-c, separator-d, title, subtitle }
778 \par
779 \group_end:
780 }
781

```

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

```

782 \DeclareTemplateCode{headformat}{runin}{6}

```

```

783 % args: keys,prefix,number,title,subtitle,quotation
784 {
785   , heading-indent    = \l__head_heading_indent_dim
786   , order             = \l__head_order_clist
787   , separator-a       = name {l__head_separator-a_tl}
788   , separator-b       = name {l__head_separator-b_tl}
789   , separator-c       = name {l__head_separator-c_tl}
790   , separator-d       = name {l__head_separator-d_tl}
791 }
792 {
793   \__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).

```

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

795   \group_begin:
796   %
797   \AssignTaggingSocketPlug{parbox/before}{noop}
798   \AssignTaggingSocketPlug{parbox/after}{noop}
799   \AssignTaggingSocketPlug{para/restore}{noop}
800   \tagpdfparaOff
801   %
802   \tl_set:Nn \l__head_prefix_tl {#2}
803   \tl_set:Nn \l__head_number_tl {#3}
804   \tl_set:Nn \l__head_title_tl {#4}

805   \normalfont

806   \interlinepenalty \OM
807   \l__head_heading_decls_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.

```

808   \bool_if:NTF \l__head_unnumbered_bool
809   { \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
810     { \skip_horizontal:N \l__head_heading_indent_dim
811       \MakeLinkTarget[\l__head_name_tl]{}
812     }
813     { \MakeLinkTarget[\l__head_name_tl]{}
814       \skip_horizontal:N \l__head_heading_indent_dim
815     }
816   }
817   { \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
818     { \skip_horizontal:N \l__head_heading_indent_dim
819       \MakeLinkTarget{\l__head_name_tl}
820     }
821     { \MakeLinkTarget{\l__head_name_tl}
822       \skip_horizontal:N \l__head_heading_indent_dim
823     }
824   }
825   \template_process_order_clist:nnn
826   { head }{ order }
827   { number, prefix, punct, separator-a,

```

```

828         separator-b, separator-c, separator-d, title, subtitle }
829 \group_end:
830 }

```

11.7.6 Internal commands used by the template code

`\__head_determine_penalty:`

```

831 \cs_new:Npn \__head_determine_penalty: {

```

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

```

832 \int_compare:nNnT \l__head_penalty_int = \c_max_int
833   { \int_set:Nn \l__head_penalty_int \@secpenalty }
834 }

```

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

`\__head_determine_number_typesetting:N`

```

835 \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.

```

836 \bool_if:NF \l__head_unnumbered_bool
837   { \bool_set:Nn \l__head_unnumbered_bool
838     { \bool_lazy_or_p:nn
839       { \int_compare_p:nNn \l__head_level_int > \c@secnumdepth }
840       { \bool_if_p:N #1 }
841     }
842   }

```

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)

```

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

```

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

```

845 \tl_set:Nn \l__head_number_tl { \NoValue }
846 \tl_set:Nn \l__head_prefix_tl { \NoValue }
847 \tl_set:Nn \l__head_punct_tl { \NoValue }
848 }

```

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.

```

849 {
850   \@kernel@refstepcounter{ \l__head_name_tl }
851   \tl_set:Nn \l__head_number_tl { \theheading }
852 }
853 }

```

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

`\_head\_vertical\_before\_spacing:`

```

854 \cs_new:Npn \_head\_vertical\_before\_spacing: {
855   \tl_if_blank:VTF \l__head\_placement\_start\_code\_tl
856   { % \if@noskipsec \leavevmode \fi \par          % <--- too late
857     \if@nobreak
858       \everypar{}
859     \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.

```

860   \sec\_add\_penalty\_with\_possible\_fil:n
861   \l__head\_penalty\_int
862   \addvspace \l__head\_before\_skip

```

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

```

863   \dim_compare:nNnF{\l__head\_after\_penalty\_skip}={0pt}
864   { \vspace*      \l__head\_after\_penalty\_skip }
865   \fi
866   }
867   {
868     \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`.

```

869   \bool_if:NF \l__head\_column\_spanning\_bool
870   {
871     \dim_compare:nNnF{\l__head\_after\_penalty\_skip}={0pt}
872     { \vspace*      \l__head\_after\_penalty\_skip }
873     }
874   }
875 }

```

(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.

```

876 \newcommand\addcontentslinebookmark[3]{}
877 \newcommand\addcontentslinebookmarkOff{}
878 \newcommand\addcontentslinebookmarkReset{}
879 \providecommand\texorpdfstring[2]{#1}

```

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

```

880 \AddToHook{package/hyperref/after}
881 {
882   \RenewCommandCopy\addcontentslinebookmark
883   \Hy@addcontentsline@addbookmark

```

```

884 \renewcommand\addcontentslinebookmarkOff
885 { \ifHy@bookmarks
886   \let\addcontentslinebookmarkReset
887     \Hy@bookmarkstrue
888   \else
889     \let\addcontentslinebookmarkReset\relax
890   \fi
891   \Hy@bookmarksfalse
892 }
893 }

```

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

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

```

894 \cs_new:Npn \_head_handle_marks_etc:nnnnn #1#2#3#4#5 {
895 %
896 \tl_set:Nn \@currentlabelname{#4}
897 \IfBlankF {#2}
898 { \_head_mark_cmd:n { #2 } }
899 \IfBlankTF {#1}

```

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

```

900 { \IfBlankF{#3}
901 {
902   \addcontentslinebookmark{toc}{\l__head_name_tl}
903   {
904     \bool_if:NF \l__head_unnumbered_bool
905     {
906       \protect\numberline{ \use:c{ the \l__head_name_tl } }
907     }
908     #3
909   }
910 }
911 }

```

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.

```

912 { \IfBlankT{#3}{\addcontentslinebookmarkOff}
913 \addcontentsline{toc}{ \l__head_name_tl }
914 {
915   \bool_if:NF \l__head_unnumbered_bool
916   {
917     \protect\numberline{ \use:c{ the \l__head_name_tl } }
918   }

```

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

```

919 \texorpdfstring{#1}{#3}
920 }
921 \addcontentslinebookmarkReset
922 }

```

perhaps this should have a hook for packages

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

```
923 \l__head_contents_extra_tl
```

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

```
924 \__head_debug_typeout:n{--->~label(s):~ \exp_not:n{#5}}
925 #5
926 }
```

(End of definition for __head_handle_marks_etc:nnnnn.)

11.8 Support for legacy classes and packages

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

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

then this gets overwritten by every class right now.

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

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

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

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

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

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

The current implementation

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

11.8.1 Instances (sample/default definitions)

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

```

927 \DeclareInstance{heading}{part}{display}
928 {
929   , name          = part
930   , level         = -1
931   , placement     = page
932   , after-penalty-vspace = 0pt
933   , prefix        = \partname
934   , number-format = \thepart
935   , heading-decls = \centering\bfseries\huge
936   , title-decls   = \Huge
937   , headformat-instance = part
938   , mark-cmd      = \partmark {#1}
939 }

```

heading chapter (*inst.*)

```

940 \DeclareInstance{heading}{chapter}{display}
941 {
942   , name          = chapter
943   , level         = 0
944   , placement     = top
945   , column-spanning = true
946   , after-penalty-vspace = 50pt
947   , after-vspace   = 40pt
948   , prefix        = \@chapapp
949   , number-format = \thechapter
950   , heading-decls = \raggedright \parindent0pt \bfseries \huge
951   , title-decls   = \Huge
952   , headformat-instance = chapter
953   , mark-cmd      = \chaptermark {#1}
954   , contents-extra = \addtocontents{lof}{\addvspace{10pt}}
955                   \addtocontents{lot}{\addvspace{10pt}}
956 }

```

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

```

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

```

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

heading section (*inst.*)

```

959 \DeclareInstance{heading}{section}{display}
960 {
961   , name          = section
962   , level         = 1
963   , mark-cmd      = \sectionmark {#1}
964   , before-vspace = 3.5ex plus 1ex minus .2ex
965   , after-vspace  = 2.3ex plus .2ex

```

```

966 , heading-decls = \normalfont\Large\bfseries
967 , headformat-instance = section
968 }

```

heading subsection (*inst.*)

```

969 \DeclareInstance{heading}{subsection}{display}
970 {
971   , name           = subsection
972   , parent-name    = section
973   , level          = 2
974   , mark-cmd       = \subsectionmark {#1}
975   , before-vspace  = 3.25ex plus 1ex minus .2ex
976   , after-vspace   = 1.5ex plus .2ex
977   , heading-decls  = \normalfont\large\bfseries
978   , headformat-instance = subsection
979 }

```

heading subsubsection (*inst.*)

```

980 \DeclareInstance{heading}{subsubsection}{display}
981 {
982   , name           = subsubsection
983   , parent-name    = subsection
984   , level          = 3
985   , before-vspace  = 3.25ex plus 1ex minus .2ex
986   , after-vspace   = 1.5ex plus .2ex
987   , heading-decls  = \normalfont\normalsize\bfseries
988   , headformat-instance = subsubsection
989 }

```

heading paragraph (*inst.*)

```

990 \DeclareInstance{heading}{paragraph}{runin}
991 {
992   , name           = paragraph
993   , parent-name    = subsubsection
994   , level          = 4
995   , before-vspace  = 3.25ex \@plus1ex \@minus.2ex
996   , after-hspace   = 1em
997   , heading-decls  = \normalfont\normalsize\bfseries
998   , mark-cmd       = \paragraphmark {#1}
999   , headformat-instance = paragraph
1000
1001 }

```

heading subparagraph (*inst.*)

```

1002 \DeclareInstance{heading}{subparagraph}{runin}
1003 {
1004   , name           = subparagraph
1005   , parent-name    = paragraph
1006   , level          = 5
1007   , before-vspace  = 3.25ex \@plus1ex \@minus.2ex
1008   , after-hspace   = 1em
1009   , heading-decls  = \normalfont\normalsize\bfseries
1010   , mark-cmd       = \subparagraphmark {#1}

```

```

1011 , headformat-instance = subparagraph
1012
1013 }

```

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

```

1014 \DeclareInstance{headformat}{part}{display}
1015 {
1016   , heading-indent      = 0pt
1017   , separator-a         = \nobreakspace           % between prefix and number
1018   , separator-b         = \par \nobreak \vspace{20pt} % between label block and title
1019 }
1020 \DeclareInstanceCopy{headformat}{chapter}{part}

```

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

```

headformat section (inst.)
headformat subsection (inst.)
headformat subsubsection (inst.)
1021 \DeclareInstance{headformat}{std}{hang}
1022 {
1023   , heading-indent = 0pt
1024 }
1025 \DeclareInstanceCopy{headformat}{section}{std}
1026 \DeclareInstanceCopy{headformat}{subsection}{std}
1027 \DeclareInstanceCopy{headformat}{subsubsection}{std}

```

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

```

headformat subparagraph (inst.)
1028 \DeclareInstance{headformat}{paragraph}{runin}
1029 {
1030   , heading-indent = 0pt
1031 }
1032 \DeclareInstance{headformat}{subparagraph}{runin}
1033 {
1034   , heading-indent = \parindent
1035 }

```

`headformat section-special` (*inst.*)

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

```

1036 \DeclareInstance{headformat}{section-special}{hang}
1037 {
1038   , heading-indent = 4em
1039   %%% , title-format = {#1!} % no longer
1040 }

```

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`.

```
1041 \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.

```
1042 \IfInstanceExistsF{heading}{#1-@startsection}{
1043     \__head_debug_typeout:n{Info:~ setting~ up~ instances~ for~
1044         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:

```
1045 \__head_prep_decls_and_format_from_tl:nNN {#6}
1046 \l__head_heading_decls_tl
1047 \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`.

```
1048 \@tempkipa #4\relax
1049 \@afterindenttrue
1050 \ifdim \@tempkipa <\z@
1051     \@tempkipa -\@tempkipa
1052     \@afterindentfalse
1053 \fi
```

... and a negative `#5` means we should produce a runin heading.

```
1054 \@tempkipb #5\relax
1055 \ifdim \@tempkipb >\z@
1056     \use:e {
1057         \DeclareInstance{heading}{#1-@startsection}{display}{
1058             , name           = #1
1059             , level          = #2
1060             , mark-cmd       = \exp_not:c {#1mark} {##1}
1061             , before-vspace = \the\@tempkipa
1062             , after-vspace  = \the\@tempkipb
1063             , 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.

```

1064         , heading-decls = \exp_not:o \l__head_heading_decls_tl
1065         , title-format  = \exp_not:o \l__head_title_format_tl {##1}
1066                     \exp_not:N \par
1067
1068     , headformat-instance = #1-@startsection
1069 }
\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.

```

1070     \else
1071       \@tempskipb=-\@tempskipb
1072       \use:e {
1073         \DeclareInstance{heading}{#1-@startsection}{runin}{
1074           , name          = #1
1075           , level         = #2
1076           , mark-cmd      = \exp_not:c {#1mark} {##1}
1077           , before-vspace = \the\@tempskipa
1078           , after-hspace  = \the\@tempskipb
1079           , heading-decls = \exp_not:o \l__head_heading_decls_tl

```

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

```

1080         , title-format  = \exp_not:o \l__head_title_format_tl {##1}
1081         , headformat-instance = #1-@startsection
1082       }
1083     \DeclareInstance{headformat}{#1-@startsection}{runin}{heading-indent = #3}
1084     \fi
1085   }
1086   \ParseLaTeXeHeading {#1-@startsection}{#7}{#8}{#9}
1087 }

```

(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.

```

1088 \cs_new:Npn \__head_prep_decls_and_format_from_tl:nNN #1#2#3
1089 { \tl_set:Nn #3 { \use:n }
1090   \exp_args:Ne
1091   \__head_prep_decls_and_auxi:nnNN
1092   { \tl_reverse:n {#1} } {#1} #2 #3
1093 }
1094 \cs_new:Npn \__head_prep_decls_and_auxi:nnNN #1#2#3#4
1095 { \tl_if_head_is_N_type:NTF {#1}
1096   { \__head_prep_decls_and_auxii:nNwNN {#2} #1 \q_stop #3 #4 }
1097   { \tl_set:Nn #3 {#2} }
1098 }

```

```

1099 \cs_new:Npn \__head_prep_decls_and_auxiii:nNwNN #1#2#3 \q_stop #4 #5
1100 { \token_if_macro:NTF #2
1101   { \exp_args:Ne
1102     \__head_prep_decls_and_auxiii:nnNnNN
1103     { \cmd_arg_spec:N #2 } {#1} #2 {#3} #4 #5
1104   }
1105   { \tl_set:Nn #4 {#1} }
1106 }

1107 \cs_new:Npn \__head_prep_decls_and_auxiii:nnNnNN #1#2#3#4#5#6 {
1108   \bool_lazy_any:nTF
1109   {

```

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

```

1110   { \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.

```

1111   { \str_if_eq_p:nn {#1} { m } }
1112   { \str_if_eq_p:nn {#1} { +m } }
1113   { \str_if_eq_p:nn {#1} { 0{ m } } }
1114   { \str_if_eq_p:nn {#1} { 0{ +m } } }

```

Another possibility are macros declared with `\DeclareRobustCommand`.

```

1115   { \bool_lazy_and_p:nn
1116     { \cs_if_exist_p:c { \cs_to_str:N #3 \c_space_tl } }
1117     { \str_if_eq_p:ee
1118       { \cs_parameter_spec:c { \cs_to_str:N #3 \c_space_tl } }
1119       { \c_hash_str 1 }
1120     }
1121   }
1122   { \bool_lazy_and_p:nn
1123     { \cs_if_exist_p:c { \token_to_str:N #3 \c_space_tl } }
1124     { \str_if_eq_p:ee
1125       { \cs_parameter_spec:c { \token_to_str:N #3 \c_space_tl } }
1126       { [ \c_hash_str 1 ] \c_hash_str 2 }
1127     }
1128   }
1129 }
1130 { \tl_set:Ne #5 { \tl_reverse:n {#4} }
1131   \tl_set:Nn #6 {#3}
1132 }
1133 { \tl_set:Nn #5 {#2} }
1134 }

```

```

1135 \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

```

1136 \NewCommandCopy\kernel@startsection\@startsection
1137 \AddToHook{class/after}[head/\@startsection]
1138 { \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

```

1139 \RenewDocumentCommand\secdef{mms0{#5}m}
1140 {
1141   \str_case:enF {\cs_to_str:N#1}
1142   {
1143     {@part}
1144     { \IfNoValueTF{#4}
1145       {\ParseLaTeXeHeading{part}{#3} {placement-start-code=\relax} {#5}}
1146       {
1147         \__head_process_shorttitle:nN{#4}\l__head_tmpa_tl
1148         \ExpandArgs{nne}\ParseLaTeXeHeading{part}{#3}
1149         {placement-start-code=\relax,\exp_not:o{\l__head_tmpa_tl}} {#5}
1150       }
1151       \@latex@warning{\noexpand\secdef~ detected~ in~ \noexpand\part
1152         command.\MessageBreak
1153         \noexpand\part~will~be~redefined~to~use~templates.\MessageBreak
1154         This~may~change~the~layout!}
1155       \DeclareDocumentCommand \part {s ={\shorttitle}o m}
1156       { \ParseLaTeXeHeading {part} {##1} {##2} {##3} }
1157     }
1158     {@chapter}
1159     { \IfNoValueTF{#4}
1160       { \ParseLaTeXeHeading{chapter}{#3} {#4} {#5} }
1161       {
1162         \__head_process_shorttitle:nN{#4}\l__head_tmpa_tl
1163         \ExpandArgs{nno}\ParseLaTeXeHeading{chapter}{#3}
1164         {\l__head_tmpa_tl} {#5}
1165       }
1166       \@latex@warning{\noexpand\secdef~ detected~ in~
1167         \noexpand\chapter command.\MessageBreak
1168         \noexpand\chapter~ will~ be~ redefined~ to~ use~
1169         templates.\MessageBreak
1170         This~may~change~the~layout!}
1171       \DeclareDocumentCommand \chapter {s ={\shorttitle}o m}
1172       { \ParseLaTeXeHeading {chapter} {##1} {##2} {##3} }
1173     }
1174   }
1175   {
1176     \tag_if_active:TF@latex@error@latex@warning
1177     {\noexpand\secdef~ with~ unknown~ argument~ \noexpand#1
1178       found.\MessageBreak
1179       The~command~can~not~be~adapted~on~the~fly~to~support~tagging.\MessageBreak
1180       The~heading~command~using~this~\noexpand\secdef~should~be~
1181       reimplemented\MessageBreak with~templates}{ }
1182     \IfBooleanTF{#3}{#2{#5}}{#1[#4]{#5}}
1183   }
1184 }

```

A helper command to reprocess the argument as key-val

```
1185 \NewDocumentCommand\__head_process_shorttitle:nN{={shorttitle}mm}
1186 { \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.

```
1187 \AddToHook{class/article/after}[head/example]
1188 {
1189   \DeclareInstance{heading}{part}{display}
1190   {
1191     , name          = part
1192     , level         = -1
1193     , before-vspace = 4ex
1194     , after-vspace  = 3ex
1195     , mark-cmd      = \partmark {#1}
1196     , prefix        = \partname
1197     , punct         =
1198     , heading-decls = \raggedright\bfseries\Large
1199     , title-decls   = \huge
1200     , headformat-instance = part
1201   }
1202   \DeclareInstance{headformat}{part}{display}
1203   {
1204     , heading-indent = 0pt
1205     , separator-a    = \nobreakspace
1206     , separator-b    = \par \nobreak \vspace{20pt}
1207   }
1208   \DeclareDocumentCommand \part {s = {shorttitle} o m}
1209   { \ParseLaTeXeHeading {part} {#1} {#2} {#3} }
1210   \__head_setup_default_heading:
1211 }

1212 \AddToHook{class/report/after}[head/example]
1213 {
1214   \DeclareDocumentCommand \part {s = {shorttitle} o m}
1215   { \ParseLaTeXeHeading {part} {#1} {#2} {#3} }
1216   \DeclareDocumentCommand \chapter {s = {shorttitle} o m}
1217   {
1218     \ParseLaTeXeHeading {chapter}{#1} {#2} {#3}
1219   }
1220   \__head_setup_default_heading:
1221 }
1222 \AddToHook{class/book/after}[head/example]
1223 {
1224   \DeclareDocumentCommand \part {s = {shorttitle} o m}
1225   { \ParseLaTeXeHeading {part} {#1} {#2} {#3} }
1226   \DeclareDocumentCommand \chapter {s = {shorttitle} o m}
1227   {
1228     \if@mainmatter
1229       \ParseLaTeXeHeading {chapter}{#1} {#2} {#3}

```

```

1230         \else
1231         \IfBooleanTF{#1}
1232         {% starred:
1233         \ParseLaTeXeHeading {chapter}{\BooleanTrue} {#2} {#3}
1234         }
1235         {\IfNoValueTF{#2}
1236         {\ParseLaTeXeHeading {chapter}{\BooleanTrue} {shorttitle={#3}} {#3}}
1237         {\ParseLaTeXeHeading {chapter}{\BooleanTrue} {#2} {#3}}
1238         }
1239         \fi
1240     }
1241     \__head_setup_default_heading:
1242 }

1243 \cs_new_protected:Npn \__head_setup_default_heading:
1244 {

\section

1245     \DeclareDocumentCommand \section {s ={\shorttitle}o m}
1246     { \ParseLaTeXeHeading {section} {##1} {##2} {##3} }

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

\subsection

1247     \DeclareDocumentCommand \subsection {s ={\shorttitle}o m}
1248     { \ParseLaTeXeHeading {subsection} {##1} {##2} {##3} }

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

\subsubsection

1249     \DeclareDocumentCommand \subsubsection {s ={\shorttitle}o m}
1250     { \ParseLaTeXeHeading {subsubsection} {##1} {##2} {##3} }

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

\paragraph

1251     \DeclareDocumentCommand \paragraph {s ={\shorttitle}o m}
1252     { \ParseLaTeXeHeading {paragraph} {##1} {##2} {##3} }

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

\subparagraph

1253     \DeclareDocumentCommand \subparagraph {s ={\shorttitle}o m}
1254     { \ParseLaTeXeHeading {subparagraph} {##1} {##2} {##3} }
1255     } %end of command

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

1256 \</package>

```

12 Issues and problems noticed along the way

12.1 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.

12.2 `\SetCurrentTemplateKeys`

Furthermore, I think I would like to have some `\SetCurrentTemplateKeys` where one does not have to specify the type nor the template name, because that is how it is always used (by me).

12.3 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.

12.4 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...`

12.5 Indexheading at least for `l3doc` is wrong now

Needs checking.

A Templates from the 2026-06-01 release of `LATEX`

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

A.1 Template documentation for -v1 templates

A.1.1 Templates of type heading

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

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

A.1.2 The heading template ‘<all>’

Attributes:

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

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

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

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

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

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

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

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`

⁴Maybe questionable

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

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

All templates should set up `\theheading` to correspond to `\the<name>`.

A.1.3 The heading template ‘display-v1’

Attributes:

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

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

penalty (*integer*) Penalty to break before the heading. The default (T_EX'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 `runin` headings so should be dropped! Default: `0pt`

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

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

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

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

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

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

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

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

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

A.1.5 Templates of type `headformat`

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

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

A.1.6 The `headformat` template ‘hang-v1’

Attributes:

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

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

prefix-number-sep (*tokenlist*) Token list that can be used in the assignment to a $\text{\TeX}$ 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 $\text{\TeX}$ 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 $\text{\LaTeX}$ 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.*)

```

1257 \DeclareTemplateInterface{heading}{display-v1}{9}
1258 {
1259   , name           : tokenlist
1260   , parent-name    : tokenlist
1261   , reset-counter  : tokenlist
1262   , level          : integer = 0
1263   , placement      : choice {page , top , normal } = normal
1264   , mark-cmd       : function(1) =
1265   , para-indent    : choice { true , false } = false
1266   , before-vspace  : skip = Opt
1267   , penalty        : integer = \c_max_int
1268   , after-penalty-vspace : skip = Opt
1269   , after-vspace   : skip = Opt
1270   , start-code     : tokenlist =      % no default values!
1271   , final-code     : tokenlist =      % no default values!
1272   , prefix         : tokenlist = \NoValue
1273   , heading-decls  : tokenlist = \normalfont
1274   , prefix-decls   : tokenlist =
1275   , number-decls   : tokenlist =
1276   , title-decls    : tokenlist =
1277   , subtitle-decls : tokenlist =
1278   , quote-decls    : tokenlist =
1279   , headformat-instance : tokenlist = std
1280   , number-format  : function(1) = \theheading
1281   , contents-extra : tokenlist =
1282 }
1283 \DeclareTemplateCode{heading}{display-v1}{9}
1284 {
1285   , name           = \l__head_name_tl
1286   , level          = \l__head_level_int
1287   , parent-name    = \l__head_pname_tl
1288   , reset-counter  = \l__head_reset_cnt_tl
1289   , placement      = {
1290     ,page = \__head_debug_typeout:n{ A~ page~ heading }
1291             \tl_set:Nn \l__head_placement_tl { page }
1292     ,top   = \__head_debug_typeout:n{ A~ top~ heading }
1293             \tl_set:Nn \l__head_placement_tl { top }
1294     ,normal = \__head_debug_typeout:n{ A~ normal~ heading }
1295             \tl_set:Nn \l__head_placement_tl { normal }
1296           }
1297   , mark-cmd       = \__head_mark_cmd:n
1298   , para-indent    = {
1299     ,true  = \@afterindenttrue
1300     ,false = \@afterindentfalse
1301   }

```

```

1302 , before-vspace = \l__head_before_skip
1303 , penalty       = \l__head_penalty_int
1304 , after-penalty-vspace = \l__head_after_penalty_skip
1305 , after-vspace  = \l__head_after_skip
1306 , start-code    = \l__head_start_code_tl
1307 , final-code    = \l__head_final_code_tl
1308 , prefix        = \l__head_typeset_prefix_tl
1309 , headformat-instance = \l__head_headformat_instance_tl
1310 , heading-decls = \l__head_heading_decls_tl
1311 , prefix-decls  = \l__head_prefix_decls_tl
1312 , number-decls  = \l__head_number_decls_tl
1313 , title-decls   = \l__head_title_decls_tl
1314 , subtitle-decls = \l__head_subtitle_decls_tl
1315 , quote-decls   = \l__head_quote_decls_tl
1316 , number-format = \__head_number_format:n
1317 , contents-extra = \l__head_contents_extra_tl
1318 }
1319 {
1320   \tl_set_eq:Nc \theheading { the \l__head_name_tl }
1321   \tl_set:Nc\l__head_saved_secnumdepth_tl{\int_use:N\c@secnumdepth}
1322   \__head_show_arguments:nnnnnnnnn
1323     {#1}{#2}{#3}{#4}{#5}{#6}{#7}{#8}{#9}
1324   \tl_if_empty:oF {#1} { \SetTemplateKeys{heading}{display}{#1} }
1325   \tl_if_empty:oT \l__head_start_code_tl
1326   { \str_case:VnF \l__head_placement_tl
1327     {
1328       { page }
1329       { \tl_set:Nn \l__head_start_code_tl
1330         {
1331           \if@openright
1332             \cleardoublepage
1333           \else
1334             \clearpage
1335           \fi
1336           \thispagestyle{plain}
1337           \if@twocolumn
1338             \onecolumn
1339             \@tempswattrue
1340           \else
1341             \@tempswafalse
1342           \fi
1343           \null\vfil
1344         }
1345       }
1346       { top }
1347       { \tl_set:Nn \l__head_start_code_tl
1348         {
1349           \if@openright\cleardoublepage\else\clearpage\fi
1350           \thispagestyle{plain}
1351           \global\@topnum\z@
1352         }
1353       }
1354     }
1355   { \tl_clear:N \l__head_start_code_tl }

```

```

1356 }
1357 \tl_if_empty:oT \l__head_final_code_tl
1358 { \str_case:VnF \l__head_placement_tl
1359   {
1360     { page }
1361     { \tl_set:Nn \l__head_final_code_tl
1362       {
1363         \vfil\newpage
1364         \if@twoside
1365           \if@openright
1366             \null
1367             \thispagestyle{empty}
1368           \newpage
1369         \fi
1370       \fi
1371       \if@tempwa
1372         \twocolumn
1373       \fi
1374     }
1375   }
1376 }
1377 { \tl_set:Nn \l__head_final_code_tl { \@afterheading } }
1378 }
1379 \__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;

```

1380 \bool_set:Nn \l__head_unnumbered_bool
1381 { \bool_lazy_or:p:nn
1382   { \int_compare_p:nNn \l__head_level_int > \c@secnumdepth }
1383   { \bool_if_p:N #2 }
1384 }
1385 \bool_if:NTF \l__head_unnumbered_bool
1386 {
1387   \int_gset:Nn\c@secnumdepth{-99}
1388   \tl_clear:N \l__head_typeset_number_tl
1389 }
1390 {
1391   \@kernel@refstepcounter{ \l__head_name_tl }
1392   \protected@edef \l__head_typeset_number_tl
1393   {
1394     \__head_number_format:n { \l__head_name_tl }
1395   }
1396 }

```

End of inlined code.

```

1397 \__head_vertical_before_spacing:
1398 \UseTaggingSocket{sec/end}{\int_use:N\l__head_level_int}
1399 \UseTaggingSocket{sec/begin}
1400   {{\int_use:N\l__head_level_int}
1401     {tag=\UseStructureName{sec/\int_use:N\l__head_level_int}}}
1402 \__head_debug_typeout:n{use~ 'headformat'~instance:~
1403   \l__head_headformat_instance_tl }
1404 \use:e {

```

```

1405 \UseInstance{headformat} { \l__head_headformat_instance_tl }
1406 { \exp_not:o \UnusedTemplateKeys }
1407 { \exp_not:o { \l__head_typeset_prefix_tl } }
1408 { \exp_not:o { \l__head_typeset_number_tl } }
1409 { \exp_not:n { #3 } }
1410 { \exp_not:o { \use_i:nn #9 } }
1411 { \exp_not:o { \use_ii:nn #9 } }
1412 }
1413 \__head_handle_marks_etc:nnnnn {#4}{#5}{#6}{#7}{#8}
1414 \int_gset:Nn\c@secnumdepth{\l__head_saved_secnumdepth_tl}
1415 \par \nobreak
1416 \skip_vertical:N \l__head_after_skip
1417 \l__head_final_code_tl
1418 \ignorespaces
1419 }

```

heading runin-v1 (*templ.*)

```

1420 \DeclareTemplateCopy{heading}{runin-v1}{display-v1}
1421 \DeclareTemplateCode{heading}{runin-v1}{9}
1422 {
1423   , name           = \l__head_name_tl
1424   , level          = \l__head_level_int
1425   , parent-name    = \l__head_pname_tl
1426   , reset-counter  = \l__head_reset_cnt_tl
1427   , placement      = {
1428     page           = \typeout{ ^^JA~ runin~ page~ placement~ heading-makes-no-sense
1429                       (top-used)}
1430     \tl_set:Nn \l__head_placement_tl { top }
1431     ,top          = \__head_debug_typeout:n{ A~ top~ heading }
1432     \tl_set:Nn \l__head_placement_tl { top }
1433     ,normal       = \__head_debug_typeout:n{ A~ normal~ heading }
1434     \tl_set:Nn \l__head_placement_tl { normal }
1435   }
1436   , mark-cmd       = \__head_mark_cmd:n
1437   , para-indent    = {
1438     ,true          = %\typeout{para-indent~ setting~ ignored}
1439     ,false         = %\typeout{para-indent~ setting~ ignored}
1440   }
1441   , before-vspace  = \l__head_before_skip
1442   , penalty        = \l__head_penalty_int
1443   , after-penalty-vspace = \l__head_after_penalty_skip
1444   , after-vspace   = \l__head_after_skip
1445   , start-code     = \l__head_start_code_tl
1446   , final-code     = \l__head_final_code_tl
1447   , prefix         = \l__head_typeset_prefix_tl
1448   , headformat-instance = \l__head_headformat_instance_tl
1449   , heading-decls  = \l__head_heading_decls_tl
1450   , prefix-decls   = \l__head_prefix_decls_tl
1451   , number-decls   = \l__head_number_decls_tl
1452   , title-decls    = \l__head_title_decls_tl
1453   , subtitle-decls = \l__head_subtitle_decls_tl
1454   , quote-decls    = \l__head_quote_decls_tl
1455   , number-format  = \__head_number_format:n
1456   , contents-extra = \l__head_contents_extra_tl

```

```

1457 }
1458 {
1459   \__head_show_arguments:nnnnnnnnnn
1460     {#1}{#2}{#3}{#4}{#5}{#6}{#7}{#8}{#9}
1461   \tl_set:Nc\l__head_saved_secnumdepth_tl{\int_use:N\c@secnumdepth}
1462   \tl_set_eq:Nc \theheading { the \l__head_name_tl }
1463   \tl_if_empty:oF {#1} { \SetTemplateKeys{heading}{runin}{#1} }
1464   \tl_if_empty:oT \l__head_start_code_tl
1465   {
1466     \str_case:Vn \l__head_placement_tl
1467     {
1468       { top }    { \tl_set:Nn \l__head_start_code_tl {\clearpage} }
1469     }
1470   }
1471   \__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;

```

1472   \bool_set:Nn \l__head_unnumbered_bool
1473     { \bool_lazy_or_p:nn
1474       { \int_compare_p:nNn \l__head_level_int > \c@secnumdepth }
1475       { \bool_if_p:N #2 }
1476     }
1477   \bool_if:NTF \l__head_unnumbered_bool
1478   {
1479     \int_gset:Nn\c@secnumdepth{-99}
1480     \tl_clear:N \l__head_typeset_number_tl
1481   }
1482   {
1483     \@kernel@refstepcounter{ \l__head_name_tl }
1484     \protected@edef \l__head_typeset_number_tl
1485       {
1486         \__head_number_format:n { \l__head_name_tl }
1487       }
1488   }

```

End of inlined code.

```

1489   \__head_vertical_before_spacing:
1490   \UseTaggingSocket{sec/end}{\int_use:N\l__head_level_int}
1491   \UseTaggingSocket{sec/begin}
1492     {{\int_use:N\l__head_level_int}{tag=\UseStructureName{sec/\int_use:N\l__head_level_int}}}
1493   \UseTaggingSocket{sec/title/init}{\int_use:N\l__head_level_int}
1494   \def \@svsechd {
1495     \__head_debug_typeout:n{use~ 'headformat'~instance:~
1496       \l__head_headformat_instance_tl }
1497     \use:e {
1498       \UseInstance{headformat} { \l__head_headformat_instance_tl }
1499       { \exp_not:o \UnusedTemplateKeys }
1500       { \exp_not:o { \l__head_typeset_prefix_tl } }
1501       { \exp_not:o { \l__head_typeset_number_tl } }
1502       { \exp_not:n { #3 } }
1503       { \exp_not:o { \use_i:nn #9 } }
1504       { \exp_not:o { \use_ii:nn #9 } }
1505     }

```

```

1506     \__head_handle_marks_etc:nnnnn {#4}{#5}{#6}{#7}{#8}
1507     \int_gset:Nn\c@secnumdepth{\l__head_saved_secnumdepth_tl}
1508   }
1509   \@nobreakfalse
1510   \global\@noskipsectrue
1511   \everypar{%
1512     \if@noskipsec
1513       \global\@noskipsecfalse
1514       {\setbox\z@\lastbox}
1515       \clubpenalty\@M
1516       \@svsechd
1517       \unskip
1518       \UseTaggingSocket{sec/title/split}
1519       \skip_horizontal:N \l__head_after_skip
1520     \else
1521       \clubpenalty \@clubpenalty
1522       \everypar{}}%
1523   \fi
1524 }
1525 \l__head_final_code_tl
1526 \ignorespaces
1527 }

```

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

```

1528 \DeclareTemplateInterface{headformat}{display-v1}{6}
1529 {
1530   , heading-indent      : length = Opt
1531   , title-format        : function(1) = #1
1532   , prefix-number-sep   : tokenlist = \WordSpaceAmount{1}
1533   , number-title-sep    : tokenlist = 20pt
1534 }

```

headformat hang-v1 (*templ.*)

```

1535 \DeclareTemplateInterface{headformat}{hang-v1}{6}
1536 {
1537   , heading-indent      : length = Opt
1538   , title-format        : function(1) = #1
1539   , prefix-number-sep   : tokenlist = \WordSpaceAmount{1}
1540   , number-title-sep    : tokenlist = 1em
1541 }

```

headformat runin-v1 (*templ.*)

```

1542 \DeclareTemplateInterface{headformat}{runin-v1}{6}
1543 {
1544   , heading-indent      : length = Opt
1545   , title-format        : function(1) = #1
1546   , prefix-number-sep   : tokenlist = \WordSpaceAmount{1}
1547   , number-title-sep    : tokenlist = 1em
1548 }

```

headformat display-v1 (*templ.*)

```

1549 \DeclareTemplateCode{headformat}{display-v1}{6}
1550   % args: keys,prefix,number,title,subtitle,quotation

```

```

1551 {
1552   , heading-indent      = \l__head_heading_indent_dim
1553   , title-format        = \__head_title_format:n
1554   , prefix-number-sep   = \l__head_prefix_number_sep_tl
1555   , number-title-sep    = \l__head_number_title_sep_tl
1556 }
1557 {
1558   \__head_show_arguments:nnnnnn {#1}{#2}{#3}{#4}{#5}{#6}
1559   \tl_if_empty:oF {#1} { \SetTemplateKeys{headformat}{display-v1}{#1} }
1560   \group_begin:
1561     \UseTaggingSocket{sec/title/begin}{\int_use:N\l__head_level_int}{#4}}
1562     \normalfont
1563     \interlinepenalty \@M
1564     \l__head_heading_decls_tl{}
1565     \bool_if:NTF \l__head_unnumbered_bool
1566     {
1567       \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
1568       {
1569         \skip_horizontal:N \l__head_heading_indent_dim
1570         \MakeLinkTarget[\l__head_name_tl]{}
1571       }
1572       {
1573         \MakeLinkTarget[\l__head_name_tl]{}
1574         \skip_horizontal:N \l__head_heading_indent_dim
1575       }
1576     }
1577     {
1578       \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
1579       {
1580         \skip_horizontal:N \l__head_heading_indent_dim
1581         \MakeLinkTarget{\l__head_name_tl}
1582       }
1583       {
1584         \MakeLinkTarget{\l__head_name_tl}
1585         \skip_horizontal:N \l__head_heading_indent_dim
1586       }
1587       \IfNoValueF{#2}
1588       {
1589         { \l__head_prefix_decls_tl #2 }
1590         \nobreak
1591         \skip_horizontal:n { \l__head_prefix_number_sep_tl }
1592       }
1593       \l__head_number_decls_tl
1594       #3
1595       \par\nobreak
1596       \skip_vertical:n { \l__head_number_title_sep_tl }
1597     }
1598     \l__head_title_decls_tl
1599     \__head_title_format:n {#4}
1600     \par
1601     \UseTaggingSocket{sec/title/end}
1602   \group_end:
1603 }

```

headformat hang-v1 (*templ.*)

```

1604 \DeclareTemplateCode{headformat}{hang-v1}{6}
1605 {
1606   , heading-indent      = \l__head_heading_indent_dim
1607   , title-format        = \__head_title_format:n
1608   , prefix-number-sep   = \l__head_prefix_number_sep_tl
1609   , number-title-sep    = \l__head_number_title_sep_tl
1610 }
1611 {
1612   \__head_show_arguments:nnnnnn {#1}{#2}{#3}{#4}{#5}{#6}
1613   \tl_if_empty:oF {#1} { \SetTemplateKeys{headformat}{hang-v1}{#1} }
1614   \group_begin:
1615     \UseTaggingSocket{sec/title/init}{\int_use:N\l__head_level_int}
1616     \normalfont
1617     \interlinepenalty \@M
1618     \l__head_heading_decls_tl{}
1619     \bool_if:NTF \l__head_unnumbered_bool
1620     {
1621       \tl_set:Nn\l__head_tmpa_tl
1622       {
1623         \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
1624         {
1625           \skip_horizontal:N \l__head_heading_indent_dim
1626           \MakeLinkTarget[\l__head_name_tl]{}
1627         }
1628         {
1629           \MakeLinkTarget[\l__head_name_tl]{}
1630           \skip_horizontal:N \l__head_heading_indent_dim
1631         }
1632       }
1633     }
1634     {
1635       \tl_set:Nn \l__head_tmpa_tl
1636       {
1637         \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
1638         {
1639           \skip_horizontal:N \l__head_heading_indent_dim
1640           \MakeLinkTarget{\l__head_name_tl}
1641         }
1642         {
1643           \MakeLinkTarget{\l__head_name_tl}
1644           \skip_horizontal:N \l__head_heading_indent_dim
1645         }
1646         \IfNoValueF{#2}
1647         {
1648           { \l__head_prefix_decls_tl #2 }
1649           \nobreak
1650           \skip_horizontal:n { \l__head_prefix_number_sep_tl }
1651         }
1652         { \l__head_number_decls_tl #3 }
1653         \skip_horizontal:n { \l__head_number_title_sep_tl }
1654       }
1655     }
1656     \UseTaggingSocket{sec/title/hang}

```

```

1657     {{\int_use:N\l__head_level_int}\l__head_unnumbered_bool{\l__head_tmpa_tl}{#4}}
1658     {
1659         \@hangfrom
1660         {
1661             \l__head_tmpa_tl
1662         }
1663     }
1664     \l__head_title_decls_tl
1665     \__head_title_format:n {#4}
1666     \par
1667 \group_end:
1668 }

```

headformat runin-v1 (*templ.*)

```

1669 \DeclareTemplateCode{headformat}{runin-v1}{6}
1670 {
1671     , heading-indent      = \l__head_heading_indent_dim
1672     , title-format       = \__head_title_format:n
1673     , prefix-number-sep = \l__head_prefix_number_sep_tl
1674     , number-title-sep  = \l__head_number_title_sep_tl
1675 }
1676 {
1677     \__head_show_arguments:nnnnnn {#1}{#2}{#3}{#4}{#5}{#6}
1678     \tl_if_empty:oF {#1} { \SetTemplateKeys{headformat}{runin-v1}{#1} }
1679     \group_begin:
1680         \normalfont
1681         \interlinepenalty \@M
1682         \l__head_heading_decls_tl{}
1683         \bool_if:NTF \l__head_unnumbered_bool
1684         {
1685             \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
1686             {
1687                 \skip_horizontal:N \l__head_heading_indent_dim
1688                 \MakeLinkTarget[\l__head_name_tl]{}
1689             }
1690             {
1691                 \MakeLinkTarget[\l__head_name_tl]{}
1692                 \skip_horizontal:N \l__head_heading_indent_dim
1693             }
1694         }
1695         {
1696             \dim_compare:nNnTF \l__head_heading_indent_dim < \c_zero_skip
1697             {
1698                 \skip_horizontal:N \l__head_heading_indent_dim
1699                 \MakeLinkTarget{\l__head_name_tl}
1700             }
1701             {
1702                 \MakeLinkTarget{\l__head_name_tl}
1703                 \skip_horizontal:N \l__head_heading_indent_dim
1704             }
1705             {
1706                 \UseTaggingSocket{sec/title/number}{\int_use:N\l__head_level_int}
1707                 {
1708                     \IfNoValueF{#2}

```

```

1709         {
1710             { \l__head_prefix_decls_tl #2 }
1711             \nobreak
1712             \skip_horizontal:n { \l__head_prefix_number_sep_tl }
1713         }
1714         \l__head_number_decls_tl
1715         #3
1716     }
1717 }
1718 \skip_horizontal:n { \l__head_number_title_sep_tl }
1719 }
1720 \l__head_title_decls_tl
1721 \__head_title_format:n {#4}
1722 \group_end:
1723 }

```

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	
\!	152, 154
\,	153, 155
\.	152, 154
\:	153, 155
\;	153, 155
\?	152, 154
\@startsection	19
_	254, 265, 276
_	16, 17
A	
\addcontents	14, 15, 63
\addcontentsline	20, 49, 913
\addcontentslinebookmark	876, 902
\addcontentslinebookmarkOff	876, 912
\addcontentslinebookmarkOn	876
\addcontentslinebookmarkReset	878, 886, 889, 921
\addpenalty	34, 37, 38, 48, 471, 475, 476
\addpenaltywithfil	468, 476
\AddPunct	157
\addtocontents	954, 955
\AddToHook	67, 511, 880, 1137, 1187, 1212, 1222
\advspace	38, 862, 954, 955
\AssignStructureRole	716
\AssignTaggingSocketPlug	666, 667, 668, 736, 737, 738, 797, 798, 799
\AtBeginDocument	957
B	
\baselineskip	14
\bfseries	935, 950, 966, 977, 987, 997, 1009, 1198
bool commands:	
\bool_gset_false:N	18
\bool_gset_true:N	13
\bool_if:NTF	24, 26, 96, 337, 427, 441, 456, 575, 678, 707, 712, 750, 808, 836, 843, 869, 904, 915, 1385, 1477, 1565, 1619, 1683
\bool_if_p:N	840, 1383, 1475
\bool_lazy_and_p:nn	1115, 1122
\bool_lazy_any:nTF	1108
\bool_lazy_or_p:nn	838, 1381, 1473
\bool_new:N	8, 59
\bool_set:Nn	837, 1380, 1472
\bool_set_false:N	84, 346, 578
\bool_set_true:N	79, 717
\BooleanFalse	10, 96
\BooleanTrue	10, 96, 1233, 1236, 1237
\box	773
C	
\centering	935
\chapter	4, 5, 9, 16, 18, 20, 50, 51, 57, 66, 1167, 1168, 1171, 1216, 1226
\chaptermark	13, 62, 953
\cleardoublepage	365, 381, 1332, 1349

`\clearpage` 12,
48, 61, 367, 381, 593, 1334, 1349, 1468
`\clubpenalty` 639, 645, 1515, 1521
cmd commands:
`\cmd_arg_spec:N` 1103
`\cs` 460
cs commands:
`\cs_generate_variant:Nn` 1135
`\cs_gset_protected:Npe` 23, 25
`\cs_if_exist:NTF` 958
`\cs_if_exist_p:N` 1116, 1123
`\cs_new:Npn` 31, 40, 124, 137, 831, 835,
854, 894, 958, 1088, 1094, 1099, 1107
`\cs_new_eq:NN` 9, 10
`\cs_new_protected:Npn` . 11, 16, 21,
28, 29, 68, 156, 157, 466, 476, 508, 1243
`\cs_parameter_spec:N`
..... 1110, 1118, 1125, 1135
`\cs_set_eq:NN` 387, 469, 473, 595
`\cs_set_protected:Npn` 505
`\cs_to_str:N` 1116, 1118, 1141
`\csname` 514

D
`\DebugHeadingsOff` 21, 28
`\DebugHeadingsOn` 21, 28
`\DeclareDocumentCommand` 1041,
1155, 1171, 1208, 1214, 1216, 1224,
1226, 1245, 1247, 1249, 1251, 1253
`\DeclareInstance`
. 60, 927, 940, 959, 969, 980, 990,
1002, 1014, 1021, 1028, 1032, 1036,
1057, 1069, 1073, 1083, 1189, 1202
`\DeclareInstanceCopy` 1020, 1025, 1026, 1027
`\DeclareRobustCommand` 56
`\DeclareTemplateCode` 30, 279, 517, 652,
720, 782, 1283, 1421, 1549, 1604, 1669
`\DeclareTemplateCopy` 30, 1420
`\DeclareTemplateInterface` . 169, 207,
245, 257, 268, 1257, 1528, 1535, 1542
`\def` 56, 152, 154, 618, 1494
dim commands:
`\dim_compare:nNnTF`
..... 430, 483, 487, 489, 497,
679, 687, 751, 759, 809, 817, 863,
871, 1567, 1578, 1623, 1637, 1685, 1696
`\dim_set:Nn` 486
`\l_tmpa_dim` 486, 494, 497, 498
`\c_zero_dim` 483, 490, 497
`\DocumentMetadata` 3, 27

E
`\EditTemplateDefaults` 30

`\else` 161, 345, 366,
373, 381, 513, 644, 859, 888, 1063,
1070, 1230, 1333, 1340, 1349, 1520
`\endcsname` 512, 514
`\everypar` ... 41, 635, 646, 858, 1511, 1522
exp commands:
`\exp_args:Ne` 1090, 1101
`\exp_not:N` 94, 1060, 1066, 1076
`\exp_not:n` 33,
34, 35, 36, 37, 38, 42, 43, 44, 45,
46, 47, 48, 49, 50, 72, 73, 95, 97,
98, 99, 100, 101, 102, 103, 104,
135, 428, 435, 436, 437, 438, 439,
440, 623, 624, 625, 626, 627, 628,
924, 1064, 1065, 1079, 1080, 1149,
1406, 1407, 1408, 1409, 1410, 1411,
1499, 1500, 1501, 1502, 1503, 1504
`\expandafter` 514
`\ExpandArgs` 1148, 1163

F
`\fbox` 4
`\fi` ... 69, 164, 165, 347, 368, 375, 381,
405, 406, 409, 515, 647, 856, 865,
890, 1053, 1063, 1084, 1239, 1335,
1342, 1349, 1369, 1370, 1373, 1523
`\filbreak` 37, 48
`\final-code` 13
`\font` 147, 148, 149
`\fontdimen` 147, 148, 149
`\frenchspacing` 27, 152

G
`\global` ... 383, 634, 637, 1351, 1510, 1513
`\glueexpr` 146
group commands:
`\group_begin:`
. 481, 665, 734, 795, 1560, 1614, 1679
`\group_end:`
. 501, 703, 779, 829, 1602, 1667, 1722

H
`\hangindent` 772
`\hbox` 749
head commands:
`\head_debug_off:` 21, 11, 16, 29
`\head_debug_on:` 21, 11, 11, 28
head internal commands:
`\l__head_after_penalty_skip`
..... 305, 430,
431, 544, 863, 864, 871, 872, 1304, 1443
`\l__head_after_skip` 306, 443,
458, 545, 643, 1305, 1416, 1444, 1519
`\l__head_before_skip`
..... 303, 542, 862, 1302, 1441

```

\l__head_bookmark_tl .....
..... 53, 78, 83, 100, 110
\l__head_column_spanning_bool ..
..... 36, 297, 337,
346, 427, 441, 456, 536, 575, 578, 869
\l__head_contents_extra_tl ....
..... 329, 568, 923, 1317, 1456
\__head_debug:n ..... 9, 9, 23
\g__head_debug_bool .. 8, 13, 18, 24, 26
\__head_debug_gset: .... 11, 14, 19, 21
\__head_debug_typeof:n 9, 10, 25,
32, 33, 34, 35, 36, 37, 38, 41, 42, 43,
44, 45, 46, 47, 48, 49, 50, 70, 71, 92,
126, 130, 135, 140, 160, 162, 288,
290, 292, 294, 341, 423, 467, 529,
531, 533, 576, 619, 924, 1043, 1290,
1292, 1294, 1402, 1431, 1433, 1495
\l__head_deprecated_end_code_tl
..... 310, 354, 357, 549, 585, 588
\l__head_deprecated_start_code_-
tl .... 309, 349, 352, 548, 580, 583
\__head_determine_number_-
typesetting:N .....
..... 68, 70, 416, 610, 835, 835
\__head_determine_penalty: ....
..... 415, 609, 831, 831, 1379, 1471
\l__head_final_code_tl .... 1307,
1357, 1361, 1377, 1417, 1446, 1525
\__head_find_label:w .... 74, 124, 124
\__head_find_label_aux:w .....
..... 26, 133, 137, 142
\__head_handle_marks_etc:nnnnn .
..... 451, 630, 894, 894, 1413, 1506
\l__head_headformat_instance_tl
..... 313, 424, 434, 552, 620,
622, 1309, 1403, 1405, 1448, 1496, 1498
\l__head_heading_decls_tl .....
314, 553, 677, 747, 807, 1046, 1064,
1079, 1310, 1449, 1564, 1618, 1682
\__head_heading_format:n 320, 559, 696
\l__head_heading_indent_dim ....
..... 655, 679, 680, 684,
687, 688, 692, 723, 751, 752, 756,
759, 760, 764, 785, 809, 810, 814,
817, 818, 822, 1552, 1567, 1569,
1574, 1578, 1580, 1585, 1606, 1623,
1625, 1630, 1637, 1639, 1644, 1671,
1685, 1687, 1692, 1696, 1698, 1703
\l__head_instance_keys_tl .....
..... 24, 60, 90, 91, 95
\l__head_label_tl .....
... 24, 25, 60, 102, 122, 127, 131, 141
\l__head_level_int .....
..... 282, 418, 420, 421, 425, 520,
612, 614, 615, 617, 839, 1286, 1382,
1398, 1400, 1401, 1424, 1474, 1490,
1492, 1493, 1561, 1615, 1657, 1706
\__head_mark_cmd:n .....
..... 298, 537, 898, 1297, 1436
\l__head_name_tl ..... 281, 332,
519, 573, 681, 683, 689, 691, 753,
755, 761, 763, 811, 813, 819, 821,
850, 902, 906, 913, 917, 1285, 1320,
1391, 1394, 1423, 1462, 1483, 1486,
1570, 1573, 1581, 1584, 1626, 1629,
1640, 1643, 1688, 1691, 1699, 1702
\l__head_nameref_tl .. 56, 86, 101, 113
\l__head_number_decls_tl .....
316, 555, 1312, 1451, 1593, 1652, 1714
\__head_number_format:n .....
.... 322, 561, 1316, 1394, 1455, 1486
\l__head_number_tag_tl .... 328, 567
\l__head_number_title_sep_tl ...
.. 1555, 1596, 1609, 1653, 1674, 1718
\l__head_number_tl ..... 47,
60, 437, 625, 672, 742, 803, 845, 851
\l__head_order_clist ... 656, 725, 786
\l__head_penalty_int .....
.. 304, 543, 832, 833, 861, 1303, 1442
\l__head_placement_end_code_tl .
... 32, 34, 41, 308, 356, 394, 398,
413, 462, 547, 587, 602, 605, 607, 649
\l__head_placement_start_code_tl
..... 32, 34,
35, 41, 48, 307, 351, 359, 363, 380,
391, 546, 582, 590, 593, 599, 855, 868
\l__head_placement_tl .....
..... 60, 289, 291, 293, 295,
339, 343, 360, 395, 528, 530, 532,
534, 591, 603, 1291, 1293, 1295,
1326, 1358, 1430, 1432, 1434, 1466
\l__head_pname_tl 284, 522, 1287, 1425
\l__head_prefix_decls_tl .....
315, 554, 1311, 1450, 1589, 1648, 1710
\__head_prefix_format:n ... 321, 560
\l__head_prefix_number_sep_tl ..
.. 1554, 1591, 1608, 1650, 1673, 1712
\l__head_prefix_tl ..... 47,
311, 436, 550, 624, 671, 741, 802, 846
\__head_prep_decls_and_auxi:nnNN
..... 1091, 1094
\__head_prep_decls_and_auxii:nNwNN
..... 1096, 1099
\__head_prep_decls_and_auxiii:nnNnNN
..... 1102, 1107
\__head_prep_decls_and_format_-
from_tl:nNN ..... 1045, 1088, 1088

```


<code>\partname</code>	933, 1196	<code>\skip_vertical:n</code>	1596
<code>\penalty</code>	37, 38, 506, 509	<code>\l_tmpa_skip</code>	482, 483, 485, 499
<code>\prevdepth</code>	38, 487, 489, 491	<code>\l_tmpb_skip</code>	38, 485, 494, 495
<code>\protect</code>	906, 917	<code>\c_zero_skip</code>	679, 687, 751, 759, 809, 817, 1567, 1578, 1623, 1637, 1685, 1696
<code>\providecommand</code>	67, 879	<code>\spacefactor</code>	27, 156, 159
<code>\ProvidesExplPackage</code>	3	<code>\specialsection</code>	54
Q		<code>\start-code</code>	13
quark commands:		str commands:	
<code>\q_no_value</code>	26, 74	<code>\c_hash_str</code>	1110, 1119, 1126
<code>\quark_if_no_value:nTF</code>	125, 139	<code>\str_case:nn</code>	360, 591, 1466
<code>\q_stop</code>	1096, 1099	<code>\str_case:nnTF</code>	395, 603, 1141, 1326, 1358
R		<code>\str_if_eq:nnTF</code>	339
<code>\raggedright</code>	950, 1198	<code>\str_if_eq_p:nn</code>	1110, 1111, 1112, 1113, 1114, 1117, 1124
<code>\refstepcounter</code>	20, 47	<code>\string</code>	1044
<code>\relax</code>	150, 158, 889, 1048, 1054, 1145, 1149	<code>\subparagraph</code>	17, 18, 65, 1253
<code>\renewcommand</code>	884	<code>\subparagraphmark</code>	1010
<code>\RenewCommandCopy</code>	882, 1138	<code>\subsection</code>	17, 18, 64, 1247
<code>\RenewDocumentCommand</code>	1139	<code>\subsectionmark</code>	974
<code>\RestoreLastSkip</code>	13, 15, 43, 62	<code>\subsubsection</code>	17, 18, 64, 1249
S		T	
<code>\SaveLastSkip</code>	13, 15, 43, 62	tag commands:	
scan commands:		<code>\tag_if_active:TF</code>	1176
<code>\scan_stop:</code>	506	<code>\tag_mc_end_push:</code>	706, 711
sec commands:		<code>\tag_struct_begin:n</code>	708, 713
<code>\sec_add_penalty_with_fil:n</code>	37, 48, 388, 466, 466, 596	<code>\tag_struct_end:</code>	707, 712
<code>\sec_add_penalty_with_possible_-</code>		tag internal commands:	
<code>fil:n</code>	34, 37, 48, 387, 466, 470, 474, 595, 860	<code>\l__tag_para_bool</code>	707, 712
<code>\secdef</code>	19	<code>\l__tag_para_flattened_bool</code>	717
<code>\secdef</code> ...	4, 5, 9, 19, 20, 50, 51, 57, 1139	<code>\tagpdfparaOff</code>	669, 739, 800
<code>\section</code> ..	5, 8, 9, 17, 18, 50, 54, 64, 1245	<code>\tagpdfparaOn</code>	718
<code>\sectionmark</code>	13, 62, 963	template commands:	
<code>\setbox</code>	638, 749, 1514	<code>\template_process_order_clist:nnn</code>	697, 766, 774, 825
<code>\SetCurrentTemplateKeys</code>	33, 60	template types:	
<code>\SetKnownTemplateKeys</code>	336, 574	<code>headformat</code>	168
<code>\SetTemplateKey</code>	33	<code>heading</code>	167
<code>\SetTemplateKeys</code>	664, 733, 794, 1324, 1463, 1559, 1613, 1678	templates:	
<code>\sfcode</code>	27, 152, 153, 154, 155	<code>headformat display</code>	245, 652
skip commands:		<code>headformat display-v1</code>	1528, 1549
<code>\skip_add:Nn</code>	494	<code>headformat hang</code>	257, 720
<code>\skip_horizontal:N</code>	643, 680, 684, 688, 692, 752, 756, 760, 764, 810, 814, 818, 822, 1519, 1569, 1574, 1580, 1585, 1625, 1630, 1639, 1644, 1687, 1692, 1698, 1703	<code>headformat hang-v1</code>	1535, 1604
<code>\skip_horizontal:n</code>	1591, 1650, 1653, 1712, 1718	<code>headformat runin</code>	268, 782
<code>\skip_set_eq:NN</code>	482, 485	<code>headformat runin-v1</code>	1542, 1669
<code>\skip_vertical:N</code>	443, 458, 1416	<code>heading display</code>	169, 279
		<code>heading display-v1</code>	1257
		<code>heading runin</code>	207, 517
		<code>heading runin-v1</code>	1420
		$\mathrm{T}_{\mathrm{E}}\mathrm{X}$ and $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X} 2_{\epsilon}$ commands:	
		<code>\@</code>	28
		<code>\@LRmoderr</code>	478

<code>\@M</code>	639,	<code>\protected@edef</code>	1392, 1484
676, 746, 806, 1515, 1563, 1617, 1681		<code>\z@</code>	383, 638, 1050, 1055, 1351, 1514
<code>\@addpunct</code>	28	tex commands:	
<code>\@afterheading</code>	35, 413, 1377	<code>\tex_lastskip:D</code>	482
<code>\@afterindentfalse</code> ..	301, 1052, 1300	<code>\texorpdfstring</code>	20, 49, 879, 919
<code>\@afterindenttrue</code> ...	300, 1049, 1299	<code>\the</code> .	147, 148, 149, 1061, 1062, 1077, 1078
<code>\@chapapp</code>	51, 948, 957	<code>\the<name></code>	13, 62
<code>\@chappap</code>	29	<code>\thechapter</code>	23, 949
<code>\@chapter</code>	4	<code>\theheading</code>	3
<code>\@clubpenalty</code>	645, 1521	<code>\theheading</code>	13, 40, 62–64,
<code>\@currentlabelname</code>	896	66, 332, 573, 851, 1280, 1320, 1462	
<code>\@firstoftwo</code>	10	<code>\thepart</code>	934
<code>\@hangfrom</code>	1659	<code>\thesection</code>	23, 63, 64
<code>\@kernel@refstepcounter</code>		<code>\thispagestyle</code>	
850, 1391, 1483			369, 382, 403, 1336, 1350, 1367
<code>\@latex@error</code>	1176	<code>\titleformat</code>	20
<code>\@latex@warning</code>	1151, 1166, 1176	<code>\titleformat</code>	20
<code>\@m</code>	159	<code>\titlespacing</code>	20
<code>\@minus</code>	995, 1007	<code>\titlespacing</code>	20
<code>\@nobreakfalse</code>	633, 1509	tl commands:	
<code>\@noskipsecfalse</code>	637, 1513	<code>\c_novalue_tl</code>	24, 87, 88
<code>\@noskipsectrue</code>	634, 1510	<code>\c_space_tl</code>	1116, 1118, 1123, 1125
<code>\@part</code>	4	<code>\tl_clear:N</code>	76, 77, 78, 90, 127,
<code>\@plus</code>	995, 1007	391, 599, 605, 607, 1355, 1388, 1480	
<code>\@secondoftwo</code>	10	<code>\tl_if_blank:nTF</code>	855
<code>\@secpenalty</code> ...	13, 14, 47, 62, 63, 833	<code>\tl_if_empty:nTF</code> ...	349, 354, 359,
<code>\@startsection</code>	4, 5, 9, 11, 19,	394, 580, 585, 590, 602, 1324, 1325,	
50, 53, 54, 56, 60, 61, 1041, 1136, 1138		1357, 1463, 1464, 1559, 1613, 1678	
<code>\@svsechd</code>	618, 640, 1494, 1516	<code>\tl_if_head_is_N_type:nTF</code>	1095
<code>\@tempboxa</code>	749, 772, 773	<code>\tl_new:N</code>	52, 53, 54,
<code>\@tempskipa</code> 1048, 1050, 1051, 1061, 1077		55, 56, 57, 58, 60, 61, 62, 63, 64, 65, 66	
<code>\@tempskipb</code> 1054, 1055, 1062, 1071, 1078		<code>\tl_put_right:Nn</code>	122, 138, 141
<code>\@tempswa</code>	34	<code>\tl_reverse:n</code>	1092, 1130
<code>\@tempswafalse</code>	374, 1341	<code>\tl_set:Nn</code>	128, 131, 132,
<code>\@tempswatrue</code>	372, 1339	289, 291, 293, 295, 333, 343, 363,	
<code>\@topnewpage</code>	33, 36, 48, 429	380, 398, 413, 528, 530, 532, 534,	
<code>\@topnum</code>	383, 1351	572, 593, 671, 672, 673, 741, 742,	
<code>\c@secnumdepth</code>	47, 333,	743, 802, 803, 804, 845, 846, 847,	
454, 572, 631, 839, 844, 1321, 1382,		851, 896, 1089, 1097, 1105, 1130,	
1387, 1414, 1461, 1474, 1479, 1507		1131, 1133, 1186, 1291, 1293, 1295,	
<code>\Hy@addcontentsline@addbookmark</code> 883		1321, 1329, 1347, 1361, 1377, 1430,	
<code>\Hy@bookmarksfalse</code>	891	1432, 1434, 1461, 1468, 1621, 1635	
<code>\Hy@bookmarkstrue</code>	887	<code>\tl_set_eq:NN</code> 81, 82, 83, 86, 87, 88,	
<code>\if@afterindent</code>	1063	332, 351, 356, 573, 582, 587, 1320, 1462	
<code>\if@mainmatter</code>	1228	token commands:	
<code>\if@nobreak</code>	857	<code>\token_if_macro:N</code>	1100
<code>\if@noskipsec</code>	69, 636, 856, 1512	<code>\token_to_str:N</code>	1123, 1125
<code>\if@openright</code>		<code>\twocolumn</code>	408, 1372
364, 381, 401, 511, 1331, 1349, 1365		<code>\typeout</code> 26, 526, 539, 540, 1428, 1438, 1439	
<code>\if@tempswa</code>	407, 1371		
<code>\if@twocolumn</code>	338, 370, 1337		
<code>\if@twoside</code>	400, 1364		
<code>\ifHy@bookmarks</code>	885		
<code>\kernel@startsection</code>	1136, 1138		

U

use commands:	642, 1398, 1399, 1490, 1491, 1493,
\use:N	906, 917 1518, 1561, 1601, 1615, 1656, 1706
\use:n	55, 93, V
426, 621, 1056, 1072, 1089, 1404, 1497	
\use_i:nn	439, 627, 1410, 1503 \vfil . 34, 37, 39, 376, 399, 509, 1343, 1363
\use_ii:nn	440, 628, 1411, 1504 \vfilneg 34, 37, 38, 509
\use_none:n	9, 10 \vskip 495, 498, 499
\UseInstance 25, 60, 94, 434, 622, 1405, 1498	\vspace . 16, 253, 431, 864, 872, 1018, 1206
\UseStructureName 203,	\vspace* 48
241, 421, 615, 708, 713, 1401, 1492	W
\UseTaggingSocket	\wd 772
. 418, 419, 425, 447, 612, 613, 617,	\WordSpaceAmount . . . 145, 1532, 1539, 1546