The titlesec, titleps and titletoc Packages*

Javier Bezos†

2019/07/16

Contents

1. Introduction

This package is essentially a replacement—partial or total—for the LaTeX macros related with sections—namely titles, headers and contents. The goal is to provide new features unavailable in current LaTeX; if you just want a more friendly interface than that of standard LaTeX but without changing the way LaTeX works you may consider using fancyhdr, by Piet van Oostrum, sectsty, by Rowland McDonnell, and tocloft, by Peter Wilson, which you can make pretty things with.[1]

Some of the new features provided are:

- Different classes and "shapes" of titles, with tools for very fancy formats. You can define different formats for left and right pages, or numbered and unnumbered titles, measure the width of the title, add a new section level, use graphics, and many more. The Appendix shows a good deal of examples, so jump forward right now!
- Headers and footers defined with no `\...mark` intermediates, and perhaps containing top, first *and* bot marks at the same time. Top marks correctly synchronized with titles, without incompatibilities with the float mechanism. Decorative elements easily added, including picture environments.

†For bug reports, comments and suggestions go to `http://www.tex-tipografia.com/contact.html`. English is not my strong point, so contact me when you find mistakes in the manual. Other packages by the same author: gloss (with José Luis Díaz), enumitem, accents, tensind, esindex, dotlessi, babeltools.

[1] Since the sectioning commands are rewritten, their behaviour could be somewhat different in some cases.

- Pretty free form contents, with the possibility of grouping entries of different levels in a paragraph or changing the format of entries in the middle of a document.

Titlesec works with the standard classes and with many others, including the AMS ones, and it runs smoothly with hyperref.[2] Unfortunately, it is not compatible with memoir, which provides its own tools with a limited subset of the features available in titlesec.

As usual, load the package in the standard way with `\usepackage`. Then, redefine the sectioning commands with the simple, predefined settings (see section "Quick Reference") or with the provided commands if you want more elaborate formats (see section "Advanced Interface.") In the latter case, you only need to redefine the commands you'll use. Both methods are available at the same time, but because `\part` is usually implemented in a non-standard way, it remains untouched by the simple settings and should be changed with the help of the "Advanced Interface."

2. Quick Reference

The easiest way to change the format is by means of a set of package options and a couple of commands. If you feel happy with the functionality provided by this set of tools, you need not go further in this manual. Just read this section and ignore the subsequent ones.

2.1. Format

There are three option groups controlling font, size and align. You need not set all of these groups, since a default is provided for each one; however, you must use at least an option from them if you want this "easy setup."

`rm sf tt md bf up it sl sc`

Select the corresponding family/series/shape. Default is `bf`.

`big medium small tiny`

Set the size of titles. Default is `big`, which gives the size of standard classes. With `tiny`, sections (except chapters) are typed in the text size. `medium` and `small` are intermediate layouts.

`raggedleft center raggedright`

Control the alignment.

2.2. Spacing

`compact`

This option is independent from those above and reduces the spacing above and below the titles.

2.3. Uppercase

`uppercase`

2.9 Uppercases titles. Depending on the class, it might not work in `\chapter` and `\part`.

2.4. Tools

`\titlelabel{`⟨*label-format*⟩`}`

Changes the label format in sections, subsections, etc. A `\thetitle` command is provided which is respectively `\thesection`, `\thesubsection`, etc. The default value in standard classes is

[2] However, be aware the AMS classes reimplement the original internal commands. These changes will be lost here. The compatibility with hyperref has been tested with dvips, dvipdfm and pdftex but it is an unsupported feature. Please, check your version of hyperref is compatible with titlesec.

```
\titlelabel{\thetitle\quad}
```

and you may add a dot after the counter simply with

```
\titlelabel{\thetitle.\quad}
```

That was done in this document.

`\titleformat*{`⟨*command*⟩`}{`⟨*format*⟩`}`

This command allows to change the ⟨*format*⟩ of a sectioning command, as for example:

```
\titleformat*{\section}{\itshape}
```

3. Advanced Interface

Two commands are provided to change the title format. The first one is used for the "internal" format, i. e., shape, font, label. . . , the second one defines the "external" format, i. e., spacing before and after, indentation, etc. This scheme is intended to easy definitions, since in most of cases you will want to modify either spacing or format.[3] That redefines existing sectioning commands, but does not create *new* ones. New sectioning levels can be added with `\titleclass`, as described below, and then their format can be set with the commands described here.

3.1. Format

A set of shapes is provided, which controls the basic distribution of elements in a title. The available shapes are:

hang is the default value, with a hanging label. (Like the standard `\section`.)

block typesets the whole title in a block (a paragraph) without additional formatting. Useful in centered titles [4] and special formatting (including graphic tools such as `picture`, `pspicture`, etc.)

display puts the label in a separate paragraph. (Like the standard `\chapter`.)

runin A run-in title, like the standard `\paragraph`.[5]

leftmargin puts the title at the left margin. Titles at the very end of a page will be moved to the next one and will not stick out in the bottom margin, which means large titles can lead to underfull pages.[6] In this case you may increase the stretchability of the page elements, use `\raggedbottom` or use the package option `nobottomtitles` described below. Since the mechanism used is independent from that of the margin pars, they can overlap. A deprecated synonymous is `margin`.

rightmargin is like `leftmargin` but at the right margin.

drop wraps the text around the title, provided the first paragraph is longer than the title (if not, they overlap). The comments in `leftmargin` also apply here.

wrap is quite similar to drop. The only difference is while the space reserved in drop for the title is fixed, in wrap is automatically readjusted to the longest line. The limitations explained below related to `calcwidth` also apply here.

frame Similar to display, but the title will be framed.

[3] Information is "extracted" from the class sectioning commands, except for chapter and part. Standard definitions with `\@startsection` are presumed—if sections have been defined without that macro, arbitrary values for the format an the spacing are provided, which you may change later. (Sadly, there is no way to catch the chapter or part formats, and one similar to that of standard classes will be used.)

[4] The label will be slightly displaced to the left if the title is two or more lines long and the `hang` shape is used, except with explicit `\\`.

[5] Well, not quite. The title is first boxed to avoid some unexpected results if, for example, there is a `\color` between the title and the text. Unfortunately, due to an optimization done by TEX discretionaries may be lost. I have found no solution, except using luatex, which works as one could expect. Anyway, if the title doesn't contain hyphen or dashes, this is not usually a real problem.

[6] However, floats following the title a couple of lines after will interfere with the page breaking used here and sometimes the title may stick out.

Note, however, some shapes do not make sense in chapters and parts.

`\titleformat{⟨command⟩}[⟨shape⟩]{⟨format⟩}{⟨label⟩}{⟨sep⟩}{⟨before-code⟩}[⟨after-code⟩]`

Here

- ⟨*command*⟩ is the sectioning command to be redefined, i. e., `\part`, `\chapter`, `\section`, `\subsection`, `\subsubsection`, `\paragraph` or `\subparagraph`.
- The paragraph shape is set by ⟨*shape*⟩, whose possible values are those described above.
- ⟨*format*⟩ is the format to be applied to the whole title—label and text. This part can contain vertical material (and horizontal with some shapes) which is typeset just after the space above the title.
- The label is defined in ⟨*label*⟩. You may leave it empty if there is no section label at that level, but this is not recommended because by doing so the number is not suppressed in the table of contents and running heads.
- ⟨*sep*⟩ is the horizontal separation between label and title body and must be a length (it must not be empty). This space is vertical in `display` shape; in `frame` it is the distance from text to frame. Both ⟨*label*⟩ and ⟨*sep*⟩ are ignored in starred versions of sectioning commands. If you are using `picture` and the like, set this parameter to 0 pt.
- ⟨*before-code*⟩ is code preceding the title body. The very last command can take an argument, which is the title text.[7] However, with the package option `explicit` the title must be given explicitly with `#1` (see below). Penalties in this argument may lead to unexpected results.
- ⟨*after-code*⟩ is code following the title body. The typeset material is in vertical mode with `hang`, `block` and `display`; in horizontal mode with `runin` and `leftmargin` (2.7 with the latter, at the beginning of the paragraph). Otherwise is ignored. Penalties in this argument may lead to unexpected results.

Penalties, marks and the like must be properly synchronized with page breaks. So, ⟨*before-code*⟩ and ⟨*after-code*⟩ are *not* the proper places for penalties. See `\sectionbreak` below.

`\chaptertitlename`

It defaults to `\chaptername` except in appendices where it is `\appendixname`. Use it instead of `\chaptername` when defining a chapter.

3.2. Spacing

`\titlespacing*{⟨command⟩}{⟨left⟩}{⟨before-sep⟩}{⟨after-sep⟩}[⟨right-sep⟩]`

The starred version kills the indentation of the paragraph following the title, except in `drop`, `wrap` and `runin` where this possibility does not make sense.

- ⟨*left*⟩ increases the left margin, except in the `...margin`, and `drop` shape, where this parameter sets the title width, in `wrap`, the maximum width, and in `runin`, the indentation just before the title. With negative value the title overhangs.[8]
- ⟨*before-sep*⟩ is the vertical space before the title.

[7] Remember font size can be changed safely between paragraphs only, and changes in the text should be made local with a group; otherwise the leading might be wrong—too large or too small.

[8] This parameter is not equal to `<indent>` of `\@startsection`, which doesn't work correctly. With a negative value in the latter and if `<indent>` is larger than the label width, the first line of the title will start in the outer margin, as expected, but the subsequent lines will not; worse, those lines will be shortened at the right margin.

- ⟨*after-sep*⟩ is the separation between title and text—vertical with `hang`, `block`, and `display`, and horizontal with `runin`, `drop`, `wrap` and `...margin`. By making the value negative, you may define an effective space of less than `\parskip`.[9]
- The `hang`, `block` and `display` shapes have the possibility of increasing the ⟨*right-sep*⟩ margin with this optional argument.

If you dislike typing the full skip values, including the `plus` and `minus` parameters, an abbreviation `*`*n* is provided. In the ⟨*before-sep*⟩ argument this is equivalent to *n* `ex` with some stretchability and a minute shrinkability. In the ⟨*after-sep*⟩ some stretchability (smaller) and no shrinkability.[10] Thus, you can write

```
\titlespacing{\section}{0pt}{*4}{*1.5}
```

The lengths `\beforetitleunit` and `\aftertitleunit` are used as units in the `*` settings and you can change them if you do not like the predefined values (or for slight changes in the makeup, for example).

Notes. `\titlespacing` does not work with either `\chapter` and `\part` unless you change its title format as well by means of `\titleformat`, the simple settings, or `\titleclass`. Arguments in `\titlespacing` must be dimensions; `\stretch` includes a command and hence raises an error.

3.3. Spacing related tools

These commands are provided as tools for `\titleformat` and `\titlespacing`.

`\filright` `\filcenter` `\filleft` `\fillast` `\filinner` `\filouter`

Variants of the `\ragged...` commands, with slight differences. In particular, the `\ragged...` commands kills the left and right spaces set by `\titlespacing`.[11] `\fillast` justifies the paragraph, except the last line which is centered.[12] These commands work in the `frame` label, too.

`\filinner` and `\filouter` are `\filleft` or `\filright` depending on the page. Because of the asynchronous TEX page breaking, these commands can be used in `\chapter` only. If you want a general tool to set different formats depending on the page, see "Extended settings" below.

`\wordsep`

The inter-word space for the current font.

`indentafter noindentafter` (Package options)

By-pass the settings for all of sectioning commands.[13]

`rigidchapters rubberchapters` (Package options)

With `rigidchapters` the space for chapter titles is always the same, and ⟨*after-sep*⟩ in `\titlespacing` does not mean the space from the bottom of the text title to the text body as described above, but from the *top* of the text title, i. e., ⟨*before-sep*⟩ + ⟨*after-sep*⟩ now is a fixed distance from the top of the page body to the main text. The default is `rubberchapters` where ⟨*after-sep*⟩ is the separation between title and text as usual. Actually, the name is misleading because it applies not only to the default chapter, but to any title of top class. (More on classes below.)

`bottomtitles nobottomtitles nobottomtitles*` (Package options)

If `nobottomtitles` is set, titles close to the bottom margin will be moved to the next page and the

[9] See Goossens, Mittelbach and Samarin: *The LaTeX Companion*, Reading, Addison Wesley, 1993, p. 25.
[10] They stand for *n* times `1ex plus .3ex minus .06ex` and `1ex plus .1ex`, respectively.
[11] Remember the package ragged2e provides some additional commands for alignment, too, like `\justifying`.
[12] Admittedly, a weird name, but it is short.
[13] Formerly `indentfirst` and `nonindentfirst`, now deprecated.

margin will be ragged. The minimal space required in the bottom margin not to move the title is set (approximately) by

`\renewcommand{\bottomtitlespace}{`⟨*length*⟩`}`

whose default value is `.2\textheight`. A simple ragged bottom on the page before is obtained with a value of 0 pt. `bottomtitles` is the default, which simply sets `\bottomtitlespace` to a negative value.

The `nobottomtitles*` option provides more accurate computations but titles of `margin`, `wrap` or `drop` shapes could be badly placed. Usually, you should use the starred version.

`aftersep largestsep` (Package options)

By default, when there are two consecutive titles the ⟨*after-sep*⟩ space from the first one is used between them. Sometimes this is not the desired behaviour, especially when the ⟨*before-sep*⟩ space is much larger than the ⟨*after-sep*⟩ one (otherwise the default seems preferable). With `largestsep` the largest of them is used. Default is `aftersep`.

`\\ \\*`
`pageatnewline` (Package option)

2.6 In version 2.6 and later, `\\` does not allow a page break and therefore is equivalent to `\\*`. Since I presume none wants a page break inside a title, this has been made the default. If for some extrange reason you want to allow page breaks inside the titles, use the package option `pageatnewline`, which is provided for backward compatibility.

`\nostruts`
`nostruts` (Package option)

2.11 The styles defined by titlesec insert some struts at certain places to make sure the vertical space is the same with relation with the baseline. This is not always the desired behavior, so the package options `nostruts` is provided. An alternative is the macro `\nostruts` when defining a section (note this macros is defined only within a title).

3.4. Rules

The package includes some tools for helping in adding rules and other stuff below or above the title. Since the margins in titles may be modified, these macros take into account the local settings to place rules properly. They also take into account the space used by marginal titles.

`\titleline[`⟨*align*⟩`]{`⟨*horizontal material*⟩`}`
`\titlerule[`⟨*height*⟩`]`
`\titlerule*[`⟨*width*⟩`]{`⟨*text*⟩`}`

The `\titleline` command allows inserting a line, which may contain text and other "horizontal" material. it is intended mainly for rules and leaders but in fact is also useful for other purposes. The line has a fixed width and hence must be filled, i.e., `\titleline{CHAPTER}` produces an underfull box. Here the optional ⟨*align*⟩ (`l`, `r` or `c`) helps, so that you simply type, say, `\titleline[c]{CHAPTER}`.[14]

Using `\titleline` in places where vertical material is not expected can lead to anomalous results. In other words, you can use it in the ⟨*format*⟩ (always) and ⟨*after-code*⟩ (`hang`, `display` and `block`) arguments; and in the `display` shape at the very beginning of the ⟨*before-code*⟩ and ⟨*label*⟩ argument as well. But try it out, because very likely it works in other places.

The `\titlerule` command, which is enclosed automatically in `\titleline` if necessary, can be used to build rules and fillers. The unstarred version draws rules of height .4 pt, or ⟨*height*⟩ if present. For example:

`\titlerule[.8pt]%`

[14] The default is the `s` parameter of the `\makebox` command.

```
\vspace{1pt}%
\titlerule
```

draws two rules of different heights with a separation of 1 pt.

The starred version makes leaders with the ⟨*text*⟩ repeated in boxes of its natural width. The width of the boxes can be changed to ⟨*width*⟩, but the first box remains with its natural width so that the ⟨*text*⟩ is aligned to the left and right edges of the space to be filled.

For instance, with

```
\titleformat{\section}[leftmargin]
  {\titlerule*[1pc]{.}%
   \vspace{1ex}%
   \bfseries}
  {... definition follows
```

leaders spanning over both main text and title precede the section.

`calcwidth` (Package option)

The `wrap` shape has the capability of measuring the lines in the title to format the paragraph. This capability may be extended to other three shapes—namely `display`, `block` and `hang`—with this package option. The length of the longest line is returned in `\titlewidth`.[15]

As far as TEX is concerned, any box is considered typeset material. If the box has been enlarged with blank space, or if conversely a box with text has been smashed, the value of `\titlewidth` may be wrong (as far as humans is concerned). The `hang` shape, for instance, uses internally such a kind of boxes, but in this case this behaviour is desired when the title is flushed right; otherwise the `block` shape produces better results. In other words, using boxes whose natural width has been overridden may be wrong.[16] Further, some commands may confuse TEX and stop parsing the title. But if you stick to text, `\\` and `\\[...]` (and it is very unlikely you might want something else), there will be no problems.

Another important point is the ⟨*before-code*⟩, ⟨*label*⟩, ⟨*sep*⟩, and ⟨*title*⟩ parameters (but not ⟨*after-code*⟩) are evaluated twice at local scope; if you increase a counter *globally*, you are increasing it twice. In most of cases, placing the conflicting assignment in the ⟨*after-code*⟩ parameter will be ok, but alternativey you can use the following macro.

`\iftitlemeasuring{`⟨*true*⟩`}{`⟨*false*⟩`}`

2.9 When the title is being measured (first pass), the ⟨*true*⟩ branch is used, and when the title is actually typeset (second pass) the ⟨*false*⟩ branch is used.

`\titleline*[`⟨*align*⟩`]{`⟨*horizontal material*⟩`}`

A variant of `\titleline` to be used only with `calcwidth`. The text will be enclosed first in a box of width `\titlewidth`; this box will be in turn enclosed in the main box with the specified alignment. There is no equivalent `\titlerule` and therefore you must enclose it explicitly in a `\titleline*` if you want the `\titlewidth` to be taken into account:

```
\titleline*[c]{\titlerule[.8pc]}
```

3.5. Page styles

2.8 You can assign a page style to levels of class `top` and `page`, as well as the default chapter with the following command:[17]

`\assignpagestyle{`⟨*command*⟩`}{`⟨*pagestyle*⟩`}`

For example, to suppress the page number in chapters write:

[15] There are two further parameters, `\titlewidthfirst` and `\titlewidthlast`, which return the length of the first and last lines. There are not specific tools for using them, but you can assign their values to `\titlewidth` and then use `\titleline*`.

[16] Which include justified lines, whose interword spacing has been enlarged.

[17] Named in the short-lived version 2.7 as `\titlepagestyle`.

```
\assignpagestyle{\chapter}{empty}
```

3.6. Breaks

`\sectionbreak`	`\subsectionbreak`	`\subsubsectionbreak`
`\paragraphbreak`	`\subparagraphbreak`	`\⟨section⟩break`

By defining these command with `\newcommand` different page breaks could be applied to different levels. In those undefined, a penalty with the internal value provided by the class is used (typically −300). For instance,

```
\newcommand{\sectionbreak}{\clearpage}
```

makes sections begin a new page. In some layouts, the space above the title is preserved even if the section begins a new page; that's accomplished with:

```
\newcommand{\sectionbreak}{%
  \addpenalty{-300}%
  \vspace*{0pt}}
```

2.6 `\⟨section⟩break` is available in the `top` class, too. Suitable values would be `\cleardoublepage` (the default if `openright`) and `\clearpage` (the default if `openany`). Thus, you can override `openright` by defining `\chapterbreak` as `\clearpage`, provided its class has been changed to `top` (in this example, parts will continue with the `openright` setting).

Note these macros apply the penalties at the right place. In other words, penalties in ⟨*before-code*⟩ and ⟨*after-code*⟩ can lead to unexpected (and even weird) results.

`\chaptertolists`

2.6 If defined, the usual white space written to lists (ie, List of Figures and List of Tables) is replaced by the code in this command. If you do not want the white space when a chapter begins, define it to empty, i.e.,

```
\newcommand{\chaptertolists}{}
```

This command is not a general tool to control spacing in lists, and is available only in titles of top class, so it will not work with the default chapters except if you change their class (on the other hand, `\...tolists` can be used in any title whose class is top).

3.7. Other Package Options

`explicit` (Package option)

2.7 With it, the title is not implicit after ⟨*before-code*⟩ but must be given explicitly with `#1` as in, for example:

```
\titleformat{\section}
 {..}
 {\thesection}{..}{#1.}
```

(Compare it with the example in section 4.4.)

`newparttoc oldparttoc` (Package options)

Standard parts write the toc entry number in a non standard way. You may change that with `newparttoc` so that titletoc or a similar package can manipulate the entry. (That works only if `\part` has been redefined.)

`clearempty` (Package options)

Modifies the behaviour of `\cleardoublepage` so that the `empty` page style will be used in empty pages.

`toctitles` (Package option)

2.6 Changes the behaviour of the optional argument in sectioning titles so that it sets only the running heads and not the TOC entries, which will be based on the full title.

`newlinetospace` (Package option)

2.6 Replaces every occurrence of `\\` or `\\*` in titles by a space in running heads and TOC entries. This way, you do not have to repeat the title just to remove a formatting command.

`notocpart*` (Package option)

2.10.1 Long ago (by the year 2000) I decided for some reason `\part*` would behave like the AMS classes and therefore there should be a contents entry for it. This is somewhat odd, indeed, but the very fact is nobody has complained until now! On the other hand, restoring the behaviour one could expect after 15 years doesn't seem a good idea. A new page/part style in on the way, but for the moment this option restores the standard behaviour.

3.8. Extended Settings

The first argument of both `\titleformat` and `\titlespacing` has an extended syntax which allows to set different formats depending on the context.[18] This argument can be a list of key/value pairs in the form:

⟨*key*⟩=⟨*value*⟩, ⟨*key*⟩=⟨*value*⟩, ⟨*key*⟩, ⟨*key*⟩, ...

Currently, only pages and unnumbered versions are taken care of, besides the sectioning command name. Thus, the available keys are:

- `name`. Allowed values are `\chapter`, `\section`, etc.
- `page`. Allowed values are `odd` or `even`.
- `numberless`. A valueless key. it is not necessary unless you want to set different numbered (without this key) and unnumbered (with `numberless`) variants.

The basic form described above with the name of a sectioning command, say

```
\titleformat{\section} ...
```

is in fact an abbreviation for

```
\titleformat{name=\section} ...
```

Let's suppose we'd like a layout with titles in the outer margin. We might set something like

```
\titleformat{name=\section,page=even}[leftmargin]
  {\filleft\scshape}{\thesection}{.5em}{}

\titleformat{name=\section,page=odd}[rightmargin]
  {\filright\scshape}{\thesection}{.5em}{}
```

[18] The keyval package is required for making use of it.

Since the page information is written to the `aux` file, at least two runs are necessary to get the desired result.

The "number" version is usually fine when generating unnumbered variants since removing the label is the only change required in most cases, but if you need some special formatting, there is the `numberless` key which defines an alternative version for sections without numbers (namely those with level below `secnumdepth`, in the front and back matters and, of course, the starred version). For instance

```
\titleformat{name=\section}{...% The normal definition follows
\titleformat{name=\section,numberless}{...% The unnumbered
% definition follows
```

Neither ⟨*label*⟩ nor ⟨*sep*⟩ are ignored in `numberless` variants.

These keys are available to both `\titleformat` and `\titlespacing`. By using `page` in one (or both) of them, odd and even pages will be formatted differently. Actually, "even" and "odd" are well established LaTeX terms, but misleading. In one side printing the "odd" pages refer to "even" pages as well (cf. `\oddsidemargin`.)

If you intend to create different odd/even *and* different numbered/unnumbered versions, it is recommended defining the four variants.

If you remove the page specifier from a sectioning command you must remove the `.aux` file.

3.9. Creating new levels and changing the class

While the shapes and the like modify the behaviour of titles related to the surrounding text, title classes allow to change the generic behaviour of them. With the help of classes you may insert, say, a new `subchapter` level between `chapter` and `section`, or creating a scheme of your own. *Making a consistent scheme and defining all of related stuff like counters, macros, format, spacing and, if there is a TOC, TOC format is left to the responsibility of the user.* There are three classes: `page` is like the book `\part`, in a single page, `top` is like `\chapter`, which begins a page and places the title at the top, and `straight` is intended for titles in the middle of text.[19]

`\titleclass{`⟨*name*⟩`}{`⟨*class*⟩`}`
`\titleclass{`⟨*name*⟩`}{`⟨*class*⟩`}[`⟨*super-level-cmd*⟩`]`

If you do not use the optional argument, you just change the ⟨*class*⟩ of ⟨*name*⟩. For example:

```
\titleclass{\part}{straight}
```

makes `part` of `straight` class.

When the second form is used, the level number is the following of ⟨*super-level-cmd*⟩. For example:

```
\titleclass{\subchapter}{straight}[\chapter]
\newcounter{subchapter}
\renewcommand{\thesubchapter}{\Alph{subchapter}}
```

creates a level under chapter (some additional code is shown as well, but you must add to it the corresponding `\titleformat` and `\titlespacing` settings).[20] If the chapter level is 0, then the subchapter one is 1; the levels below are increased by one (section is 2, subsection is 3, and so on).

There are two sectioning commands which perform some extra actions depending of its name and ignoring the class:

- `\chapter` logs the string defined in `\chaptertitlename` and the matter is taken into account.
- `\part` does not encapsulates the label in the toc entry, except if you use the `newparttoc` option.

`loadonly` (Package option)

Let us suppose you want to create your sectioning commands from scratch. This package option ignores any previous definitions, if any, and hence removes the possibility of using the options described in "Quick Reference." Then you use the titlesec tools, and define the corresponding counters and labels.

[19] There is an further class named `part` to emulate the article `\part`, but you should not use it at all. Use the `straight` class instead. Remember some features rely in these classes and titlesec does not change by default the definition of `\part` and `\chapter`.

[20] Regarding counters, the remreset package can be useful.

`\titleclass{⟨name⟩}[⟨start-level-num⟩]{⟨class⟩}`

Here, the ⟨*name*⟩ title is considered the top level, with number ⟨*start-level-num*⟩ (typically 0 or −1). It should be used only when creating sectioning commands from scratch with the help of `loadonly`, and there must be exactly one (no more, no less) declaration of this kind. After it, the rest of levels are added as explained above.

4. Additional Notes

This part describes briefly some LaTeX commands, useful when defining sectioning titles.

4.1. Fixed Width Labels

The `\makebox` command allows to use fixed width label, which makes the left margin of the actual title (not the label) to lie in the same place. For instance (only the relevant code is provided):

```
\titleformat{\section}
  {..}
  {\makebox[2em]{\thesection}}{..}{..}
```

See your LaTeX manual for further reference on boxing commands.

4.2. Starred Versions

Using sectioning commands in the starred version is strongly discouraged. Instead, you can use a set of markup oriented commands which are easy to define and modify, if necessary. Thus, you can test different layouts before choosing amongst them.

Firstly remember if you say

```
\setcounter{secnumdepth}{0}
```

sections will be not numbered but they will be included in both toc and headers.

Now, let's suppose you want to include some sections with a special content; for example, a section (or more) with exercises. We will use an environment named `exercises` whose usage is:

```
\section{A section}
Text of a normal section.

\begin{exercises}
\section{Exercises A}
Some exercises

\section{Exercises B}
Some exercises
\end{exercises}
```

The following definition suppresses numbers but neither toc lines nor headers.

```
\newenvironment{exercises}
  {\setcounter{secnumdepth}{0}}
  {\setcounter{secnumdepth}{2}}
```

The following one adds a toc line but headers will remain untouched:

```
\newenvironment{exercises}
 {\setcounter{secnumdepth}{0}%
  \renewcommand\sectionmark[1]{}}
 {\setcounter{secnumdepth}{2}}
```

The following one updates the headers but there will be no toc line:

```
\newenvironment{exercises}
 {\setcounter{secnumdepth}{0}%
  \addtocontents{toc}{\protect\setcounter{tocdepth}{0}\ignorespaces}}
 {\setcounter{secnumdepth}{2}%
  \addtocontents{toc}{\protect\setcounter{tocdepth}{2}\ignorespaces}}
```

(I find the latter a bit odd in this particular example; the first and second options are more sensible. The `\ignorespaces` is not very important, and you need not it unless there is unwanted space in the toc.)

That works with standard classes, but if you are using fancyhdr or titlesec to define headers you need further refinement to kill the section number. In titlesec that's accomplished with `\ifthesection` (see below).

As you can see, there are no `\addcontentsline`, no `\markboth`, no `\section*`, just logical structure. Of course you may change it as you wish; for example if you decide these sections should be typeset in small typeface, include `\small`, and if you realize you do not like that, remove it.

While the standard LaTeX commands are easier and more direct for simple cases, I think the proposed method above is far preferable in large documents.

4.3. Variants

Let's suppose we want to mark some sections as "advanced topics" with an asterisk after the label. The following code does the job:

```
\newcommand{\secmark}{}
\newenvironment{advanced}
  {\renewcommand{\secmark}{*}}
  {}
\titleformat{\section}
  {..}
  {\thesection\secmark\quad}{..}{..}
```

To mark the sections write

```
\begin{advanced}
\section{...}
...
\end{advanced}
```

That marks sections but not subsections. If you like being redundant and marking the subsection level as well, you must define it accordingly.

4.4. Putting a Dot after the Section Title

Today this styling is not used, but formerly it was fairly common. The basic technique was described above, but here is a reminder:

```
\newcommand{\periodafter}[1]{#1.}
\titleformat{\section}
 {..}
 {\thesection}{..}{..\periodafter}
```

If you had to combine this dot with some command (perhaps an underlining), you can say:

```
\newcommand{\periodafter}[2]{#1{#2.}}
\titleformat{\section}
 {..}
 {\thesection}{..}{..\periodafter{\ul}} % \ul from soul package
```

However, you might prefer the package option `explicit`.

5. titleps and Page Styles

The titleps package provides tools for one-stage setting of page styles (headlines and footlines). A higher-level interface is used, where the mark mechanism is hidden and there is no need to deal with `\leftmark`s and `\rightmark`s – just use a command or variable registered as a "mark" as the expected value will be returned, i.e., those when the mark was emitted, either by a sectioning command or explicitly with `\chaptermark`, `\sectionmark`, etc. A simple example, whose meaning should be obvious, is:

```
\newpagestyle{main}{
  \sethead[\thepage][\chaptertitle][(\thesection] % even
          {\thesection)}{\sectiontitle}{\thepage}} % odd
\pagestyle{main}
```

Other features are:

- Working top marks, compatible with floats (unlike the standard `\topmark`, which does not work correctly in LaTeX).
- Access to top, first and bot marks in a single headline/footline (e.g., the first and last section numbers).
- Marks for more than 2 sectioning levels.
- Simple (and not so simple) headrules and footrules.
- Headlines and footlines for pages with floats.
- Headlines and footlines for specific floats (a sort of `\thispagestyle` for floats).
- Multiple sets of marks (named here *markset*s and *extra marks*).

It can be used without titlesec, but you will get most of it when used together. To load it as a separate package, use the customary `\usepackage{titleps}`, but with titlesec you have to load it with:

```
\usepackage[pagestyles]{titlesec}
```

Please, read `titleps.pdf` (or typeset `titleps.tex`) for further information.

6. Contents: The titletoc package

This package is a companion to the titlesec package and it handles toc entries. However, it is an independent package and you can use it alone. The philosophy is similar to that of titlesec—instead of hooking the commands as defined by standard LaTeX and classes, there are new commands which you can format the toc entries with in a generic way. This means you have to learn just two new basic command and a couple of tools, no more, and you have access to new features. Paragraph format and fonts are set with commands like `\\`, `\makebox`, `\large`, `\itshape`, and so on, and entries are not shaped in any fashion because they are pretty free form.

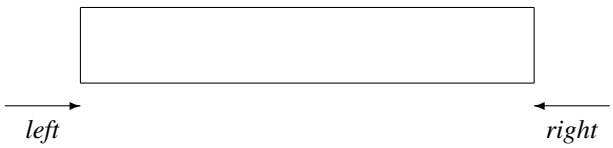
The behaviour of entries defined with titletoc are different at some points to those created with the standard commands. In particular:

- Pages are never broken between entries if the first one is of an higher level than the second one as, for instance, between a section and a subsection. If both of them are of the same level, the break is allowed, and if the first is lower than the second, it is considered a good place for a page break.
- The symbols in the leaders are not centered but flushed right. That is usually more convenient.

I would like to note no attempt to handle tocs can be complete because the standard LaTeX commands write directly some formatting commands which cannot be removed. This is particularly important in lists of figures and tables, and in the `\part` command.[21]

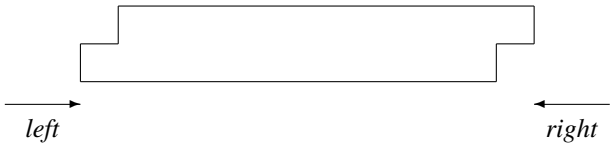
6.1. A ten-minute guide to titletoc

Toc entries are treated as rectangular areas where the text and probably a filler will be written. Let's draw such an area (of course, the lines themselves are not printed):

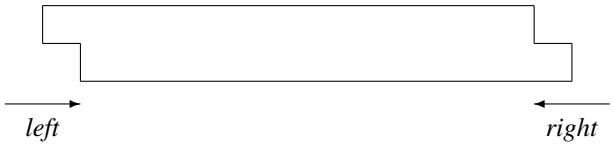


The space between the left page margin and the left edge of the area will be named ⟨*left*⟩; similarly we have ⟨*right*⟩. You are allowed to modify the beginning of the first line and the ending of the last line. For example by "taking up" both places with `\hspace*{2pc}` the area becomes:

[21] But some of these issues are fixed by titlesec.



And by "clearing" space in both places with `\hspace*{-2pc}` the area becomes:



If you have seen tocs, the latter should be familiar to you– the label at the very beginning, the page at the very end:

```
3.2  This is an example showing that toc
     entries fits in that scheme . . . .   4
```

All you need is to put these elements in the right way. If you have reserved the space with `\hspace*{-2pc}`, simply put a box 2 pc width containing the section label or page so that this space will be retrieved; this layout is used so often that two commands are provided which does that for you:

- `\contentslabel{`⟨*length*⟩`}` creates the space at the beginning and prints the section number.
- `\contentspage` creates a space at the end of length ⟨*right*⟩ and prints the page number aligned at the right.

Now, we are about to show the three basic commands:

`\dottedcontents{`⟨*section*⟩`}[`⟨*left*⟩`]{`⟨*above-code*⟩`}`
`{`⟨*label width*⟩`}{`⟨*leader width*⟩`}`

Here:

- ⟨*section*⟩ is the section name without backslash: `part`, `chapter`, `section`, etc. `figure` and `table` are allowed, too. (The backlash is omitted because we are dealing with the concept and not the `\part`, `\section`, etc. macros themselves. Furthermore, `figure` and `table` are environments.)
- ⟨*above-code*⟩ is code for the global formatting of the entry. Vertical material is allowed. At this point the value of `\thecontentslabel` (see below) is known which enables you to take decisions depending on its value (with the help of the ifthen package). You may use the titlesec `\filleft`, `\filright`, `\filcenter` and `\fillast` commands.
- ⟨*left*⟩ even if bracketed is currently mandatory and it sets the left margin from the left page margin.
- ⟨*label width*⟩ is the width of the space created for the label, as described above.
- ⟨*leader width*⟩ is the width of the box containing the char to be used as filler, as described below.

The definitions for section and subsection entries in the book class are roughly equivalent to:

```
\contentsmargin{2.55em}
\dottedcontents{section}[3.8em]{}{2.3em}{1pc}
\dottedcontents{subsection}[6.1em]{}{3.2em}{1pc}
```

`\titlecontents{`⟨*section*⟩`}[`⟨*left*⟩`]{`⟨*above-code*⟩`}`
`{`⟨*numbered-entry-format*⟩`}{`⟨*numberless-entry-format*⟩`}`
`{`⟨*filler-page-format*⟩`}[`⟨*below-code*⟩`]`

Here ⟨*section*⟩, ⟨*left*⟩ and ⟨*above-code*⟩ like above, and

- ⟨*numbered-entry-format*⟩ is in horizontal mode and it will be used just before the entry title. As in `\titleformat`, the last command can take an argument with the title.
- ⟨*numberless-entry-format*⟩ is like the above if there is, well, no label.
- ⟨*filler-page-format*⟩ is self explanatory. Fillers are created with the `\titlerule` command which is shared by that package and titlesec. However, when used in this context its behaviour changes a little to fit the needs of toc leaders.[22] You might prefer a `\hspace` instead.
- And finally ⟨*below-code*⟩ is code following the entry for, say, vertical space.

When defining entries, use `\addvspace` if you want to add vertical space, and `\\*` instead of `\\` for line breaks.

This command can be used in the middle of a document to change the format of toc/lot/lof entries at any point. The new format is written to the toc file and hence two runs are necessary to see the changes.

`\contentsmargin{`⟨*right*⟩`}`

The value set is used in all of sections. If you are wondering why, the answer is quite simple: in most of cases the ⟨*right*⟩ margin will be constant. However, you are allowed to change it locally in the ⟨*before-code*⟩ arguments. Note as well that the default space in standard classes does not leave room to display boldfaced page number above 100 and therefore you might want to set a larger margin with this command.

The book class formats section entries similarly (but not equally) to:

```
\titlecontents{section}
              [3.8em] % ie, 1.5em (chapter) + 2.3em
              {}
              {\contentslabel{2.3em}}
              {\hspace*{-2.3em}}
              {\titlerule*[1pc]{.}\contentspage}
```

Compare this definition with that given above and you will understand how `\dottedcontents` is defined.

Although standard classes use font dependent units (mainly em), it is recommended using absolute ones (pc, pt, etc.) to ensure they entries are aligned correctly.

6.2. And more

Strict typographical rules state full text lines shouldn't surpass the last dot of the leaders; ideally they should be aligned. Surprisingly enough, TEX lacks of a tool for doing that automatically—when you fill a box with leading dots, they can be centered in the box with the `\cleaders` primitive , "justified" with `\xleaders` or aligned with the outermost enclosing box with `\leaders`, but there is no way to align them with the "current" margin.

So, the only way to get a fine layout is by hand. To do , you can use the an optional argument in the `\contentsmargin` command whose syntax in full is the following:

`\contentsmargin[`⟨*correction*⟩`]{`⟨*right*⟩`}`

The ⟨*correction*⟩ length is added to the ⟨*right*⟩ one in all of lines except the last one, where the leaders are placed. For instance, if the text lines are 6 pt longer than the last dot, you should rewrite the `\contentsmargin` command to add a ⟨*correction*⟩ of 6 pt.[23] Unlike the standard LATEX tools, the `\titlerule*` command has been designed so that the ⟨*correction*⟩ will have the minimum value possible.

`\thecontentslabel` `\thecontentspage`

Contains the text with the label and the page with no additional formatting, except written by the class.

[22] For TEXnicians, the default `\xleaders` becomes `\leaders`.

[23] Usefully, many dvi previewers allow to get the coordinates of the pointed location.

`\contentslabel[`⟨*format*⟩`]{`⟨*space*⟩`}`
`\contentspage[`⟨*format*⟩`]`

As described above, but with different ⟨*format*⟩s. The defaults are just `\thecontentslabel` and `\thecontentspage`, respectively.

`\contentspush{`⟨*text*⟩`}`

Prints the ⟨*text*⟩ and increases ⟨*left*⟩ by the width of ⟨*text*⟩. It is similar to the hang shape of titlesec.

`\titlecontents*{`⟨*section*⟩`}[`⟨*left*⟩`]{`⟨*above-code*⟩`}`
`{`⟨*numbered-entry-format*⟩`}{`⟨*numberless-entry-format*⟩`}`
`{`⟨*filler-page-format*⟩`}[`⟨*separator*⟩`]`
or ...`{`⟨*filler-page-format*⟩`}[`⟨*separator*⟩`][`⟨*end*⟩`]`
or ...`{`⟨*filler-page-format*⟩`}[`⟨*begin*⟩`][`⟨*separator*⟩`][`⟨*end*⟩`]`

This starred version groups the entries in a single paragraph. The ⟨*separator*⟩ argument is the separator between entries, and there is a further optional argument with an ending punctuation. For example, this document sets:

```
\titlecontents*{subsection}[1.5em]
  {\small}
  {\thecontentslabel. }
  {}
  {, \thecontentspage}
  [.---][.]
```

whose result is showed in the contents at the very beginning of this document. Note the paragraph format must be written in the ⟨*above-code*⟩ argument.

Let us explain how the optional arguments works. First note the number of them determines their meaning—since there should be a separator between entries this one is always present; on the other hand, ⟨*begin*⟩ is rarely used and hence it has the lowest "preference." The simplest case is when the titles are of the same level; in this case the ⟨*sepatator*⟩ and the ⟨*end*⟩ parameters (which default to empty) are inserted between consecutive entries and at the end of the block, respectively. ⟨*before-code*⟩ is executed just once at the very beginning of the block and its declarations are local to the whole set of entries.

Now suppose we want to group entries of two levels; in this case a nesting principle applies. To fix ideas, we will use section and subsection. When a subsection entry begins after a section one, ⟨*before-code*⟩ is executed and ⟨*begin*⟩ of subsection is inserted, which should contain text format only. Then subsections are added inserting separators as explained above. When a section arrives, the ending punctuation of subsection and the separator of section is added (except if the block is finished by a subsection, where the ending of section is added instead). We said "after a section" because a subsection never begins a block.[24] The subsection entries are nested inside the section ones, and declarations are again local.

An example will illustrate that.

```
\titlecontents*{section}[0pt]
  {\small\itshape}{}{}
  {}[ \textbullet\ ][.]

\titlecontents*{subsection}[0pt]
  {\upshape}{}{}
  {, \thecontentspage}[ ()[. ][)]
```

produces something similar to:

The first section • *The second one* • *The third one* (A subsection in it, 1. Another, 2) • *A fourth section* (A subsection in it, 1. Another, 2).

[24] In rare cases that could be necessary, yet.

`\contentsuse{`⟨*name*⟩`}{`⟨*ext*⟩`}`

Makes titletoc aware of the existence of a contents file with ⟨*ext*⟩ extension. Mainly, it makes sure the command `\contentsfinish` is added at the end of the corresponding contents (and which must be added at the end of tocs made by hand). The package performs

```
\contentsuse{figure}{lof}
\contentsuse{table}{lot}
```

`leftlabels` `rightlabels` (Package options)

These package options set how the labels are aligned in `\contentslabel`. Default is `rightlabels`. With `leftlabels` the default ⟨*format*⟩ for `\contentslabel` becomes `\thecontenstlabel\enspace`.

`dotinlabels` (Package option)

With this package option, a dot is added after the label in `\contentslabel`.

6.3. Partial TOC's

`\startcontents[`⟨*name*⟩`]`

At the point where this command is used, a partial toc begins (note the document doesn't require a `\tableofcontents` for partial tocs to work). The ⟨*name*⟩ argument allows different sets of tocs and it defaults to `default`. These sets may be intermingled, but usually will be nested. For example, you may want two kinds of partial tocs: by part and by chapter (besides the full toc, of course). When a part begins, write `\startcontents[parts]`, and when a chapter `\startcontents[chapters]`. This way a new toc is started at each part and chapter.[25]

`\stopcontents[`⟨*name*⟩`]`
`\resumecontents[`⟨*name*⟩`]`

Stops the partial toc of ⟨*name*⟩ kind, which may be resumed. Since partial contents are stopped by `\startcontents` if necessary, those macros will not be used very often.

`\printcontents[`⟨*name*⟩`]{`⟨*prefix*⟩`}{`⟨*start-level*⟩`}[`⟨*toc-depth*⟩`]{`⟨*toc-code*⟩`}`

Print the current partial toc of ⟨*name*⟩ kind. The format of the main toc entries are used, except if there is a ⟨*prefix*⟩. In such a case, the format of ⟨*prefix*⟩⟨*level*⟩ is used, provided it is defined. For example, if prefix is `l` and the format of `lsection` is defined, then this definition will be used; otherwise, the format is that of `section`. The ⟨*start-level*⟩ parameter sets the top level of the tocs—for a part toc it would be `0` (chapter), for a chapter toc `1` (section), and so on. The ⟨*toc-code*⟩ is local code for the current toc; it may be used to change the `\contentsmargin`, for instance. **New 2.11** Finally, ⟨*toc-depth*⟩ sets the `tocdepth` locally (in former versions it was suggested setting this value with `\setcounter` in the last argument, but that was wrong, because this command set counters globally).

A simple usage might look like (provided you are using titlesec as well):

```
\titleformat{\chapter}[display]
  {...}{...}{...}  % Your definitions come here
  [\vspace*{4pc}%
   \startcontents
   \printcontents{l}{1}[2]{}]

\titlecontents*{lsection}[0pt]
  {\small\itshape}{}{}
  {}[ \textbullet\ ][.]
```

The included entries are those in levels 1 to 2 inclusive (i.e., 1 and 2).

[25] *All* partial tocs are stored in a single file with extension `.ptc`.

6.4. Partial lists 2.6

You may want to create partial LOFs and LOTs. The systax is similar to that of partial TOCs and what was said for them can be applied here. The commands are:

```
\startlist[⟨name⟩]{⟨list⟩}
\stoplist[⟨name⟩]{⟨list⟩}
\resumelist[⟨name⟩]{⟨list⟩}
\printlist[⟨name⟩]{⟨list⟩}{⟨prefix⟩}[⟨toc-depth⟩]{⟨toc-code⟩}
```

Here ⟨*list*⟩ is either `lof` or `lot`. Note as well `\printlist` does not have the ⟨*start-level*⟩ argument, because figures and tables have not levels. Currently, only those two float lists are supported, but in a future release support for more kinds of float lists will be added. Unfortunately, many classes write some formatting commands to these lists (more precisely, `\addvspace`s in chapters); I'm still not sure how to remove these commands without removing as well others which can be wanted, but for the time being a quick trick to remove these spaces is to redefine `\addvspace` in the ⟨*toc-code*⟩ with `\renewcommand\addvspace[1]{}`.

6.5. Examples

```
\titlecontents{chapter}
              [0pt]
              {\addvspace{1pc}%
               \itshape}%
              {\contentsmargin{0pt}%
               \bfseries
               \makebox[0pt][r]{\huge\thecontentslabel\enspace}%
               \large}
              {\contentsmargin{0pt}%
               \large}
              {\quad\thepage}
              [\addvspace{.5pc}]
```

The chapter number is out at the edge of the page margin, in a font larger than the font of the title. If the chapter lacks of number (because, say, it is the preface or the bibliography) it is not boldfaced. The page number follows the title without fillers, but after an em-space.

```
\titlecontents{chapter}
              [3pc]
              {\addvspace{1.5pc}%
               \filcenter}
              {CHAPTER \thecontentslabel\\*[.2pc]%
               \huge}
              {\huge}
              {}  % That is, without page number
              [\addvspace{.5pc}]
```

The chapter title is centered with the chapter label on top of it. There is no page number.

6.6. Inserting a figure in the contents

The `\addtocontents` command is still available and you may use it to perform special operation, like inserting a figure just before or after of an entry. Sadly, fragile arguments are not allowed and writing complex code could be a mess. The trick is to define a command to perform the required operations which in turn is written with `\protect`.

Let's suppose we want to insert a figure before an entry.

```
\newcommand{\figureintoc}[1]{
  \begin{figure}
    \includegraphics{#1}%
  \end{figure}}
```

makes the dirty work.

In the place where a figure is inserted write:

```
\addtocontents{\protect\figureintoc{myfig}}
```

6.7. Marking entries with asterisks

Let's now resume a problem explained in relation with titlesec: marking sections with asterisks to denote an "advanced topic" unless the star should be printed in the toc as well. Here is the code:

```
\newcommand{\secmark}{}
\newcommand{\marktotoc}[1]{\renewcommand{\secmark}{#1}}
\newenvironment{advanced}
  {\renewcommand{\secmark}{*}%
   \addtocontents{toc}{\protect\marktotoc{*}}}
  {\addtocontents{toc}{\protect\marktotoc{}}}
\titleformat{\section}
  {..}
  {\thesection\secmark}{..}{..}
\titlecontents{section}[..]{..}
  {\contentslabel[\thecontentslabel\secmark]{1.5pc}}{..}{..}
```

7. The titlesec philosophy

Once you have read the documentation it should be clear this is not a package for the casual user who likes the standard layout and wants to make simple changes. This is a tool for the serious typographer who has a clear idea of what layout wants and do not have the skill to get it. No attempt is made to improve your taste in section formatting.

8. Appendix

The following examples will be illustrative. In this part, the `\parskip` is 0 pt.

■

9 **This is an example of the section command defined below and, what's more, this is an example of the section command defined below**

```
\titleformat{\section}[block]
  {\normalfont\bfseries\filcenter}{\fbox{\itshape\thesection}}{1em}{}
```

■

SECTION 10

A framed title

```
\titleformat{\section}[frame]
  {\normalfont}
  {\filright
   \footnotesize
   \enspace SECTION \thesection\enspace}
  {8pt}
  {\Large\bfseries\filcenter}
```

■

11. A Ruled Title

```
\titleformat{\section}
  {\titlerule
   \vspace{.8ex}%
   \normalfont\itshape}
  {\thesection.}{.5em}{}
```

■

12

Another Ruled Title

```
\titleformat{\section}[block]
  {\normalfont\sffamily}
  {\thesection}{.5em}{\titlerule\\[.8ex]\bfseries}
```

■

. . . . . .

13 The length of the “rule” above is that of the longest line in this title increased by two picas

. . . . . .

14 This one is shorter

```
\titleformat{\section}[block]
  {\filcenter\large
   \addtolength{\titlewidth}{2pc}%
   \titleline*[c]{\titlerule*[.6pc]{\tiny\textbullet}}%
   \addvspace{6pt}%
   \normalfont\sffamily}
  {\thesection}{1em}{}
\titlespacing{\section}
  {5pc}{*2}{*2}[5pc]
```

■

SECTION 15

This is an example of the section command defined below and, what’s more, this is an example of the section command defined below. Let us repeat it. This is an example of the section command defined below and, what’s more, this is an example of the section command defined below

```
\titleformat{\section}[display]
  {\normalfont\fillast}
  {\scshape section \oldstylenums{\thesection}}
  {1ex minus .1ex}
  {\small}
\titlespacing{\section}
  {3pc}{*3}{*2}[3pc]
```

■

THIS PART IS THE TITLE ITSELF and this part is the section body...

```
\titleformat{\section}[runin]
  {\normalfont\scshape}
  {}{0pt}{}
\titlespacing{\section}
  {\parindent}{*2}{\wordsep}
```

■

16. A Simple Example of the “wrap” Section Shape Which is followed by some text to show the result. Which is followed by some text to show the result. Which is followed by some text to show the result. Which is followed by some text to show the result. Which is followed by some text to show the result. Which is followed by some text to show the result. Which is followed by some text to show the result. Which is followed by some text to show the result.

17. And another Note how the text wraps the title and the space reserved to it is readjusted automatically. And it is followed by some text to show the result. Which is followed by some text to show the result.

```
\titleformat{\section}[wrap]
  {\normalfont\fontseries{b}\selectfont\filright}
  {\thesection.}{.5em}{}
\titlespacing{\section}
  {12pc}{1.5ex plus .1ex minus .2ex}{1pc}
```

■

§ 18. Old-fashioned runin title.—Of course, you would prefer just a dot after the title. In this case the optional argument should be `[.]` and the space after a sensible value (1em, for example).

```
\titleformat{\section}[runin]
  {\normalfont\bfseries}
  {\S\ \thesection.}{.5em}{}[.---]
\titlespacing{\section}
  {\parindent}{1.5ex plus .1ex minus .2ex}{0pt}
```

■

Example of margin section

Which is followed by some text to show the result. But do not stop reading, because the following example illustrates how to take advantage of other packages. The last command in the last argument can take an argument, which is the title with no additional command inside it. We just give the code, but you may try it by yourself. Thus, with the soul package you may say

```
\newcommand{\secformat}[1]{\MakeLowercase{\so{#1}}}
   % \so spaces out letters
\titleformat{\section}[block]
  {\normalfont\scshape\filcenter}
  {\thesection}
  {1em}
  {\secformat}
```

The margin title above was defined:

```
\titleformat{\section}[leftmargin]
  {\normalfont
   \titlerule*[.6em]{\bfseries.}%
   \vspace{6pt}%
   \sffamily\bfseries\filleft}
  {\thesection}{.5em}{}
\titlespacing{\section}
  {4pc}{1.5ex plus .1ex minus .2ex}{1pc}
```

■

The following examples are intended for chapters. However, this document lacks of `\chapter` and are showed using `\section`s with slight changes.

CHAPTER 19

The Title

```
\titleformat{\chapter}[display]
```

```
  {\normalfont\Large\filcenter\sffamily}
  {\titlerule[1pt]%
   \vspace{1pt}%
   \titlerule
   \vspace{1pc}%
   \LARGE\MakeUppercase{\chaptertitlename} \thechapter}
  {1pc}
  {\titlerule
   \vspace{1pc}%
   \Huge}
```

■

CHAPTER XX

The Title

```
\renewcommand{\thechapter}{\Roman{chapter}}
\titleformat{\chapter}[display]
  {\bfseries\Large}
  {\filleft\MakeUppercase{\chaptertitlename} \Huge\thechapter}
  {4ex}
  {\titlerule
   \vspace{2ex}%
   \filright}
  [\vspace{2ex}%
   \titlerule]
```

9.1. A full example

Now an example of a complete title scheme follows.

```
\documentclass[twoside]{report}
\usepackage[sf,sl,outermarks]{titlesec}

% \chapter, \subsection...: no additional code

\titleformat{\section}
  {\LARGE\sffamily\slshape}
  {\thesection}{1em}{}
\titlespacing{\section}
  {-6pc}{3.5ex plus .1ex minus .2ex}{1.5ex minus .1ex}

\titleformat{\paragraph}[leftmargin]
  {\sffamily\slshape\filright}
  {}{}{}
\titlespacing{\paragraph}
  {5pc}{1.5ex minus .1 ex}{1pc}

% 5+1=6, ie, the negative left margin in section

\widenhead{6pc}{0pc}
```

```
\renewpagestyle{plain}{}

\newpagestyle{special}[\small\sffamily]{
   \headrule
   \sethead[\usepage][\textsl{\chaptertitle}][]
           {}{\textsl{\chaptertitle}}{\usepage}}

\newpagestyle{main}[\small\sffamily]{
   \headrule
   \sethead[\usepage][\textsl{\thechapter. \chaptertitle}][]
           {}{\textsl{\thesection. \sectiontitle}}{\usepage}}

\pagestyle{special}

\begin{document}

---TOC

\pagestyle{main}

---Body

\pagestyle{special}

---Index
\end{document}
```

9.2. Standard Classes

Now follows, for your records, how sectioning commands of standard classes could be defined.

```
\titleformat{\chapter}[display]
  {\normalfont\huge\bfseries}{\chaptertitlename\ \thechapter}{20pt}{\Huge}
\titleformat{\section}
  {\normalfont\Large\bfseries}{\thesection}{1em}{}
\titleformat{\subsection}
  {\normalfont\large\bfseries}{\thesubsection}{1em}{}
\titleformat{\subsubsection}
  {\normalfont\normalsize\bfseries}{\thesubsubsection}{1em}{}
\titleformat{\paragraph}[runin]
  {\normalfont\normalsize\bfseries}{\theparagraph}{1em}{}
\titleformat{\subparagraph}[runin]
  {\normalfont\normalsize\bfseries}{\thesubparagraph}{1em}{}

\titlespacing*{\chapter}      {0pt}{50pt}{40pt}
\titlespacing*{\section}      {0pt}{3.5ex plus 1ex minus .2ex}{2.3ex plus .2ex}
\titlespacing*{\subsection}   {0pt}{3.25ex plus 1ex minus .2ex}{1.5ex plus .2ex}
\titlespacing*{\subsubsection}{0pt}{3.25ex plus 1ex minus .2ex}{1.5ex plus .2ex}
\titlespacing*{\paragraph}    {0pt}{3.25ex plus 1ex minus .2ex}{1em}
\titlespacing*{\subparagraph} {\parindent}{3.25ex plus 1ex minus .2ex}{1em}
```

9.3. Chapter Example

A final example shows how to take advantage of the `picture` environment for fancy sectioning formats. Even with the simple tools provided by standard LaTeX you may create impressive titles but you may devise more elaborated ones with, for instance, `pspicture` (PSTricks package) or by including graphics created with the help of external programs.

```
\usepackage[dvips]{color}
\usepackage[rigidchapters,explicit]{titlesec}

\DeclareFixedFont{\chapterfont}{T1}{phv}{bx}{n}{11cm}
```

```
\titlespacing{\chapter}{0pt}{0pt}{210pt}
% Most of titles have some depth. The total space is
% a bit larger than the picture box.

\titleformat{\chapter}[block]
  {\begin{picture}(330,200)}
  {\put(450,80){%
     \makebox(0,0)[rb]{%
       \chapterfont\textcolor[named]{SkyBlue}{\thechapter}}}
   \put(0,230){%
     \makebox(0,0)[lb]{%
       \Huge\sffamily\underline{Chapter \thechapter}}}}
  {0pt}
  {\put(0,190){\parbox[t]{300pt}{%
     \Huge\sffamily\filright#1}}}
  [\end{picture}]
```

(The exact values to be used depend on the text area, class, `\unitlength`, paper size, etc.)