The KOMA-Script package
tocstyle*

Markus Kohm

2009/11/09

While the main classes of the KOMA-Script bundle were made, there where several ideas for formating the table of contents and lists of floats, but almost none of them where implemented. One reason was, that the KOMA-Script author didn't like to change the LaTeX kernel at a class, because this may result in serveral problems with other packages. The package tocstyle will fill the gap. If it conflicts with another package, you simply may decide not to use it.

Contents

*This is version v0.2d-alpha of file `tocstyle.dtx`.

1. How It Works

Loading the package tocstyle will redefine the kernel macro `\@starttoc`. Using the redefined `\@starttoc` will redefine `\@dottedtocline`, `\l@part` down to `\l@subparagraph`, `\l@figure`, and `\l@table`, if and only if tocstyle wasn't deactivated for all TOCs or this TOC. Usage the redefined `\@dottedtocline` will redefine `\numberline`.

Redefining `\@starttoc`, `\@dottedtocline`, and `\numberline` will activate the features of tocstyle for all lists that uses these, e.g. table of contents, list of figures and list of tables at the standard or the KOMA-Script classes. But while not all classes uses `\@dottedtocline` and `\@numberline` for all entries to table of contents and list of floats the package redefines some other macros that are typically used for those entries. These are e.g.

\l@part, \l@chapter and some more. If the class even does not use those macros, you may not use tocstyle to change the lists. The term TOC will be used for all kind of list, that may be processed by tocstyle. The package tests wether the original kernel macros \@starttoc, \@dottedtocline, and \numberline were used or not and warns if not.

Package tocstyle needs some more information. For the standard and the KOMA-Script classes these informations may be detected by the package. If the result is not the expected, you may configure these informations manually.

The entries of every TOC hat a depth. See the counter tocdepth for more information about the depth. You may change several settings for the entries of either all depths of all TOCs, all depths of one TOC, or one depth of one TOC.

But most users will not need to set up tocstyle at this low level. They simply will select one of the predefined styles and maybe select one of the optional features.

2. Optional Features

Optional features will be selected using a package option while loading the package or using the package option as a global option loading the class using \documentclass. Optional features change generall behaviour of all TOCs.

tocindentauto
tocindentmanual

With option tocindentauto all widths at the TOCs are calculated by tocstyle. The calculation of the width needs at least one LaTeX run with all TOC entries. So you need at least three LaTeX runs:

- one to write all the TOC entries to the TOC file
- one with the known TOC entries from the TOC file but unknown widths
- one with the known TOC entries from the TOC file and known widths

If the TOC entries changed between the second and the third run — e.g. because of page numbers changed — you'll need one more run (and so on).

Note: The widths of all entries of same depth and same TOC are same. Don't ask for less width of page numbers at the first than the last TOC page!

tocgraduated
tocflat
tocfullflat

The option tocgraduated selects the graduated version of all TOCs. You know the graduated version from the standard classes. Entries of lower depth are indented against entries of higher depth. This may e.g. look like:

The option `tocflat` selects the flat — aka left aligned — version of all TOCs. You know the flat version from the KOMA-Script classes using option `tocleft`. This may e.g. look like:

The option `tocfullflat` is similar to flat version of all TOCs, but there is even no box of same width for the numbers of all entries. This may e.g. look like:

Default is option `tocgraduated`.

`tocbreaksstrict` `tocbreakscareless` Default option `tocbreaksstrict` sets a lot of penalties before and after TOC entries to avoid page break between a TOC entry and it's parent. But sometimes you may like to allow more page breaks. You may use option `tocbreakscareless` for this.

`toctextentriesindented` `toctextentriesleft` With default option `toctextentriesleft` unnumbered TOC entries, e.g. from KOMA-Script command `\addchap`, are indented only as wide as the number of numbered TOC entries of the same level are. But with option `toctextentriesindented` these are indented as if they have an empty number.

3. Using TOC Styles

Package tocstyle hat several predefined toc styles. Most users will never need to define their own toc style but only select one of the predefined and maybe configure it by one of the options described at the previsous section.

`\usetocstyle` You may set the style of one or all TOCs. If you want to set the style of all TOCs, you'd simply say `\usetocstyle{`⟨*style*⟩`}`. This will set all settings of the given style to all TOCs. Individual settings will overwrite this generall setting.

If you use `\usetocstyle[`⟨*TOC*⟩`]{`⟨*style*⟩`}`, only the style of the given TOC will be set. This will be done *after* the general setting. Only individual settings of single features may overwrite the setting of the style.

The table 1 shows the predefined styles, that may be used as mandatory argument of `\tocstyle`. The optional argument ⟨*TOC*⟩ is the shortcut (file extension) of the TOC. Examples of known shortcuts are shown at table 2.

Note: Before you're setting a style the style of the TOCs are unspecified. This means that some entries may be set using tocstyle others may not.

`\deactivatetocstyle` `\reactivatetocstyle` Both commands have one optional argument ⟨*TOC*⟩. You may deactive the influence of tocstyle for a TOC and reactivate it. If you use `\deactivatetocstyle` without the optional argument or empty optional argument, the influence of tocstyle for all TOCs will be deactivated and may reactivated only using `\reactivatetocstyle` without the optional argument or empty optional argument too.

After deactivation of tocstyle for one TOC or all TOCs you may continue configuring TOCs. All these changes will be used after reactivation.

4. Setting-up Single Features

At the previous section you've learned how to select a predefined TOC style. You were also told, that you may change one ore more features against the

Table 1: Predefined TOC Styles

`standard`
: A style similar to the standard classes. All width are predefined to the width of the standard classes, but may be overwritten by the general options (see section 2). The depth -1 (part) and 0 (chapter) are set in bold face (`\bfseries`). If no depth 0 was found at the TOC, depth 1 (section) will be set in bold face, too. All other depth will be set in normal font. Depth -1 (part) will be set using `\large`. The font changes are valid for the page numbers, too.

`KOMAlike`
: A style similar to the KOMA-Script classes. This is almost the same like `standard`, but instead of bold face `\usekomafont{disposition}` will be used if `\usekomafont` was defined and sans serif, bold face (`\sffamily\bfseries`) if not.

`classic`
: Like `KOMAlike` but all page numbers are set using normal font.

`allwithdot`
: Like `classic` but dots between entry text and page numbers are used at all depths.

`noonewithdot`
: Like `classic` but not dots between entry text and page numbers are used.

`nopagecolumn`
: Like `noonewithdot` but also the gap between text and page numbers is omited. This means, that the page numbers are set 1 em after the text.

Table 2: Known TOC Shortcuts

`toc`
: Table of contents of almost all known classes.

`lof`
: List of figures of almost all known classes.

`lot`
: List of tables of almost all known classes.

`lol`
: List of listings of package listings. Currently the usability of listings with tocstyle is not recomended. Maybe it works, maybe not. Maybe you should try `\deactivatetocstyle[lol]`.

Table 3: Features that May Be Set

`dothook`	will be executed before any dot of the dot line
`entryhook`	hook before the entry will be set
`entryvskip`	initial vertical skip amount (if not set 0pt plus .2pt will be used)
`leaders`	commands for fillin the gap between entry text and page number (if not set the default leaders command with dots will be used)
`pagenumberbox`	the box command for setting the page number (if not set the default box of with `\@pnumwidth` will be used); note, that this has to be a command with exactly one argument
`pagenumberhook`	hook before the page number will be set at the page number box
`parfillskip`	add this amount to the default value of `\parfillskip` after setting up all lengths
`raggedhook`	the only allowed values here are `\raggedright` or nothing
`spaceafternumber`	amount of minimum space after the entry number, if the needed width will be calculated automaticly

used predefined TOC style for one or all depth of one ore all TOCs. Now you will learn how to do this.

`\settocfeature`
`\settocstylefeature`

These commands are used to set a single feature eiher of all depth of all TOCs (`\settocfeature {`⟨*feature*⟩`} {`⟨*command*⟩`}` or `\settocstylefeature {`⟨*feature*⟩`} {`⟨*commands*⟩`}`), or of all depth of a single TOC (`\settocfeature [`⟨*TOC*⟩`] {`⟨*feature*⟩`} {`⟨*commands*⟩`}`), or of a single depth of all TOCs (`\settocstylefeature [`⟨*depth*⟩`] {`⟨*feature*⟩`} {`⟨*commands*⟩`}`), or of a single depth of a single TOC (`\settocfeature [`⟨*TOC*⟩`] [`⟨*depth*⟩`] {`⟨*feature*⟩`} {`⟨*commands*⟩`}`).

Parameter ⟨*commands*⟩ is a list of commands. In most cases these must not be commands, that need an argument. So you should e.g. not use `\textbf` but `\bfseries` to switch to bold face. Parameter ⟨*feature*⟩ is the feature, that may be configured with parameter ⟨*commands*⟩. All known features are show at table 3.

The order of used commands for a feature is

1. commands for all depths of all TOCs,

2. commands for all depth of a single TOC,

3. commands for a single depth of all TOCs,

4. commands for a single depth of a single TOC,

and settings of `\usetocstyle` may be overwritten by `\settocfeature` und `\settocstylefeature`.

5. Defining New TOC Styles

Now you know how to select a prefedined TOC style and how to change single features. But wouldn't it be nice to define your own TOC style?

`\newtocstyle` You may do this using `\newtocstyle[`⟨*parent style*⟩`] [`⟨*exclude features*⟩`] {`⟨*style name*⟩`} {`⟨`\settocstylefeature`-*commands*⟩`}`. If you used the optional argument ⟨*parent style*⟩ all features of the parent style will be part of the new style, before overwriting them with the features of the ⟨`\settocstylefeature`-*commands*⟩. You should not use any other commands at the last argument. But at `\newtocstyle` the command `\settocfeature` becomes an alias for `\settocstylefeature` to avoid to much mistakes.

The second optional argument is a comma seperated list of feature names. If it is used, these features of the parent style (and all ancestors of the parent) will not be part of the new style.

`\aliastoc` Using `\aliastoc{`⟨*original-TOC*⟩`}{`⟨*alias-TOC*⟩`}` you may define an alias for a TOC. The first argument is the original TOC for that the second argument should be the alias. An alias-TOC will be processed with all settings, that were done for the original-TOC. Internally this command is used as default for the optional, first argument of `\showtoc`.

6. Processing a TOC

While LaTeX inputs a toc file it processes the commands of the TOC. These commands mainly produce the entries of the toc. Some commands are only available or valid while a TOC is processed. But be carefull: Some of these are read-only commands. Changing such a read-only command may result in various errors!

`\@starttoc` The internal comand `\@startoc` is defined by the LaTeX kernel. It is used
`\showtoc` by package and class authors to build commands like `\tableofcontents` or `\listoffigures`. Without using it you will not get a toc file. tocstyle redefines it, to add pre- and post-processing commands. The original definition found by tocstyle will be used inside the redefinition.

`\showtoc [`⟨*preprocession*⟩`]{`⟨*TOC*⟩`}` is an addition of `tocstyle`. Using it will procude a copy of TOC and process this copy. The copy

will be done just after creating the original TOC. The copy will be an alias for the original file. The extension of the copy is the generated alias if ⟨*TOC*⟩. You may generate the alias using `\aliastoc` at the optional argument of `\showtoc`. The default for this optional argument will be `\aliastoc\tocstyleTOC\tocstyleAliasTOC` and the default alias `\tocstyleAliasTOC` will be `\tocstyleTOC` extended by a number. The first TOC example at section 2 was made using

```
\showtoc[{%
  \aliastoc{\tocstyleTOC}{toc}%
  \usetocstyle[toc]{standard}%
  \settocfeature[toc]{raggedhook}{\raggedright}%
  \selecttocstyleoption{tocgraduated}%
}]{toc}
```

If you want to show a copy the table of contents, that shows only depth 1 of the headlines you may simply use:

```
\showtoc[{%
  \expandafter\value{tocdepth}=1\relax
  \aliastoc{\tocstyleTOC}{toc}%
}]{toc}
```

or

```
\newcounter{savedtocdepth}
\setcounter{savedtocdepth}{\value{tocdepth}}
\setcounter{tocdepth}{1}
\showtoc{toc}
\setcounter{tocdepth}{\value{savedtocdepth}}
```

`\tocstyleTOC` `\tocstyleAliasTOC` These are read-only macros. While processing a TOC using `\@starttoc` or `\showtoc`, `\tocstyleAliasTOC` is the shortcut, that is valid for the features and `\tocstyleTOC` is valid for the file extension to be used.

`\tocstyledepth` This is a read-only macro. While processing a single toc entry with `\@dottedtocline` this is the depth (first argument of `\@dottedtocline`) of this entry. Most users will never need this, but it is often used internaly. Because of this *you should never change it!*

`\iftochasdepth` Using `\iftochasdepth{`⟨*TOC*⟩`}{`⟨*depth*⟩`}{`⟨*true*⟩`} {`⟨*false*⟩`}` you may test, if an entry of a given depth was already output to a TOC. If so the commands of argument ⟨*true*⟩ will be processed. If not so the commands of argument ⟨*false*⟩ will be processed.

7. Configuration file

There's another feature for new toc styles. If there's a file `tocstyle.cfg` it will be loaded at the end of the package. This is usefull to define your own toc styles.

8. Implementation

```
\PackageWarningNoLine{tocstyle}{%
  THIS IS AN ALPHA VERSION!\MessageBreak
  USAGE OF THIS VERSION IS ON YOUR OWN RISK!\MessageBreak
  EVERYTHING MAY HAPPEN!\MessageBreak
  EVERYTHING MAY CHANGE IN FUTURE!\MessageBreak
  THERE IS NO SUPPORT, IF YOU USE THIS PACKAGE!\MessageBreak
  Maybe it would be better, not to load this package%
}
```

8.1. Option

Options change general behaviour of TOCs.

\selecttocstyleoption

```
\newif\if@tocstyle@penalties
\newif\iftocstyle@autolength
\newif\iftocstyle@indentnotnumbered
\newcount\tocstyle@indentstyle\tocstyle@indentstyle=\z@
\newcommand*{\selecttocstyleoption}[1]{%
  \begingroup
    \edef\@tempa{#1}%
    \edef\@tempb{tocbreaksstrict}%
    \ifx\@tempa\@tempb\aftergroup\@tocstyle@penaltiestrue\else
      \edef\@tempb{tocbreakscareless}%
      \ifx\@tempa\@tempb\aftergroup\@tocstyle@penaltiesfalse\else
        \edef\@tempb{tocindentauto}
        \ifx\@tempa\@tempb\aftergroup\tocstyle@autolengthtrue\else
          \edef\@tempb{tocindentmanual}%
          \ifx\@tempa\@tempb\aftergroup\tocstyle@autolengthfalse\else
            \edef\@tempb{tocgraduated}%
            \ifx\@tempa\@tempb
              \aftergroup\tocstyle@indentstyle\aftergroup\z@
            \else
              \edef\@tempb{tocflat}%
              \ifx\@tempa\@tempb
                \aftergroup\tocstyle@indentstyle\aftergroup\@ne
                \aftergroup\relax
              \else
                \edef\@tempb{tocfullflat}%
                \ifx\@tempa\@tempb
                  \aftergroup\tocstyle@indentstyle\aftergroup\tw@
                  \aftergroup\relax
                \else
                  \edef\@tempb{toctextentriesindented}%
                  \ifx\@tempa\@tempb\aftergroup\tocstyle@indentnotnumberedtrue
                  \else
                    \edef\@tempb{toctextentriesleft}%
                    \ifx\@tempa\@tempb
```

```
                    \aftergroup\tocstyle@indentnotnumberedfalse
                  \else
                    \PackageError{tocstyle}{unknown option ‘#1’}{%
                      You’ve told me to select toc style option
                      ‘#1’,\MessageBreak
                      but tocstyle doesn’t know an option named ‘#1’}%
                  \fi
                \fi
              \fi
            \fi
          \fi
        \fi
      \fi
    \fi
  \endgroup
}
```

chapter nochapter \iftochaschapter

Do we have \chapter and \l@chapter?

```
\newif\iftochaschapter\tochaschapterfalse
\ifcsname l@chapter\endcsname
  \ifcsname chapter\endcsname
    \tochaschaptertrue
  \fi
\fi
```

tocbreaksstrict tocbreakscareless

Switch on extended pernalties.

```
\DeclareOption{tocbreaksstrict}{\selecttocstyleoption\CurrentOption}
\DeclareOption{tocbreakscareless}{\selecttocstyleoption\CurrentOption}
```

tocindentauto tocindentmanual

```
\DeclareOption{tocindentauto}{\selecttocstyleoption\CurrentOption}
\DeclareOption{tocindentmanual}{\selecttocstyleoption\CurrentOption}
```

toctextentriesindented toctextentriesleft

```
\DeclareOption{toctextentriesindented}{\selecttocstyleoption\CurrentOption}
\DeclareOption{toctextentriesleft}{\selecttocstyleoption\CurrentOption}
```

tocgraduated tocflat tocfullflat

```
\DeclareOption{tocgraduated}{\selecttocstyleoption\CurrentOption}
\DeclareOption{tocflat}{\selecttocstyleoption\CurrentOption}
\DeclareOption{tocfullflat}{\selecttocstyleoption\CurrentOption}
```

Defaults and others:

```
\ExecuteOptions{tocbreaksstrict,tocindentauto,tocgraduated,%
  toctextentriesleft}
\ProcessOptions\relax
```

```
\ifcsname if@tocleft\endcsname
  \expandafter\let\csname if@tempswa\expandafter\endcsname
  \csname if@tocleft\endcsname
\else
  \@tempswafalse
\fi
\if@tempswa
  \PackageWarningNoLine{tocstyle}{%
    You should not use class option ‘toc=flat’!\MessageBreak
    This may result in errors or unexpected results.\MessageBreak
    I’ll try to deactivate ‘toc=flat’, now.\MessageBreak
    You may use package options ‘tocflat’ and\MessageBreak
    ‘tocindentauto’ instead of ‘toc=flat’}%
  \csname @tocleftfalse\endcsname
\fi
```

8.2. Body

There are two parts at tocstyle:

- redefining internal LaTeX kernel macros,
- defining new macros and redefining class macros.

Redefining LaTeX kernel macros may not be switched of. But redefining class macros will only be on demand.

8.2.1. Redefining LaTeX Kernel Macros

Some LaTeX kernel macros must be redefined to add the new functionality. Before redefining them, we test against the definition at kernel 2005/12/01

`\@starttoc` `\tocstyle@saved@@starttoc` The original definition will be extended by defaults for `\parskip`, `\parindent` and `\parfillskip` and storage of the shortcut of the current TOC.

```
\newcommand*\tocstyle@saved@starttoc{}
\let\tocstyle@saved@starttoc\@starttoc
\renewcommand*{\@starttoc}[1]{%
  \tocstyle@pre@starttoc{#1}%
  \tocstyle@saved@starttoc{#1}%
  \tocstyle@post@starttoc{#1}%
}
```

`\tocstyle@saved@dottedtocline` For saving the unchanged definition (at `\begindocument`):

```
\newcommand*{\tocstyle@saved@dottedtocline}{}
```

`\tocstyle@dottedtocline` Implement new definition and redefine:

```
\newcommand*{\tocstyle@dottedtocline}[5]{%
  \let\numberline\tocstyle@numberline
  \ifnum #1>\c@tocdepth \else
```

Penalty feature: no page break between higher and lower depths.

```
    \if@tocstyle@penalties
      \begingroup
        \@tempcnta 20010
        \advance \@tempcnta by -#1
        \ifnum \@tempcnta>\lastpenalty
          \aftergroup\penalty\aftergroup\@lowpenalty
        \fi
      \endgroup
    \fi
```

Activation of all features for this TOC and depth:

```
    \edef\tocstyledepth{#1}%
    \tocstyle@activate@features
```

Similar to kernel command but if feature `entryvskip` was set use `\addvspace`:

```
    \ifx\tocstyle@feature@entryvskip\relax
      \vskip \z@ \@plus.2\p@
    \else
      \addvspace{\tocstyle@feature@entryvskip}%
    \fi
    {%
```

Preinitialization of lengths and skips and then call a hook

```
      \parskip \z@ \parindent \z@ \leftskip \z@ \rightskip \z@
      \tocstyle@feature@raggedhook
```

Set number indent to `\@tempdimb` and text indent to `\@tempdima`.

```
      \@tempdima #3\relax
      \@tempdimb #2\relax
⟨trace⟩      \typeout{number indent by \string\l@... (\tocstyleTOC, \tocstyledepth
⟨trace⟩      \typeout{text indent by \string\l@... (\tocstyleTOC, \tocstyledepth):
```

Calc auto lengths. Use max. of last run of parents if available.

```
      \ifnum #1>\z@\relax
        \@tempcnta #1\relax \advance\@tempcnta \m@ne
        \ifcsname tocstyle@maxskipwidth@\tocstyleTOC @\the\@tempcnta\endcsname
          \ifcsname tocstyle@maxnumwidth@\tocstyleTOC @\the\@tempcnta\endcsname
            \@tempdimb
            \csname tocstyle@maxskipwidth@\tocstyleTOC @\the\@tempcnta\endcsname
            \advance\@tempdimb
            \csname tocstyle@maxnumwidth@\tocstyleTOC @\the\@tempcnta\endcsname
          \fi
        \fi
      \fi
⟨trace⟩      \typeout{number indent by parent (\tocstyleTOC, \tocstyledepth): \spa
      \ifcsname tocstyle@skipwidth@\tocstyleTOC @#1\endcsname
        \ifdim \@tempdimb>
        \csname tocstyle@skipwidth@\tocstyleTOC @#1\endcsname\relax
          \expandafter\xdef\csname tocstyle@skipwidth@\tocstyleTOC
          @#1\endcsname{\the\@tempdimb}%
```

```
        \fi
      \else
        \expandafter\xdef\csname tocstyle@skipwidth@\tocstyleTOC
        @#1\endcsname{\the\@tempdimb}%
      \fi
      \iftocstyle@autolength
        \ifcsname tocstyle@maxskipwidth@\tocstyleTOC @#1\endcsname
          \@tempdimb \csname tocstyle@maxskipwidth@\tocstyleTOC @#1\endcsname
          \relax
        \fi
        \ifcsname tocstyle@maxnumwidth@\tocstyleTOC @#1\endcsname
          \@tempdima \csname tocstyle@maxnumwidth@\tocstyleTOC @#1\endcsname
          \relax
        \fi
⟨trace⟩       \typeout{text indent calculated (\tocstyleTOC, \tocstyledepth): \th
⟨trace⟩       \typeout{number indent calculated (\tocstyleTOC, \tocstyledepth): \
      \else
        \@tempdimb #2\relax
⟨trace⟩       \typeout{number indent explicite (\tocstyleTOC, \tocstyledepth): \t
      \fi
      \ifcsname tocstyle@unumwidth@\tocstyleTOC @\endcsname
        \ifdim \@tempdima>
        \csname tocstyle@unumwidth@\tocstyleTOC @\endcsname\relax
          \expandafter\xdef\csname tocstyle@unumwidth@\tocstyleTOC
          @\endcsname{\the\@tempdima}%
        \fi
      \else
        \expandafter\xdef\csname tocstyle@unumwidth@\tocstyleTOC
        @\endcsname{\the\@tempdima}%
      \fi
      \ifcase\tocstyle@indentstyle\relax\else
        \@tempdimb \z@
        \ifcsname tocstyle@maxunumwidth@\tocstyleTOC @\endcsname
          \@tempdima \csname tocstyle@maxunumwidth@\tocstyleTOC @\endcsname
          \relax
        \fi
⟨trace⟩       \typeout{text noindent (\tocstyleTOC, \tocstyledepth): \the\@tempdi
⟨trace⟩       \typeout{number noindent (\tocstyleTOC, \tocstyledepth): \the\@temp
      \fi
```

Advance instead of set, because of the hook above:

```
      \advance\parindent \@tempdimb\@afterindenttrue
      \advance\leftskip \parindent
      \advance\rightskip \@tocrmarg
      \parfillskip -\rightskip
      \ifx\tocstyle@feature@parfillskip\relax\else
        \advance\parfillskip \tocstyle@feature@parfillskip\relax
      \fi
      \interlinepenalty\@M
      \leavevmode
```

```
      \advance\leftskip \@tempdima
      \null\nobreak
```

`\hskip\-leftskip` optional moved to `\numberline`

```
      \iftocstyle@indentnotnumbered\else
        \hskip -\leftskip
      \fi
```

Change at start of the entry

```
      \tocstyle@feature@entryhook
```

Similar to kernel command but if feature `leaders` was set use this instead of the default leaders. And if feature `dothook` was set (default is `\normalfont`) use this at the default leaders.

```
      {#4}\nobreak
      \ifx\tocstyle@feature@leaders\relax
        \leaders\hbox{$\m@th
          \mkern \@dotsep mu\hbox{\tocstyle@feature@dothook .}%
          \mkern \@dotsep mu$}\hfill
      \else
        \tocstyle@feature@leaders
      \fi
      \nobreak
      \ifx\tocstyle@feature@pagenumberbox\relax
        \hb@xt@\@pnumwidth{\hfil\tocstyle@feature@pagenumberhook #5}%
      \else
        \tocstyle@feature@pagenumberbox{\tocstyle@feature@pagenumberhook #5}%
      \fi
      \par
    }%
```

Last change is, another penalty change:

```
    \if@tocstyle@penalties
      \bgroup
        \@tempcnta 20009
        \advance\@tempcnta by -#1
        \edef\reserved@a{\egroup\penalty\the\@tempcnta\relax}%
      \reserved@a
    \fi
  \fi}
```

`\tocstyle@saved@numberline`
`\tocstyle@numberline`

Define a new `\numberline`, that will do all the job after `\begindocument` and one to save the original definition.

```
\newcommand*{\tocstyle@saved@numberline}{}
\newcommand*{\tocstyle@numberline}[1]{%
  \begingroup
    \ifx\tocstyle@feature@spaceafternumber\relax
      \settowidth\@tempdima{\tocstyle@@numberline{#1}\enskip}%
    \else
      \settowidth\@tempdima{\tocstyle@@numberline{#1}}%
```

```
      \advance \@tempdima \tocstyle@feature@spaceafternumber\relax
    \fi
    \ifcsname tocstyle@numwidth@\tocstyleTOC @\tocstyledepth\endcsname
      \ifdim \@tempdima >
      \csname tocstyle@numwidth@\tocstyleTOC @\tocstyledepth\endcsname\relax
        \expandafter\xdef\csname tocstyle@numwidth@\tocstyleTOC
        @\tocstyledepth\endcsname{\the\@tempdima}%
      \fi
    \else
      \expandafter\xdef\csname tocstyle@numwidth@\tocstyleTOC
      @\tocstyledepth\endcsname{\the\@tempdima}%
    \fi
  \endgroup
  \iftocstyle@indentnotnumbered
    \hskip -\leftskip
  \fi
  \ifcase \tocstyle@indentstyle
    \hb@xt@\@tempdima{\tocstyle@@numberline{#1}\hfil}%
  \or
    \hb@xt@\@tempdima{\tocstyle@@numberline{#1}\hfil}%
  \else
    \ifx\tocstyle@feature@spaceafternumber\relax
      \hbox{\tocstyle@@numberline{#1}\enskip}%
    \else
      \hbox{\tocstyle@@numberline{#1}\hskip
        \tocstyle@feature@spaceafternumber\relax}%
    \fi
  \fi
}
```

`\tocstyle@@numberline` Do the main work!

```
\newcommand*{\tocstyle@@numberline}[1]{%
  #1\csname autodot\endcsname
}
```

8.2.2. Redefining Class Macros

`\l@part` `\l@chapter` `\l@section` `\l@subsection` `\l@subsubsection` `\l@paragraph` `\l@subparagraph` `\l@table` `\l@figure` Try to redefine the toc commands at startup.

```
\AtBeginDocument{%
  \ifcsname l@part\endcsname
    \setbox\@tempboxa\vbox{\hsize\maxdimen
      \l@part{\tocstyle@l@define{part}{-1}}{}}%
  \fi
  \ifcsname l@chapter\endcsname
    \setbox\@tempboxa\vbox{\hsize\maxdimen
      \l@chapter{\tocstyle@l@define{chapter}{0}}{}}%
  \fi
  \ifcsname l@section\endcsname
    \setbox\@tempboxa\vbox{\hsize\maxdimen
```

```
    \l@section{\tocstyle@l@define{section}{1}}{}}%
\fi
\ifcsname l@subsection\endcsname
  \setbox\@tempboxa\vbox{\hsize\maxdimen
    \l@subsection{\tocstyle@l@define{subsection}{2}}{}}%
\fi
\ifcsname l@subsubsection\endcsname
  \setbox\@tempboxa\vbox{\hsize\maxdimen
    \l@subsubsection{\tocstyle@l@define{subsubsection}{3}}{}}%
\fi
\ifcsname l@paragraph\endcsname
  \setbox\@tempboxa\vbox{\hsize\maxdimen
    \l@paragraph{\tocstyle@l@define{paragraph}{4}}{}}%
\fi
\ifcsname l@subparagraph\endcsname
  \setbox\@tempboxa\vbox{\hsize\maxdimen
    \l@subparagraph{\tocstyle@l@define{subparagraph}{5}}{}}%
\fi
\ifcsname l@table\endcsname
  \setbox\@tempboxa\vbox{\hsize\maxdimen
    \l@table{\tocstyle@l@define{table}{1}}{}}%
\fi
\ifcsname l@figure\endcsname
  \setbox\@tempboxa\vbox{\hsize\maxdimen
    \l@figure{\tocstyle@l@define{figure}{1}}{}}%
\fi
```

`\@dottedtocline` This will be used even for undotted toc lines. First check the definition, then redefine.

```
\def\@tempa#1#2#3#4#5{%
  \ifnum #1>\c@tocdepth \else
    \vskip \z@ \@plus.2\p@
    {\leftskip #2\relax \rightskip \@tocrmarg \parfillskip -\rightskip
     \parindent #2\relax\@afterindenttrue
     \interlinepenalty\@M
     \leavevmode
     \@tempdima #3\relax
     \advance\leftskip \@tempdima \null\nobreak\hskip -\leftskip
     {#4}\nobreak
     \leaders\hbox{$\m@th
        \mkern \@dotsep mu\hbox{.}\mkern \@dotsep
        mu$}\hfill
     \nobreak
     \hb@xt@\@pnumwidth{\hfil \normalfont \normalcolor #5}%
     \par}%
  \fi}%
\ifx\@dottedtocline\@tempa\else
  \tocstyle@macrochangewarning\@dottedtocline
\fi
```

```
  \let\tocstyle@saved@dottedtocline\@dottedtocline
```

`\numberline` This macro needed to be redefined to calculate the with of the numbers. First of all: check the definition. This is a bit more difficult, because of respecting KOMA-Script:

```
  \def\@tempa#1{\hb@xt@\@tempdima{#1\autodot\hfil}}%
  \ifx\numberline\@tempa\else
    \def\@tempa#1{\hb@xt@\@tempdima{#1\hfil}}%
    \ifx\numberline@tempa\else
      \tocstyle@macrochangewarning\numberline
    \fi
  \fi
  \let\tocstyle@saved@numberline\numberline
}
```

`\tocstyle@macrochangewarning`

```
\newcommand*{\tocstyle@macrochangewarning}[1]{%
  \PackageWarningNoLine{tocstyle}{%
    unexpected \string#1\space definition!\MessageBreak
    You are either using an unknown LaTeX kernel\MessageBreak
    version, an unknown class or package, that redefines\MessageBreak
    \string#1, or a \string#1\space
    redefinition\MessageBreak
    at the document preamble.\MessageBreak
    Because of this you may get unexpected results!\MessageBreak
    Maybe it would be better not to use package tocstyle}%
  \PackageInfo{tocstyle}{Unexpected definition is:\MessageBreak
    \meaning#1}%
}
```

`\tocstyle@l@define`
`\tocstyle@activate@all@l`

```
\newcommand*{\tocstyle@activate@all@l}{}
\newcommand*{\tocstyle@l@define}[2]{%
  \advance\leftskip-\@tempdima
  \edef\@tempa{%
    \noexpand\global\noexpand\let
    \expandafter\noexpand\csname tocstyle@saved@l@#1\endcsname
    \expandafter\noexpand\csname l@#1\endcsname
    \noexpand\gdef
    \expandafter\noexpand\csname tocstyle@l@#1\endcsname{%
      \noexpand\@dottedtocline{#2}{\the\leftskip}{\the\@tempdima}}%
    \noexpand\g@addto@macro\noexpand\tocstyle@activate@all@l{%
      \noexpand\let\expandafter\noexpand\csname l@#1\endcsname
      \expandafter\noexpand\csname tocstyle@l@#1\endcsname
    }%
  }%
  \PackageInfo{tocstyle}{prepare \expandafter\string
    \csname l@#1\endcsname\space for redefinition}%
  \@tempa
```

```
}
```

8.2.3. New Macros

\showtoc

```
\newcommand*{\showtoc}[2][\aliastoc\tocstyleTOC\tocstyleAliasTOC]{%
  \ifcsname tocstyle@copyname@#2\endcsname
    \@tempcnta \csname tocstyle@copyname@#2\endcsname\relax
    \advance\@tempcnta \@ne
    \expandafter\xdef\csname tocstyle@copyname@#2\endcsname{\the\@tempcnta}%
  \else
    \expandafter\xdef\csname tocstyle@copyname@#2\endcsname{1}%
  \fi
  \ifx\@dofilelist\relax\let\@dofilelist\@empty\fi
  \edef\@tempa{\noexpand\g@addto@macro\noexpand\@dofilelist{%
      \noexpand\tocstyle@copy@toc{#2}{\csname
        tocstyle@copyname@#2\endcsname}}%
  }\@tempa%
  \begingroup
    \edef\tocstyleAliasTOC{#2}%
    \edef\tocstyleTOC{#2\csname tocstyle@copyname@#2\endcsname}%
    #1
    \tocstyle@pre@starttoc{#2\csname tocstyle@copyname@#2\endcsname}%
    \makeatletter
    \@input{\jobname.#2\csname tocstyle@copyname@#2\endcsname}%
    \@nobreakfalse
    \tocstyle@post@starttoc{#2\csname tocstyle@copyname@#2\endcsname}%
  \endgroup
}
```

\tocstyle@copy@toc

```
\newcommand*{\tocstyle@copy@toc}[2]{%
  \if@filesw
    \begingroup
      \endlinechar=\m@ne
% While \LaTeX{} does not close the files, we have to do it know.
      \immediate\closeout\csname tf@#1\endcsname
      \immediate\openin\@inputcheck \jobname.#1
      \immediate\openout\@partaux \jobname.#1#2
      \loop\unless\ifeof\@inputcheck
        \immediate\readline\@inputcheck to \@tempa
        \immediate\write\@partaux{\@tempa}%
      \repeat
      \immediate\closeout\@partaux
      \immediate\closein\@inputcheck
    \endgroup
  \fi
}
```

\aliastoc Internal use not the real TOC shortcut but another one.

```
\newcommand*{\aliastoc}[2]{%
  \expandafter\edef\csname tocstyle@alias@TOC@#1\endcsname{#2}%
}
```

\tocstyle@pre@starttoc Commands before and after the original \@starttoc.
\tocstyle@post@starttoc

```
\newcommand*{\tocstyle@pre@starttoc}[1]{%
  \begingroup
    \expandafter\ifx\csname tocstyle@deactivated@\endcsname\relax
      \expandafter\ifx\csname tocstyle@deactivated@#1\endcsname\relax\relax
        \tocstyle@activetrue
      \else
        \tocstyle@activefalse
      \fi
    \else
      \tocstyle@activefalse
    \fi
    \iftocstyle@active
      \let\@dottedtocline\tocstyle@dottedtocline
      \parskip \z@
      \parindent \z@
      \parfillskip \z@\@plus 1fil
      \ifcsname tocstyle@alias@TOC@#1\endcsname
        \edef\tocstyleAliasTOC{\csname tocstyle@alias@TOC@#1\endcsname}%
      \else
        \edef\tocstyleAliasTOC{#1}%
      \fi
      \edef\tocstyleTOC{#1}%
      \tocstyle@activate@all@l
    \fi
}
\newcommand*{\tocstyle@post@starttoc}[1]{%
    \iftocstyle@active
      \if@filesw
        \ifcsname tocstyle@unumwidth@#1@\endcsname
          \protected@write\@auxout{}{%
            \protect\tocstyle@set@width{unum}{#1}{}{%
              \csname tocstyle@unumwidth@#1@\endcsname}%
          }%
        \fi
        \expandafter\let\expandafter\@tempa
          \csname tocstyle@depthlist@#1\endcsname
        \ifx\@tempa\relax\else
          \expandafter\@for \expandafter\@tempa\expandafter:\expandafter=\@tempa
          \do {%
            \ifcsname tocstyle@numwidth@#1@\@tempa\endcsname
              \protected@write\@auxout{}{%
                \protect\tocstyle@set@width{num}{#1}{\@tempa}{%
                  \csname tocstyle@numwidth@#1@\@tempa\endcsname}%
```

```
              }%
            \fi
            \ifcsname tocstyle@skipwidth@#1@\@tempa\endcsname
              \protected@write\@auxout{}{%
                \protect\tocstyle@set@width{skip}{#1}{\@tempa}{%
                  \csname tocstyle@skipwidth@#1@\@tempa\endcsname}%
              }%
            \fi
          }%
        \fi
      \fi
    \fi
  \endgroup
}
```

`tocstyle@set@width` Some classes do not use `\numberline`. This may result in negativ widths (esp. negativ skips). Following special handling of negativ values improves the toc handling of the standard classes. Nevertheless indentation of not numbered entries does not work with such classes!

```
\newcommand*{\tocstyle@set@width}[4]{%
  \iftocstyle@indentnotnumbered
    \ifdim #4<\z@
      \expandafter\gdef\csname tocstyle@max#1width@#2@#3\endcsname{%
        \dimexpr #4/2\relax}%
    \else
      \expandafter\gdef\csname tocstyle@max#1width@#2@#3\endcsname{#4}%
    \fi
  \else
    \ifdim #4<\z@
      \expandafter\gdef\csname tocstyle@max#1width@#2@#3\endcsname{\z@}%
    \else
      \expandafter\gdef\csname tocstyle@max#1width@#2@#3\endcsname{#4}%
    \fi
  \fi
}
```

`\tocstyleTOC` Shortcut of the current processed TOC. Empty outside of TOCs.

`\tocstyleAliasTOC`

```
\newcommand*{\tocstyleTOC}{}
\newcommand*{\tocstyleAliasTOC}{}
```

`\tocstyledepth` Current depth of the current processed TOC entry.

```
\newcommand*{\tocstyledepth}{}
```

`\deactivatetocstyle` You may (de)activate all influence of tocstyle either for one or all TOCs.

`\reactivatetocstyle`

```
\newif\iftocstyle@active
\newcommand*{\deactivatetocstyle}[1][]{%
  \expandafter\let\csname tocstyle@deactivated@#1\endcsname\@empty}
\newcommand*{\reactivatetocstyle}[1][]{%
  \expandafter\let\csname tocstyle@deactivated@#1\endcsname\relax}
```

\settocfeature The primary command to set the features of a depth of a TOC.

\@settocfeature
\@@@settocfeature

```
\newcommand*{\@settocfeature}[1][]{%
  \@ifnextchar[ {\@@settocfeature[{#1}]}{\@@settocfeature[{#1}][]}
}
\def\@@settocfeature[#1][#2]#3#4{%
⟨trace⟩  \typeout{exclude: \tocstyle@feature@excludelist}%
  \@expandtwoargs\in@{,#3,}{,\tocstyle@feature@excludelist,}%
  \ifin@\else
    \expandafter\ifcsname tocstyle@feature@#3\endcsname
      \@namedef{tocstyle@feature@#3@#1@#2}{#4}%
      \begingroup
        \expandafter\let\expandafter\@tempa
        \csname tocstyle@commandlist@#1\endcsname
        \@expandtwoargs\in@{,tocstyle@feature@#3@#1@#2,}{,\@tempa,}%
        \ifin@\let\@tempa\endgroup\else
          \edef\@tempa{\endgroup
            \noexpand\expandafter\noexpand\ifx
            \noexpand\csname tocstyle@commandlist@#1\noexpand\endcsname\relax
              \noexpand\expandafter\noexpand\expandafter\noexpand\expandafter
              \noexpand\def
            \noexpand\else
              \noexpand\expandafter\noexpand\expandafter\noexpand\expandafter
              \noexpand\l@addto@macro
            \noexpand\fi
            \noexpand\csname tocstyle@commandlist@#1\noexpand\endcsname%
            {tocstyle@feature@#3@#1@#2,}}%
        \fi
      \@tempa
    \else
      \PackageError{tocstyle}{unkown feature ‘#3’}{%
        You’ve told me to set up toc style feature ‘#3’,\MessageBreak
        but I don’t know this feature.\MessageBreak
        See the tocstyle manual for all known feature.\MessageBreak
      }%
    \fi
  \fi
}
\newcommand*{\settocfeature}{}
\let\settocfeature\@settocfeature
```

\l@addto@macro Something like \g@addto@macro but only with local effect. While other packages or classes may also define this, \providecommand will be used.

```
\providecommand{\l@addto@macro}[2]{%
  \edef#1{\unexpanded\expandafter{#1#2}}%
}%
```

\settocstylefeature Same as above without TOC argument.

\@settocstylefeature

```
\newcommand*{\@settocstylefeature}{%
  \@ifnextchar[ {\@settocfeature[]}{\@settocfeature[][]}%
```

```
}
\newcommand*{\settocstylefeature}{}
\let\settocstylefeature\@settocstylefeature
```

Different commands will be defined:

`ocstyle@feature@!<feature!>@@` Global feature (all TOCs all depths).

`@feature@@!<feature!>@!<TOC!>@` All depth feature for one TOC.

`eature@@!<feature!>@@!<depth!>` All TOCs feature for one depth.

`@!<feature!>@!<TOC!>@!<depth!>` One depth of one TOC feature.

`\tocstyle@activate@features` Activates the features

```
\newcommand*{\tocstyle@activate@features}{%
  \expandafter\ifx\csname tocstyle@depthlist@\tocstyleTOC\endcsname\relax
    \expandafter\xdef\csname tocstyle@depthlist@\tocstyleTOC\endcsname{%
      \tocstyledepth}%
  \else
    \expandafter\let\expandafter\@tempa
    \csname tocstyle@depthlist@\tocstyleTOC\endcsname
    \@expandtwoargs\in@{,\tocstyledepth,}{,\@tempa,}%
    \ifin@\else
      \expandafter\xdef\csname tocstyle@depthlist@\tocstyleTOC\endcsname{%
        \csname tocstyle@depthlist@\tocstyleTOC\endcsname,\tocstyledepth}%
    \fi
  \fi
  \expandafter\@for \expandafter\@tempa
  \expandafter:\expandafter=\tocstyle@featurelist \do
  {%
    \@ifundefined{tocstyle@feature@\@tempa @\tocstyleAliasTOC @\tocstyledepth}{%
      \@ifundefined{tocstyle@feature@\@tempa @@\tocstyledepth}{%
        \@ifundefined{tocstyle@feature@\@tempa @\tocstyleAliasTOC @}{%
          \@ifundefined{tocstyle@feature@\@tempa @@}{%
            \expandafter\let\csname tocstyle@feature@\@tempa\endcsname\relax
          }{%
            \expandafter\let\csname tocstyle@feature@\@tempa
            \expandafter\endcsname
            \csname tocstyle@feature@\@tempa @@\endcsname
          }%
        }{%
          \expandafter\let\csname tocstyle@feature@\@tempa
          \expandafter\endcsname
          \csname tocstyle@feature@\@tempa @\tocstyleAliasTOC @\endcsname
        }%
      }{%
        \expandafter\let\csname tocstyle@feature@\@tempa
        \expandafter\endcsname
        \csname tocstyle@feature@\@tempa @@\tocstyledepth\endcsname
```

```
      }%
    }{%
      \expandafter\let\csname tocstyle@feature@\@tempa
      \expandafter\endcsname
      \csname tocstyle@feature@\@tempa @\tocstyleAliasTOC @\tocstyledepth\endcsna
    }%
  }%
}
```

\newtocstyle Defining a new TOC style. First optional argument is a TOC style, that will be activated before the new definitions. Note that all new definitions will overwirte the parent's definitions. So a new TOC style, that defines all features doesn't need a parent.

```
\newcommand*{\newtocstyle}{%
  \@ifnextchar [{\@newtocstyle}{\@newtocstyle[]}}
\newcommand*{\@newtocstyle}{}
\def\@newtocstyle[#1]{%
  \@ifnextchar [{\@@newtocstyle[{#1}]}{\@newtocstyle[{#1}][]}}
\newcommand*{\@@newtocstyle}{}
\def\@@newtocstyle[#1][#2]#3#4{%
  \@ifundefined{tocstyle@style@#3}{%
    \@ifundefined{tocstyle@style@#1}{%
      \ifx \relax#1\relax\else
        \PackageError{tocstyle}{unknown parent TOC style `#1'}{%
          You've told me to inheritate parent TOC style `#1',\MessageBreak
          but there's no TOC style `#1' defined.}%
      \fi
      \expandafter\def\csname tocstyle@style@#3\endcsname{#4}%
    }{%
      \expandafter\def\csname tocstyle@style@#3\endcsname{%
        \edef\reserved@a{%
          \noexpand\l@addto@macro\noexpand\tocstyle@feature@excludelist{#2}%
          \noexpand\@usetocstyle{#1}%
          \noexpand\def\noexpand\tocstyle@feature@excludelist{%
            \tocstyle@feature@excludelist}%
        }\reserved@a
        #4%
      }%
    }%
  }{%
    \PackageError{tocstyle}{TOC style `#3' already defined}{%
      You've tried to define a new TOC style `#3',\MessageBreak
      but there's already a TOC style named `#3'.}%
  }%
}
\newcommand*{\tocstyle@feature@excludelist}{}
```

\usetocstyle
\@usetocstyle Use the predefined TOC style. You may define `\tocstyle@deprecated@style@foo` to mark TOC style `foo` to be deprecated. If `\tocstyle@deprecated@style@foo`

is `\@empty` TOC style `deprecated@foo` will be used instead almost silently. Otherwise TOC style `\tocstyle@deprecated@style@foo` will be used instead and the user will be told about this change.

```
\newcommand*{\usetocstyle}[2][]{%
  \@ifundefined{tocstyle@deprecated@style@#2}{%
    \@ifundefined{tocstyle@style@#2}{%
      \PackageError{tocstyle}{unknown TOC style `#2'}{%
        You've told me to use TOC style `#2',\MessageBreak
        but there's no TOC style `#2' defined.}%
    }{%
      \def\settocfeature{%
        \@ifnextchar[ {\@@settocfeature[{#1}]}{\@@settocfeature[{#1}][]}%
      }%
      \let\settocstylefeature\settocfeature
```

Deactivate all known features for this TOC

```
      \expandafter\ifx\csname tocstyle@commandlist@#1\endcsname\relax
      \else
        \expandafter\expandafter\expandafter\@for
        \expandafter\expandafter\expandafter\@tempa
        \expandafter\expandafter\expandafter:%
        \expandafter\expandafter\expandafter=%
        \csname tocstyle@commandlist@#1\endcsname
        \do{%
          \expandafter\let\csname \@tempa\endcsname\relax
        }%
```

So there are no more known features for this TOC.

```
        \expandafter\let\csname tocstyle@commandlist@#1\endcsname\relax
      \fi
```

Activate all known features for this style and TOC

```
      \@usetocstyle{#2}%
      \let\settocfeature\@settocfeature
      \let\settocstylefeature\@settocstylefeature
    }%
  }{%
    \expandafter\ifx\csname tocstyle@deprecated@style@#2\endcsname\@empty
      \PackageWarning{tocstyle}{%
        deprecated TOC style `#2'!\MessageBreak
        You should not longer use this style,\MessageBreak
        because it will be removed soon.\MessageBreak
        You should select another TOC style}%
      \usetocstyle[{#1}]{deprecated@#2}%
    \else
      \PackageWarning{tocstyle}{%
        deprecated TOC style `#2'!\MessageBreak
        You should use TOC style '\csname
        tocstyle@deprecated@style@#2\endcsname'\MessageBreak
        instead of `#2'}%
```

```
    \fi
  }%
}
\newcommand*{\@usetocstyle}[1]{%
  \csname tocstyle@style@#1\endcsname
}
```

\tocstyle@featurelist Comma seperated list of all known features

```
\newcommand*{\tocstyle@featurelist}{%
  pagenumberhook,entryhook,dothook,entryvskip,leaders,raggedhook,%
  spaceafternumber,parfillskip,pagenumberbox,%
}
```

\tocstyle@feature@pagenumberhook
\tocstyle@feature@pagenumberhook
\tocstyle@feature@entryhook
\tocstyle@feature@dothook
\tocstyle@feature@entryvskip
\tocstyle@feature@leaders
\tocstyle@feature@parfillskip
\tocstyle@feature@raggedhook
\tocstyle@feature@spaceafternumber

```
\newcommand*{\tocstyle@feature@pagenumberhook}{}
\let\tocstyle@feature@pagenumberhook\relax
\newcommand*{\tocstyle@feature@pagenumberbox}{}
\let\tocstyle@feature@pagenumberbox\relax
\newcommand*{\tocstye@feature@entryhook}{}
\let\tocstyle@feature@entryhook\relax
\newcommand*{\tocstye@feature@dothook}{}
\let\tocstyle@feature@dothook\relax
\newcommand*{\tocstye@feature@entryvskip}{}
\let\tocstyle@feature@entryvskip\relax
\newcommand*{\tocstye@feature@leaders}{}
\let\tocstyle@feature@leaders\relax
\newcommand*{\tocstye@feature@parfillskip}{}
\let\tocstyle@feature@parfillskip\relax
\newcommand*{\tocstye@feature@raggedhook}{}
\let\tocstyle@feature@raggedhook\relax
\newcommand*{\tocstye@feature@spaceafternumber}{}
\let\tocstyle@feature@spaceafternumber\relax
```

\iftochasdepth Uses `\tocstyle@depthlist@`⟨*TOC*⟩ to test, if the TOC has the depth already.

```
\newcommand*{\iftochasdepth}[2]{%
  \begingroup
    \expandafter\let\expandafter\@tempa\csname tocstyle@depthlist@#1\endcsname
    \ifx\@tempa\relax
      \aftergroup\@secondoftwo
    \else
      \@expandtwoargs\in@{,#2,}{,\@tempa}%
      \expandafter\aftergroup\ifin@
        \@firstoftwo
      \else
        \@secondoftwo
      \fi
    \fi
  \endgroup
```

```
}
```

8.2.4. Defining Some TOC Styles

```
\newtocstyle{standard}{%
  \settocfeature{dothook}{\normalfont}%
  \settocfeature[-1]{entryhook}{\bfseries}%
  \settocfeature[-1]{entryvskip}{2.25em\@plus\p@}%
  \settocfeature[-1]{leaders}{\hfill}%
  \settocfeature[0]{entryvskip}{1em\@plus\p@}%
  \settocfeature[0]{leaders}{\hfill}%
  \settocfeature[0]{entryhook}{%
    \begingroup
      \edef\@tempa{toc}%
      \ifx\tocstyleAliasTOC\@tempa\aftergroup\bfseries\fi
    \endgroup
  }%
  \iftochaschapter\else
    \settocfeature[1]{entryvskip}{1em\@plus\p@}%
    \settocfeature[1]{leaders}{\hfill}%
    \settocfeature[1]{entryhook}{%
      \begingroup
        \edef\@tempa{toc}%
        \ifx\tocstyleAliasTOC\@tempa\aftergroup\bfseries\fi
      \endgroup
    }%
  \fi
}
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname sectfont\endcsname\relax
  \newtocstyle{KOMAlike}{%
    \settocfeature{dothook}{\normalfont}%
    \settocfeature[-1]{entryhook}{\sffamily\bfseries}%
    \settocfeature[-1]{entryvskip}{2.25em\@plus\p@}%
    \settocfeature[-1]{leaders}{\hfill}%
    \settocfeature[0]{entryvskip}{1em\@plus\p@}%
    \settocfeature[0]{leaders}{\hfill}%
    \settocfeature[0]{entryhook}{%
      \begingroup
        \edef\@tempa{toc}%
        \ifx\tocstyleAliasTOC\@tempa\aftergroup\sffamily\bfseries\fi
      \endgroup
    }%
    \iftochaschapter\else
      \settocfeature[1]{entryvskip}{1em\@plus\p@}%
      \settocfeature[1]{leaders}{\hfill}%
      \settocfeature[1]{entryhook}{%
        \begingroup
          \edef\@tempa{toc}%
```

```
          \ifx\tocstyleAliasTOC\@tempa\aftergroup\sffamily\bfseries\fi
        \endgroup
      }%
    \fi
  }
\else
  \newtocstyle{KOMAlike}{%
    \settocfeature{dothook}{\normalfont}%
    \settocfeature[-1]{entryhook}{\sectfont}%
    \settocfeature[-1]{entryvskip}{2.25em\@plus\p@}%
    \settocfeature[-1]{leaders}{\hfill}%
    \settocfeature[0]{entryvskip}{1em\@plus\p@}%
    \settocfeature[0]{leaders}{\hfill}%
    \settocfeature[0]{entryhook}{%
      \begingroup
        \edef\@tempa{toc}%
        \ifx\tocstyleAliasTOC\@tempa\aftergroup\sectfont\fi
      \endgroup
    }%
    \iftochaschapter\else
      \settocfeature[1]{entryvskip}{1em\@plus\p@}%
      \settocfeature[1]{leaders}{\hfill}%
      \settocfeature[1]{entryhook}{%
        \begingroup
          \edef\@tempa{toc}%
          \ifx\tocstyleAliasTOC\@tempa\aftergroup\sectfont\fi
        \endgroup
      }%
    \fi
  }
\fi
\newcommand*{\tocstyle@deprecated@style@KOMAScript}{KOMAlike}%
\newtocstyle[KOMAlike]{classic}{%
  \settocfeature{pagenumberhook}{\normalfont\normalcolor}%
  \settocfeature{raggedhook}{\raggedright}%
}
\newtocstyle[classic][leaders]{allwithdot}{}
\newtocstyle[allwithdot]{noonewithdot}{%
  \settocfeature{leaders}{\hfill}%
}
\newtocstyle[classic][leaders]{nopagecolumn}{%
  \settocfeature{leaders}{\quad}%
  \settocfeature{parfillskip}{\z@ plus 1fil}%
  \settocfeature{pagenumberbox}{\hbox}%
}
```

8.2.5. Defining Some TOC Styles

Loading a optional configuration file.

```
\InputIfFileExists{tocstyle.cfg}{%
```

```
  \PackageInfo{tocstyle}{using tocstyle.cfg}%
}{%
  \PackageInfo{tocstyle}{no tocstyle.cfg found}%
}
```

A. Examples for the Different TOC Styles

Here you will find the table of contents of this document set in the different TOC styles. All are set with option `tocindentauto`.

A.1. Graduated Versions

First of all all graduated versions of the table of contents

A.1.1. `standard` with Option `tocgraduated`

A.1.2. `KOMAlike` with Option `tocgraduated`

A.1.3. `classic` with Option `tocgraduated`

A.1.4. `allwithdot` with Option `tocgraduated`

A.1.5. `noonewithdot` with Option `tocgraduated`

A.2. Flat Versions

Now, all flat versions of the table of contents

A.2.1. `standard` with Option `tocflat`

A.2.2. KOMAlike with Option tocflat

A.2.3. `classic` with Option `tocflat`

A.2.4. `allwithdot` with Option `tocflat`

A.2.5. `noonewithdot` with Option `tocflat`

A.3. Fullflat Versions

Now, all full-flat versions of the table of contents

A.3.1. `standard` with Option `tocfullflat`

A.3.2. KOMAlike with Option tocfullflat

A.3.3. `classic` with Option `tocfullflat`

A.3.4. `allwithdot` with Option `tocfullflat`

A.3.5. `noonewithdot` **with Option** `tocfullflat`

A.3.6. `nopagecolumn` with Option `tocfullflat`

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

U

Change History

v0.1

General: start of new package .. 1

v0.2a

\@usetocstyle: extended for deprecated TOC styles ... 25

v0.2d

\tocstyle@dottedtocline: fix: use of max-values 14

v2.4

\l@part: part level is always -1 17

v3.05

chapter: New 12
\iftochaschapter: New 12
nochapter: New 12
tocstyle@set@width: improve handling of standard classes 22