The Implementation of the caption kernel[*]

Axel Sommerfeldt
axel.sommerfeldt@f-m.fm

2011/10/09

Abstract

The caption kernel consists of two parts – the kernel (`caption3.sty`) and the main package (`caption.sty`).

The kernel provides all the user commands and internal macros which are necessary for typesetting captions and setting parameters regarding these. While the standard LaTeX document classes provide an internal command called `\@makecaption` and no options to control its behavior (except the vertical skips above and below the caption itself), we provide similar commands called `\caption@make` and `\caption@@make`, but with a lot of options which can be selected with `\captionsetup`. Loading the kernel part do not change the output of a LaTeX document – it just provides functionality which can be used by LaTeX 2ε packages which typesets captions, for example the caption and subfig packages.

[*] This package has version number v1.4a, last revised 2011/11/01.

Contents

1 Identification

```
\NeedsTeXFormat{LaTeX2e}[1994/12/01]
\ProvidesPackage{caption3}[2011/11/01 v1.4a caption3 kernel (AR)]
```

2 Generic helpers

\@nameundef

This is the opposite to `\@namedef` which is offered by the LaTeX kernel. We use it to remove the definition of some commands and keyval options after `\begin{document}` (to save TeX memory) and to remove caption options defined with `\captionsetup[`⟨*type*⟩`]`.

```
\providecommand*\@nameundef[1]{%
  \expandafter\let\csname #1\endcsname\@undefined}
```

\l@addto@macro

The LaTeX 2_ε kernel offers the internal helper macro `\g@addto@macro` which globally adds tokens to existing macros, like in `\AtBeginDocument`. This is the same but it works local, not global (using `\edef` instead of `\xdef`).

```
\providecommand\l@addto@macro[2]{%
  \begingroup
    \toks@\expandafter{#1#2}%
    \edef\@tempa{\endgroup\def\noexpand#1{\the\toks@}}%
  \@tempa}
```

\bothIfFirst
\bothIfSecond

`\bothIfFirst` tests if the first argument is not empty, `\bothIfSecond` tests if the second argument is not empty. If yes both arguments get typeset, otherwise none of them.

```
\def\bothIfFirst#1#2{%
  \protected@edef\caption@tempa{#1}%
  \ifx\caption@tempa\@empty \else
    #1#2%
  \fi}
```

```
\def\bothIfSecond#1#2{%
  \protected@edef\caption@tempa{#2}%
  \ifx\caption@tempa\@empty \else
    #1#2%
  \fi}
```

\caption@ifundefined

Similar to `\@ifundefined` offered by the LaTeXkernel, but does not define the undefined macro as `\relax`.

```
\newcommand*\caption@ifundefined[1]{%
  \ifx#1\@undefined
    \expandafter\@firstoftwo
  \else\ifx#1\relax
    \expandafter\expandafter\expandafter\@firstoftwo
  \else
    \expandafter\expandafter\expandafter\@secondoftwo
  \fi\fi}
```

\caption@ifinlist

This helper macro checks if the first argument is in the comma separated list which is offered as second argument. So for example

```
\caption@ifinlist{frank}{axel,frank,olga,steven}{yes}{no}
```

would expand to `yes`.

```
\newcommand*\caption@ifinlist{%
  \@expandtwoargs\caption@@ifinlist}
```

```
\newcommand*\caption@@ifinlist[2]{%
  \begingroup
  \def\@tempa##1,#1,##2\@nil{%
    \endgroup
    \ifx\relax##2\relax
      \expandafter\@secondoftwo
    \else
      \expandafter\@firstoftwo
    \fi}%
  \@tempa,#2,#1,\@nil}%
```

\caption@ifin@list `\caption@ifin@list{`⟨*cmd*⟩`}{`⟨*list entry*⟩`}{`⟨*yes*⟩`}{`⟨*no*⟩`}`

```
\newcommand*\caption@ifin@list[2]{%
  \caption@ifempty@list#1%
    {\@secondoftwo}%
    {\@expandtwoargs\caption@@ifinlist{#2}{#1}}}
```

\caption@g@addto@list `\caption@g@addto@list{`⟨*cmd*⟩`}{`⟨*list entry*⟩`}`

```
\newcommand*\caption@g@addto@list[2]{%
  \caption@ifempty@list#1{\gdef#1{#2}}{\g@addto@macro#1{,#2}}}
```

\caption@l@addto@list `\caption@l@addto@list{`⟨*cmd*⟩`}{`⟨*list entry*⟩`}`

```
\newcommand*\caption@l@addto@list[2]{%
  \caption@ifempty@list#1{\def#1{#2}}{\l@addto@macro#1{,#2}}}
```

caption@g@removefrom@list `\caption@g@removefrom@list{`⟨*cmd*⟩`}{`⟨*list entry*⟩`}`

```
\newcommand*\caption@g@removefrom@list[2]{%
  \caption@l@removefrom@list#1{#2}%
  \global\let#1#1}
```

caption@l@removefrom@list `\caption@l@removefrom@list{`⟨*cmd*⟩`}{`⟨*list entry*⟩`}`

Caveat: ⟨*cmd*⟩ will be expanded during this process since `\@removeelement` is using `\edef` to build the new list!

```
\newcommand*\caption@l@removefrom@list[2]{%
  \caption@ifempty@list#1{}{\@expandtwoargs\@removeelement{#2}#1#1}}
```

\caption@for@list `\caption@for@list{`⟨*code with #1*⟩`}`

```
\newcommand*\caption@for@list[2]{%
  \caption@ifempty@list#1{}{%
    \def\caption@tempb##1{#2}%
    \@for\caption@tempa:=#1\do{%
      \expandafter\caption@tempb\expandafter{\caption@tempa}}}}
```

\caption@ifempty@list `\caption@ifempty@list{`⟨*cmd*⟩`}{`⟨*true*⟩`}{`⟨*false*⟩`}`

```
\newcommand*\caption@ifempty@list[1]{%
  \ifx#1\@undefined
    \expandafter\@firstoftwo
  \else\ifx#1\relax
    \expandafter\expandafter\expandafter\@firstoftwo
  \else\ifx#1\@empty
    \expandafter\expandafter\expandafter\expandafter
      \expandafter\expandafter\expandafter\@firstoftwo
  \else
    \expandafter\expandafter\expandafter\expandafter
      \expandafter\expandafter\expandafter\@secondoftwo
  \fi\fi\fi}
```

\caption@setbool
\caption@set@bool
\caption@ifbool
\caption@undefbool

For setting and testing boolean options we offer these three helper macros:

```
\caption@setbool{⟨name⟩}{⟨value⟩}
                  (with value = false/true/no/yes/off/on/0/1)
\caption@ifbool{⟨name⟩}{⟨if-clause⟩}{⟨else-clause⟩}
\caption@undefbool{⟨name⟩}
```

```
\newcommand*\caption@setbool[1]{%
  \expandafter\caption@set@bool\csname caption@if#1\endcsname}
```

```
\newcommand*\caption@set@bool[2]{%
  \caption@ifinlist{#2}{1,true,yes,on}{%
    \let#1\@firstoftwo
  }{\caption@ifinlist{#2}{0,false,no,off}{%
    \let#1\@secondoftwo
  }{%
    \caption@Error{Undefined boolean value `#2'}%
  }}}
```

```
\newcommand*\caption@ifbool[1]{\@nameuse{caption@if#1}}
```

```
\newcommand*\caption@undefbool[1]{\@nameundef{caption@if#1}}
```

\caption@teststar

```
\caption@teststar{⟨cmd⟩}{⟨star arg⟩}{⟨non-star arg⟩}
\caption@teststar@{⟨cmd⟩}{⟨star arg⟩}{⟨non-star arg⟩}
```

```
\newcommand*\caption@teststar[3]{\@ifstar{#1{#2}}{#1{#3}}}
```

```
\newcommand*\caption@teststar@[3]{%
  \@ifstar{#1{#2}}{\caption@ifatletter{#1{#2}}{#1{#3}}}}
\AtBeginDocument{\let\caption@teststar@\caption@teststar}
```

```
\newcommand*\caption@ifatletter{%
  \ifnum\the\catcode`\@=11
    \expandafter\@firstoftwo
  \else
    \expandafter\@secondoftwo
  \fi}
\AtBeginDocument{\let\caption@ifatletter\@secondoftwo}
```

\caption@withoptargs

```
\caption@withoptargs{⟨cmd⟩}
```

```
\newcommand*\caption@withoptargs[1]{%
  \@ifstar
    {\def\caption@tempa{*}\caption@@withoptargs#1}%
    {\def\caption@tempa{}\caption@@withoptargs#1}}
```

```
\def\caption@@withoptargs#1{%
  \@ifnextchar[%]
    {\caption@@@withoptargs#1}%
    {\caption@@@@withoptargs#1}}
```

```
\def\caption@@@withoptargs#1[#2]{%
  \l@addto@macro\caption@tempa{[{#2}]}%
  \caption@@withoptargs#1}
```

```
\def\caption@@@@withoptargs#1{%
  \expandafter#1\expandafter{\caption@tempa}}
```

\caption@gobble

\caption@gobble*[⟨*arg*⟩][⟨...⟩]{⟨*arg*⟩}
Same as \@gobble, but gobbles optional arguments as well.

```
\DeclareRobustCommand*\caption@gobble{%
  \caption@withoptargs\@gobbletwo}
```

\caption@CheckCommand
\caption@IfCheckCommand

\caption@CheckCommand{⟨*macro*⟩}{⟨*definition of macro*⟩}
checks if a command already exists, with the same definition. It can be used more-than-once to check if one of multiple definitions will finally match. (It redefines itself later on to \@gobbletwo if the two commands match fine, making further checks harmless.)
\caption@IfCheckCommand{⟨*true*⟩}{⟨*false*⟩}
will execute the ⟨*true*⟩ code if one match was finally given, the ⟨*false*⟩ code otherwise. (It simply checks if \caption@CheckCommand is \@gobbletwo and restores the starting definition of \caption@CheckCommand.)

```
\newcommand\caption@DoCheckCommand[2]{%
  \begingroup
    \let\@tempa#1%
    #2%
    \ifx\@tempa#1%
      \endgroup
      \let\caption@CheckCommand\@gobbletwo
    \else
      \endgroup
    \fi}
\@onlypreamble\caption@DoCheckCommand
```

```
\let\caption@CheckCommand\caption@DoCheckCommand
\@onlypreamble\caption@CheckCommand
```

```
\newcommand*\caption@IfCheckCommand{%
  \ifx\caption@CheckCommand\@gobbletwo
    \let\caption@CheckCommand\caption@DoCheckCommand
    \expandafter\@firstoftwo
  \else
    \expandafter\@secondoftwo
  \fi}
\@onlypreamble\caption@IfCheckCommand
```

\caption@AtBeginDocument

\caption@AtBeginDocument*{⟨*code*⟩}
Same as \AtBeginDocument but the execution of code will be surrounded by two \PackageInfos. The starred variant causes the code to be executed after all code specified using the non-starred variant.

```
\let\caption@begindocumenthook\@empty
\let\caption@@begindocumenthook\@empty
```

```
\def\caption@AtBeginDocument{%
  \caption@teststar\g@addto@macro
    \caption@@begindocumenthook\caption@begindocumenthook}
%\@onlypreamble\caption@AtBeginDocument
```

```
\AtBeginDocument{%
   \caption@InfoNoLine{Begin \noexpand\AtBeginDocument code}%
```

```
  \def\caption@AtBeginDocument{%
    \@ifstar{\g@addto@macro\caption@@begindocumenthook}\@firstofone}%
  \caption@begindocumenthook
  \let\caption@begindocumenthook\relax

  \def\caption@AtBeginDocument{%
    \@ifstar\@firstofone\@firstofone}%
  \caption@@begindocumenthook
  \let\caption@@begindocumenthook\relax

  \caption@InfoNoLine{End \noexpand\AtBeginDocument code}}
```

3 Information, Warnings, and Errors

`\caption@Info` `\caption@Info{⟨message⟩}`

```
\newcommand*\caption@Info[1]{%
  \PackageInfo{caption}{#1}}
```

`\caption@InfoNoLine` `\caption@InfoNoLine{⟨message⟩}`

Note: The `\@gobble` at the end of the 2nd argument of `\PackageInfo` suppresses the line number info. See TLC2[**?**], A.4.7, p885 for details.

```
\newcommand*\caption@InfoNoLine[1]{%
  \PackageInfo{caption}{#1\@gobble}}
```

`\caption@Warning` `\caption@Warning{⟨message⟩}`

```
\newcommand*\caption@Warning[1]{%
  \caption@WarningNoLine{#1\on@line}}
```

`\caption@WarningNoLine` `\caption@WarningNoLine{⟨message⟩}`

```
\newcommand*\caption@WarningNoLine[1]{%
  \PackageWarning{caption}{#1.^^J\caption@wh\@gobbletwo}}

\newcommand*\caption@wh{%
  See the caption package documentation for explanation.}
```

`\caption@Error` `\caption@Error{⟨message⟩}`

```
\newcommand*\caption@Error[1]{%
  \PackageError{caption}{#1}\caption@eh}

\newcommand*\caption@eh{%
  If you do not understand this error, please take a closer look\MessageBreak
  at the documentation of the ‘caption’ package, especially the\MessageBreak
  section about errors.\MessageBreak\@ehc}
```

`\caption@KV@err`

```
\let\caption@KV@err\caption@Error
```

4 Using the keyval package

We need the keyval package for option handling, so we load it here.

```
\RequirePackage{keyval}[1997/11/10]
```

\undefine@key

\undefine@key{⟨*family*⟩}{⟨*key*⟩}
This helper macro is the opposite of \define@key, it removes a keyval definition.

```
\providecommand*\undefine@key[2]{%
  \@nameundef{KV@#1@#2}\@nameundef{KV@#1@#2@default}}
```

\@onlypreamble@key

\onlypreamble@key{⟨*family*⟩}{⟨*key*⟩}
Analogous to \@onlypreamble from LaTeX 2ε.

```
\providecommand*\@preamble@keys{}
\providecommand*\@onlypreamble@key[2]{\@cons\@preamble@keys{{#1}{#2}}}
\@onlypreamble\@onlypreamble@key
\@onlypreamble\@preamble@keys
```

```
\providecommand*\@notprerr@key[1]{\KV@err{Can be used only in preamble}}
```

```
\caption@AtBeginDocument*{%
  \def\@elt#1#2{\expandafter\let\csname KV@#1@#2\endcsname\@notprerr@key}%
  \@preamble@keys
  \let\@elt\relax}
```

\DeclareCaptionOption

\DeclareCaptionOption{⟨*option*⟩}[⟨*default value*⟩]{⟨*code*⟩}
\DeclareCaptionOption*{⟨*option*⟩}[⟨*default value*⟩]{⟨*code*⟩}
We declare our options using these commands (instead of using \DeclareOption offered by LaTeX 2ε), so the keyval package is used. The starred form makes the option available during the lifetime of the current package only, so they can be used with \usepackage, but *not* with \captionsetup later on.

```
\newcommand*\DeclareCaptionOption{%
  \caption@teststar\caption@declareoption\AtEndOfPackage\@gobble}
\@onlypreamble\DeclareCaptionOption
```

```
\newcommand*\caption@declareoption[2]{%
  #1{\undefine@key{caption}{#2}}\define@key{caption}{#2}}
\@onlypreamble\caption@declareoption
```

clareCaptionOptionNoValue

\DeclareCaptionOptionNoValue{⟨*option*⟩}{⟨*code*⟩}
\DeclareCaptionOptionNoValue*{⟨*option*⟩}{⟨*code*⟩}
Same as \DeclareCaptionOption but issues an error if a value is given.

```
\newcommand*\DeclareCaptionOptionNoValue{%
  \caption@teststar\caption@declareoption@novalue\AtEndOfPackage\@gobble}
\@onlypreamble\DeclareCaptionOptionNoValue
```

```
\newcommand\caption@declareoption@novalue[3]{%
  \caption@declareoption{#1}{#2}[\KV@err]{%
    \caption@option@novalue{#2}{##1}{#3}}}
\@onlypreamble\caption@declareoption@novalue
```

```
\newcommand*\caption@option@novalue[2]{%
  \ifx\KV@err#2%
    \expandafter\@firstofone
  \else
    \KV@err{No value allowed for #1}%
    \expandafter\@gobble
  \fi}
```

\ifcaptionsetup@star

If the starred form of \captionsetup is used, this will be set to true. (It will be reset to false at the end of \caption@setkeys.)

```
\newif\ifcaptionsetup@star
```

\captionsetup

\captionsetup[⟨*type*⟩]{⟨*keyval-list of options*⟩}
\captionsetup*[⟨*type*⟩]{⟨*keyval-list of options*⟩}
If the optional argument 'type' is specified, we simply save or append the option list, otherwise we 'execute' it with \setkeys. (The non-starred variant issues a warning if ⟨*keyval-list of options*⟩ is not used later on.)
Note: The starred variant will be used inside packages automatically.

```
\newcommand*\captionsetup{%
  \caption@teststar@\@captionsetup\@gobble\@firstofone}
```

```
\newcommand*\@captionsetup[1]{%
  \captionsetup@startrue#1\captionsetup@starfalse
  \@ifnextchar[\caption@setup@options\caption@setup}
```

```
\newcommand*\caption@setup{\caption@setkeys{caption}}
```

```
\def\caption@setup@options[#1]#2{%
  \@bsphack
    \ifcaptionsetup@star\captionsetup@starfalse\else\caption@addtooptlist{#1}\fi
    \expandafter\caption@l@addto@list\csname caption@opt@#1\endcsname{#2}%
  \@esphack}
```

\clearcaptionsetup

\clearcaptionsetup[⟨*option*⟩]{⟨*type*⟩}
\clearcaptionsetup*[⟨*option*⟩]{⟨*type*⟩}
This removes the saved option list associated with ⟨*type*⟩. If ⟨*option*⟩ is given, only this option will be removed from the list. (The starred variant does not issue warnings.)
Note: The starred variant will be used inside packages automatically.

```
\newcommand*\clearcaptionsetup{%
  \caption@teststar@\@clearcaptionsetup\@gobble\@firstofone}
```

```
\newcommand*\@clearcaptionsetup[1]{%
  \let\caption@tempa#1%
  \@testopt\@@clearcaptionsetup{}}
```

```
\def\@@clearcaptionsetup[#1]#2{%
  \@bsphack
    \expandafter\caption@ifempty@list\csname caption@opt@#2\endcsname
      {\caption@tempa{\caption@Warning{Option list `#2' undefined}}}%
      {\ifx,#1,%
         \caption@clearsetup{#2}%
       \else
         \caption@@removefromsetup{#1}{#2}%
       \fi}%
  \@esphack}
```

```
\newcommand*\caption@clearsetup[1]{%
  \caption@removefromoptlist{#1}%
  \@nameundef{caption@opt@#1}}
```

```
\newcommand*\caption@removefromsetup{%
  \let\caption@tempa\@gobble
  \caption@@removefromsetup}
```

```
\newcommand*\caption@@removefromsetup[2]{%
  \expandafter\let\expandafter\@tempa\csname caption@opt@#2\endcsname
  \expandafter\let\csname caption@opt@#2\endcsname\@undefined
  \def\@tempb##1=##2\@nil{##1}%
  \edef\@tempc{#1}%
  \@for\@tempa:=\@tempa\do{%
    \edef\@tempd{\expandafter\@tempb\@tempa=\@nil}%
    \ifx\@tempd\@tempc
      \let\caption@tempa\@gobble
    \else
      \expandafter\expandafter\expandafter\caption@l@addto@list
        \expandafter\csname caption@opt@#2\expandafter\endcsname
        \expandafter{\@tempa}%
    \fi}%
  \expandafter\caption@ifempty@list\csname caption@opt@#2\endcsname
    {\caption@removefromoptlist{#2}}{}%
  \caption@tempa{\caption@Warning{%
    Option `#1' was not in list `#2'\MessageBreak}}}
```

\showcaptionsetup

\showcaptionsetup[⟨*package*⟩]{⟨*type*⟩}

This comes for debugging issues: It shows the saved option list which is associated with ⟨*type*⟩.

```
\newcommand*\showcaptionsetup[2][\@firstofone]{%
  \@bsphack
    \GenericWarning{}{%
      #1 Caption Info: Option list on `#2'\MessageBreak
      #1 Caption Data: \@ifundefined{caption@opt@#2}{%
        -none-%
      }{%
        {\expandafter\expandafter\expandafter\strip@prefix
           \expandafter\meaning\csname caption@opt@#2\endcsname}%
      }}%
  \@esphack}
```

```
\DeclareCaptionOption{options}{\caption@setoptions{#1}}
\DeclareCaptionOption{options*}{\caption@setoptions*{#1}}
```

\caption@setoptions

\caption@setoptions*{⟨*type or environment or...*⟩}

Caption options which have been saved with \captionsetup[⟨*type*⟩] can be executed by using this command. It simply executes the saved option list (and clears it afterwards), if there is any. (The starred variant do not clear the option list.)

```
\newcommand*\caption@setoptions{%
  \caption@teststar\caption@set@options\@gobble\@firstofone}
```

```
\newcommand*\caption@set@options[2]{%
  \caption@Debug{options=#2}%
  \expandafter\let\expandafter\caption@opt\csname caption@opt@#2\endcsname
  \ifx\caption@opt\relax \else
    \caption@xsetup\caption@opt
    #1{\caption@clearsetup{#2}}% #1 = \@firstofone -or- \@gobble
  \fi}
```

```
\newcommand*\caption@xsetup[1]{\expandafter\caption@setup\expandafter{#1}}
```

\caption@addtooptlist
caption@removefromoptlist

\caption@addtooptlist{⟨*type*⟩}
\caption@removefromoptlist{⟨*type*⟩}

Adds or removes an ⟨*type*⟩ to the list of unused caption options. Note that the catcodes of ⟨*type*⟩ are sanitized here so removing ⟨*type*⟩ from the list do not fail when the float package is used (since \float@getstyle gives a result which tokens have catcode 12 = “other”).

```
\newcommand*\caption@addtooptlist[1]{%
  \@ifundefined{caption@opt@#1@lineno}{%
    \caption@dooptlist\caption@g@addto@list{#1}%
    \expandafter\xdef\csname caption@opt@#1@lineno\endcsname{\the\inputlineno}%
  }{}}
```

```
\newcommand*\caption@removefromoptlist[1]{%
  \caption@dooptlist\caption@g@removefrom@list{#1}%
  \global\expandafter\let\csname caption@opt@#1@lineno\endcsname\@undefined}
```

```
\newcommand*\caption@dooptlist[2]{%
  \begingroup
    \edef\@tempa{#2}\@onelevel@sanitize\@tempa
    \expandafter#1\expandafter\caption@optlist\expandafter{\@tempa}%
  \endgroup}
```

```
\AtEndDocument{%
  \caption@for@list\caption@optlist{%
    \caption@WarningNoLine{%
      Unused \string\captionsetup[#1]
      on input line \csname caption@opt@#1@lineno\endcsname}}}
```

`\caption@setkeys`

`\caption@setkeys[`⟨*package*⟩`]{`⟨*family*⟩`}{`⟨*key-values*⟩`}`
This one simply calls `\setkeys{`⟨*family*⟩`}{`⟨*key-values*⟩`}` but lets the error messages not refer to the keyval package, but to the ⟨*package*⟩ package instead.

```
\newcommand*\caption@setkeys{\@dblarg\caption@@setkeys}
```

```
\long\def\caption@@setkeys[#1]#2#3{%
  \@bsphack

  \expandafter\let\csname ORI@KV@err\caption@keydepth\endcsname\KV@err
  \expandafter\let\csname ORI@KV@errx\caption@keydepth\endcsname\KV@errx
  \expandafter\let\csname ORI@XKV@err\caption@keydepth\endcsname\XKV@err
  \edef\caption@keydepth{\caption@keydepth i}%

  \expandafter\let\expandafter\KV@err\csname #1@KV@err\endcsname
  \ifx\KV@err\relax
    \def\KV@err##1{\PackageError{#1}{##1}{%
      See the #1 package documentation for explanation.}}%
  \fi
  \def\KV@errx{\KV@err}%
  \def\XKV@err{\let\@tempa\XKV@tkey\KV@err}%

  \caption@Debug{\protect\setkeys{#2}{#3}}%
  \setkeys{#2}{#3}%

  \edef\caption@keydepth{\expandafter\@gobble\caption@keydepth}%
  \expandafter\let\expandafter\KV@err\csname ORI@KV@err\caption@keydepth\endcsnam
  \expandafter\let\expandafter\KV@errx\csname ORI@KV@errx\caption@keydepth\endcsn
  \expandafter\let\expandafter\XKV@err\csname ORI@XKV@err\caption@keydepth\endcsn

  \ifx\caption@keydepth\@empty \captionsetup@starfalse \fi

  \@esphack}
```

```
\let\caption@keydepth\@empty
```

`\caption@ExecuteOptions`

`\caption@ExecuteOptions{`⟨*package*⟩`}{`⟨*key-values*⟩`}`
We execute our options using the keyval interface, so we use this one instead of `\ExecuteOptions` offered by LaTeX 2ε.

```
\newcommand*\caption@ExecuteOptions[2]{%
  \expandafter\@expandtwoargs\csname caption@setkeys@#1\endcsname{#1}{#2}}%
\@onlypreamble\caption@ExecuteOptions
```

\caption@ProcessOptions

\caption@ProcessOptions*{⟨*package*⟩}
We process our options using the keyval package, so we use this one instead of \ProcessOptions offered by LaTeX 2ε. The starred variant do not process the global options. (This code was taken from the hyperref package[2] *v6.74* and improved.)

```
\newcommand*\caption@ProcessOptions{%
  \caption@teststar\caption@@ProcessOptions\@gobble\@firstofone}
\@onlypreamble\caption@ProcessOptions
```

```
\newcommand\caption@@ProcessOptions[2]{%
  \let\@tempc\relax
  \let\caption@tempa\@empty
  #1{% \@firstofone -or- \@gobble
    \@for\CurrentOption:=\@classoptionslist\do{%
      \@ifundefined{KV@#2@\CurrentOption}{}{%
        \@ifundefined{KV@#2@\CurrentOption @default}{%
          \PackageInfo{#2}{Global option `\CurrentOption' ignored}%
        }{%
          \PackageInfo{#2}{Global option `\CurrentOption' processed}%
          \edef\caption@tempa{\caption@tempa\CurrentOption,}%
          \@expandtwoargs\@removeelement\CurrentOption
            \@unusedoptionlist\@unusedoptionlist
        }%
      }%
    }%
    \let\CurrentOption\@empty
  }%
  \caption@ExecuteOptions{#2}{\caption@tempa\@ptionlist{\@currname.\@currext}}%
  \AtEndOfPackage{\let\@unprocessedoptions\relax}}
\@onlypreamble\caption@@ProcessOptions
```

\caption@SetupOptions

\caption@SetupOptions{⟨*package*⟩}{⟨*code*⟩}
After calling this macro \caption@ExecuteOptions and \usepackage[⟨*options*⟩]{⟨*package*⟩} will both be mapped to ⟨*code*⟩ with ⟨*package*⟩ and ⟨*options*⟩ as arguments #1 and #2.
(This helps avoiding "Option clash" errors.)

```
\newcommand*\caption@packagelist{}
\@onlypreamble\caption@packagelist
```

```
\newcommand\caption@SetupOptions[2]{%
  \@namedef{caption@setkeys@#1}##1##2{#2}%
  \expandafter\@onlypreamble\csname caption@setkeys@#1\endcsname
  \@cons\caption@packagelist{{#1}}}
\@onlypreamble\caption@SetupOptions
```

```
\let\caption@onefilewithoptions\@onefilewithoptions
\def\@onefilewithoptions#1[#2]{%
  \begingroup
  \def\@tempa{%
    \endgroup
    \caption@onefilewithoptions{#1}[{#2}]}%
  \def\@tempb{#1}%
  \def\@elt##1{%
```

```
    \def\@tempc{##1}%
    \ifx\@tempb\@tempc
      \def\@tempa{%
        \endgroup
        \caption@ExecuteOptions{#1}{#2}%
        \caption@onefilewithoptions{#1}[]}%
    \fi}
  \caption@packagelist
  \@tempa}
\@onlypreamble\caption@onefilewithoptions
```

5 Margin resp. width

\captionmargin
\captionwidth

`\captionmargin` and `\captionwidth` contain the extra margin resp. the total width used for captions. Please never set these values in a direct way, they are just accessible in user documents to provide compatibility to *v1.x*.
Note that we can only set one value at a time, 'margin' *or* 'width'. If `\captionwidth` is not zero we will take this value afterwards, otherwise `\captionmargin` and `\captionmargin@`.

```
\newdimen\captionmargin
\newdimen\captionmargin@
\newdimen\captionwidth
```

```
\DeclareCaptionOption{margin}{\setcaptionmargin{#1}}
\DeclareCaptionOption{margin*}{\setcaptionmargin*{#1}}
\DeclareCaptionOption{width}{\setcaptionwidth{#1}}
\DeclareCaptionOption{width*}{\setcaptionwidth*{#1}}
```

```
\DeclareCaptionOption{calcmargin}{\caption@setcalcmargin{#1}}
\DeclareCaptionOption{calcmargin*}{\caption@setcalcmargin*{#1}}
\DeclareCaptionOption{calcwidth}{\caption@setcalcwidth{#1}}
\DeclareCaptionOption{calcwidth*}{\caption@setcalcwidth*{#1}}
```

```
\DeclareCaptionOption{twoside}[1]{\caption@set@bool\caption@iftwoside{#1}}
\DeclareCaptionOptionNoValue{oneside}{\caption@set@bool\caption@iftwoside0}
```

```
\DeclareCaptionOption{minmargin}{\caption@setoptcmd\caption@minmargin{#1}}
\DeclareCaptionOption{maxmargin}{\caption@setoptcmd\caption@maxmargin{#1}}
```

\setcaptionmargin

`\setcaptionmargin{`⟨*amount*⟩`}`
`\setcaptionmargin*{`⟨*amount*⟩`}`
Please never use this in user documents, it's just there to provide compatibility to the caption2 package.

```
\newcommand*\setcaptionmargin{%
  \caption@resetcalcmargin
  \caption@setmargin}
```

```
\newcommand*\caption@setmargin{%
  \caption@teststar\caption@@setmargin\@gobble\@firstofone}
```

```
\newcommand*\caption@@setmargin[2]{%
  #1{\captionwidth\z@}%
  \caption@@@setmargin#2,#2,\@nil}
```

```
\def\caption@@@setmargin#1,#2,#3\@nil{%
  \setlength\captionmargin@{#2}%
  \setlength\captionmargin{#1}%
  \addtolength\captionmargin@{-\captionmargin}}
```

\setcaptionwidth

\setcaptionwidth{⟨*amount*⟩}
\setcaptionwidth*{⟨*amount*⟩}

Please never use this in user documents, it's just there to provide compatibility to the caption2 package.

```
\newcommand*\setcaptionwidth{%
  \caption@resetcalcmargin
  \caption@setwidth}
```

```
\newcommand*\caption@setwidth{%
  \caption@teststar\caption@@setwidth\@gobble\@firstofone}
```

```
\newcommand*\caption@@setwidth[2]{%
  #1{\captionmargin\z@\captionmargin@\z@}%
  \setlength\captionwidth{#2}}%
```

\caption@resetcalcmargin

```
\newcommand*\caption@resetcalcmargin{%
  \let\caption@calcmargin@hook\@empty}
```

\caption@setcalcmargin

```
\newcommand*\caption@setcalcmargin{%
  \caption@teststar{\caption@@setcalcmargin\caption@setmargin}%
    \@secondoftwo\@firstoftwo}
```

```
\newcommand*\caption@@setcalcmargin[3]{%
  #2{\caption@resetcalcmargin
     \l@addto@macro\caption@calcmargin@hook{#1{#3}}}%
    {\l@addto@macro\caption@calcmargin@hook{#1*{#3}}}}
```

\caption@setcalcwidth

```
\newcommand*\caption@setcalcwidth{%
  \caption@teststar{\caption@@setcalcmargin\caption@setwidth}%
    \@secondoftwo\@firstoftwo}
```

\caption@counter

This counter numbers the captions. At the moment it will be used inside \caption@ifoddpage only.

```
\newcommand*\caption@thecounter{0}
```

```
\newcommand*\caption@stepcounter{%
  \@tempcnta\caption@thecounter
  \advance\@tempcnta\@ne
  \xdef\caption@thecounter{\the\@tempcnta}}
```

\caption@newlabel

This command is a modified version of \newlabel from LaTeX2e. It will be written to the .aux file to pass label information from one run to another. (We use it inside \caption@ifoddpage and \caption@ragged.)

```
\newcommand*\caption@newlabel{\@newl@bel{caption@r}}
```

\caption@thepage

This command is a modified version of \thepage from LaTeX2e. It will be used inside \caption@ifoddpage only.

```
\newcommand*\caption@thepage{\the\c@page}
```

`\caption@label`

This command is a modified version of `\label` from LaTeX2e. It will be used inside `\caption@ifoddpage` and `\FP@helpNote`.

```
\newcommand*\caption@label[1]{%
  \caption@@label
  \protected@write\@auxout{\let\caption@thepage\relax}%
        {\string\caption@newlabel{#1}{\caption@thepage}}}
```

```
\newcommand*\caption@@label{%
  \global\let\caption@@label\relax
  \protected@write\@auxout{}%
    {\string\providecommand*\string\caption@newlabel[2]{}}}
```

`\caption@pageref`

This command is a modified version of `\pageref` from LaTeX2e. It will be used inside `\caption@ifoddpage` and `\FP@helpNote`.

```
\newcommand*\caption@pageref[1]{%
  \expandafter\ifx\csname caption@r@#1\endcsname\relax
    \G@refundefinedtrue % => 'There are undefined references.'
    \@latex@warning{Reference `#1' on page \thepage \space undefined}%
  \else
    \expandafter\let\expandafter\caption@thepage\csname caption@r@#1\endcsname
  \fi}
```

`\caption@ifoddpage`

At the moment this macro uses an own label...ref mechanism, but an alternative implementation method would be using the refcount package[3] and `\ifodd\getpagerefnumber{...}`.
Note: This macro re-defines itself so the `.aux` file will only be used once per group.

```
\newcommand*\caption@ifoddpage{%
  \caption@iftwoside{%
    \caption@label\caption@thecounter
    \caption@pageref\caption@thecounter
    \ifodd\caption@thepage
      \let\caption@ifoddpage\@firstoftwo
    \else
      \let\caption@ifoddpage\@secondoftwo
    \fi
  }{\let\caption@ifoddpage\@firstoftwo}%
```

```
  \caption@ifoddpage}
```

`\caption@setoptcmd`

`\caption@setoptcmd{`⟨*cmd*⟩`}{`⟨*off -or- value*⟩`}`

```
\newcommand*\caption@setoptcmd[2]{%
  \caption@ifinlist{#2}{0,false,no,off}{\let#1\@undefined}{\def#1{#2}}}
```

6 Indentions

`\caption@indent`
`\caption@parindent`
`\caption@hangindent`

These are the indentions we support.

```
\newdimen\caption@indent
\newdimen\caption@parindent
\newdimen\caption@hangindent
```

```
\DeclareCaptionOption{indent}[\leftmargini]{% obsolete!
      \setlength\caption@indent{#1}}
\DeclareCaptionOption{indention}[\leftmargini]{%
      \setlength\caption@indent{#1}}
```

```
\DeclareCaptionOption{parindent}{%
      \setlength\caption@parindent{#1}}
\DeclareCaptionOption{hangindent}{%
      \setlength\caption@hangindent{#1}}
\DeclareCaptionOption{parskip}{%
      \l@addto@macro\caption@@par{\setlength\parskip{#1}}}
```

There is an option clash between the KOMA-Script document classes and the caption kernel, both define the options `parindent` and `parskip` but with different meaning. Furthermore the ones defined by the caption kernel take a value as parameter but the KOMA-Script ones do not. So we need special versions of the options `parindent` and `parskip` here which determine if a value is given (and therefore should be treated as our option) or not (and therefore should be ignored by us).[1]

```
\providecommand*\caption@ifkomaclass{%
  \caption@ifundefined\scr@caption\@gobble\@firstofone}
\@onlypreamble\caption@ifkomaclass
```

```
\caption@ifkomaclass{%
```

```
  \let\caption@KV@parindent\KV@caption@parindent
  \DeclareCaptionOption{parindent}[]{%
    \ifx,#1,%
      \caption@Debug{Option `parindent' ignored}%
    \else
      \caption@KV@parindent{#1}%
    \fi}%
```

```
  \let\caption@KV@parskip\KV@caption@parskip
  \DeclareCaptionOption{parskip}[]{%
    \ifx,#1,%
      \caption@Debug{Option `parskip' ignored}%
    \else
      \caption@KV@parskip{#1}%
    \fi}%
```

```
}
```

7 Styles

`\DeclareCaptionStyle`

`\DeclareCaptionStyle{`⟨*name*⟩`}[`⟨*single-line-list-of-KV*⟩`]{`⟨*list-of-KV*⟩`}`

```
\newcommand*\DeclareCaptionStyle[1]{%
  \@testopt{\caption@declarestyle{#1}}{}}
\@onlypreamble\DeclareCaptionStyle
```

```
\def\caption@declarestyle#1[#2]#3{%
  \global\@namedef{caption@sls@#1}{#2}%
  \global\@namedef{caption@sty@#1}{#3}}
\@onlypreamble\caption@declarestyle
```

```
\DeclareCaptionOption{style}{\caption@setstyle{#1}}
\DeclareCaptionOption{style*}{\caption@setstyle*{#1}}
\DeclareCaptionOption{singlelinecheck}[1]{\caption@set@bool\caption@ifslc{#1}}
\DeclareCaptionOption{slc}[1]{\KV@caption@singlelinecheck{#1}}
```

[1] This problem was completely solved due a change of `\caption@ProcessOptions` in the caption kernel *v1.0h*, but we still need this workaround since these options would otherwise still collide with the current version 1.3 of the subfig package (Sigh!)

`\caption@setstyle`

`\caption@setstyle{`⟨*name*⟩`}`
`\caption@setstyle*{`⟨*name*⟩`}`
Selecting a caption style means saving the additional ⟨*single-line-list-of-KV*⟩ (this will be done by `\caption@sls`), resetting the caption options to the base ones (this will be done using `\caption@resetstyle`) and executing the ⟨*list-of-KV*⟩ options (this will be done using `\caption@setup`).
The starred version will give no error message if the given style is not defined.

```
\newcommand*\caption@setstyle{%
  \caption@teststar\caption@@setstyle\@gobble\@firstofone}
```

```
\newcommand*\caption@@setstyle[2]{%
  \@ifundefined{caption@sty@#2}%
    {#1{\caption@Error{Undefined style ‘#2’}}}%
    {\expandafter\let\expandafter\caption@sty\csname caption@sty@#2\endcsname
     \ifx\caption@setstyle@flag\@undefined
       \let\caption@setstyle@flag\relax
       \caption@resetstyle
       \caption@xsetup\caption@sty
       \let\caption@setstyle@flag\@undefined
     \else
       \caption@xsetup\caption@sty
     \fi
     \expandafter\let\expandafter\caption@sls\csname caption@sls@#2\endcsname
     \expandafter\caption@l@addto@list\expandafter\caption@opt@singleline
       \expandafter{\caption@sls}}}
```

`\caption@resetstyle`

This resets (nearly) all caption options to the base ones. *Note that this does not touch the skips and the positioning!*

```
\newcommand*\caption@resetstyle{%
  \caption@setup{%
    format=plain,labelformat=default,labelsep=colon,textformat=simple,%
    justification=justified,font=,size=,labelfont=,textfont=,%
    margin=0pt,minmargin=0,maxmargin=0,%
    indent=0pt,parindent=0pt,hangindent=0pt,%
    slc,rule,strut}%
  \caption@clearsetup{singleline}}
```

Currently there are two pre-defined styles, called ‘base’ & ‘default’. The first one is a perfect match to the behavior of `\@makecaption` offered by the standard LaTeX document classes (and was called ‘default’ in the caption kernel *v1.0*), the second one matches the document class actually used.

```
\DeclareCaptionStyle{base}[indent=0pt,justification=centering]{}
\DeclareCaptionStyle{default}[indent=0pt,justification=centering]{%
  format=default,labelsep=default,textformat=default,%
  justification=default,font=default,labelfont=default,textfont=default}
```

8 Formats

`\DeclareCaptionFormat`

`\DeclareCaptionFormat{`⟨*name*⟩`}{`⟨*code with #1, #2, and #3*⟩`}`
`\DeclareCaptionFormat*{`⟨*name*⟩`}{`⟨*code with #1, #2, and #3*⟩`}`
The starred form causes the code being typeset in vertical (instead of horizontal) mode, but does not support the `indention=` option.

```
\newcommand*\DeclareCaptionFormat{%
  \caption@teststar\caption@declareformat\@gobble\@firstofone}
\@onlypreamble\DeclareCaptionFormat
```

```
\newcommand*\caption@declareformat[2]{%
  \@dblarg{\caption@@declareformat#1{#2}}}
\@onlypreamble\caption@declareformat
```

```
\long\def\caption@@declareformat#1#2[#3]#4{%
  \global\expandafter\let\csname caption@ifh@#2\endcsname#1%
  \global\long\@namedef{caption@slfmt@#2}##1##2##3{#3}%
  \global\long\@namedef{caption@fmt@#2}##1##2##3{#4}}
\@onlypreamble\caption@@declareformat
```

```
\DeclareCaptionOption{format}{\caption@setformat{#1}}
```

\caption@setformat

`\caption@setformat{`⟨*name*⟩`}`

Selecting a caption format simply means saving the code (in `\caption@fmt`) and if the code should be used in horizontal or vertical mode (`\caption@ifh`).

```
\newcommand*\caption@setformat[1]{%
  \@ifundefined{caption@fmt@#1}%
    {\caption@Error{Undefined format `#1'}}%
    {\expandafter\let\expandafter\caption@ifh\csname caption@ifh@#1\endcsname
     \expandafter\let\expandafter\caption@slfmt\csname caption@slfmt@#1\endcsname
     \expandafter\let\expandafter\caption@fmt\csname caption@fmt@#1\endcsname}}
```

clareCaptionDefaultFormat

```
\newcommand*\DeclareCaptionDefaultFormat[1]{%
  \expandafter\def\expandafter\caption@fmt@default\expandafter
    {\csname caption@fmt@#1\endcsname}%
  \expandafter\def\expandafter\caption@slfmt@default\expandafter
    {\csname caption@slfmt@#1\endcsname}%
  \expandafter\def\expandafter\caption@ifh@default\expandafter
    {\csname caption@ifh@#1\endcsname}}
\@onlypreamble\DeclareCaptionDefaultFormat
```

There are two pre-defined formats, called ‘plain’ and ‘hang’.

```
\DeclareCaptionFormat{plain}{#1#2#3\par}
```

```
\DeclareCaptionFormat{hang}[#1#2#3\par]{%
  \caption@ifin@list\caption@lsepcrlist\caption@lsepname
    {\caption@Error{%
       The option `labelsep=\caption@lsepname' does not work\MessageBreak
       with `format=hang'}}%
    {\@hangfrom{#1#2}%
     \advance\caption@parindent\hangindent
     \advance\caption@hangindent\hangindent
     \caption@@par#3\par}}
```

‘default’ usually maps to ‘plain’.

```
\DeclareCaptionDefaultFormat{plain}
```

9 Label formats

DeclareCaptionLabelFormat

\DeclareCaptionLabelFormat{⟨*name*⟩}{⟨*code with #1 and #2*⟩}

```
\newcommand*\DeclareCaptionLabelFormat[2]{%
  \global\@namedef{caption@lfmt@#1}##1##2{#2}}
\@onlypreamble\DeclareCaptionLabelFormat
```

```
\DeclareCaptionOption{labelformat}{\caption@setlabelformat{#1}}
```

\caption@setlabelformat

\caption@setlabelformat{⟨*name*⟩}
Selecting a caption label format simply means saving the code (in \caption@lfmt).

```
\newcommand*\caption@setlabelformat[1]{%
  \@ifundefined{caption@lfmt@#1}%
    {\caption@Error{Undefined label format `#1'}}%
    {\expandafter\let\expandafter\caption@lfmt\csname caption@lfmt@#1\endcsname}}
```

There are four pre-defined label formats, called 'empty', 'simple', 'parens', and 'brace'.

```
\DeclareCaptionLabelFormat{empty}{}
\DeclareCaptionLabelFormat{simple}{\bothIfFirst{#1}{\nobreakspace}#2}
\DeclareCaptionLabelFormat{parens}{\bothIfFirst{#1}{\nobreakspace}(#2)}
\DeclareCaptionLabelFormat{brace}{\bothIfFirst{#1}{\nobreakspace}#2)}
```

'default' usually maps to 'simple'.

```
\def\caption@lfmt@default{\caption@lfmt@simple}
```

10 Label separators

lareCaptionLabelSeparator

\DeclareCaptionLabelSeparator{⟨*name*⟩}{⟨*code*⟩}
\DeclareCaptionLabelSeparator*{⟨*name*⟩}{⟨*code*⟩}
The starred form causes the label separator to be typeset *without* using \captionlabelfont.

```
\newcommand\DeclareCaptionLabelSeparator{%
  \caption@teststar\caption@declarelabelseparator\@gobble\@firstofone}
\@onlypreamble\DeclareCaptionLabelSeparator
```

```
\newcommand\caption@declarelabelseparator[3]{%
  \global\@namedef{caption@iflf@#2}{#1}%
  \global\long\@namedef{caption@lsep@#2}{#3}%
  \caption@@declarelabelseparator{#2}#3\\\@nil}
\@onlypreamble\caption@declarelabelseparator
```

```
\long\def\caption@@declarelabelseparator#1#2\\#3\@nil{%
  \def\@tempa{#3}\ifx\@tempa\@empty \else
    \caption@g@addto@list\caption@lsepcrlist{#1}%
  \fi}
\@onlypreamble\caption@@declarelabelseparator
```

```
\DeclareCaptionOption{labelsep}{\caption@setlabelseparator{#1}}
\DeclareCaptionOption{labelseparator}{\caption@setlabelseparator{#1}}
```

caption@setlabelseparator

\caption@setlabelseparator{⟨*name*⟩}
Selecting a caption label separator simply means saving the code (in \caption@lsep).

```
\newcommand*\caption@setlabelseparator[1]{%
  \@ifundefined{caption@lsep@#1}%
    {\caption@Error{Undefined label separator `#1'}}%
    {\edef\caption@lsepname{#1}%
     \expandafter\let\expandafter\caption@iflf\csname caption@iflf@#1\endcsname
     \expandafter\let\expandafter\caption@lsep\csname caption@lsep@#1\endcsname}}
```

There are seven pre-defined label separators, called ‘none’, ‘colon’, ‘period’, ‘space’, ‘quad’, ‘newline’, and ‘endash’.

```
\DeclareCaptionLabelSeparator{none}{}
\DeclareCaptionLabelSeparator{colon}{: }
\DeclareCaptionLabelSeparator{period}{. }
\DeclareCaptionLabelSeparator{space}{ }
\DeclareCaptionLabelSeparator*{quad}{\quad}
\DeclareCaptionLabelSeparator*{newline}{\\}
\DeclareCaptionLabelSeparator*{endash}{\space\textendash\space}
```

aption@setdefaultlabelsep

```
\newcommand*\caption@setdefaultlabelsep[1]{%
  \ifx\caption@lsep\caption@lsep@default
    \caption@set@default@labelsep{#1}%
    \caption@setlabelseparator{default}%
  \else
    \caption@set@default@labelsep{#1}%
  \fi}
```

```
\newcommand*\caption@set@default@labelsep[1]{%
  \def\caption@lsep@default{\@nameuse{caption@lsep@#1}}%
  \def\caption@iflf@default{\@nameuse{caption@iflf@#1}}}
```

‘default’ usually maps to ‘colon’.

```
\caption@set@default@labelsep{colon}
```

11 Text formats

\DeclareCaptionTextFormat

\DeclareCaptionTextFormat{⟨*name*⟩}{⟨*code with #1*⟩}

```
\newcommand*\DeclareCaptionTextFormat[2]{%
  \global\long\@namedef{caption@tfmt@#1}##1{#2}}
\@onlypreamble\DeclareCaptionTextFormat
```

```
\DeclareCaptionOption{textformat}{\caption@settextformat{#1}}
\DeclareCaptionOption{strut}[1]{\caption@set@bool\caption@ifstrut{#1}}
```

\caption@settextformat

\caption@settextformat{⟨*name*⟩}

Selecting a caption text format simply means saving the code (in \caption@tfmt).

```
\newcommand*\caption@settextformat[1]{%
  \@ifundefined{caption@tfmt@#1}%
    {\caption@Error{Undefined text format `#1'}}%
    {\expandafter\let\expandafter\caption@tfmt\csname caption@tfmt@#1\endcsname}}
```

There are three pre-defined text formats, called ‘empty’, ‘simple’ and ‘period’.

```
\DeclareCaptionTextFormat{empty}{}
\DeclareCaptionTextFormat{simple}{#1}
\DeclareCaptionTextFormat{period}{#1.}
```

‘default’ usually maps to ‘simple’.

```
\def\caption@tfmt@default{\caption@tfmt@simple}
```

12 Fonts

\DeclareCaptionFont

\DeclareCaptionFont{⟨*name*⟩}{⟨*code*⟩}

```
\newcommand*\DeclareCaptionFont[2]{%
  \define@key{caption@fnt}{#1}[]{\l@addto@macro\caption@fnt{#2}}}
\@onlypreamble\DeclareCaptionFont
```

DeclareCaptionDefaultFont

\DeclareCaptionDefaultFont{⟨*name*⟩}{⟨*code*⟩}

```
\newcommand*\DeclareCaptionDefaultFont[2]{%
  \global\@namedef{caption#1@default}{#2}}
\@onlypreamble\DeclareCaptionDefaultFont
```

```
\DeclareCaptionOption{font}{\caption@setfont{font}{#1}}
\DeclareCaptionOption{font+}{\caption@addtofont{font}{#1}}
\DeclareCaptionDefaultFont{font}{}
```

```
\DeclareCaptionOption{labelfont}{\caption@setfont{labelfont}{#1}}
\DeclareCaptionOption{labelfont+}{\caption@addtofont{labelfont}{#1}}
\DeclareCaptionDefaultFont{labelfont}{}
```

```
\DeclareCaptionOption{textfont}{\caption@setfont{textfont}{#1}}
\DeclareCaptionOption{textfont+}{\caption@addtofont{textfont}{#1}}
\DeclareCaptionDefaultFont{textfont}{}
```

\caption@setfont

\caption@setfont{⟨*name*⟩}{⟨*keyval-list of names*⟩}

Selecting a caption font means saving all the code snippets in \caption⟨*name*⟩.

```
\newcommand*\caption@setfont[1]{%
  \expandafter\let\csname caption#1\endcsname\@empty
  \caption@addtofont{#1}}
```

\caption@addtofont

\caption@addtofont{⟨*name*⟩}{⟨*keyval-list of names*⟩}

Like \caption@setfont, but adds the code snippets to \caption⟨*name*⟩.
Because we use \setkeys recursive here we need to do this inside an extra group.

```
\newcommand*\caption@addtofont[2]{%
  \begingroup
    \expandafter\let\expandafter\caption@fnt\csname caption#1\endcsname
    \define@key{caption@fnt}{default}[]{%
      \l@addto@macro\caption@fnt{\csname caption#1@default\endcsname}}%
    \caption@setkeys[caption]{caption@fnt}{#2}%
    \global\let\caption@tempa\caption@fnt
  \endgroup
  \expandafter\let\csname caption#1\endcsname\caption@tempa}
```

\caption@font

\caption@font{⟨*keyval-list of names*⟩}
\caption@font*{⟨*keyval-code*⟩}

Sets the given font, e.g. \caption@font{small,it} is equivalent to \small\itshape.

```
\newcommand*\caption@font{%
  \caption@teststar\caption@@font\@firstofone
          {\caption@setkeys[caption]{caption@fnt}}}
\newcommand*\caption@@font[2]{%
  \begingroup
  \def\caption@fnt{\endgroup}%
  #1{#2}%
  \caption@fnt}
```

These are the pre-defined font code snippets.

```
\DeclareCaptionFont{normalcolor}{\normalcolor}
\DeclareCaptionFont{color}{\color{#1}}
```

```
\DeclareCaptionFont{normalfont}{\normalfont}
\DeclareCaptionFont{up}{\upshape}
\DeclareCaptionFont{it}{\itshape}
\DeclareCaptionFont{sl}{\slshape}
\DeclareCaptionFont{sc}{\scshape}
\DeclareCaptionFont{md}{\mdseries}
\DeclareCaptionFont{bf}{\bfseries}
\DeclareCaptionFont{rm}{\rmfamily}
\DeclareCaptionFont{sf}{\sffamily}
\DeclareCaptionFont{tt}{\ttfamily}
```

```
\DeclareCaptionFont{scriptsize}{\scriptsize}
\DeclareCaptionFont{footnotesize}{\footnotesize}
\DeclareCaptionFont{small}{\small}
\DeclareCaptionFont{normalsize}{\normalsize}
\DeclareCaptionFont{large}{\large}
\DeclareCaptionFont{Large}{\Large}
```

```
\DeclareCaptionFont{sansmath}{\sansmath}
```

```
\DeclareCaptionFont{singlespacing}{%
  \caption@ifundefined\setspace@singlespace{}{%
    \setstretch\setspace@singlespace}}% normally 1
\DeclareCaptionFont{onehalfspacing}{\onehalfspacing}
\DeclareCaptionFont{doublespacing}{\doublespacing}
\DeclareCaptionFont{stretch}{\setstretch{#1}}
```

```
%\DeclareCaptionFont{normal}{%
%  \caption@font{normalcolor,normalfont,normalsize,singlespacing}
\DeclareCaptionFont{normal}{%
  \caption@font*{%
    \KV@caption@fnt@normalcolor\@unused
    \KV@caption@fnt@normalfont\@unused
    \KV@caption@fnt@normalsize\@unused
    \KV@caption@fnt@singlespacing\@unused}}
```

The old versions *v1.x* of the caption kernel offered this command to setup the font size used for captions. We still do so old documents will work fine.

```
\DeclareCaptionOption{size}{\caption@setfont{size}{#1}}
\DeclareCaptionDefaultFont{size}{}
```

13 Justifications

clareCaptionJustification

`\DeclareCaptionJustification{`⟨*name*⟩`}{`⟨*code*⟩`}`

```
\newcommand*\DeclareCaptionJustification[2]{%
  \global\@namedef{caption@hj@#1}{#2}% for compatibility to v1.0
  \DeclareCaptionFont{#1}{#2}}
\@onlypreamble\DeclareCaptionJustification
```

ptionDefaultJustification

`\DeclareCaptionDefaultJustification{`⟨*code*⟩`}`

```
\newcommand*\DeclareCaptionDefaultJustification[1]{%
```

```
  \global\@namedef{caption@hj@default}{#1}% for compatibility to v1.0
  \DeclareCaptionDefaultFont{@hj}{#1}}
\@onlypreamble\DeclareCaptionDefaultJustification
```

```
\DeclareCaptionOption{justification}{\caption@setjustification{#1}}
\DeclareCaptionDefaultJustification{}
```

\caption@setjustification

`\caption@setjustification{`⟨*name*⟩`}`
Selecting a caption justification simply means saving the code (in `\caption@hj`).

```
\newcommand*\caption@setjustification{\caption@setfont{@hj}}
```

These are the pre-defined justification code snippets.

```
\DeclareCaptionJustification{justified}{}
\DeclareCaptionJustification{centering}{\centering}
\DeclareCaptionJustification{centerfirst}{\centerfirst}
\DeclareCaptionJustification{centerlast}{\centerlast}
\DeclareCaptionJustification{raggedleft}{\raggedleft}
\DeclareCaptionJustification{raggedright}{\raggedright}
```

\centerfirst

Please blame Frank Mittelbach for the code of `\centerfirst :-)`

```
\providecommand\centerfirst{%
  \let\\\@centercr
  \edef\caption@normaladjust{%
    \leftskip\the\leftskip
    \rightskip\the\rightskip
    \parfillskip\the\parfillskip\relax}%
  \leftskip\z@\@plus -1fil%
  \rightskip\z@\@plus 1fil%
  \parfillskip\z@skip
  \noindent\hskip\z@\@plus 2fil%
  \@setpar{\@@par\@restorepar\caption@normaladjust}}
```

\centerlast

This is based on code from Anne Brüggemann-Klein[1]

```
\providecommand\centerlast{%
  \let\\\@centercr
  \leftskip\z@\@plus 1fil%
  \rightskip\z@\@plus -1fil%
  \parfillskip\z@\@plus 2fil\relax}
```

13.1 The ragged2e package

We also support the upper-case commands offered by the ragged2e package. Note that these just map to their lower-case variants if the ragged2e package is not available.

```
\DeclareCaptionJustification{Centering}{%
  \caption@ragged\Centering\centering}
\DeclareCaptionJustification{RaggedLeft}{%
  \caption@ragged\RaggedLeft\raggedleft}
\DeclareCaptionJustification{RaggedRight}{%
  \caption@ragged\RaggedRight\raggedright}
```

\caption@ragged

`\caption@ragged` will be basically defined as

```
\AtBeginDocument{\IfFileExists{ragged2e.sty}%
  {\RequirePackage{ragged2e}\let\caption@ragged\@firstoftwo}%
  {\let\caption@ragged\@secondoftwo}}
```

but with an additional warning if the ragged2e package is not loaded (yet). (This warning will be type out only one time per option, that's why we need the `caption\string#1` stuff.) Furthermore we load the ragged2e package, if needed and available.

```
\newcommand*\caption@ragged{%
  \caption@Debug{We need ragged2e}%
  \protected@write\@auxout{}{\string\caption@newlabel{ragged2e}{}}%
  \global\let\caption@ragged\caption@@ragged
  \caption@ragged}
```

```
\caption@AtBeginDocument{%
  \@ifundefined{caption@r@ragged2e}{%
    \newcommand*\caption@@ragged{%
      \caption@Warning{%
        `ragged2e' support has been changed.\MessageBreak
        Rerun to get captions right}%
      \global\let\caption@ragged\@secondoftwo % suppress further warnings
      \caption@ragged}%
  }{%
    \caption@Debug{We load ragged2e}%
    \IfFileExists{ragged2e.sty}{%
      \RequirePackage{ragged2e}%
      \let\caption@@ragged\@firstoftwo
    }{%
      \newcommand*\caption@@ragged[2]{%
        \@ifundefined{caption\string#1}{%
          \caption@Warning{%
            `ragged2e' package not loaded, therefore\MessageBreak
            substituting \string#2 for \string#1\MessageBreak}%
          \global\@namedef{caption\string#1}}{}%
        #2}%
    }%
  }}
```

14 Vertical spaces before and after captions

`\abovecaptionskip`
`\belowcaptionskip`

Usually these skips are defined within the document class, but some document classes don't do so.

```
\caption@ifundefined\abovecaptionskip{%
  \newlength\abovecaptionskip\setlength\abovecaptionskip{10\p@}}{}
\caption@ifundefined\belowcaptionskip{%
  \newlength\belowcaptionskip\setlength\belowcaptionskip{0\p@}}{}
```

```
\DeclareCaptionOption{aboveskip}{\setlength\abovecaptionskip{#1}}
\DeclareCaptionOption{belowskip}{\setlength\belowcaptionskip{#1}}
\DeclareCaptionOption{skip}{\setlength\abovecaptionskip{#1}}
```

`\caption@rule`

`\caption@rule`
Draws an invisible rule to adjust the "skip" setting.

```
\newcommand*\caption@rule{\caption@ifrule\caption@hrule\relax}
```

```
\newcommand*\caption@hrule{\hrule\@height\z@}
```

```
\DeclareCaptionOption{rule}[1]{\caption@set@bool\caption@ifrule{#1}}
```

15 Positioning

These macros handle the right position of the caption. Note that the position is actually *not* controlled by the caption3 kernel options, but by the user (or a specific package like the float package) instead. The user can put the `\caption` command wherever he likes! So this stuff is only to give us a *hint* where to put the right skips, the user usually has to take care for himself that this hint actually matches the right position.

```
\DeclareCaptionOption{position}{\caption@setposition{#1}}
```

`\caption@setposition`

`\caption@setposition{`⟨*position*⟩`}`
Selecting the caption position means that we put `\caption@position` to the right value. *Please do **not** use the internal macro `\caption@position` in your own package or document, but use the wrapper macro `\caption@iftop` instead.*

```
\newcommand*\caption@setposition[1]{%
  \caption@ifinlist{#1}{d,default}{%
    \let\caption@position\caption@defaultpos
  }{\caption@ifinlist{#1}{t,top,above}{%
    \let\caption@position\@firstoftwo
  }{\caption@ifinlist{#1}{b,bottom,below}{%
    \let\caption@position\@secondoftwo
  }{\caption@ifinlist{#1}{a,auto}{%
    \let\caption@position\@undefined
  }{%
    \caption@Error{Undefined position `#1'}%
  }}}}}
```

`\caption@defaultpos`

The default 'position' is 'auto', this means that the caption kernel will try to guess the current position of the caption. (But in many cases, for example in `longtable`s, this is doomed to fail!)
The setting 'bottom' correspondents to the `\@makecaption` implementation in the standard LaTeX document classes, but 'auto' should give better results in most cases.

```
%\caption@setdefaultpos{a}% default = auto
\let\caption@defaultpos\@undefined
```

`\caption@iftop`

`\caption@iftop{`⟨*true-code*⟩`}{`⟨*false-code*⟩`}`
(If the `position=` is set to `auto` we assume a `bottom` position here.)

```
\newcommand*\caption@iftop{%
  \ifx\caption@position\@undefined
    \let\caption@position\@secondoftwo
%   = \caption@setposition b%
  \fi
  \caption@position}
```

`\caption@fixposition`

`\caption@fixposition`
This macro checks if the 'position' is set to 'auto'. If yes, `\caption@autoposition` will be called to set `\caption@position` to a proper value we can actually use.

```
\newcommand*\caption@fixposition{%
```

```
  \ifx\caption@position\@undefined
    \caption@autoposition
  \fi}
```

`\caption@autoposition`
We guess the current position of the caption by checking `\prevdepth`.
A different solution would be setting the `\spacefactor` to something not much less than 1000 (for example 994) in `\caption@start` and checking this value here by `\ifnum\spacefactor=994`. (It's implemented in the threeparttable package[4] this way.)
Another idea would be checking `\@ifminipage`, but since some packages typeset the caption within a simple `\vbox` this does not seem to be a good one.

```
\newcommand*\caption@autoposition{%
  \ifvmode
    \edef\caption@tempa{\the\prevdepth}%
    \caption@Debug{\protect\prevdepth=\caption@tempa}%
    \ifdim\prevdepth>-\p@
      \let\caption@position\@secondoftwo
    \else
      \let\caption@position\@firstoftwo
    \fi
%   = \caption@setposition{\ifdim\prevdepth>-\p@ b\else t\fi}%
  \else
    \caption@Debug{no \protect\prevdepth}%
    \let\caption@position\@secondoftwo
%   = \caption@setposition b%
  \fi}
```

`\caption@setautoposition{`⟨*position*⟩`}`
replaces the above algorithm by a different one (or a fixed position setting).

```
\newcommand*\caption@setautoposition[1]{%
  \def\caption@autoposition{\caption@setposition{#1}}}
```

16 Hooks

`\AtBeginCaption {`⟨*code*⟩`}`
`\AtEndCaption {`⟨*code*⟩`}`
These hooks can be used analogous to `\AtBeginDocument` and `\AtEndDocument`.

```
\newcommand*\caption@beginhook{}
\newcommand*\caption@endhook{}
\newcommand*\AtBeginCaption{\l@addto@macro\caption@beginhook}
\newcommand*\AtEndCaption{\l@addto@macro\caption@endhook}
```

17 Lists

```
\DeclareCaptionOption{list}[1]{\caption@setlist{#1}}
\DeclareCaptionOption{listof}[1]{\caption@setlist{#1}}
```

`\caption@setlist{`⟨*boolean*⟩`}`

```
\newcommand*\caption@setlist{\caption@set@bool\caption@iflist}
```

```
\DeclareCaptionOption{listtype}{\caption@setlisttype{#1}}
\DeclareCaptionOption{listtype+}{\caption@setlisttype@ext{#1}}
```

\caption@setlisttype

\caption@setlisttype{⟨*type*⟩}

```
\newcommand*\caption@setlisttype{%
  \caption@setlisttype@ext{}%
  \caption@@setlisttype\caption@listtype}
```

```
\newcommand*\caption@@setlisttype[2]{%
  \edef#1{#2}%
  \ifx#1\@empty \let#1\@undefined \fi}
```

\caption@setlisttype@ext

\caption@setlisttype@ext{⟨*type extension*⟩}

```
\newcommand*\caption@setlisttype@ext{%
  \caption@@setlisttype\caption@listtype@ext}
```

\DeclareCaptionListFormat

\DeclareCaptionListFormat{⟨*name*⟩}{⟨*code with #1 and #2*⟩}

```
\newcommand*\DeclareCaptionListFormat[2]{%
  \global\@namedef{caption@lstfmt@#1}##1##2{#2}}
\@onlypreamble\DeclareCaptionListFormat
```

```
\DeclareCaptionOption{listformat}{\caption@setlistformat{#1}}
```

\caption@setlistformat

\caption@setlistformat{⟨*name*⟩}

Selecting a caption list format simply means saving the code (in `\caption@lstfmt`).

```
\newcommand*\caption@setlistformat[1]{%
  \@ifundefined{caption@lstfmt@#1}%
    {\caption@Error{Undefined list format `#1'}}%
    {\expandafter\let\expandafter\caption@lstfmt
       \csname caption@lstfmt@#1\endcsname}}
```

There are five pre-defined list formats, taken from the subfig package.

```
\DeclareCaptionListFormat{empty}{}
\DeclareCaptionListFormat{simple}{#1#2}
\DeclareCaptionListFormat{parens}{#1(#2)}
\DeclareCaptionListFormat{subsimple}{#2}
\DeclareCaptionListFormat{subparens}{(#2)}
```

tion@setdefaultlistformat

```
\newcommand*\caption@setdefaultlistformat[1]{%
  \ifx\caption@lstfmt\caption@lstfmt@default
    \caption@set@default@listformat{#1}%
    \caption@setlistformat{default}%
  \else
    \caption@set@default@listformat{#1}%
  \fi}
```

```
\newcommand*\caption@set@default@listformat[1]{%
  \def\caption@lstfmt@default{\@nameuse{caption@lstfmt@#1}}}
```

‘default’ usually maps to ‘subsimple’.

```
\caption@set@default@listformat{subsimple}
```

18 Debug option

```
\DeclareCaptionOption{debug}[1]{%
  \caption@set@bool\caption@ifdebug{#1}%
  \caption@ifdebug
    {\let\caption@Debug\caption@Info}%
    {\let\caption@Debug\@gobble}}
```

```
\DeclareOption{debug}{\setkeys{caption}{debug}}
```

```
\setkeys{caption}{debug=0}
```

19 Document classes & Babel support

19.1 The standard LaTeX classes

```
\caption@CheckCommand\@makecaption{%
  % article|report|book [2005/09/16 v1.4f Standard LaTeX document class]
  \long\def\@makecaption#1#2{%
    \vskip\abovecaptionskip
    \sbox\@tempboxa{#1: #2}%
    \ifdim \wd\@tempboxa >\hsize
      #1: #2\par
    \else
      \global \@minipagefalse
      \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
    \fi
    \vskip\belowcaptionskip}}
```

19.2 The $\mathcal{AMS}$ & SMF classes

\caption@ifamsclass

```
\providecommand*\caption@ifamsclass{%
  \caption@ifundefined\@captionheadfont\@gobble\@firstofone}
\@onlypreamble\caption@ifamsclass
```

```
\caption@ifamsclass{%
```

```
  \caption@CheckCommand\@makecaption{%
    % amsart|amsproc|amsbook [2004/08/06 v2.20]
    \long\def\@makecaption#1#2{%
      \setbox\@tempboxa\vbox{\color@setgroup
        \advance\hsize-2\captionindent\noindent
        \@captionfont\@captionheadfont#1\@xp\@ifnotempty\@xp
            {\@cdr#2\@nil}{.\@captionfont\upshape\enspace#2}%
        \unskip\kern-2\captionindent\par
        \global\setbox\@ne\lastbox\color@endgroup}%
      \ifhbox\@ne % the normal case
        \setbox\@ne\hbox{\unhbox\@ne\unskip\unskip\unpenalty\unkern}%
      \fi
      \ifdim\wd\@tempboxa=\z@ % this means caption will fit on one line
        \setbox\@ne\hbox to\columnwidth{\hss\kern-2\captionindent\box\@ne\hss}%
      \else % tempboxa contained more than one line
        \setbox\@ne\vbox{\unvbox\@tempboxa\parskip\z@skip
            \noindent\unhbox\@ne\advance\hsize-2\captionindent\par}%
      \fi
      \ifnum\@tempcnta<64 % if the float IS a figure...
        \addvspace\abovecaptionskip
        \hbox to\hsize{\kern\captionindent\box\@ne\hss}%
      \else % if the float IS NOT a figure...
        \hbox to\hsize{\kern\captionindent\box\@ne\hss}%
        \nobreak
        \vskip\belowcaptionskip
      \fi
    \relax
    }}
  \caption@CheckCommand\@makecaption{%
    % smfart|smfbook [1999/11/15 v1.2f Classe LaTeX pour les articles publies par
    \long\def\@makecaption#1#2{%
      \ifdim\captionindent>.1\hsize \captionindent.1\hsize \fi
      \setbox\@tempboxa\vbox{\color@setgroup
        \advance\hsize-2\captionindent\noindent
        \@captionfont\@captionheadfont#1\@xp\@ifnotempty\@xp
            {\@cdr#2\@nil}{\@addpunct{.}\@captionfont\upshape\enspace#2}%
        \unskip\kern-2\captionindent\par
        \global\setbox\@ne\lastbox\color@endgroup}%
      \ifhbox\@ne % the normal case
        \setbox\@ne\hbox{\unhbox\@ne\unskip\unskip\unpenalty\unkern}%
      \fi
      \ifdim\wd\@tempboxa=\z@ % this means caption will fit on one line
        \setbox\@ne\hbox to\columnwidth{\hss\kern-2\captionindent\box\@ne\hss}%
        \@tempdima\wd\@ne\advance\@tempdima-\captionindent
        \wd\@ne\@tempdima
      \else % tempboxa contained more than one line
        \setbox\@ne\vbox{\rightskip=0pt plus\captionindent\relax
            \unvbox\@tempboxa\parskip\z@skip
            \noindent\unhbox\@ne\advance\hsize-2\captionindent\par}%
      \fi
      \ifnum\@tempcnta<64 % if the float IS a figure...
        \addvspace\abovecaptionskip
        \noindent\kern\captionindent\box\@ne
      \else % if the float IS NOT a figure...
        \noindent\kern\captionindent\box\@ne
        \nobreak
        \vskip\belowcaptionskip
      \fi
    \relax
    }}
  \let\captionmargin\captionindent % set to 3pc by AMS class
  \begingroup\edef\@tempa{\endgroup
    \noexpand\caption@g@addto@list\noexpand\caption@sty@default
      {margin=\the\captionmargin
       \caption@ifundefined\smf@makecaption{}{,maxmargin=.1\linewidth}}}
  \@tempa
  \caption@g@addto@list\caption@sls@default{margin*=.5\captionmargin}
  \DeclareCaptionLabelSeparator{default}{.\enspace}
  \DeclareCaptionDefaultFont{font}{\@captionfont}
  \DeclareCaptionDefaultFont{labelfont}{\@captionheadfont}
  \DeclareCaptionDefaultFont{textfont}{\@captionfont\upshape}
  \captionsetup[figure]{position=b}
```

```
  \captionsetup[table]{position=t}
}
```

19.3 The beamer class (Part one)

\caption@ifbeamerclass

```
\providecommand*\caption@ifbeamerclass{%
  \@ifclassloaded{beamer}\@firstofone\@gobble}
\@onlypreamble\caption@ifbeamerclass
```

```
\caption@ifbeamerclass{%
```

```
  \caption@CheckCommand\beamer@makecaption{%
    % beamerbaselocalstructure.sty,v 1.53 2007/01/28 20:48:21 tantau
    \long\def\beamer@makecaption#1#2{%
      \def\insertcaptionname{\csname#1name\endcsname}%
      \def\insertcaptionnumber{\csname the#1\endcsname}%
      \def\insertcaption{#2}%
      \nobreak\vskip\abovecaptionskip\nobreak
      \sbox\@tempboxa{\usebeamertemplate**{caption}}%
      \ifdim \wd\@tempboxa >\hsize
        \usebeamertemplate**{caption}\par
      \else
        \global \@minipagefalse
        \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
      \fi
      \nobreak\vskip\belowcaptionskip\nobreak}}
```

\caption@ifbeamertemplate

```
\newcommand*\caption@ifbeamertemplate[1]{%
  \begingroup
    \let\beamer@@tmpl@caption@ORI\beamer@@tmpl@caption
    \@nameuse{beamer@@tmpop@caption@#1}%
    \ifx\beamer@@tmpl@caption@ORI\beamer@@tmpl@caption
      \endgroup\expandafter\@firstoftwo
    \else
      \endgroup\expandafter\@secondoftwo
    \fi}
```

```
  \DeclareCaptionLabelFormat{default}{%
    #1\caption@ifbeamertemplate{numbered}{~#2}{}}
  \caption@declarelabelseparator
    {\caption@ifbeamertemplate{caption name own line}\@gobble\@firstofone}
    {default}
    {\caption@ifbeamertemplate{caption name own line}{\\}{: }}
  \DeclareCaptionDefaultFont{font}{%
    \usebeamerfont*{caption}%
    \usebeamercolor[fg]{caption}}
  \DeclareCaptionDefaultFont{labelfont}{%
    \usebeamercolor[fg]{caption name}%
    \usebeamerfont*{caption name}}
  \DeclareCaptionDefaultJustification{\raggedright}
  \DeclareOption{beamerclass}{%
    \renewcommand\caption@ifslc{%
```

```
      \caption@ifbeamertemplate{caption name own line}\@secondoftwo\@firstoftwo}
    % Since the beamer class do not offer a ‘list of figures’ we switch this supp
    \captionsetup{list=0}}
  \PassOptionsToPackage{beamerclass}{caption3}
```

If the beamer document class is used, we offer a beamer template called ‘caption3’ which can be used with option ‘beamer’ or `\setbeamertemplate{caption}[caption3]`. (Note that this is of no use when the caption package is used, too.)

```
  \defbeamertemplate{caption}{caption3}{%
    \caption@make\insertcaptionname\insertcaptionnumber\insertcaption}
```

```
  \DeclareOption{beamer}{%
    % \usebeamertemplate**{caption} will set font
    \DeclareCaptionDefaultFont{font}{}%
    \setbeamertemplate{caption}[caption3]}
```

```
%
%    \begin{macrocode}
}
```

19.4 The KOMA-Script classes

\caption@ifkomaclass

```
\providecommand*\caption@ifkomaclass{%
  \caption@ifundefined\scr@caption\@gobble\@firstofone}
\@onlypreamble\caption@ifkomaclass
```

```
\caption@ifkomaclass{%
```

```
  \caption@CheckCommand\@makecaption{%
    % scrartcl|scrreprt|scrbook [2007/03/07 v2.97a KOMA-Script document class]
    \long\def\@makecaption#1#2{%
      \if@captionabove
        \vskip\belowcaptionskip
      \else
        \vskip\abovecaptionskip
      \fi
      \@@makecaption\@firstofone{#1}{#2}%
      \if@captionabove
        \vskip\abovecaptionskip
      \else
        \vskip\belowcaptionskip
      \fi}}
```

```
  \DeclareCaptionFormat{default}[#1#2#3\par]{%
    \ifdofullc@p
      \caption@ifin@list\caption@lsepcrlist\caption@lsepname
        {\caption@Error{%
           The option ‘labelsep=\caption@lsepname’ does not work\MessageBreak
           with \noexpand\setcaphanging (which is set by default)}}%
        {\caption@fmt@hang{#1}{#2}{#3}}%
    \else
      #1#2%
      \ifdim\cap@indent<\z@
        \par
        \noindent\hspace*{-\cap@indent}%
      \else\if@capbreak
        \par
      \fi\fi
      #3\par
    \fi}
  \DeclareCaptionLabelSeparator{default}{\captionformat}
  \DeclareCaptionDefaultFont{font}{\scr@fnt@caption}
  \DeclareCaptionDefaultFont{labelfont}{\scr@fnt@captionlabel}
```

```
}
```

19.5 The NTG Dutch classes

\caption@ifntgclass

```
\providecommand*\caption@ifntgclass{%
  \caption@ifundefined\CaptionFonts\@gobble\@firstofone}
\@onlypreamble\caption@ifntgclass
```

```
\caption@ifntgclass{%
```

```
  \caption@CheckCommand\@makecaption{%
    % artikel|rapport|boek [2004/06/07 v2.1a NTG LaTeX document class]
    \long\def\@makecaption#1#2{%
      \vskip\abovecaptionskip
      \sbox\@tempboxa{{\CaptionLabelFont#1:} \CaptionTextFont#2}%
      \ifdim \wd\@tempboxa >\hsize
        {\CaptionLabelFont#1:} \CaptionTextFont#2\par
      \else
        \global \@minipagefalse
        \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
      \fi
      \vskip\belowcaptionskip}}
```

```
  \DeclareCaptionDefaultFont{labelfont}{\CaptionLabelFont}
  \DeclareCaptionDefaultFont{textfont}{\CaptionTextFont}
```

```
}
```

19.6 The thesis class

\caption@ifthesisclass

```
\providecommand*\caption@ifthesisclass{%
  \caption@ifundefined\cph@font
    {\@gobble}%
    {\caption@ifundefined\cpb@font\@gobble\@firstofone}}
```

```
\caption@ifthesisclass{%
```

```
  \caption@CheckCommand\@makecaption{%
    % thesis.cls 1996/25/01 1.0g LaTeX document class (wm).
    \long\def\@makecaption#1#2{%
     \vskip\abovecaptionskip
     \setbox\@tempboxa\hbox{{\cph@font #1:} {\cpb@font #2}}%
     \ifdim \wd\@tempboxa >\hsize
        \@hangfrom{\cph@font #1: }{\cpb@font #2\par}%
     \else
```

```
        \hbox to\hsize{\hfil\box\@tempboxa\hfil}%
     \fi
     \vskip\belowcaptionskip}}
  \DeclareCaptionDefaultFormat{hang}
  \DeclareCaptionDefaultFont{labelfont}{\cph@font}
  \DeclareCaptionDefaultFont{textfont}{\cpb@font}
}
```

19.7 The frenchb Babel option

```
\caption@ifundefined\FB@makecaption{}{%
  \caption@CheckCommand\@makecaption{%
    % frenchb.ldf [2005/02/06 v1.6g French support from the babel system]
    % frenchb.ldf [2007/10/05 v2.0e French support from the babel system]
    \long\def\@makecaption#1#2{%
      \vskip\abovecaptionskip
      \sbox\@tempboxa{#1\CaptionSeparator #2}%
      \ifdim \wd\@tempboxa >\hsize
        #1\CaptionSeparator #2\par
      \else
        \global \@minipagefalse
        \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
      \fi
      \vskip\belowcaptionskip}}
  \ifx\@makecaption\STD@makecaption
    \DeclareCaptionLabelSeparator{default}{\CaptionSeparator}
    \def\caption@frenchb{% supress frenchb warning
      \let\STD@makecaption\@makecaption
      \let\FB@makecaption\@makecaption}
  \else
    \ifx\@makecaption\@undefined\else
      \caption@InfoNoLine{%
        The definition of \protect\@makecaption\space
        has been changed,\MessageBreak
        frenchb will NOT customize it}%
    \fi
  \fi
}
```

19.8 The frenchle/pro package

```
\caption@ifundefined\frenchTeXmods{}{%
  \caption@CheckCommand\@makecaption{%
    % french(le).sty [2006/10/03 The french(le) package /V5,9991/]
    % french(le).sty [2007/06/28 The french(le) package /V5,9994/]
    \def\@makecaption#1#2{%
      \ifFTY%
        \def\@secondofmany##1##2\void{##2}%
        \def\@tempa{\@secondofmany#2\void}%
        \ifx\@tempa\empty%
          \let\captionseparator\empty%
        \fi%
        \@mcORI{#1}{\relax\captionfont{#2}}%
      \else
        \@mcORI{#1}{#2}%
      \fi}}%
  \caption@CheckCommand\@makecaption{%
    % french(le).sty [2007/02/11 The french(le) package /V5,9993/]
    \def\@makecaption#1#2{%
      \ifFTY%
        \def\@secondofmany##1##2\void{##2}%
        \protected@edef\@tempa{\@secondofmany#2\void}%
        \ifx\@tempa\empty%
          \let\captionseparator\empty%
        \fi%
        \@mcORI{#1}{\relax\captionfont{#2}}%
      \else
        \@mcORI{#1}{#2}%
      \fi}}%
  \DeclareCaptionDefaultFont{textfont}{\itshape}%
  \DeclareCaptionLabelSeparator{default}{\captionseparator\space}%
}
```

19.9 The hungarian and magyar Babel option

```
\DeclareCaptionListFormat{subperiod}{#2.}
\caption@ifundefined\hunnewlabel{}{%
  \caption@CheckCommand\@makecaption{%
    % magyar.ldf [2005/03/30 v1.4j Magyar support from the babel system]
    \def\@makecaption#1#2{%
      \vskip\abovecaptionskip
      \sbox\@tempboxa{#1. #2}%
      \ifdim \wd\@tempboxa >\hsize
        {#1. #2\csname par\endcsname}
      \else
        \global \@minipagefalse
        \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
      \fi
      \vskip\belowcaptionskip}}}
\def\caption@tempa#1{\@ifundefined{extras#1}{}{%
  \expandafter\addto\csname extras#1\endcsname{%
     % change default labelsep and listformat
     \caption@setdefaultlabelsep{period}%
     \caption@setdefaultlistformat{subperiod}}%
  \expandafter\addto\csname noextras#1\endcsname{%
     % change default labelsep and listformat
     \caption@setdefaultlabelsep{colon}%
     \caption@setdefaultlistformat{subsimple}}%
}}
\caption@tempa{hungarian}
\caption@tempa{magyar}
```

19.10 Unknown document class (or package)

```
\caption@IfCheckCommand{%
  \caption@setbool{documentclass}{1}%
}{%
  \caption@setbool{documentclass}{0}%
  \caption@InfoNoLine{%
         Unknown document class (or package),\MessageBreak
         standard defaults will be used}%
  \caption@Debug{\string\@makecaption\space=\space\meaning\@makecaption\@gobble}%
}
```

20 Execution of options

```
\captionsetup{style=default,position=default,%
              list,listformat=default,twoside=\if@twoside 1\else 0\fi}
```

```
\ProcessOptions*
```

21 Making an 'List of' entry

\caption@addcontentsline

\caption@addcontentsline{⟨*type*⟩}{⟨*list entry*⟩}
Makes an entry in the list-of-whatever, if requested, i.e. the argument ⟨*list entry*⟩ is not empty and listof= was set to true.

```
\newcommand\caption@addcontentsline[2]{%
  \caption@ifcontentsline{#2}{%
    \begingroup
      \let\@tempa\@gobble
      \caption@ifundefined\caption@listtype
        {\edef\caption@listtype{#1}}%
        {\let\@tempa\@firstofone}%
      \caption@ifundefined\caption@listtype@ext
        {}%
        {\edef\caption@listtype{\caption@listtype\caption@listtype@ext}%
         \let\@tempa\@firstofone}%
      \@tempa
        {\caption@Debug{addcontentsline: #1 => \caption@listtype}%
%        \caption@setoptions*\caption@listtype
         \@namedef{the\caption@listtype}{\@nameuse{the#1}}}%
      \expandafter\caption@@addcontentsline\expandafter{\caption@listtype}{#2}%
    \endgroup}}
```

```
\newcommand\caption@@addcontentsline[2]{%
  {\let\\\space
   \@ifundefined{ext@#1}%
     {\caption@Error{No float type '#1' defined}}%
     {\caption@@@addcontentsline
       {\csname ext@#1\endcsname}%
       {#1}%
       {\caption@lstfmt{\@nameuse{p@#1}}{\@nameuse{the#1}}}%
       {\ignorespaces #2}}}}
```

```
\newcommand*\caption@@@addcontentsline[4]{%
  \addcontentsline{#1}{#2}{\protect\numberline{#3}{#4}}}
```

```
\newcommand\caption@ifcontentsline[1]{%
  \caption@iflist
    {\def\@tempa{#1}}%
    {\let\@tempa\@empty}%
  \ifx\@tempa\@empty
    \expandafter\@gobble
  \else
    \expandafter\@firstofone
  \fi}
```

22 Typesetting the caption

\ifcaption@star

If the starred form of `\caption` is used, this will be set to `true`. (It will be reset to `false` at the end of `\caption@@make`.)

```
\newif\ifcaption@star
```

\caption@fnum

`\caption@fnum{`⟨*float type*⟩`}`
Typesets the caption label; as replacement for `\fnum@`⟨*float type*⟩.

```
\newcommand*\caption@fnum[1]{\caption@lfmt{\@nameuse{#1name}}{\@nameuse{the#1}}}
```

\caption@make

`\caption@make{`⟨*float name*⟩`}{`⟨*ref. number*⟩`}{`⟨*text*⟩`}`
Typesets the caption.

```
\newcommand\caption@make[2]{\caption@@make{\caption@lfmt{#1}{#2}}}
```

\caption@@make

`\caption@@make{`⟨*caption label*⟩`}{`⟨*caption text*⟩`}`

```
\newcommand\caption@@make[2]{%
  \begingroup
  \caption@stepcounter
  \caption@beginhook
```

Check margin, if `\caption@minmargin` or `\caption@maxmargin` is set

```
% TODO: Move this to \caption@calcmargin!?
  \ifx\caption@maxmargin\@undefined \else
    \ifdim\captionmargin>\caption@maxmargin\relax
      \captionmargin\caption@maxmargin\relax
    \fi
  \fi
  \ifx\caption@minmargin\@undefined \else
    \ifdim\captionmargin<\caption@minmargin\relax
      \captionmargin\caption@minmargin\relax
    \fi
  \fi
```

Special single-line treatment (option `singlelinecheck=`)

```
  \caption@ifslc{\caption@slc{#1}{#2}\captionwidth\relax}{}%
```

Typeset the left margin (option `margin=`)

```
  \caption@calcmargin
  \@tempdima\captionmargin
  \ifdim\captionmargin@=\z@ \else
    \caption@ifoddpage{}{\advance\@tempdima\captionmargin@}%
  \fi
  \caption@ifh{\advance\@tempdima\caption@indent}%
  \hspace\@tempdima
```

We actually use a `\vbox` of width `\captionwidth - \caption@indent` to typeset the caption.
Note: `\captionindent` is *not* supported if the caption format was defined with `\DeclareCaptionFormat*`.

```
  \@tempdima\captionwidth
  \caption@ifh{\advance\@tempdima-\caption@indent}%
  \caption@parbox\@tempdima{%
```

Typeset the indention (option `indention=`)
Bugfix 04-05-05: `\hskip-\caption@indent` replaced by `\ifdim\caption@indent=\z@`...

```
    \caption@ifh{%
      \ifdim\caption@indent=\z@
        \leavevmode
      \else
        \hskip-\caption@indent
      \fi}%
```

Typeset the caption itself and close the `\caption@parbox`

```
    \caption@@@make{#1}{#2}}%
```

Typeset the right margin (option `margin=`)

```
  \@tempdima\captionmargin
  \ifdim\captionmargin@=\z@ \else
    \caption@ifoddpage{\advance\@tempdima\captionmargin@}{}%
  \fi
  \hspace\@tempdima

  \caption@endhook
  \endgroup

  \global\caption@starfalse}
```

`\caption@calcmargin`

`\caption@calcmargin`
Calculate `\captionmargin` & `\captionwidth`, so both contain valid values.

```
\newcommand*\caption@calcmargin{%
  \caption@calcmargin@hook
  \ifdim\captionwidth=\z@
    \captionwidth\linewidth
    \advance\captionwidth by -2\captionmargin
    \advance\captionwidth by -\captionmargin@
  \else
    \captionmargin\linewidth
    \advance\captionmargin by -\captionwidth
    \divide\captionmargin by 2
    \captionmargin@\z@
  \fi

  \caption@Debug{%
    \string\hsize=\the\hsize,
    \string\linewidth=\the\linewidth,\MessageBreak
    \string\leftmargin=\the\leftmargin,
    \string\rightmargin=\the\rightmargin,\MessageBreak
    \string\margin=\the\captionmargin,
    \string\margin@=\the\captionmargin@,
    \string\width=\the\captionwidth}%

}
```

\caption@slc

\caption@slc{⟨*label*⟩}{⟨*text*⟩}{⟨*width*⟩}{⟨*extra code*⟩}
This one does the single-line-check.

```
\newcommand\caption@slc[4]{%
  \caption@@slc{#1}{#2}{#3}{\caption@singleline#4}{}}
```

```
\newcommand\caption@@slc[5]{%
  \caption@Debug{Begin SLC}%
  \begingroup
  \caption@singleline
  \let\caption@hj\@empty
  \caption@calcmargin % calculate #3 if necessary
  \caption@prepareslc
  \sbox\@tempboxa{\caption@@@make{#1}{#2}}%
  \ifdim\wd\@tempboxa>#3%
    \endgroup
    #5%
  \else
    \endgroup
    #4%
  \fi
  \caption@Debug{End SLC}}
```

```
\newcommand*\caption@singleline{%
  \caption@xsetup\caption@opt@singleline
  \let\caption@fmt\caption@slfmt}
```

\caption@prepareslc

\caption@prepareslc
Re-define anything which would disturb the single-line-check.

```
\newcommand*\caption@prepareslc{%
  \let\label\caption@gobble
```

```
  \let\caption@footnotemark@ORI\footnotemark
  \def\footnote{\caption@withoptargs\caption@footnote}%
  \def\footnotemark{\caption@withoptargs\caption@footnotemark}%
  \let\@footnotetext\caption@gobble
```

```
  \let\@endnotetext\caption@gobble
```

```
  \let\pagenote\caption@gobble
```

```
}
```

```
\newcommand\caption@footnote[2]{%
  \caption@footnotemark{#1}}
\newcommand\caption@footnotemark[1]{%
  \begingroup
    \let\stepcounter\caption@l@stepcounter
    \caption@footnotemark@ORI#1%
  \endgroup}
```

```
\newcommand*\caption@l@stepcounter[1]{%
  \advance\csname c@#1\endcsname\@ne\relax}
```

\caption@parbox

\caption@parbox{⟨*width*⟩}{⟨*contents*⟩}
This macro defines the box which surrounds the caption paragraph.

```
\newcommand*\caption@parbox{\parbox[b]}
```

\caption@applyfont

\caption@applyfont
This macro executes the font relevant macros, i.e. by default the options set by justification=, font=, and size=.

```
\newcommand*\caption@applyfont{%
  \caption@hj\captionfont\captionsize}
```

\caption@@@make

\caption@@@make{⟨*caption label*⟩}{⟨*caption text*⟩}
This one finally typesets the caption paragraph, without margin and indention.

```
\newcommand\caption@@@make[2]{%
```

If the label is empty, we use no caption label separator.

```
  \sbox\@tempboxa{#1}%
  \ifdim\wd\@tempboxa=\z@
    \let\caption@lsep\relax
%   \@capbreakfalse
  \fi
```

If the text is empty, we use no caption label separator, too. (And no text format either.)

```
  \caption@ifempty{#2}{%
    \let\caption@lsep\@empty
    \let\caption@tfmt\@firstofone
%   \@capbreakfalse
%   \let\caption@ifstrut\@secondoftwo
  }%
```

Take care that \caption@parindent and \caption@hangindent will be used to typeset the paragraph.

```
  \@setpar{\@@par\caption@@par}\caption@@par
```

Finally typeset the caption.

```
  \caption@applyfont
  \caption@fmt
    {\ifcaption@star\else{\captionlabelfont#1}\fi}%
    {\ifcaption@star\else{\caption@iflf\captionlabelfont\caption@lsep}\fi}%
    {{\captiontextfont
      \caption@ifstrut{\vrule\@height\ht\strutbox\@width\z@}{}%
      \nobreak\hskip\z@skip % enable hyphenation
      \caption@tfmt{#2}%
%     \caption@ifstrut{\vrule\@height\z@\@depth\dp\strutbox\@width\z@}{}%
      \caption@ifstrut{\ifhmode\@finalstrut\strutbox\fi}{}%
      \par}}}
```

\caption@ifempty

\caption@ifempty{⟨*text*⟩}{⟨*true*⟩} (*no* ⟨*false*⟩)
This one tests if the ⟨*text*⟩ is actually empty.
Note: This will be done without expanding the text, therefore this is far away from being bulletproof.
Note: This macro is re-defining itself so only the first test (in a group) will actually be done.

```
\newcommand\caption@ifempty[1]{%
  \caption@if@empty{#1}%
  \caption@ifempty\@unused}
```

```
\newcommand\caption@if@empty[1]{%
  \def\caption@tempa{#1}%
  \ifx\caption@tempa\@empty
    \let\caption@ifempty\@secondoftwo
  \else
    \expandafter\def\expandafter\caption@tempa\expandafter{%
      \caption@car#1\caption@if@empty\caption@nil}%
    \def\caption@tempb{\caption@if@empty}%
    \ifx\caption@tempa\caption@tempb
      \let\caption@ifempty\@secondoftwo
    \else
      \def\caption@tempb{\ignorespaces}%
      \ifx\caption@tempa\caption@tempb
        \expandafter\caption@if@empty\expandafter{\@gobble#1}%
      \else
        \def\caption@tempb{\label}%
        \ifx\caption@tempa\caption@tempb
          \expandafter\caption@if@empty\expandafter{\@gobbletwo#1}%
        \else
          \def\caption@tempb{\index}%
          \ifx\caption@tempa\caption@tempb
            \expandafter\caption@if@empty\expandafter{\@gobbletwo#1}%
          \else
            \def\caption@tempb{\glossary}%
            \ifx\caption@tempa\caption@tempb
              \expandafter\caption@if@empty\expandafter{\@gobbletwo#1}%
            \else
              \let\caption@ifempty\@gobbletwo
            \fi
          \fi
        \fi
      \fi
    \fi
  \fi}
```

```
\long\def\caption@car#1#2\caption@nil{#1}% same as \@car, but \long
```

\caption@@par

`\caption@@par`
This command will be executed with every `\par` inside the caption.

```
\newcommand*\caption@@par{%
  \parindent\caption@parindent\hangindent\caption@hangindent}%
```

23 Types & sub-types

\DeclareCaptionType

`\DeclareCaptionType[`⟨*options*⟩`]{`⟨*environment*⟩`}[`⟨*name*⟩`][`⟨*list name*⟩`]`

```
\newcommand*\DeclareCaptionType{%
  \RequirePackage{newfloat}%
  \DeclareFloatingEnvironment}
\@onlypreamble\DeclareCaptionType
```

\caption@ForEachType

`\caption@ForEachType{`⟨*code*⟩`}` will execute the given code for all (known) floating environments.

```
\newcommand\caption@ForEachType[1]{%
  \caption@ifundefined\ForEachFloatingEnvironment
    {\def\@elt##1{#1}%
      \caption@ifundefined\c@figure\@gobble\@elt{figure}%
      \caption@ifundefined\c@table\@gobble\@elt{table}%
      \let\@elt\relax
      \newfloat@addtohook{#1}}%
    {\ForEachFloatingEnvironment{#1}}}
```

```
\providecommand\newfloat@addtohook[1]{%
  \toks@=\expandafter{\newfloat@hook{##1}#1}%
  \edef\@tempa{\def\noexpand\newfloat@hook####1{\the\toks@}}%
  \@tempa}
```

```
\providecommand*\newfloat@hook[1]{}
```

\@stpelt

We patch \@stpelt so a list of ‘connected’ counters will be reset, too. (Like \stepcounter does in ltcounts.dtx.)

```
\newcommand*\caption@patch@stpelt{%
  \let\caption@stpelt\@stpelt
  \def\@stpelt##1{%
    \caption@stpelt{##1}%
    \begingroup
      \let\@elt\caption@stpelt
      \csname caption@cl@##1\endcsname
    \endgroup}%
  \let\caption@patch@stpelt\relax}
\@onlypreamble\caption@patch@stpelt
```

\caption@addtoreset

Like \@addtoreset from ltcounts.dtx

```
\newcommand*\caption@addtoreset[2]{%
  \caption@patch@stpelt
  \@ifundefined{caption@cl@#2}{\@namedef{caption@cl@#2}{}}{}%
  \expandafter\@cons\csname caption@cl@#2\endcsname{{#1}}}
\@onlypreamble\caption@addtoreset
```

\caption@removefromreset

Like \@removefromreset from remreset.sty

```
\newcommand*\caption@removefromreset[2]{%
  \begingroup
    \expandafter\let\csname c@#1\endcsname\caption@removefromreset
    \def\@elt##1{%
      \expandafter\ifx\csname c@##1\endcsname\caption@removefromreset
      \else
        \noexpand\@elt{##1}%
      \fi}%
    \expandafter\xdef\csname caption@cl@#2\endcsname{%
      \csname caption@cl@#2\endcsname}%
  \endgroup}
\@onlypreamble\caption@removefromreset
```

\DeclareCaptionSubType

\DeclareCaptionSubType[⟨*numbering scheme*⟩]{⟨*type*⟩}
\DeclareCaptionSubType*[⟨*numbering scheme*⟩]{⟨*type*⟩}
The starred variant provides the numbering format ⟨*type*⟩.⟨*subtype*⟩ while the non-starred variant simply uses ⟨*subtype*⟩.

```
\newcommand*\DeclareCaptionSubType{%
  \caption@teststar\caption@declaresubtype\@firstoftwo\@secondoftwo}
\@onlypreamble\DeclareCaptionSubType
```

```
\newcommand*\caption@declaresubtype[1]{%
  \@testopt{\caption@@declaresubtype{#1}}{alph}}
\@onlypreamble\caption@declaresubtype
```

```
\def\caption@@declaresubtype#1[#2]#3{%
  \@ifundefined{c@#3}%
    {\caption@Error{No float type ’#3’ defined}}%
    {\@ifundefined{c@sub#3}%
       {\caption@Debug{New subtype ‘sub#3’}%
        \newcounter{sub#3}%
        \caption@addtoreset{sub#3}{#3}%
        \@namedef{ext@sub#3}{\csname ext@#3\endcsname}%
        \caption@declaresublistentry{#3}%
        \@cons\caption@subtypelist{{#3}}}%
       {\caption@Debug{Modify caption ‘sub#3’}}%
```

Support of titletoc package

```
     \caption@ifundefined\contentsuse{}{%
       \contentsuse{sub#3}{\csname ext@sub#3\endcsname}}%
     \@namedef{sub#3name}{}%
     \@namedef{sub#3autorefname}{\csname #3name\endcsname}%
     #1% is \@firstoftwo in star form, and \@secondoftwo otherwise
     {\@namedef{p@sub#3}{}%
      \@namedef{thesub#3}{\csname the#3\endcsname.\@nameuse{#2}{sub#3}}}%
     {\@namedef{p@sub#3}{\csname the#3\endcsname}%
      \@namedef{thesub#3}{\@nameuse{#2}{sub#3}}}%
     \@namedef{theHsub#3}{\csname theH#3\endcsname.\arabic{sub#3}}%
    }}
\@onlypreamble\caption@@declaresubtype
\newcommand*\caption@declaresublistentry{%
  \caption@ifundefined\l@chapter
    {\caption@@declaresublistentry\l@subsubsection}%
    {\caption@@declaresublistentry\l@subsection}}
\@onlypreamble\caption@declaresublistentry
\newcommand*\caption@@declaresublistentry[2]{%
  \ifx#1\@undefined
    \caption@@@declaresublistentry\relax\@dottedtocline\caption@nil{#2}%
  \else
    \expandafter\caption@@@declaresublistentry#1{}{}\@dottedtocline\caption@nil{#
  \fi}
\@onlypreamble\caption@@declaresublistentry
\long\def\caption@@@declaresublistentry#1\@dottedtocline#2\caption@nil#3{%
  \def\@tempa{#1}%
% Does \l@(sub)subsection start with \@dottedtocline?
  \ifx\@tempa\@empty
% Yes
    \caption@@@@declaresublistentry{#3}#2\caption@nil
  \else
% No
    \caption@@@@declaresublistentry{#3}@{3.8em}{3.2em}\caption@nil
  \fi}
\@onlypreamble\caption@@@declaresublistentry
\def\caption@@@@declaresublistentry#1#2#3#4#5\caption@nil{%
  \expandafter\caption@@@@@declaresublistentry\expandafter
    {\csname @dotted\csname ext@#1\endcsname line\endcsname}{#1}{#3}{#4}}
\@onlypreamble\caption@@@@declaresublistentry
```

```
\newcommand*\caption@@@@@declaresublistentry[4]{%
  \@namedef{l@sub#2}{#1{2}{#3}{#4}}%
  \caption@@@@@@declaresublistentry#1{c@\csname ext@#2\endcsname depth}}
\@onlypreamble\caption@@@@@declaresublistentry
```

```
\newcommand*\caption@@@@@@declaresublistentry[2]{
  \ifx#1\relax
    \def#1##1{%
      \def\next{\@dottedtocline{##1}}%
      \@ifundefined{#2}{}{%
        \ifnum ##1>\@nameuse{#2}\relax
          \let\next\@gobblefour
        \fi}%
      \next}%
  \fi}
\@onlypreamble\caption@@@@@@declaresublistentry
```

\caption@subtypelist An `\@elt`-list containing the subtypes defined with `\DeclareCaptionSubType`.

```
\newcommand*\caption@subtypelist{}
```

\caption@For `\caption@For{`⟨*elt-list*⟩`}{`⟨*code with #1*⟩`}`
`\caption@For*{`⟨*elt-list*⟩`}{`⟨*code with #1*⟩`}`

```
\newcommand*\caption@For{\caption@withoptargs\caption@@For}
%\@onlypreamble\caption@For
```

```
\newcommand\caption@@For[3]{%
  \caption@AtBeginDocument#1{%
    \def\@elt##1{#3}%
    \@nameuse{caption@#2}%
    \let\@elt\relax}}%
%\@onlypreamble\caption@@For
```

24 subfig package adaptions

Since the subfig package is not maintained anymore, we have to make several adaptions to the caption kernel *v1.1* here. Please note that we only support the version 1.3 of the subfig package here. So older versions do not work with this version of the caption kernel, and never versions are expected to be adapted.

```
\caption@AtBeginDocument{%
  \def\@tempa{2005/06/28 ver: 1.3 subfig package}%
  \expandafter\ifx\csname ver@subfig.sty\endcsname\@tempa
    \caption@InfoNoLine{subfig package v1.3 is loaded}%
```

```
    \let\caption@setfloattype\@gobble
    \let\@dottedxxxline\sf@NEW@dottedxxxline
    \let\sf@subfloat\sf@NEW@subfloat
```

```
  \fi
  \let\sf@NEW@dottedxxxline\@undefined
  \let\sf@NEW@subfloat\@undefined}
```

```
\def\sf@NEW@dottedxxxline#1#2#3#4#5#6#7{%
  \begingroup
    \caption@setfloattype{#1}%
    \caption@setoptions{subfloat}%
    \caption@setoptions{sub#1}%
    \ifnum #3>\@nameuse{c@#2depth}\else
      \@dottedtocline{\z@}{#4}{#5}{#6}{#7}%
    \fi
  \endgroup}
\def\sf@NEW@subfloat{%
  \begingroup
    \caption@setfloattype\@captype
    \sf@ifpositiontop{%
      \maincaptiontoptrue
    }{%
      \maincaptiontopfalse
    }%
    \caption@setoptions{subfloat}%
    \caption@setoptions{sub\@captype}%
    \let\sf@oldlabel=\label
    \let\label=\subfloat@label
    \ifmaincaptiontop\else
      \advance\@nameuse{c@\@captype}\@ne
    \fi
    \refstepcounter{sub\@captype}%
    \setcounter{sub\@captype @save}{\value{sub\@captype}}%
    \@ifnextchar [%  %] match left bracket
      {\sf@@subfloat}%
      {\sf@@subfloat[\@empty]}}
```

References

[1] Anne Brüggemann-Klein:
Einführung in die Dokumentverarbeitung,
B.G. Teubner, Stuttgart, 1989

[2] Sebastian Rahtz & Heiko Oberdiek:
Hypertext marks in LaTeX,
November 12, 2007

[3] Heiko Oberdiek:
The refcount package,
2006/02/20

[4] Donald Arseneau:
Three part tables: title, tabular environment, notes,
2003/06/13