The (obsolete) caption2 package[*]

Axel Sommerfeldt
caption@sommerfee.de

2007/11/28

THIS PACKAGE IS OBSOLETE!

The caption2 package used to be an experimental side-version of the regular caption package. It was made public as beta test version without documentation in 1995 because of the strong demand for new features and adaptations to other packages like the longtable and subfigure package.

But within the next years I found no time to reintegrate some of the well-tried features into the regular caption package. So I decided to release a version 2.1 of the caption2 package in 2002 instead, which included some minor bug fixes and adaptations to the new version 2.1 of the subfigure package. Furthermore I started to write a documentation for this package, but unfortunately did not get very far with this. . .

In 2003 I finally found some (more) time, so a new regular release 3.0 of the caption package could be build in cooperation with Frank Mittelbach and Steven Cochran. It was released in December 2003 and superseded the neglected caption2 package.

(In parallel, Steven Cochran released the subfig package which superseded the subfigure package.)

So please don't use this package for new documents. It's old, it's obsolete and it starts to begin smell bad! Please ignore all hints in books or other documents which try to tell you that the caption2 package should be used instead of the caption package – these hints are outdated since December 2003.

[*] This package has version number v2.2c, last revised 2008/07/01.

How to migrate to the regular caption package?

Usually replacing caption2 by caption is sufficient because the caption package emulates most of the options and commands offered by the caption2 package. If you get some errors or wired results afterwards, please take a closer look at the caption package documentation which will hopefully help you clearing these problems. You will also find a section called ‘Compatibility to older versions’ there which should help you with the migration process. If all this should fail you can write me an e-mail asking for help.

What will happen to this package?

The caption2 package is still some kind of supported, that means it will be part of future releases of the caption package bundle, and bugs will still be fixed so existing documents using this package will still compile. But it will *not* be enhanced in the future.

This means migrating to the actual caption package should not be necessary for old documents – they should still compile fine as they are. If not, please don’t hesitate to write me an e-mail asking for maintainance.

1 The Implementation

1.1 Identificaton

```
\NeedsTeXFormat{LaTeX2e}[1994/12/01]
\ProvidesPackage{caption2}[2008/07/01 v2.2c Customising captions (AS)]
```

```
\PackageWarning{caption2}{%
  ****************************************************\MessageBreak
  THIS PACKAGE IS OBSOLETE:\MessageBreak
  This package attempts to provide an ‘caption2’\MessageBreak
  package v2.0/2.1 author environment so that OLD\MessageBreak
  documents can be successfully processed. It should\MessageBreak
  NOT be used for NEW documents! New documents should\MessageBreak
  use the regular ‘caption’ package v3.x instead.\MessageBreak
  ****************************************************\@gobbletwo}%
```

1.2 Loading the caption3 kernel

```
\RequirePackage{caption3}[2007/09/01] % needs v3.1 or newer
```

1.3 Check against the regular caption package

```
\@ifpackageloaded{caption}{%
  \PackageError{caption2}{%
    You can’t use both, the (current) caption *and*\MessageBreak
    the (obsolete) caption2 package}\caption@eh
  \endinput
}{}
```

1.4 Preliminary declarations

\captionlabeldelim
\captionlabelsep

\captionlabeldelim & \captionlabelsep will hold the iterim space between caption label and text. (\captionlabeldelim will be typeset within \captionlabelfont, \captionlabelsep not.)

```
\newcommand*\captionlabeldelim{}
\newcommand*\captionlabelsep{}
```

\ifcaptionwidth

Either \captionmargin (with specifies an extra margin) or \captionwidth (with specifies an explicit width) can be set, therefore we need the flag \ifcaptionwidth to determine with parameter we should pay attention to.

```
\newcommand*\ifcaptionwidth{\ifdim\captionwidth>\z@}
```

\captionindent

\captionindent will be used in caption style indent and specifies the indention after the first line.

```
\let\captionindent\caption@indent
```

\ifcaptionlabel

If \ifcaptionlabel is not set the caption label should be suppressed; we need this flag to support the \caption* command.

```
\def\ifcaptionlabel{%
  \ifcaption@star
    \@tempswafalse
  \else
    \@tempswatrue
  \fi
  \if@tempswa}
\def\captionlabeltrue{\caption@starfalse}
\def\captionlabelfalse{\caption@startrue}
```

\ifonelinecaptions

If \ifonelinecaptions is set we support the LaTeX base style 'one line captions', that means the caption will be typeset centered if it fits to one line.

```
\def\ifonelinecaptions{%
  \caption@ifslc{\@tempswatrue}{\@tempswafalse}%
  \if@tempswa}
\def\onelinecaptionstrue{\caption@setbool{slc}{1}}
\def\onelinecaptionsfalse{\caption@setbool{slc}{0}}
```

\ifignoreLTcapwidth

If \ifignoreLTcapwidth is set we ignore the \LTcapwidth of longtable.

```
\newif\ifignoreLTcapwidth
```

\normalcaptionparams

\normalcaptionparams resets all caption related parameters to it's normal default values. \captionfont will be set to \captionsize so setting the obsolete \captionsize will still work. Same story with \captiondelim and the obsolete \captionlabeldelim.

```
\newcommand*\normalcaptionparams{%
  \let\captionsize\@empty
  \renewcommand*\captionfont{\captionsize}%
  \let\captionlabelfont\@empty
  \renewcommand*\captionlabeldelim{:}%
  \renewcommand*\captionlabelsep{\space}%
  \setcaptionmargin\z@
  \setlength\captionindent\z@
  \onelinecaptionstrue}
```

\defcaptionstyle
\newcaptionstyle
\renewcaptionstyle

These macros will define a new caption style. \newcaptionstyle and \renewcaptionstyle will additionally check if the caption style already exists or not.

```
\newcommand*\defcaptionstyle[1]{%
  \@namedef{caption@@#1}}
%
\newcommand*\newcaptionstyle[1]{%
  \expandafter\ifx\csname caption@@#1\endcsname\relax
    \expandafter\defcaptionstyle
  \else
    \PackageError{caption2}{Caption style `#1' already defined}{\caption@eh}%
    \expandafter\@gobbletwo
  \fi
  {#1}}
%
\newcommand*\renewcaptionstyle[1]{%
  \expandafter\ifx\csname caption@@#1\endcsname\relax
    \PackageError{caption2}{Caption style `#1' undefined}{\caption@eh}%
    \expandafter\@gobbletwo
  \else
```

```
    \expandafter\defcaptionstyle
  \fi
  {#1}}
```

\dummycaptionstyle

This macro will also define a new caption style, but a one which is based on the actual set caption style. Therefore you can't set a caption style made with this command with `\captionstyle` – we check this to avoid an endless recursion.

```
\newcommand*\dummycaptionstyle[2]{%
  \defcaptionstyle{#1}{%
    \expandafter\ifx\csname caption@@\caption@style\expandafter\endcsname%
                    \csname caption@@#1\endcsname
      \PackageError{caption2}{You can't use the caption style '#1' directy}{%
        The caption style '#1' is only a dummy and does not really exists.%
        \MessageBreak You have to redefine it (with \protect\renewcaptionstyle)
        before you can select\MessageBreak it with \protect\captionstyle.
        \space\caption@eh}%
    \else
      #2\usecaptionstyle\caption@style
    \fi}}
```

\captionstyle

`\captionstyle` sets the actual caption style. It includes a check if the given caption style is defined or not.

```
\newcommand*\captionstyle[1]{%
  \expandafter\ifx\csname caption@@#1\endcsname\relax
    \PackageError{caption2}{Undefined caption style '#1'}{\caption@eh}%
  \else
    \def\caption@style{#1}%
  \fi}
```

style 'normal'
style 'center'
style 'centerlast'
style 'flushleft'
style 'flushright'
style 'hang'
style 'indent'

The predefined caption styles 'normal', 'center', 'flushleft', 'flushright', 'centerlast', 'hang', 'hang+X', and 'indent'. Because they are quite similar they all are based on the macro `\caption@make`.

```
\newcaptionstyle{normal}{\caption@make{normal}}
\newcaptionstyle{center}{\caption@make{center}}
\newcaptionstyle{centerlast}{\caption@make{centerlast}}
\newcaptionstyle{flushleft}{\caption@make{flushleft}}
\newcaptionstyle{flushright}{\caption@make{flushright}}
\newcaptionstyle{hang}{\caption@make{hang}}
\newcaptionstyle{hang+center}{\caption@make{hang@center}}
\newcaptionstyle{hang+centerlast}{\caption@make{hang@centerlast}}
\newcaptionstyle{hang+flushleft}{\caption@make{hang@flushleft}}
\newcaptionstyle{indent}{\caption@make{indent}}
```

\caption@makecaption

Our predefined caption styles. `\caption@makecaption` takes the style name as parameter, it does the common stuff and calls a macro (build out of the style name) to do the uncommon stuff if neccessary.

```
\newcommand*\caption@makecaption[1]{%
  \usecaptionmargin
```

```
  \ifcaptionlabel
    \def\caption@label{%
```

```
      {\captionlabelfont\captionlabel\captionlabeldelim}\captionlabelsep}%
  \else
    \let\caption@label\@empty
  \fi
```

```
  \captionfont
  \onelinecaption
    {\caption@label\captiontext}%
    {\parbox[b]\captionlinewidth{\strut\@nameuse{caption@@@#1}\par}\par}}
```

\caption@@@normal

The ‘normal’ caption style. Just typeset caption (label & text) as paragraph.

```
\newcommand*\caption@@@normal{%
  \caption@label\captiontext}
```

\caption@@@center

The ‘center’ caption style. Typeset the caption centered within a parbox.

```
\newcommand*\caption@@@center{%
  \centering\caption@label\captiontext}%
```

\caption@@@centerlast

The ‘centerlast’ caption style. The idea how to do this was taken from Brüggemann-Klein[6], it is also mentioned in Kopka[7, p227].

```
\newcommand*\caption@@@centerlast{%
  \centerlast\caption@label\captiontext}
```

\caption@@@flushleft

The ‘flushleft’ caption style. Typeset the caption raggedright within a parbox.

```
\newcommand*\caption@@@flushleft{%
  \raggedright\caption@label\captiontext}%
```

\caption@@@flushright

The ‘flushright’ caption style. Typeset the caption raggedleft within a parbox.

```
\newcommand*\caption@@@flushright{%
  \raggedleft\caption@label\captiontext}%
```

\caption@@@hang
\caption@hangplus

The ‘hang’ caption style. This code was taken from The LaTeX Companion[5, p155] and modified.

```
\newcommand*\caption@@@hang{%
  \sbox\@tempboxa{\caption@label}%
  \hangindent\wd\@tempboxa\noindent
  \usebox\@tempboxa\caption@hangplus\captiontext}
```

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

\caption@@@hang@center

The ‘hang+flushleft’ caption style.

```
\newcommand*\caption@@@hang@center{%
  \let\caption@hangplus\centering\caption@@@hang}
```

caption@@@hang@centerlast

The ‘hang+flushleft’ caption style.

```
\newcommand*\caption@@@hang@centerlast{%
  \let\caption@hangplus\centerlast\caption@@@hang}
```

`\caption@@@hang@flushleft` The ‘hang+flushleft’ caption style.

```
\newcommand*\caption@@@hang@flushleft{%
  \let\caption@hangplus\raggedright\caption@@@hang}
```

`\caption@@@indent` The ‘indent’ caption style. Is is quite like the ‘hang’ style but the indention is given as `\captionindent`.

```
\newcommand*\caption@@@indent{%
  \hangindent\captionindent\noindent
  \caption@label\captiontext}
```

1.5 Declaration of options

`normal`
`center`
`centerlast,anne`
`flushleft`
`flushright`
`hang,isu`
`indent`

These options will set the caption style. (‘normal’ is the default one.) The options ‘anne’ and ‘isu’ are for backward compatibility only.

```
\DeclareOption{normal}{\captionstyle{normal}}
\DeclareOption{center}{\captionstyle{center}}
\DeclareOption{centerlast}{\captionstyle{centerlast}}
\DeclareOption{flushleft}{\captionstyle{flushleft}}
\DeclareOption{flushright}{\captionstyle{flushright}}
\DeclareOption{anne}{\ExecuteOptions{centerlast}}
\DeclareOption{hang}{\captionstyle{hang}}
\DeclareOption{hang+center}{\captionstyle{hang+center}}
\DeclareOption{hang+centerlast}{\captionstyle{hang+centerlast}}
\DeclareOption{hang+flushleft}{\captionstyle{hang+flushleft}}
\DeclareOption{isu}{\ExecuteOptions{hang}}
\DeclareOption{indent}{\captionstyle{indent}}
```

`scriptsize`
`footnotesize`
`small`
`normalsize`
`large,Large`

These options will set the caption size. We use `\g@addto@macro` so more that one option can be set.

```
\DeclareOption{scriptsize}{\g@addto@macro\captionsize\scriptsize}
\DeclareOption{footnotesize}{\g@addto@macro\captionsize\footnotesize}
\DeclareOption{small}{\g@addto@macro\captionsize\small}
\DeclareOption{normalsize}{\g@addto@macro\captionsize\normalsize}
\DeclareOption{large}{\g@addto@macro\captionsize\large}
\DeclareOption{Large}{\g@addto@macro\captionsize\Large}
```

`up,it,sl,sc`
`md,bf`
`rm,sf,tt`

These options will set the caption label.

```
\DeclareOption{up}{\g@addto@macro\captionlabelfont\upshape}
\DeclareOption{it}{\g@addto@macro\captionlabelfont\itshape}
\DeclareOption{sl}{\g@addto@macro\captionlabelfont\slshape}
\DeclareOption{sc}{\g@addto@macro\captionlabelfont\scshape}
\DeclareOption{md}{\g@addto@macro\captionlabelfont\mdseries}
\DeclareOption{bf}{\g@addto@macro\captionlabelfont\bfseries}
\DeclareOption{rm}{\g@addto@macro\captionlabelfont\rmfamily}
\DeclareOption{sf}{\g@addto@macro\captionlabelfont\sffamily}
\DeclareOption{tt}{\g@addto@macro\captionlabelfont\ttfamily}
```

`oneline`
`nooneline`

These options will set the ‘oneline’ flag. (‘oneline’ is the default.)

```
\DeclareOption{oneline}{\onelinecaptionstrue}
\DeclareOption{nooneline}{\onelinecaptionsfalse}
```

\caption@setpackage

A helper macro, a value of 1 within parameter #2 will activate the support of the package given in parameter #1, a value of 0 will deactivate it.

```
\newcommand*\caption@setpackage[1]{\@namedef{caption@pkt@#1}}
```

float
longtable
subfigure

These options will enable or suppress the support of the packages float, longtable, and subfigure.

```
\DeclareOption{float}{%
  \caption@twozerofalse\caption@setpackage{float}{1}}
\DeclareOption{longtable}{%
  \caption@twozerofalse\caption@setpackage{longtable}{1}}
\DeclareOption{subfigure}{%
  \caption@twozerofalse\caption@setpackage{subfigure}{1}}
```

none
all

These options will enable or suppress the support of all the above packages.

```
\DeclareOption{none}{\caption@twozerofalse
  \caption@setpackage{float}{0}\caption@setpackage{longtable}{0}%
  \caption@setpackage{subfigure}{0}}
\DeclareOption{all}{\ExecuteOptions{float,longtable,subfigure}}
```

ruled
boxed

The option ‘ruled’ introduced in caption v1.2 is obsolete now, but we will still support it. The option ‘boxed’ was introduced in version 2.0 and is obsolete now, too.

```
\newif\ifcaption@ruled
\DeclareOption{ruled}{\caption@ruledtrue}
\DeclareOption{boxed}{}
```

ignoreLTcapwidth

This option will make the caption code ignore the setting of \LTcapwidth and use the setting of \setcaptionmargin or \setcaptionwidth instead.

```
\DeclareOption{ignoreLTcapwidth}{\ignoreLTcapwidthtrue}
```

debug

This option will put additional debug information in the log file.

```
\DeclareOption{debug}{\captionsetup{debug}}
```

1.6 Execution of options

Now we set the default values and start processing the options. (If \caption@twozero is set to true (default) we will emulate the package load algorithm of caption v2.0: If the package is already loaded patch it, otherwise do nothing.)

```
\newif\ifcaption@twozero
\normalcaptionparams
\ExecuteOptions{none,normal}
\caption@twozerotrue
\ProcessOptions*
\ifcaption@twozero
  \PackageInfo{caption2}{Running in caption2 v2.0 compatibility mode}
\fi
```

1.7 More declarations

\captionof
\captionof*

\captionof resp. \captionof* will just set \@captype and do the normal \caption resp. \caption*, so we can also typeset captions outside floating environments.

```
\def\captionof{\@ifstar{\caption@of{\caption*}}{\caption@of\caption}}
\newcommand*\caption@of[2]{\def\@captype{#2}#1}
```

\abovecaptionskip
\belowcaptionskip

Not all document classes define \abovecaptionskip and \belowcaptionskip (like ucthesis), so we do it here if not already done.

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

\captionlinewidth
\captionlabel
\captiontext

These values are only set and used within the caption code itself. \captionlinewidth will be set to the given vertical space for the caption, normally this is \linewidth. (This value was called \realcaptionwidth within caption2 2.0, so we will offer this, too.)

\captionlabel and \captiontext will be set to the caption label resp. the caption text. (Because \captionlabel and \captiontext will be locally defined with \def we do not need to define them here.)

```
\newdimen\captionlinewidth
\newdimen\realcaptionwidth
```

\usecaptionmargin

A helper macro for caption style authors: It calculates \leftskip and \rightskip out of \captionlinewidth and \captionmargin resp. \captionwidth. Also \captionlinewidth will be corrected to the appropriate value.

```
\newcommand*\usecaptionmargin{%
  \ifcaptionwidth
    \leftskip\captionlinewidth
    \advance\leftskip by -\captionwidth
    \divide\leftskip by 2
    \rightskip\leftskip
    \captionlinewidth\captionwidth
  \else
    \leftskip\captionmargin
    \rightskip\captionmargin
    \advance\captionlinewidth by -2\captionmargin
  \fi
  \realcaptionwidth\captionlinewidth}
```

\onelinecaption

This macro definition helps setting captions the LaTeX base classes way: If \ifonelinecaptions is set and the 1st argument fits within \captionlinewidth, we typeset it centered – otherway we typeset the 2nd argument. (We use the savebox \@tempboxa as helper for this.)

```
\newcommand\onelinecaption[1]{%
  \let\next\@firstofone
  \ifonelinecaptions
    \sbox\@tempboxa{#1}%
    \ifdim\wd\@tempboxa >\captionlinewidth
    \else
      \def\next{{\centering\usebox\@tempboxa\par}\@gobble}%
    \fi
  \fi\next}
```

\usecaptionstyle First we check if we are inside a caption – if \captiontext is undefined we are not. If we are we call the appropriate caption definition.

```
\newcommand*\usecaptionstyle[1]{%
  \@ifundefined{captiontext}{%
    \PackageError{caption2}{You can't use \protect#1
      in normal text}{The usage of \protect#1 is only
      allowed inside code declared with\MessageBreak \protect\defcaptionstyle,
      \protect\newcaptionstyle \space or \protect\renewcaptionstyle.
      \space\caption@eh}
  }{%
    \@ifundefined{caption@@#1}%
      {\PackageError{caption2}{Caption style `#1' undefined}{\caption@eh}}%
      {\let\caption@make\caption@makecaption
       \@nameuse{caption@@#1}}%
  }}
```

\@makecaption This is the heart of the caption2 package – the redefinition of the core caption code. It was taken from the LaTeX 2ε standard classes and modified. It's very easy – apart from using \abovecaptionskip and \belowcaptionskip we just set \captionlinewidth, \captionlabel and \captiontext to its appropriate values and using the code of the actual caption style via \usecaptionstyle.

```
\renewcommand\@makecaption[2]{%
  \vskip\abovecaptionskip
  \captionlinewidth\hsize
  \realcaptionwidth\hsize
  \def\captionlabel{#1}%
  \def\captiontext{#2}%
  \usecaptionstyle\caption@style
  \vskip\belowcaptionskip}
```

1.8 Support of other packages

\caption@ifpackage This macro will execute the code needed to support the package named within argument #1. The parameter #2 is the command which shows if the package is loaded – it is defined, it is already loaded, otherwise not. The parameter #3 contains code which will be executed if no support is required – this is for cleanup purposes. The final parameter #4 contains the code itself.

```
\newcommand*\caption@ifpackage[3]{%
  \if1\@nameuse{caption@pkt@#1}%
    \@ifundefined{#2}%
      {\let\next\AtBeginDocument}%
      {\let\next\@firstofone}%
  \else\ifcaption@twozero
    \@ifundefined{#2}%
      {#3\let\next\@gobble}%
      {\let\next\@firstofone}%
  \else
    #3\let\next\@gobble
  \fi\fi
  \expandafter\let\csname caption@pkt@#1\endcsname\undefined
  \caption@ifdebug{%
    \ifx\next\@gobble\PackageInfo{caption2}{#1 => gobble}%
    \else\ifx\next\@firstofone\PackageInfo{caption2}{#1 => firstofone}%
    \else\ifx\next\AtBeginDocument\PackageInfo{caption2}{#1 => AtBeginDocument}%
    \fi\fi\fi}{}%
  \next}
```

1.8.1 Support of the float package

```
\caption@ifpackage{float}{floatc@plain}{}{%
  \ifx\floatc@plain\relax
    \PackageWarning{caption2}{%
      Option `float' was set but there is no float package loaded}
  \else
    \PackageInfo{caption2}{float package v1.2 (or newer) detected}
```

\caption@floatc

First we define a helper macro to typeset the caption via \usecaptionstyle, the 1st parameter is the caption style name, the 2nd and 3rd are the caption label and text.

caption2 has the goal not to modify the output just by loading it (without options), therefore we have to be tricky here to support \@fs@cfont which is in fact the same as our \captionlabelfont. So we test if a \captionlabelfont has been set by the user – if not \@fs@cfont will be used, otherwise \captionlabelfont.

```
    \newcommand\caption@floatc[3]{%
      \ifx\captionlabelfont\@empty
        \let\captionlabelfont\@fs@cfont
      \fi
      \captionlinewidth\hsize
      \realcaptionwidth\hsize
      \def\captionlabel{#2}%
      \def\captiontext{#3}%
      \usecaptionstyle{#1}}
```

\floatc@plain

Now we can redefine the caption code of the float package. Here we redefine \floatc@plain to use our caption code, so plain and boxed float types will use the actual caption style set by the user.

```
    \renewcommand*\floatc@plain{\caption@floatc{\caption@style}}
```

\floatc@ruled

The support of the ruled float type is a little more complex. First we define a caption style 'ruled' so the end-user can change this caption style afterwards. If the (obsolete) option 'ruled' is set, we define it in a caption v1.x compatible way, otherwise we define it in a float compatible way.

Then we redefine \floatc@ruled so the caption style 'ruled' will be used.

```
    \ifcaption@ruled
      \dummycaptionstyle{ruled}{\onelinecaptionsfalse\setcaptionmargin{\z@}}%
    \else
      \newcaptionstyle{ruled}{%
    \ifcaptionlabel
     {\@fs@cfont\captionlabel}\space%
    \fi\captiontext\par}%
  \fi
```

```
  \renewcommand*\floatc@ruled{\caption@floatc{ruled}}
```

\caption@of

Typesetting captions outside floats is not so easy with redefined floats, because

- The caption code of the float package needs not only \@captype defined, but \@fs@capt (the command which will typeset the caption itself) either.
- The caption is only saved within a \vbox, so the float package can typeset the caption later at it's float style specific place (that means at top or at the bottom of the float).

Here is the new code: First we check if it's a restyled float by checking if \fst@<floattype> is defined. If yes, we use this command (it will define \@fs@capt). Then we execute \@float@setevery, if it exists (that means we are dealing with the float package 1.3 or newer here). Now comes the basic trick: We redefine the caption typesetting command \@fs@capt, so it will close the \vbox, typeset the caption outside the vbox and finally start the group again so the original \@fs@capt is happy with closing the group.

```
  \renewcommand*\caption@of[2]{\def\@captype{#2}%
   \@ifundefined{fst@#2}{}{%
    \@nameuse{fst@#2}%
    \@ifundefined{@float@setevery}{}{\@float@setevery{#2}}%
    \let\caption@fs@capt\@fs@capt
    \let\@fs@capt\caption@of@float}%
   #1}
```

```
  \newcommand\caption@of@float[2]{\egroup
   \vskip\abovecaptionskip
   \normalsize\caption@fs@capt{#1}{#2}%
   \vskip\belowcaptionskip
   \bgroup}%
```

```
 \fi}
```

1.8.2 Support of the longtable package

```
\caption@ifpackage{longtable}{LT@makecaption}{}{%
 \ifx\LT@makecaption\relax
  \PackageWarning{caption2}{%
   Option ‘longtable’ was set but there is no longtable package loaded}
 \else
  \PackageInfo{caption2}{longtable package v3.15 (or newer) detected}
```

\LT@makecaption

David Carlisle was so kind to introduce a macro called \LT@makecaption in version 3.15 of the longtable package which typeset the caption and can be easily redefined.

This is the original definition:

```
\def\LT@makecaption#1#2#3{%
 \LT@mcol\LT@cols c{\hbox to\z@{\hss\parbox[t]\LTcapwidth{%
```

```
      ⟨typeset #1{#2: }#3 as caption⟩
      \endgraf\vskip\baselineskip}%
    \hss}}}
```

So we do here: First we define a new (dummy) caption style ‘longtable’, than we redefine `\LT@makecaption` so this style will be used. (Remember: `#1` is `\@gobble` in star form of `\caption`, and `\@firstofone` otherwise.)

```
    \dummycaptionstyle{longtable}{}

    \renewcommand\LT@makecaption[3]{%
      \LT@mcol\LT@cols c{\hbox to\z@{\hss\parbox[t]\hsize{%
        \ifignoreLTcapwidth
        \else
          \setcaptionwidth\LTcapwidth
        \fi
        \captionlinewidth\hsize
        \realcaptionwidth\hsize
        \captionlabelfalse#1\captionlabeltrue
        \def\captionlabel{#2}%
        \def\captiontext{#3}%
        \usecaptionstyle{longtable}%
        \endgraf\vskip\baselineskip}%
      \hss}}}

  \fi}
```

1.8.3 Support of the subfigure package

Some of the following code will not work within `\if`, because of the (yet) undefined `\if`*xxx*s. So we simply define the critical code within the helper commands `\setsubcapstyle` and `\caption@makesubcaption` already here.

`\setsubcapstyle` This sets the subcaptionstyle to a appropriate value.

If `\ifsubcapraggedright` is undefined (it was introduced into v2.1 of the subfigure package) we define it first.

```
\newcommand*\setsubcapstyle{%
  \@ifundefined{subcapraggedrightfalse}{%
    \newif\ifsubcapraggedright}{}%
  \ifsubcaphang
    \ifsubcapcenter
      \subcapstyle{hang+center}%
    \else\ifsubcapcenterlast
      \subcapstyle{hang+centerlast}%
    \else\ifsubcapraggedright
      \subcapstyle{hang+flushleft}%
    \else
      \subcapstyle{hang}%
    \fi\fi\fi
  \else\ifsubcapcenter
    \subcapstyle{center}%
  \else\ifsubcapcenterlast
    \subcapstyle{centerlast}%
  \else\ifsubcapraggedright
    \subcapstyle{flushleft}%
  \else
    \subcapstyle{normal}%
  \fi\fi\fi\fi}
```

`\caption@makesubcaption`

This will typeset the subcaption. We just set all our `\caption`*xxx* values to the values of `\subcap`*xxx* and typeset the caption like subfigure within a `\hbox`, but with the help of `\usecaptionstyle`.

But this is not as easy as it seems. We typeset the caption like this:

```
\captionfont
  {\capionlabelfont\captionlabel\captionlabeldelim}%
\captionlabelsep\captiontext
```

Within subfigure 2.0 the caption will be set quite similar to:

```
\subcapsize
  {\subcaplabelfont\captionlabel}%
\space\captiontext
```

But within subfigure 2.1 this has changed to:

```
\subcapsize
  {\subcaplabelfont\captionlabel}%
\hskip\subfiglabelskip
  {\subcapfont\captiontext}}
```

So we have to be tricky here: We set `\captionlabelfont` to `\normalfont` plus `\subcapsize` & `\subcaplabelfont`, so the font setting in `\captionfont` will not affect the caption label in subfigure captions.

Note that `\hfil` has changed to `\hss` from subfigure 2.0 to 2.1, so we use `\caption@subfig@hss` instead. (We will define this later on.)

```
\newcommand\caption@makesubcaption[2]{%
  \renewcommand*\captionfont{\subcapsize\subcapfont}%
  \renewcommand*\captionlabelfont{\normalfont\subcapsize\subcaplabelfont}%
  \let\captionlabeldelim\subcaplabeldelim
  \let\captionlabelsep\subcaplabelsep
  \ifsubfigcapwidth
    \setcaptionwidth\subfigcapwidth
  \else
    \setcaptionmargin\subfigcapmargin
  \fi
  \setlength\captionmargin\subfigcapmargin
  \setlength\captionwidth\subfigcapwidth
  \captionindent\subcapindent
  \ifsubcapnooneline
    \onelinecaptionsfalse
  \else
    \onelinecaptionstrue
  \fi
  \hbox to\@tempdima{%
    \caption@subfig@hss\parbox[t]\@tempdima{%
      \captionlinewidth\@tempdima
      \realcaptionwidth\@tempdima
      \captionlabeltrue
      \def\captionlabel{#1}%
      \def\captiontext{\ignorespaces #2}%
      \usecaptionstyle\caption@substyle}%
    \caption@subfig@hss}}
```

If the subfigure support is not needed, we throw the helper macros in the garbage can.

```
\caption@ifpackage{subfigure}{@makesubfigurecaption}{%
  \let\setsubcapstyle\undefined
  \let\caption@makesubcaption\undefined}{%
  \ifx\@makesubfigurecaption\relax
    \PackageWarning{caption2}{%
      Option `subfigure' was set but there is no subfigure package loaded}
    \let\setsubcapstyle\undefined
    \let\caption@makesubcaption\undefined
  \else
```

Some stuff has changed from version 2.0 to 2.1 of the subfigure package, so we make a branch here. If `\subcapfont` is undefined we assume v2.0, otherwise we assume v2.1 or newer.

```
    \ifx\subcapfont\undefined
      \PackageInfo{caption2}{subfigure package v2.0 detected}
```

`\subcapfont` We define `\subcapfont` here so we can use it later in common code for subfigure v2.0 and v2.1 (or newer).

```
      \let\subcapfont\@empty
```

`\subfigcapwidth` `\setsubcapmargin` `\setsubcapwidth` Analogous to `\captionwidth`, `\setcaptionmargin`, and `\setcaptionwidth` we define `\subfigcapwidth`, `\setsubcapmargin`, and `\setsubcapwidth`.

Note: `\subfigcapmargin` is a command in v2.0 of subfigure. So we make `\subfigcapwidth` a command, too.

```
      \newcommand*\subfigcapwidth{\z@}
      \newcommand*\setsubcapmargin{%
        \subfigcapwidthfalse
        \renewcommand*\subfigcapmargin}
      \newcommand*\setsubcapwidth{%
        \subfigcapwidthtrue
        \renewcommand*\subfigcapwidth}
```

`\subcaplabelsep` Analogous to `\captionlabelsep` we define `\subcaplabelsep`.

```
      \newcommand*\subcaplabelsep{\space}
```

`\caption@subfig@hss` This will be uses within the caption code itself.

```
      \let\caption@subfig@hss\hfil
```

```
  \else
    \PackageInfo{caption2}{subfigure package v2.1 (or newer) detected}
```

\subfigcapwidth
\setsubcapmargin
\setsubcapwidth

Analogous to \captionwidth, \setcaptionmargin, and \setcaptionwidth we define \subfigcapwidth, \setsubcapmargin, and \setsubcapwidth. Note: \subfigcapmargin is a length in v2.1 of subfigure. So we make \subfigcapwidth a length, too.

```
    \newdimen\subfigcapwidth
    \newcommand*\setsubcapmargin{%
      \subfigcapwidthfalse
      \setlength\subfigcapmargin}
    \newcommand*\setsubcapwidth{%
      \subfigcapwidthtrue
      \setlength\subfigcapwidth}
```

\subcaplabelsep

Analogous to \captionlabelsep we define \subcaplabelsep.

```
    \newcommand*\subcaplabelsep{\hskip\subfiglabelskip}
```

\caption@subfig@hss

This will be uses within the caption code itself.

```
    \let\caption@subfig@hss\hss
```

```
  \fi
```

Here starts the common code for subfigure v2.0 and v2.1.

\ifsubfigcapwidth
\subcapindent
\subcaplabeldelim

Analogous to \ifcaptionwidth, \captionindent & \captionlabeldelim we define \ifsubfigcapwidth, \subcapindent & \subcaplabeldelim

```
  \newif\ifsubfigcapwidth
  \newdimen\subcapindent
  \newcommand*\subcaplabeldelim{}
```

\subcapstyle

Analogous to \captionstyle we define \subcapstyle and set it (via \setsubcapstyle) to a appropriate value.

```
  \newcommand*\subcapstyle[1]{%
    \expandafter\ifx\csname caption@@#1\endcsname\relax
      \PackageError{caption2}{Undefined caption style `#1'}{\caption@eh}%
    \else
      \def\caption@substyle{#1}%
    \fi}
  \setsubcapstyle
```

\@thesubfigure
\@thesubtable

The subfigure package makes use of \subcaplabelfont and \subfiglabelskip within its \@thesub*xxx* macros. This is totally in contrast to the way the caption2 package handle these settings. So we redefine the \@thesub*xxx* to be just the plain label and nothing else.

```
  \renewcommand*\@thesubfigure{\thesubfigure}
  \renewcommand*\@thesubtable{\thesubtable}
```

\@makesubfigurecaption
\@makesubtablecaption

Now we are ready to redefine \@makesubfigurecaption.

```
    \let\@makesubfigurecaption\caption@makesubcaption
    \let\@makesubtablecaption\caption@makesubcaption
```

```
  \fi}
```

That’s all folks!

```
\let\caption@ifpackage\undefined
```

References

[1] Anselm Lingnau: *An Improved Environment for Floats*, 2001/11/08

[2] David Carlisle: *The longtable package*, 2000/10/22

[3] Sebastian Rahtz and Leonor Barroca: *A style option for rotated objects in LaTeX*, 1997/09/26

[4] Steven D. Cochran: *The subfigure package*, 2002/02/14

[5] Michel Goossens, Frank Mittelbach and Alexander Samarin: *The LaTeX Companion*, Addison-Wesley, Reading, Massachusetts, 1994.

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

[7] Helmut Kopka: *LaTeX– Erweiterungsmöglichkeiten*, 3. überarbeitete Auflage, Addison-Wesley, Bonn, 1991