

The `ccaption` package*

Peter Wilson
Catholic University of America
Now at `peter.r.wilson@boeing.com`

2001/01/02

Abstract

The `ccaption` package provides for ‘continuation’ captions, unnumbered captions, bilingual captions, and an ‘anonymous’ caption (a legend) that can be used in any environment. It also provides commands to define captions that can be used outside float environments. Further, it provides a mechanism for creating new types of float environments and subfloats.

The package has been tested in conjunction with the `rotating`, `caption2`, `subfigure` and `endfloat` packages.

Contents

1	Introduction	1
2	The <code>ccaption</code> package	2
2.1	Continuation captions and legends	2
2.2	Bilingual captions	6
2.3	Use with the <code>subfigure</code> package	7
2.4	Use with the <code>endfloat</code> package	9
2.5	New float environments	10
2.6	How L ^A T _E X makes captions	11
3	The package code	14
3.1	Continuation captions and legends	15
3.2	Non-float captions	17
3.3	Bilingual captions	17
3.4	The <code>subfigure</code> options	19
3.4.1	Option <code>subfigure20</code>	20
3.4.2	Option <code>subfigure21</code>	23
3.5	New floats	26
A	The perils of empty	29

*This file has version number v2.6d, last revised 2001/01/02.

List of Tables

1	A multi-part table	3
2	Another table	4
	A legendary table	4
	A legendary table	4

List of Figures

1	Long	7
1	Lang	7
2	Long English	7
	Lang Deutsch	7
3	Short English	7
4	A picture is worth a thousand words	13
5	A different kind of figure caption	13

1 Introduction

Some publishers require that documents that include multi-part tables use a *continuation caption* on all but the first part of the multi-part table. For the times where such a table is specified by the author as a set of tables, the `ccaption` package provides a simple ‘continuation’ caption command to meet this requirement. It also provides a facility for an ‘anonymous’ caption which can be used in any float environment. The package has been tested with the `rotating`, `caption2`, `subfigure` (v2.0 and v2.1) and the `endfloat` packages.

Captions can be defined that are suitable for use in non-float environments, such as placing a picture in a minipage and captioning it just as though it had been put into a normal figure environment. Further, a mechanism is provided for defining new float environments.

These facilities were originally developed in support of a suite for typesetting ISO international standard [Wil96], but they are generally applicable. This manual is typeset according to the conventions of the L^AT_EX DOCSTRIP utility which enables the automatic extraction of the L^AT_EX macro source files [GMS94].

Section 2 provides a short overview of the commands in the package and shows some examples of their use, while §2.6 gives examples of how L^AT_EX’s captioning style can be changed without the use of any package. The implementation is given in Section 3.

2 The `ccaption` package

The package takes one option, namely either `subfigure20`, for use in conjunction with version 2.0 of Steven Douglas Cochran’s `subfigure` package [Coc95], or `subfigure21` for use with version 2.1 of the package¹.

¹Available from <http://almond.srv.cs.cmu.edu/afs/cs/usr/sdc/www/latex/subfigure.html>

Table 1: A multi-part table

just a single line	1
--------------------	---

Table 1: Continued

just a single line	2
--------------------	---

The L^AT_EX `\caption` command is used within either a `table` or a `figure` environment to produce appropriately labelled and numbered captions. The captioning text may also be listed via a `\listoftables` and/or `\listoffigures` command.

2.1 Continuation captions and legends

`\contcaption` The `\contcaption{<text>}` command can be used to put a ‘continuation’ or ‘concluded’ caption into a float environment. It neither increments the float number nor makes any entry into a float listing, but it does repeat the numbering of the previous `\caption` command.

Table 1 illustrates the use of the `\contcaption` command. The table was produced from the following code.

```
\begin{table}
\centering
\caption{A multi-part table} \label{tab:m}
\begin{tabular}{lc} \hline
just a single line & 1 \\ \hline
\end{tabular}
\end{table}

\begin{table}
\centering
\contcaption{Continued}
\begin{tabular}{lc} \hline
just a single line & 2 \\ \hline
\end{tabular}
\end{table}

\begin{table}
\centering
\contcaption{Concluded}
\begin{tabular}{lc} \hline
just a single line & 3 \\ \hline
\end{tabular}
\end{table}
```

`\legend` The `\legend{<text>}` command can be used to put an anonymous caption into

Table 1: Concluded
just a single line 3

<u>Table 2: Another table</u>	
A legendary table	5
with two lines	6

The legend

a float environment.

For example, the following code was used to produce the two-line table 2. The `\legend` command can be used within a float independently of any `\caption` command.

```
\begin{table}
\centering
\caption{Another table} \label{tab:legend}
\begin{tabular}{lc} \hline
A legendary table & 5 \\
with two lines    & 6 \\ \hline
\end{tabular}
\legend{The legend}
\end{table}
```

Title legend

This is a marginal note with a legend.

Captioned floats are usually thought of in terms of the `table` and `figure` environments. There can be other kinds of float. As perhaps a more interesting example, the following code produces the titled marginal note which should be displayed near here.

```
\marginpar{\legend{Title legend}
This is a marginal note with a legend.}
```

You can even

Legend in running text

use the `\legend` command in running text, as has been done in this sentence, but I'm not sure why one might want to do that as \LaTeX already provides the `center` environment.

`\formatlegend` The macro `\formatlegend{<text>}` is responsible for specifying how the `<text>` of a `\legend` is to be typeset. The `\legend` command can be used to make unnumbered captions in a `figure` or `table` environment. For example, in a `table` environment you could do something like:

```
\begin{table}
...
\renewcommand{\formatlegend}[1]{\tablename: #1}
...
\end{table}
```

In addition, if you wanted the title text to be included in the List of Tables, use the `\addcontentsline` command in conjunction with the `\legend`. For example:

```
\addcontentsline{lot}{table}{Titling text} % left justified
```

Table A legendary table

An unnumbered table	5
with two lines	6

```
\addcontentsline{lot}{table}{\protect\numberline{}Titling text} % indented
```

The first of these forms will align the first line of the titling text under the normal table numbers. The second form will align the first line of the titling text under the normal table titles. In either case, second and later lines of a multi-line title will be aligned under the normal title lines.

As an example, the last table is produced by the following code:

```
\begin{table}
\centering
\renewcommand{\formatlegend}[1]{\textbf{\tablename} #1}
\legend{A legendary table}
\addcontentsline{lot}{table}{A legendary table}
\addcontentsline{lot}{table}{\protect\numberline{}A legendary table}
\begin{tabular}{lc} \hline
An unnumbered table & 5 \\
with two lines      & 6 \\ \hline
\end{tabular}
\end{table}
```

Look at the List of Tables to see how the two forms of `\addcontentsline` are typeset.

`\abovelegendskip` Correspondingly to the `\abovecaptionskip` and `\belowcaptionskip` commands associated with the `\caption` command, the spacing before and after a legend is controlled by the `\abovelegendskip` and `\belowlegendskip` commands. If necessary, these can be modified via the `\setlength` command. By default these are defined to give a half baseline spacing before and after the legend.

`\belowlegendskip` As a convenience, the `\namedlegend[<short-title>]{<long-title>}` command is like the `\caption` command except that it does not number the caption and, by default, puts no entry into a listof file. Like the `\caption` command, it picks up the name to be prepended to the title text from the float environment in which it is called (e.g., it uses `\tablename` if called within a `table` environment).

The two examples of tables below are equivalent:

```
\begin{table}
\renewcommand{\formatlegend}[1]{\tablename: #1}
\legend{Long title text}
...
\end{table}

\begin{table}
\namedlegend[Short title text]{Long title text}
...
\end{table}
```

`\flegtoc@type` The macro `\flegtoc@type{<title>}`, where `type` is the name of a float environment (e.g., `table`) is called by the `\namedlegend` macro. It is provided as a hook that can be used to add `<title>` to the `listof` file. By default it is defined to do nothing, and can be changed via `\renewcommand`. For instance, it could be changed for tables as:

```
\makeatletter
\renewcommand{\flegtoc@table}[1]{%
  \addcontentsline{lot}{table}{#1}}
\makeatother
```

`\newfixedcaption` The `\legend` command produces a plain, unnumbered heading. It can also
`\renewfixedcaption` be useful sometimes to have named and numbered captions outside a floating
`\providefixedcaption` environment, perhaps in a `minipage` if you want the table or picture to appear at a precise location in your document.

The `\newfixedcaption[<capcommand>]{<command>}{<env>}` command, and its friends, can be used to create a new captioning `<command>` that may be used outside the float environment `<env>`. Both the environment `<env>` and a captioning command, `<capcommand>`, for that environment must have been defined before calling `\newfixedcaption`. Note that `\namedlegend` can be used as `<capcommand>`.

The `\renewfixedcaption` and `\providefixedcaption` commands take the same arguments as `\newfixedcaption`; the three commands are analagous to those in the `\newcommand` family.

For example, to define a new `\figcaption` command for captioning pictures outside the `figure` environment, do

```
\newfixedcaption{\figcaption}{figure}
```

The optional `<capcommand>` argument is the name of the float captioning command that is being aliased. It defaults to `\caption`. As another example, where the optional argument is required, if you want to create a new continuation caption command for non-floating tables, say `\ctabcaption`, then do

```
\newfixedcaption[\contcaption]{\ctabcaption}{table}
```

Captioning commands created by `\newfixedcaption` will be named and numbered in the same style as the original `<capcommand>`, can be given a `\label`, and will appear in the appropriate **List of ...**. They can also be used within floating environments, but will not use the environment name as a guide to the caption name or entry into the **List of ...**. For example, using `\ctabcaption` in a `figure` environment will still produce a **Table...** named caption.

Sometimes captions are required on the opposite page to a figure, and `\newfixedcaption` can be useful in this context. For example, if figure captions should be placed on an otherwise empty page immediately before the actual figure, then this can be accomplished by the following hack:

```
\newfixedcaption{\figcaption}{figure}
...
\afterpage{% fill current page then flush pending floats
  \clearpage
  \begin{midpage} % vertically center the caption
  \figcaption{The caption} % the caption
  \end{midpage}
  \clearpage
```

EXAMPLE FIGURE WITH BITWONUMCAPTION

Figure 1: Long

Bild 1: Lang

```
\begin{figure}THE FIGURE, NO CAPTION HERE\end{figure}
\clearpage
} % end of \afterpage
```

Note that the `afterpage` package is required, which is part of the required tools bundle. The `midpage` package supplies the `midpage` environment, which can be simply defined as:

```
\newenvironment{midpage}{\vspace*{\fill}}{\vspace*{\fill}}
```

The code might need adjusting to meet your particular requirements. The `nextpage` package might also be useful in this context as it provides a `\cleartoevenpage` command which ensures that you get to the next even-numbered page (the `\cleardoublepage` gets you to the next odd-numbered page and `\clearpage` gets you to the next page which may be odd or even).

2.2 Bilingual captions

Some documents require bilingual (or more) captions. The package provides a set of commands for bilingual captions. Extensions to the set, perhaps to support trilingual captioning, are left as an exercise for the document author.

`\bitwონumcaption` Bilingual captions can be typeset by the `\bitwონumcaption` command. This
`\bionenumcaption` takes 6 arguments as:

```
\bitwონumcaption[⟨label⟩]{⟨short-1⟩}{⟨long-1⟩}{⟨NAME⟩}{⟨short-2⟩}{⟨long-2⟩}
```

The first, optional argument `⟨label⟩`, is the name of a label, if required. `⟨short-1⟩` and `⟨long-1⟩` are the short (i.e., equivalent to the optional argument to the `\caption` command) and long caption texts for the main language of the document. The value of the `⟨NAME⟩` argument is used as the caption name for the second language caption, while `⟨short-2⟩` and `⟨long-2⟩` are the short and long caption texts for the second language. For example, if the main and secondary languages are English and German and a figure is being captioned:

```
\bitwონumcaption{Short}{Long}{Bild}{Kurz}{Lang}
```

If the short title text(s) is not required, then leave the appropriate argument(s) either empty or as one or more spaces, like:

```
\bitwონumcaption[fig:bi1]{}{Long}{Bild}{ }{Lang}
```

Both language texts are entered into the appropriate List of ..., and both texts are numbered.

Figure 1 is an example of using the above code.

The `\bionenumcaption` command takes the same arguments as `\bitwონumcaption`. The difference between the two commands is that `\bionenumcaption` does not number the second language text in the List of. Figure 2 is an example of using `\bionenumcaption`.

`\bicaption` When bilingual captions are typeset via the `\bicaption` command the second

EXAMPLE FIGURE WITH BIONENUMCAPTION

Figure 2: Long English

Bild 2: Lang Deutsch

EXAMPLE FIGURE WITH BICAPTION

Figure 3: Longingly

Bild 3: Langlauf

language text is not put into the List of The command takes 5 arguments as:
`\bicaption[⟨label⟩]{⟨short-1⟩}{⟨long-1⟩}{⟨NAME⟩}{⟨long-2⟩}`

The optional `⟨label⟩` is for a label if required. `⟨short-1⟩` and `⟨long-1⟩` are the short and long caption texts for the main language of the document. The value of the `⟨NAME⟩` argument is used as the caption name for the second language caption. The last argument, `⟨long-2⟩`, is the caption text for the second language (which is not put into the List of). For example, if the main and secondary languages are English and German:

`\bicaption{Short}{Long}{Bild}{Langlauf}`

If the short title text is not required, then leave the appropriate argument either empty or as one or more spaces.

Figure 3 is an example of using `\bicaption`.

`\bicontcaption` Bilingual continuation captions can be typeset via the `\bicontcaption` command. In this case, neither language text is put into the List of This command takes 3 arguments as:

`\bicontcaption{⟨long-1⟩}{⟨NAME⟩}{⟨long-2⟩}`

`⟨long-1⟩` is the caption text for the main language of the document. The value of the `⟨NAME⟩` argument is used as the caption name for the second language caption. The last argument, `⟨long-2⟩`, is the caption text for the second language. For example, if the main and secondary languages are again English and German:

`\bicontcaption{Continued}{Bild}{Fortgefahren}`

2.3 Use with the subfigure package

The `subfigure` package enables the captioning of sub-figures within a larger figure, and similarly for tables. If a figure that includes sub-figures is itself continued then it may be desirable to continue the captioning of the sub-figures. For example, if Figure 3 has three sub-figures, say A, B and C, and Figure 3 is continued then the sub-figures in the continuation should be D, E, etc.

`\contsubtop` The command `\contsubtop[⟨subcaption⟩]{⟨text⟩}` will continue the sub-
`\contsubbottom` caption numbering scheme across (continued) floats, putting the `⟨subcaption⟩` at the top of the `⟨text⟩`. The `\contsubbottom` command is similar but puts the `⟨subcaption⟩` at the bottom of the `⟨text⟩`. In either case, the main caption can be at the top or bottom of the float.

`\subconcluded` The `\subconcluded` command is used to indicate that the continued (sub) float has been concluded and the numbering scheme is reinitialized. The command should be placed immediately before the end of the last continued environment.

`\subtop` The command `\subtop[⟨subcaption⟩]{⟨text⟩}` is in addition to the `subfig-`
`\subbottom` ure package commands `\subfigure` and `\subtable`. It puts the `⟨subcaption⟩`

at the top of the $\langle text \rangle$, and similarly `\subbottom[$\langle subcaption \rangle$]{ $\langle text \rangle$ }` puts $\langle subcaption \rangle$ at the bottom of the $\langle text \rangle$.

For example:

```
\begin{figure}
\subbottom{...} % captioned as (a) below
\subbottom{...} % captioned as (b) below
\caption{...}
\end{figure}
\begin{figure}
\contsubtop{...} % captioned as (c) above
\contsubtop{...} % captioned as (d) above
\contcaption{Concluded}
\subconcluded
\end{figure}
...
\begin{table}
\caption{...}
\subtop{...} % captioned as (a) above
\subbottom{...} % captioned as (b) below
\end{table}
```

Depending on the age of your L^AT_EX distribution, you may find that you have either version 2.0 or version 2.1 of the `subfigure` package (currently version 2.1 has not been released and is only available from the author). If you have version 2.1, then call the `ccaption` package as:

`\usepackage[subfigure21]{ccaption}`, otherwise as:

`\usepackage[subfigure20]{ccaption}`.

Version 2.1 of the `subfigure` package uses many package options some of which had been provided as commands in version 2.0. The `ccaption` commands just described apply to version 2.0. They also apply to version 2.1 except that the `\...top` and `\...bottom` commands take a second optional argument as: `\...top[$\langle list-entry \rangle$][$\langle subcaption \rangle$]{ $\langle text \rangle$ }` (see the `subfigure` documentation for an explanation of the arguments).

Both versions of `subfigure` provide the commands `\subfigure` and `\subtable` which may be used with the `ccaption` package (which also provides matching `\contsubfigure` and `\contsubtable` commands) but I recommend using the generic `\...top` and `\...bottom` commands instead. One reason being that the generic commands can be used for subcaptions in new kinds of floats, whereas the specific `\...figure` and `\...table` commands cannot. In version 2.1 of `subfigure` the placement (top or bottom) of the subcaptions and the expected placement of the main caption are set by package options. Using these options in conjunction with the `ccaption` package may cause unexpected results, which is another reason for using the generic subcaption commands.

Note that the `subfigure21` option may change after the general release of version 2.1 of the `subfigure` package.

2.4 Use with the `endfloat` package

The `endfloat` package [McCG95] has the capability of putting all floats at the end of the printed document and inserting comments in the main text that a float

should be placed about *there*. There is a slight problem if continuation captions are used in conjunction with the package, as `endfloat` effectively numbers each float whether or not it is captioned, and thus will increment the numbering for and continued float.

One way of getting `endfloat` and `ccaption` continued captions to cooperate is to put the following in the document preamble (modifying or extending it to suit):

```
\newcommand{\contendfloat}{}
\renewcommand{\tableplace}{%
  \begin{center}
    [\tablename~\theposttbl\ \contendfloat\ about here.]
  \end{center}
}
\newenvironment{conttable}{%
  \addtocounter{posttbl}{-1}%
  \def\contendfloat{(continued)}}{}
\renewcommand{\figureplace}{%
  \begin{center}
    [\figurename~\thepostfig\ \contendfloat\ about here.]
  \end{center}
}
\newenvironment{configure}{%
  \addtocounter{postfig}{-1}%
  \def\contendfloat{(continued)}}{}
```

and then, for a table, in the document:

```
...
\begin{table}
\caption{...}
...
\end{table}
...
\begin{conttable}
\begin{table}
\contcaption{Continued}
...
\end{table}
\end{conttable}
```

and similarly for any continued figures.

2.5 New float environments

The commands in the previous sections have been tested with the `caption2` and `rotating` packages. They will most likely fail if used with the `float` package because of the way this package redefines the basic `\caption` command.

The `float` package, developed by Anselm Lingnau [Lin95], provides a simple scheme for creating new kinds of floats with a variety of captioning styles. Unfortunately the package does not effectively separate the float creation aspects and the captioning styles. I have therefore included in the `ccaption` package a poor man's version of some aspects of the float creation elements that are in `float`. Both the commands and their coding differ from those in the `float` package.

`\newfloatenv` The command `\newfloatenv` [*within*] {*fenv*} {*ext*} {*capname*} creates a

new kind of floating environment called $\langle fenv \rangle$. A caption within the environment will be written out to a file with extension $\langle ext \rangle$. The caption, if present, will start with $\langle capname \rangle$. For example, if this command had been used to create the **figure** environment for the **article** class it would have been used as (remembering that L^AT_EX uses `\figurename` to store the ‘Figure’ text):

```
\newfloatenv{figure}{lof}{\figurename}
```

The optional $\langle within \rangle$ argument can be used if you want the captions to be numbered within a particular document division, as figures are within the **book** and **report** classes with the numbering starting afresh with each new chapter. Creating the figure environment for either of these classes would have used:

```
\newfloatenv[chapter]{figure}{lof}{\figurename}
```

The captioning style for floats defined with `\newfloatenv` is the same as for figures and tables in the standard classes.

`\listfloats` The `\listfloats{ $\langle fenv \rangle$ }{ $\langle heading \rangle$ }` command can be used to typeset a listing of the float captions for the float environment $\langle fenv \rangle$. The listing will be preceded by $\langle heading \rangle$ which will be in the same style as the headings for `\listoftables` and `\listoffigures`. So, just to finish off the figure example, instead of putting `\listoffigures` where you want the listing to appear, you would have had to use (remembering that L^AT_EX uses `\listfigurename` to store the ‘List of Figures’ text)

```
\listfloats{figure}{\listfigurename}
```

As a fuller example, suppose you wanted both figures (which come with the standard classes), and diagrams. You could then do something like the following.

```
\usepackage{ccaption}
...
\newfloatenv{diagram}{dgm}{Diagram}
\newfixedcaption{\fdiagcaption}{diagram}
\begin{document}
...
\listoffigures
\listfloats{diagram}{List of diagrams}
...
\begin{diagram}
\caption{A diagram} \label{diag1}
...
\end{diagram}
As diagram~\ref{diag1} shows ...
\begin{minipage}{.9\textwidth}
\fdiagcaption{Another diagram} \label{diag2}
...
\end{minipage}
```

In contrast to `diagram~\ref{diag1}`, `diagram~\ref{diag2}` provides ...

As a matter of style, though, it would be more in keeping with L^AT_EX to use the following for diagrams:

```
...
\newcommand{\diagramname}{Diagram}
\newcommand{\listdiagramname}{List of Diagrams}
\newfloatenv{diagram}{dgm}{\diagramname}
```

```

...
\listfloats{diagram}{\listdiagramname}
...

```

As a word of warning, if you mix both floats and fixed environments with the same kind of caption you have to ensure that they get printed in the correct order in the final document. If you do not do this, then the `\list...` of captions will come out in the wrong order (the lists are ordered according the page number in the typeset document, *not* your source input order).

`\newsubfloat` The `\newsubfloat{fenv}` command, which is only of use with the `subfigure` package and the `subfigure20` or `subfigure21` option, creates subcaptions (`\subtop` and `\subbottom`, together with their continued forms) for use within the float environment *fenv* previously defined via `\newfloatenv{fenv}`.

2.6 How L^AT_EX makes captions

This section provides an overview of how L^AT_EX creates captions and gives some examples of how to change the captioning style without having to use any package. The section need not be looked at more than once unless you like reading L^AT_EX code or you want to make changes to L^AT_EX's style of captioning.

The L^AT_EX kernel provides tools to help in the definition of captions, but it is the particular class that decides on their format.

`\caption` The kernel (in `lfloat.dtx`) defines the caption command via
`\def\caption{\refstepcounter\@capytype \@dblarg{\@caption\@capytype}}`
`\@capytype` `\@capytype` is defined by the code that creates a new float environment and is set to the environment's name (see the code for `\xfloat` in `lfloat.dtx`). For a `figure` environment, there is an equivalent to
`\def\@capytype{figure}`.

`\@caption` The kernel also provides the `\@caption{<type>}[<short-title>]{<full-title>}` command as:

```

\long\def\@caption#1[#2]#3{%
  \par
  \addcontentsline{\csname ext@#1\endcsname}{#1}% <----
    {\protect\numberline{\csname the#1\endcsname}{\ignorespaces #2}}%
  \begingroup
    \@parboxrestore
    \if@minipage
      \setminipage
    \fi
    \normalsize
    \@makecaption{\csname fnum@#1\endcsname}{\ignorespaces #3}\par % <----
  \endgroup}

```

where *<type>* is the name of the environment in which the caption will be used. Putting these three commands together results in the user's view of the caption command as `\caption[<short-title>]{<full-title>}`.

It is the responsibility of the class (or package) which defines floats to provide definitions for `\ext@type`, `\fnum@type` and `\@makecaption` which appear in the definition of `\@caption` (in the lines marked <---- above).

A THOUSAND WORDS...

FIGURE 4: A picture is worth a thousand words

`\ext@type` This macro holds the name of the extension for a ‘List of...’ file. For example for the `figure` float environment there is the definition equivalent to `\newcommand{\ext@figure}{lof}`.

`\fnum@type` This macro is responsible for typesetting the caption number. For example, for the `figure` environment there is the definition equivalent to `\newcommand{\fnum@figure}{\figurename~\thefigure}`.

`\@makecaption` The `\@makecaption{<number>}{<text>}`, where `<number>` is a string such as ‘Table 5.3’ and `<text>` is the caption text, performs the typesetting of the caption, and is defined in the standard classes (in `classes.dtx`) as the equivalent of:

```
\newcommand{\@makecaption}[2]{%
  \vskip\abovecaptionskip      % <- 1
  \sbox\@tempboxa{#1: #2}%      % <- 2
  \ifdim \wd\@tempboxa >\hsize
    #1: #2\par                  % <- 3
  \else
    \global \@minipagefalse
    \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
  \fi
  \vskip\belowcaptionskip      % <- 4
}
```

`\abovecaptionskip` Vertical space is added before and after a caption (lines marked 1 and 4 in the code for `\@makecaption` above) and the amount of space is given by the lengths `\abovecaptionskip` and `\belowcaptionskip`. The standard classes set these to 10pt and 0pt respectively. If you want to change the space before or after a caption, use `\setlength` to change the values.

The actual typesetting of a caption is effectively performed by the code in lines marked 2 and 3 in the code for `\@makecaption`; note that these are where the colon that is typeset after the number is specified. If you want to make complex changes to the default captioning style you may have to create your own version of `\@caption` using `\renewcommand`. On the other hand, many such changes can be achieved by changing the definition of the the appropriate `\fnum@type` command(s). For example, to make the figure name and number bold:

```
\renewcommand{\fnum@figure}{\textbf{\figurename~\thefigure}}
```

REMEMBER: If you are doing anything involving commands that include the `@` character, and it’s not in a class or package file, you have to do it within a `\makeatletter` and `\makeatother` pairing. So, if you modify the `\fnum@figure` command anywhere in your document it has to be done as:

```
\makeatletter
\renewcommand{\fnum@figure}{.....}
\makeatother
```

As an example, Figure 4 was created by the following code:

```
\makeatletter
\renewcommand{\fnum@figure}{\textsc{\figurename~\thefigure}}
```

ANOTHER THOUSAND WORDS...

Figure 5 — A different kind of figure caption

```
\makeatother
\begin{figure}
\centering
A THOUSAND WORDS\ldots
\caption{A picture is worth a thousand words}\label{fig:sc}
\end{figure}
```

As another example, suppose that you needed to typeset the `\figurename` and its number in a bold font, replace the colon that normally appears after the number by a long dash, and typeset the actual title text in a sans-serif font, as is illustrated by the caption for Figure 5. The following code does this.

```
\makeatletter
\renewcommand{\fnum@figure}[1]{\textbf{\figurename~\thefigure} --- \sffamily}
\makeatother
\begin{figure}
\centering
ANOTHER THOUSAND WORDS\ldots
\caption{A different kind of figure caption}\label{fig:sf}
\end{figure}
```

Perhaps a little description of how this works is in order. Doing a little bit of TeX's macro processing by hand, the typesetting lines in `\@makecaption` (lines 2 and 3) get instantiated like:

```
\fnum@figure{\figurename~\thefigure}: text
Redefining \fnum@figure to take one argument and then not using the value of
the argument essentially gobbles up the colon. Using
\textbf{\figurename~\thefigure}
in the definition causes \figurename and the number to be typeset in a bold font.
After this comes the long dash. Finally, putting \sffamily at the end of the
redefinition causes any following text (i.e., the actual title) to be typeset using the
sans-serif font.
```

If you do modify `\@makecaption`, then spaces in the definition may be important; also you must use the comment (%) character in the same places as I have done above.

You may also want to take a look at the `caption2` package by Harald Axel Sommerfeldt which provides a ready-made set of differing captioning styles. This basically works by redefining the `\@makecaption` command to provide some hooks. As the captions made by the `ccaption` package use `\@makecaption` they should match any style that you pick from `caption2`.

In the `article` class and its derivatives, captions are numbered continuously throughout the document, while in the `book` and `report` classes, numbering starts anew in each chapter.

If you want captions to be numbered anew with sections in the `article` class you can do this:

```

\makeatletter
\@addtoreset{table}{section}
\renewcommand{\thetable}{\thesection.\arabic{table}}
\makeatletter

```

and similarly for all the other float environments.

If you are using the `book` or `report` class and you want the captions to be numbered consecutively throughout the document you can do this:

```

\makeatletter
\@removefromreset{table}{chapter}
\renewcommand{\thetable}{\arabic{table}}
\makeatother

```

and similarly for all the other float environments. Note that you will need the `remreset` package² which provides the definition of `\@removefromreset`.

3 The package code

Announce the name and version of the package, which requires L^AT_EX 2_ε.

```

1 (*usc)
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesPackage{ccaption}[2001/01/02 v2.6d Extended captioning and new floats]

```

In an attempt to avoid name clashes with other packages, all internal commands include the string `@cont`.

Do the options first.

```

\if@contsubfigxx These three \if... are used to remember if the subfigure20 or subfigure21 option
\if@contsubfigxxi has been given.
\if@contsubfig
4 \newif\if@contsubfigxx
5 \@contsubfigxxfalse
6 \newif\if@contsubfigxxi
7 \@contsubfigxxifalse
8 \newif\if@contsubfig
9 \@contsubfigfalse
10 \DeclareOption{subfigure20}{\@contsubfigxxtrue\@contsubfigxxifalse\@contsubfigtrue}
11 \DeclareOption{subfigure21}{\@contsubfigxxfalse\@contsubfigxxitruetrue\@contsubfigtrue}
12 \ProcessOptions
13

```

3.1 Continuation captions and legends

`\contcaption` `\contcaption{<text>}` is a user-level command. It is a simplified version of the normal `\caption` command as it doesn't have to deal with numbering or list of ... entries.

```

14 \newcommand{\contcaption}{\@contcaption\@capttype}
15

```

²Available on CTAN in `tex-archive/macros/latex/contrib/supported/carlise`.

`\@contcaption` This is the workhorse for the `\contcaption` command. In turn, it uses the `\@makecaption` command (defined in the usual classes) to do most of its work. It uses the number of the previous `\caption` command in the same type of float and its implementation includes much of the code used in the L^AT_EX `\caption` command.

```

16 \long\def\@contcaption#1#2{%
17   \par
18   \begingroup
19     \@parboxrestore
20     \if@minipage
21       \setminipage
22     \fi
23     \normalsize
24     \@makecaption{\csname fnum@#1\endcsname}{\ignorespaces #2}\par
25   \endgroup}
26

```

`\abovelegendskip` These two lengths control the vertical spacing before and after a legend. We will
`\belowlegendskip` give these values such that a legend will occupy an integral number of lines.

```

27 \newlength{\abovelegendskip}
28 \setlength{\abovelegendskip}{0.5\baselineskip}
29 \newlength{\belowlegendskip}
30 \setlength{\belowlegendskip}{\abovelegendskip}
31

```

`\legend` The command is called as `\legend{<text>}`. It is intended to be used in a float environment for an ‘anonymous’ caption.

The implementation is similar to the `\caption` command.

```

32 \newcommand{\legend}[1]{%
33   \par
34   \begingroup
35     \@parboxrestore
36     \if@minipage
37       \setminipage
38     \fi
39     \normalsize
40     \@makelegend{}{\ignorespaces \formatlegend{#1}}\par
41   \endgroup}
42

```

`\formatlegend` `\formatlegend{<text>}` is responsible for typesetting a legend’s text.

```

43 \newcommand{\formatlegend}[1]{#1}
44

```

`\@makelegend` The implementation of this command is similar to that of the `\@makecaption` command in the standard classes, or to the redefined version of `\@makecaption` if the `caption2` package has been loaded.

```

45 \AtBeginDocument{%
46   \@ifpackageloaded{caption2}{%
47     \newcommand{\@makelegend}[2]{%
48       \vskip\abovelegendskip
49       \realcaptionwidth\linewidth

```



```

50     \def\captionlabel{#1}%
51     \def\captiontext{#2}%
52     \usecaptionstyle{\caption@style}%
53     \vskip\belowlegendskip}}{%
54     \newcommand{\@makelegend}[2]{%
55         \vskip\abovelegendskip
56         \sbox\@tempboxa{#1 #2}%
57         \ifdim \wd\@tempboxa >\hsize
58             #1 #2\par
59         \else
60             \global \@minipagefalse
61             \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
62         \fi
63         \vskip\belowlegendskip}}
64 }
65

```

`\namedlegend` `\namedlegend[(short-title)]{(long-title)}` is like the `\caption` command except that it does not number the caption.

```

66 \newcommand{\namedlegend}{\@dblarg{\@legend\@capttype}}
67

```

`\@legend` `\@legend{(type)}[(short-title)]{(long-title)}` is the workhorse for the `\namedlegend` command. In turn, it calls `\@makelegend`. It requires two commands to have been defined, namely `\flegtoc@type` and `\fleg@type`. The command `\flegtoc@type{(text)}` is responsible for writing a title text to the appropriate listof file. `\fleg@type` is responsible for typesetting the name of the legend.

```

68 \long\def\@legend#1[#2]#3{%
69     \par
70     \csname flegtoc@#1\endcsname{#2}%
71     \begingroup
72         \@parboxrestore
73         \if@minipage
74             \@setminipage
75         \fi
76         \normalsize
77         \@makelegend{\csname fleg@#1\endcsname}{\formatlegend{#3}}\par
78     \endgroup}
79

```

`\flegtoc@table` `\flegtoc@figure` These macros write a `\namedlegend` title to the respective listof file. By default they do nothing.

```

80 \newcommand{\flegtoc@table}[1]{ }
81 \newcommand{\flegtoc@figure}[1]{ }
82

```

`\fleg@table` `\fleg@figure` These macros typeset the name before the title of a `\namedlegend`. By default they are defined to mimic the normal captioning style.

```

83 \newcommand{\fleg@table}{\tablename: }
84 \newcommand{\fleg@figure}{\figurename: }
85

```

3.2 Non-float captions

`\newfixedcaption` These commands are defined in terms of their `\...command` counterparts.
`\renewfixedcaption` Call as `\...fixedcaption[<capcommand>]{<command>}{<env>}`
`\providefixedcaption` 86 `\newcommand{\newfixedcaption}[3][\caption]{%`
87 `\newcommand{#2}{\def\@capttype{#3}#1}}`
88 `\newcommand{\renewfixedcaption}[3][\caption]{%`
89 `\renewcommand{#2}{\def\@capttype{#3}#1}}`
90 `\newcommand{\providefixedcaption}[3][\caption]{%`
91 `\providecommand{#2}{\def\@capttype{#3}#1}}`
92

3.3 Bilingual captions

The bilingual caption commands all use internal grouping so that any changes are kept local. This has the unfortunate side-effect that any `\label` command must be within the grouping otherwise the wrong number is picked up. To make the coding, if not necessarily the use, of the commands simpler, I have not used the traditional style of square brackets for optional caption text arguments. Instead, empty ‘required’ arguments are used as the implementation means.

`\@if@contemptyarg` For dealing with empty arguments. `\@if@contemptyarg{<testarg>}{<YES>}{<NO>}` checks if *<testarg>* is empty (consists of zero or more spaces only). If it is empty then the *<YES>* argument is processed otherwise the *<NO>* argument is processed. The implementation uses code suggested by Donald Arseneau (see section A for some background on this).
93 `\begingroup`
94 `\catcode'\Q=3`
95 `\long\gdef\@if@contemptyarg#1{\@xif@contmt#1QQ\@secondoftwo\@firstoftwo\@nil}`
96 `\long\gdef\@xif@contmt#1#2Q#3#4#5\@nil{#4}`
97 `\endgroup`
98

`\bitwonumcaption` The 6 arguments are: optional label, short and long in language 1, name in language 2, and short and long in language 2. Both texts are put into the List of as numbered entries.

99 `\newcommand{\bitwonumcaption}[6][\@empty]{%`
100 `\begingroup`
Check if the first language argument is vacuous, then call the normal `\caption` for language 1.
101 `\@if@contemptyarg{#2}{\caption{#3}}{\caption[#2]{#3}}`
Do the optional labeling.
102 `\ifx #1\@empty\else`
103 `\label{#1}`
104 `\fi`
Remove any extra spacing between the captions, and set the NAME for the second caption. Use a command to transfer the NAME to the `renewell` code to avoid circularity if for example, we are trying to redefine `\tablename` as `\tablename`. Decrement the caption counter.
105 `\setlength{\abovecaptionskip}{0pt}`

```

106 \setlength{\belowcaptionskip}{0pt}
107 \edef\@tempc{#4}
108 \expandafter \renewcommand \csname \@captype name\endcsname{\@tempc}
109 \addtocounter{\@captype}{-1}
    Now repeat for the second language caption.
110 \@if@contemptyarg{#5}{\caption{#6}}{\caption[#5]{#6}}
111 \endgroup
112 }
113

```

\bionenumcaption The 6 arguments are: optional labelling, short and long in language 1, name in language 2, and short and long in language 2. Both texts are put into the List of, but only the first is numbered.

```

114 \newcommand{\bionenumcaption}[6][\@empty]{%
115   \begingroup
    Check if the first language argument is vacuous, then call the normal \caption
    for language 1.
116   \@if@contemptyarg{#2}{\caption{#3}}{\caption[#2]{#3}}
    Do the optional labeling.
117   \ifx #1\@empty\else
118     \label{#1}
119   \fi
    Do the between captions code.
120   \setlength{\abovecaptionskip}{0pt}
121   \setlength{\belowcaptionskip}{0pt}
122   \edef\@tempc{#4}
123   \expandafter \renewcommand \csname \@captype name\endcsname{\@tempc}
    Use a continuation caption for the second language, not forgetting to add the
    appropriate unnumbered text to the List.
124   \contcaption{#6}
125   \@if@contemptyarg{#5}{%
126     \addcontentsline{\csname ext@\@captype\endcsname}{\@captype}%
127       {\protect\numberline{}{\ignorespaces #6}}}%
128     \addcontentsline{\csname ext@\@captype\endcsname}{\@captype}%
129       {\protect\numberline{}{\ignorespaces #5}}}
130   \endgroup
131 }
132

```

\bicaption The 5 arguments are: optional labelling, short and long in language 1, name in language 2, and long in language 2. Only the first text is put into the List.

```

133 \newcommand{\bicaption}[5][\@empty]{%
134   \begingroup
    Check if the first language argument is vacuous, then call the normal \caption
    for language 1.
135   \@if@contemptyarg{#2}{\caption{#3}}{\caption[#2]{#3}}
    Do the optional labeling.
136   \ifx #1\@empty\else
137     \label{#1}
138   \fi

```

Do the between captions code and finally just use `\contcaption` for the second language.

```

139 \setlength{\abovecaptionskip}{0pt}
140 \setlength{\belowcaptionskip}{0pt}
141 \edef\@tempc{#4}
142 \expandafter \renewcommand \csname \@captype name\endcsname{\@tempc}
143 \contcaption{#5}
144 \endgroup
145 }
146

```

`\bicontcaption` The 3 arguments are long in language 1, name in language 2, and long in language 2.

```

147 \newcommand{\bicontcaption}[3]{%
148 \beginingroup
    Call \contcaption for language 1.
149 \contcaption{#1}
    Do the between captions code and use \contcaption for the second language.
150 \setlength{\abovecaptionskip}{0pt}
151 \setlength{\belowcaptionskip}{0pt}
152 \edef\@tempc{#2}
153 \expandafter \renewcommand \csname \@captype name\endcsname{\@tempc}
154 \contcaption{#3}
155 \endgroup
156 }
157

```

3.4 The subfigure options

`\@contkeep` These are common to both subfigure options. `\@contkeep` stores the current subfigure/table number in counter `@contsubnum` and `\@contset` sets the subfigure/table number to the value of `@contsubnum`. `\subconcluded` sets the subfigure/table number to zero. The original definition of `\@contcaption` is kept in `\subfigold@contcaption`.

```

158 \if@contsubfig
159 \newcounter{@contsubnum}
160 \newcommand{\@contkeep}{\setcounter{@contsubnum}{\value{sub\@captype}}}
161 \newcommand{\@contset}{\setcounter{sub\@captype}{\value{@contsubnum}}}
162 \newcommand{\subconcluded}{\setcounter{sub\@captype}{0}}
163 \let\subfigold@contcaption\@contcaption
164 \fi

```

`\if@contmaincaption` This is set TRUE after the (cont)caption in a float has been processed. (A `\newif` cannot be used within an `\if... \fi` construct.)

```

165 \newif\if@contmaincaption
166 \@contmaincaptionfalse

```

`\if@contbotsub` A flag indicating whether the subcaption is to be at the bottom or top of the subfigure/subtable; TRUE for the subcaption at the bottom.

```

167 \newif\if@contbotsub
168 \@contbotsubtrue
169

```

3.4.1 Option subfigure20

In order to eliminate an ordering dependency between the `subfigure` and `ccaption` packages, modifications to the original `subfigure` code have to be done at the start of the document after all packages have been loaded. First for `subfigure 2.0`, if it is called for.

```
170 \AtBeginDocument{%
171 \if@contsubfigxx
```

```
\caption These original commands are all modified to set the value of \if@contmaincaption.
\contcaption The (cont)caption commands set it to TRUE and the float commands set it
\@float FALSE. Additionally, the \@float and \@dbflt commands are modified to zero
\@dbflt the subfloat counter, if it is defined.

172 \let\@contoldc\caption
173 \renewcommand{\caption}{\@contmaincaptiontrue\@contoldc}
174 \let\@contoldcont\contcaption
175 \renewcommand{\contcaption}{\@contmaincaptiontrue\@contoldcont}
176 \let\@contoldf\@float
177 \renewcommand{\@float}[1]{\@contmaincaptionfalse
178 \ifundefined{c@sub#1}{\csname c@sub#1\endcsname = 0\relax}
179 \@contoldf{#1}}
180 \let\@contoldff\@dbflt
181 \renewcommand{\@dbflt}[1]{\@contmaincaptionfalse
182 \ifundefined{c@sub#1}{\csname c@sub#1\endcsname = 0\relax}
183 \@contoldff{#1}}
184
```

`\@subfloat` This macro from `subfigure v2.0` is modified to enable subcaptions to be placed at either the top or bottom of the sub... (the original only placed them at the bottom). First, the `subfigure/table` is set in a box.

```
185 \def\@subfloat#1[#2]#3{%
186 \setbox\@tempboxa \hbox{#3}%
187 \@tempdima=\wd\@tempboxa
188 \if@contbotsub
```

The subcaption is to be put at the bottom, so typeset the figure, followed by the caption, if any.

```
189 \vtop{%
190 \vbox{\vskip\subfigtopskip
191 \box\@tempboxa}%
192 \ifx \@empty#2\relax \else
193 \vskip\subfigcapskip
194 \@subcaption{#1}{#2}%
195 \fi
196 \vskip\subfigbottomskip}%
197 \else
```

The subcaption is to be put at the top, so typeset the caption if any, followed by the figure.

```
198 \vtop{%
199 \ifx \@empty#2\relax \else
200 \vskip\subfigcapskip
201 \begin{group}\@subcaption{#1}{#2}\endgroup%
202 \fi
```

```

203     \vbox{\vskip\subfigtopskip
204           \box\@tempboxa}%
205     \vskip\subfigbottomskip}%
206   \fi
207   \egroup}
208

```

`\@subcaption` The original `\@subcaption` command produces unexpected results in the ToC (i.e., `numberline` appears instead of `\numberline` because of the original internal definition of `\protect`). I have also modified it so that when a top main caption is being used, it adds the subcaption to the ToC directly.

Sebastien Derriere found that there were problems when fragile commands were used within a continued subcaption. Steven Douglas Cochran kindly provided a fix for this.

```

209   \renewcommand{\@subcaption}[2]{%
210     \begingroup
211       \let\label\@gobble
212       \let\protect\string      % SDC mod
213       \ifcontmaincaption
214         \addcontentsline{\csname ext@#1\endcsname}{#1}%
215           {\protect\numberline{\csname p@#1\endcsname\csname the#1\endcsname}%
216             {\ignorespaces #2}}%
217         \gdef\@subfigcaptionlist{}
218       \else
219         \xdef\@subfigcaptionlist{%
220           \@subfigcaptionlist,%
221           % \string\numberline {\@currentlabel}%    % SDC mod
222           {\protect\numberline {\@currentlabel}%    % SDC mod
223             \noexpand{\ignorespaces #2}}}%
224       \fi
225     \endgroup
226     \@nameuse{@make#1caption}{\@nameuse{@the#1}}{#2}}
227

```

`\subfigure` These are revised versions of the original commands. They are now aliases for `\subbottom` and `\subtop` respectively. In their original form they were both effectively aliases for `\subbottom` only.

```

228   \let\subfigure\subbottom
229   \let\subtable\subtop
230 \fi
231 }

```

The end of the `\AtBeginDocument` code for `subfigure20`.

Do the remaining code for the `subfigure20` option, if called for.

```

232 \if@contsubfigxx

```

`\subbottom` `\subbottom[<caption>]{<text>}` typesets a subcaption when the main caption is at the end of the float environment. The code is a slight modification of the original `\subfigure` command in that the bottom flag is added and set to true and the subcaption number is stored. The caption number must be locally advanced if the main caption has not yet been processed (i.e., is at the bottom of the float). As

most of the code is common with `\subtop` it is placed into the `\@contsubbody` macro.

```

233 \newcommand{\subbottom}{%
234   \@contbotsubtrue
235   \@contsubbody}
236
237 \newcommand{\@contsubbody}{%
238   \bgroup
239   \if@contmaincaption\else
240     \advance\csname c@\@capttype\endcsname\@ne
241   \fi
242   \refstepcounter{sub\@capttype}\@contkeep%
243   \leavevmode
244   \ifnextchar [%
245     {\@subfloat{sub\@capttype}}
246     {\@subfloat{sub\@capttype}[\@empty]}}
247

```

`\contsubbottom` The continued version of `\subbottom`. It restores the kept subcaption number before incrementing and keeping it. As most of the code is common with `\contsubtop` it is kept in the `\subbody@cont`.

```

248 \newcommand{\contsubbottom}{%
249   \@contbotsubtrue
250   \subbody@cont}
251
252 \newcommand{\subbody@cont}{%
253   \bgroup
254   \@contset
255   \refstepcounter{sub\@capttype}\@contkeep%
256   \leavevmode
257   \ifnextchar [%
258     {\@subfloat{sub\@capttype}}
259     {\@subfloat{sub\@capttype}[\@empty]}}
260

```

`\subtop` `\subtop[<caption>]{<text>}` typesets a subcaption at the top of the subfigure/table. This is almost identical to `\subbottom`.

```

261 \newcommand{\subtop}{%
262   \@contbotsubfalse
263   \@contsubbody}
264

```

`\contsubtop` The continued version of `\subtop`.

```

265 \newcommand{\contsubtop}{%
266   \@contbotsubfalse
267   \subbody@cont}
268

```

`\@contcaption` The `\@contcaption` command must be modified to add the listed subcaptions (if any, and there should be none for top main captions) to the ToC. A simplified version of the subfigure redefinition of `\@caption`.

```

269 \long\def\@contcaption#1#2{%
270   \subfigold@contcaption{#1}{#2}%

```

```

271 \@for \@tempa:=\@subfigcaptionlist \do {%
272 \ifx\@empty\@tempa\relax \else
273 \addcontentsline
274 {\@nameuse{ext@sub#1}}%
275 {sub#1}%
276 {\@tempa}%
277 \fi}%
278 \gdef\@subfigcaptionlist{}}
279

```

`\contsubtable` Aliases for `\contsubtop` and `\contsubbottom`, respectively.

```

\contsubfigure 280 \let\contsubtable\contsubtop
281 \let\contsubfigure\contsubbottom
282

```

The end of the `subfigure20` option code.

```

283 \fi
284

```

This is the end of the version 2.0 code.

3.4.2 Option `subfigure21`

`\do@contsubfig` The `\do@contsubfig` command redefines the `subfigure v2.1 \subfigure` command. The revised version stores the subcounter. The redefinition cannot be done directly within an `\if... \fi` group because of the internal `\csname if\@capytype...`

```

\subfigure
285 \newcommand{\do@contsubfig}{%
286 \renewcommand{\subfigure}{%
287 \bgroup
288 \csname if\@capytype topcap\endcsname
289 \else
290 \advance\csname c@\@capytype\endcsname\@ne
291 \fi
292 \refstepcounter{sub\@capytype}\@contkeep% % <- change here
293 \@ifnextchar [%
294 {\@subfigure}%
295 {\@subfigure[\@empty]}
296 }

```

`\@subfloat` This is a modified version of the `subfigure v2.1 \@subfloat` command. Essentially the `\csname if#1topcap\endcsname` constructs are replaced by `\if@contbotsub`. This is actually only required for user-defined floats where I haven't been able to work out if it is possible to create new `\if#1...` commands within a command that has a parameter `#1`.

```

297 \def\@subfloat##1[##2][##3]##4{%
298 \if@minipage
299 \@tempcnta=\z@
300 \else
301 \ifdim\lastskip=\z@
302 \@tempcnta=\@ne
303 \else
304 \@tempcnta=\tw@
305 \fi
306 \fi

```



```

307 \if@contbotsub
308 \def\subfig@top{\subfigtopskip}%
309 \def\subfig@bottom{\subfigbottomskip}%
310 \else
311 \def\subfig@top{\subfigbottomskip}%
312 \def\subfig@bottom{\subfigtopskip}%
313 \fi
314 \setbox\@tempboxa \hbox{##4}%
315 \@tempdima=\wd\@tempboxa
316 \vtop\bgroup
317 \vbox\bgroup
318 \ifcase\@tempcnta
319 \or
320 \vskip\subfig@top
321 \or
322 \@tempskipb\subfig@top\@xaddvskip
323 \fi
324 \if@contbotsub
325 \box\@tempboxa\egroup
326 \ifx \@empty##3\relax \else
327 \vskip\subfigcapskip
328 \@subcaption{##1}{##2}{##3}%
329 \fi
330 \else
331 \ifx \@empty##3\relax \else
332 \@subcaption{##1}{##2}{##3}%
333 \vskip\subfigcapskip
334 \fi\egroup
335 \box\@tempboxa
336 \fi
337 \vskip\subfig@bottom
338 \egroup
339 \egroup}
340
341 } % end of \do@contsubfig
342

```

\cont@subfig@oldcaption Keep the definition of \@caption.

```

343 \let\cont@subfig@oldcaption\@caption
344

```

Now these can be used within the \AtBeginDocument code.

```

345 \AtBeginDocument{%
346 \if@contsubfigxxi
347 \do@contsubfig
348 \let\subfigure\subbottom
349 \let\subtable\subtop
350 \let\contsubfigure\contsubbottom
351 \let\contsubtable\contsubtop
352 \long\def\@caption#1[#2]#3{%
353 \cont@subfig@oldcaption{#1}[#2]{#3}}
354 \fi
355 }
356

```

The remainder of the subfigure21 option code.

`\doxxi@contcaption` This command redefines the `\@contcaption` command to flush out any pending subcaptions. The redefinition cannot be done within `\if...\fi` because of the internal `\if...` creation. The code is simplified from the subfigure v2.1 redefinition of `\@caption`.

```

357 \newcommand{\doxxi@contcaption}{%
358   \long\def\@contcaption##1##2{%
359     \if@contbotsub
360       \@listsubcaptions{##1}%
361       \subfigold@contcaption{##1}{##2}
362     \else
363       \subfigold@contcaption{##1}{##2}
364       \@listsubcaptions{##1}%
365     \fi}
366 }
367

```

We can now call the rest of the subfigure21 code, if required.

```

368 \if@contsubfigxxi
369

```

`\subbottom` `\subbottom[<list-entry>][<subcaption>]{<text>}` typesets a subcaption below the `\@contsubbody` `<text>`. Most of the work is performed by the `\@contsubbody` macro.

```

370 \newcommand{\subbottom}{%
371   \@contbotsubtrue
372   \@contsubbody}
373
374 \newcommand{\@contsubbody}{%
375   \bgroup
376   \if@contmaincaption\else
377     \advance\csname c@\@captype\endcsname\@ne
378   \fi
379   \refstepcounter{sub\@captype}\@contkeep%
380   \leavevmode
381   \@ifnextchar [%
382     {\@subfigure}%
383     {\@subfigure[\@empty]}}
384

```

`\contsubbottom` These are the continued versions of `\subbottom` and `\@contsubbody`.

```

\subbody@cont 385 \newcommand{\contsubbottom}{%
386   \@contbotsubtrue
387   \subbody@cont}
388
389 \newcommand{\subbody@cont}{%
390   \bgroup
391   \@contset
392   \refstepcounter{sub\@captype}\@contkeep%
393   \leavevmode
394   \@ifnextchar [%
395     {\@subfigure}%
396     {\@subfigure[\@empty]}}
397

```

`\subtop` These are similar to `\subbottom` and `\contsubbottom` except that they put the subcaption on top of the *text*.

`\contsubtop`

```

398 \newcommand{\subtop}{%
399   \@contbotsubfalse
400   \@contsubbody}
401
402 \newcommand{\contsubtop}{%
403   \@contbotsubfalse
404   \subbody@cont}
405
\contsubfigure This a simplified version of \subfigure in that the main caption counter is not
incremented (we should be in a continued float), and the subcounter is restored
before being incremented.
406 \newcommand{\contsubfigure}{%
407   \bgroup
408   \@contset
409   \refstepcounter{sub\@capttype}\@contkeep%
410   \@ifnextchar [%
411     {\@subfigure}%
412     {\@subfigure[\@empty]}}
413
And call for the new version of \@contcaption.
414 \doxxi@contcaption

\contsubtable Subtables are treated just like subfigures.
415 \let\contsubtable\contsubfigure

The end of the subfigure21 option code.
416 \fi
417
```

3.5 New floats

To define a float environment, say `fenv`, the following macros must be defined:

- `\fps@fenv` The default placement specifier (normally `tbp`).
- `\ftype@fenv` The type number which is an integer and a power of 2.
- `\ext@fenv` The file extension for the contents list.
- `\c@fenv` A counter for the environment (for caption numbering).
- `\fnum@fenv` A macro to generate the caption ‘number’.
- `\l@fenv` A macro to produce an entry in a list of. . .
- `\flegtoc@fenv` A macro to write a `\namedlegend` title to a listof file.
- `\fleg@fenv` A macro to typeset the name of a `\namedlegend`.

Note that the last two are only required for the `ccaption` package while the others are required for any new float, whether or not the `ccaption` package is being used.

`newflo@tctr` A counter for the type number of a new float. Normally figures are of type 1, tables type 2, and the next float type is then 4, and so on.

```
418 \newcounter{newflo@tctr}
419 \@ifundefined{c@figure}{\setcounter{newflo@tctr}{1}}{
420   \@ifundefined{c@table}{\setcounter{newflo@tctr}{2}}{
421     \setcounter{newflo@tctr}{4}}
422
```

`\newfloatenv` The command `\newfloatenv[<within>]{<fenv>}{<ext>}{<capname>}` creates a new float environment called *<fenv>*, with captions named *<capname>*. Caption titles will be written to file `\jobname.<ext>`. *<within>* can be used for numbering within the document *<within>* division.

Start off by checking if this is not a new float, by testing the float counter. We can rely on the normal L^AT_EX command/environment making code to handle the case when *<fenv>* has been defined previously.

```
423 \newcommand{\newfloatenv}[4][\@empty]{%
424   \@ifundefined{c@#2}{%
```

It hasn't been defined before, so check if numbering is going to be within a some other numbering scheme.

```
425     \ifx \@empty#1\relax
426       \newcounter{#2}
427     \else
```

We are numbering within, so check that there is a within counter.

```
428       \@ifundefined{c@#1}{\PackageWarning{ccaption}%
429         {#1 has no counter for use as a 'within'}}
430       \newcounter{#2}}%
```

There is a valid within counter. Do complicated things to define the `\thefenv`. With just `\newcounter{#2}` `\the#2` is automatically set to arabic numbering. We have to set it to `\thewithin.\the#2`.

```
431       {\newcounter{#2}[#1]%
432         \expandafter\edef\csname the#2\endcsname{%
433           \expandafter\noexpand\csname the#1\endcsname.\noexpand\arabic{#2}}}
434       \fi
```

Next we set the type for the new float, using `newflo@tctr`, and doubling it afterwards.

```
435     \@namedef{ftype@#2}{\value{newflo@tctr}} % float type
436     \addtocounter{newflo@tctr}{\value{newflo@tctr}}
```

Having set up the counters, the rest is easy. We just define the commands that are needed for floats.

```
437     \@namedef{fps@#2}{tbp} % default positioning
438     \@namedef{ext@#2}{#3} % file extension
439     \@namedef{fnum@#2}{#4~\@nameuse{the#2}} % caption naming
440     \@namedef{fleg@#2}{#4: } % named legend naming
441     \@namedef{flegtoc@#2}##1{} % legend title to listof
```

Define the macro that will format the entries. This is a straight copy from the book/article class code for figures (tables).

```
442     \@namedef{l@#2}{\@dottedtocline{1}{1.5em}{2.3em}}%
```

Finally, define the new (starred) floating environment.

```
443 \newenvironment{#2}{\@float{#2}}{\end@float}
444 \newenvironment{#2*}{\@dblfloat{#2}}{\end@dblfloat}
445 }{%
```

Error handling for previously defined environment.

```
446 \PackageError{ccaption}{#2 has been previously defined}{\@eha}
447 }
448 }
449
```

`\listfloats` The `\listfloats{<fenv>}{<heading>}` command can be used to produce a list of floats for `<fenv>`. The listing has title `<heading>`. First check if there is a file extension for `<fenv>`.

```
450 \newcommand{\listfloats}[2]{%
451 \ifundefined{ext@#1}{\PackageError{ccaption}%
452 {File extension for #1 is undefined}{\@eha}}{%
```

Decide on the heading style. For classes with chapters this will be a `\chapter*` heading, otherwise a `\section*` heading. Copy the code from the book or article class as appropriate. Section style heading is easy.

```
453 \ifundefined{chapter}{%
454 \section*{#2\@mkboth{\MakeUppercase{#2}}{\MakeUppercase{#2}}}}{%
```

Chapter style is more tedious.

```
455 \if@twocolumn
456 \@restonecoltrue\onecolumn
457 \else
458 \@restonecolfalse
459 \fi
460 \chapter*{#2\@mkboth{\MakeUppercase{#2}}{\MakeUppercase{#2}}}}{%
```

And start reading the file.

```
461 \@starttoc{\@nameuse{ext@#1}}
```

At the end, if it was a `\chapter*` style heading we may have to restore `twocolumn` typesetting.

```
462 \ifundefined{chapter}{\if@restonecol\twocolumn\fi}
463 }
464
```

To define subcaptions for use in a new float environment, say `fenv`, the following macros must be defined [Coc95]:

- A new counter `subfenv` for subcaption numbering.
- A new counter `extdepth`, where `ext` is the file extension for the contents list of `fenv`, for setting the contents depth.
- `\thesubfenv` for the formatting of the subcaption number.
- `\@thesubfenv` for typesetting the number.
- `\p@subfenv` for prepending to the subcaption number when it is referenced.
- `\ext@subfenv` the file extension for the contents list.

- `\l@subfenv` for formatting the contents list entry.
- `\@makesubfenvcaption` for typesetting the subcaption.

`\newsfloat` The macros necessary for subcaptions for a float defined via `\newfloatenv{⟨fenv⟩}` are created by `\newsfloat{⟨fenv⟩}`. It is an error if the `⟨fenv⟩` float environment has not been previously defined.

```

465 \newcommand{\newsfloat}[1]{%
466   \ifundefined{c@#1}{\PackageError{caption}%
467     {Float environment #1 has not been previously defined}{\@eha}}{%
468     \newcounter{thesub#1}[#1]
469     \@namedef{thesub#1}{\alph{sub#1}}
470     \@namedef{@thesub#1}{\subcaplabelfont\@nameuse{thesub#1}}\space}
471     \@namedef{p@sub#1}{\csname the#1\endcsname}
472     \@namedef{ext@sub#1}{\csname ext@#1\endcsname}
473     \@namedef{l@sub#1}{\dottedxxxline{\@nameuse{ext@sub#1}}{2}{3.9em}{2.3em}}
474     \newcounter{\@nameuse{ext@sub#1}depth}
475     \setcounter{\@nameuse{ext@sub#1}depth}{1}
476     \@namedef{@makesub#1caption}{\@makesubfigurecaption}
477   }}
478
    The end of this package.
479 \</usc>

```

A The perils of empty

My original code for the `\if@contemptyarg` command was as follows:

```

\newcommand{\if@contemptyarg}[3]{%
  \edef\@conttemp{\zap@space#1 \@empty}
  \ifx\@empty\@conttemp\relax #2\else #3\fi}

```

This uses the `\zap@space` kernel command and I wrote the code after looking at various code bits in the kernel and other packages, but I can't now remember which ones.

Donald Arseneau kindly pointed out the error of my ways and provided the robust solution which is used in the body of this package. The following is a slightly edited version of an email he sent me on the subject.

I'm not sure how exactly it is *supposed* to work because there are cases for which it will fail spectacularly. [These involved testing an argument that included macros of various forms]

There are several errors I am sure of though:

- You used `\edef` which is *not* allowed in \LaTeX — this creates a moving argument without any protection from `\protect`. Fragile commands will produce stack overflows and other errors. Even if you use `\protected@edef`, as is correct, you still make a moving argument to no purpose.

- `\zap@space` is not valid for general arguments. It fails if it ever sees an empty macro following a space. [e.g., `\def\none{}` used as `\if@contemptyarg{ \none}{ }{ }`]
- By making `\if@contemptyarg` skip over one of its parameters (`#2`, `#3`) you make it fail for nesting tabular or array environments.
- `\if@contemptyarg` is itself a fragile command, and will require `\protect` if it ever appears in a title or other moving argument. Since it is possible to do the test by expandable operations alone, it should be done that way.

I suggest you read `CTAN:tex-archive/info/aro-bend/answer.002` for a past discussion of detecting empty arguments, and then use a definition of `\if@contemptyarg` based on that discussion. You'll find it in `amsgen.dtx`, or use instead the improved version

The definition of `\if@contemptyarg` is based on the improved version that Donald supplied, only the macro names being changed.

References

- [Coc95] Steven Douglas Cochran. *The subfigure package*. March 1995. (Available from CTAN as file `subfigure.dtx`)
- [GMS94] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion*. Addison-Wesley Publishing Company, 1994.
- [Lin95] Anselm Lingnau. *An Improved Environment for Floats*. March 1995. (Available from CTAN as file `float.dtx`)
- [McCG95] James Darrell McCauley and Jeff Goldberg. *The endfloat package*. October 1995. (Available from CTAN as file `endfloat.dtx`)
- [Wil96] Peter R. Wilson. *LaTeX for standards: The LaTeX package files user manual*. NIST Report NISTIR, June 1996.