The ccaption package

Author: Peter Wilson, Herries Press
Maintainer: Will Robertson
will dot robertson at latex-project dot org

v3.2c 2011/08/07

Abstract

The ccaption package enables restyling of captions and 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 as well as a mechanism for creating new types of float environments and subfloats.

The package has been tested in conjunction with the tocloft, rotating, caption2, sidecap, subfigure, endfloat, longtable, xtab and hyperref packages.

Contents

List of Tables

List of Figures

1 Introduction

Some publishers require and some authors prefer captioning styles other than the one style provided by LaTeX. The ccaption package provides the tools to design your own captioning styles.

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, sidecap, subfigure (v2.0 and the current version), endfloat, longtable, xtab and the hyperref 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 LaTeX DOCSTRIP utility which enables the automatic extraction of the LaTeX macro source files [GMS94].

Section 2 provides a short overview of the commands in the package and shows some examples of their use. This section also gives examples of how LaTeX's captioning style can be changed to a limited extent without the use of any package and provides general information on floats. For a more comprehensive description of floats read Keith Reckdahl's excellent *Using Imported Graphics in LaTeX2e* [Rec97], although this was written before the advent of the ccaption package. The implementation is given in Section 5.

2 The ccaption package

2.1 Options

The package may take one or more options, depending on which other packages are used in conjunction with ccaption. The options are designed so that the package loading order does not matter. The current options are:

subfigure for use with the current version of Steven Douglas Cochran's subfigure package [Coc95].

subfigure20 when used together with the old version 2.0 of the subfigure package.

caption2 when used together with Harald Axel Sommerfeldt's caption or caption2 package [Som95].

titles When new floats, and their corresponding 'List of...', are defined, the list headings may be individually configured. The titles option disables the configuration mechanism. This may be useful if, say, the fncychap package is used to redefine the appearance of chapter titles.

For example, if the package is being used with both subfigure version 2.1 and caption then it should be called as:

```
\usepackage[subfigure,caption2]{ccaption}
```

2.2 Changing the caption style

The discussion in §3 includes example methods for changing the typeset appearance of captions without the benefit of any package. The caption2 and caption packages provides a set of predefined captioning styles, and the ccaption package also provides an easy means of changing the style.

The style of subcaptions is controlled by the subfigure package.

Note that if the caption2 option is used then it is assumed that the caption(2) package is being used and the facilities described in this section are unavailable.

`\captiondelim` The default captioning style is to put a delimeter in the form of a colon between the caption number and the caption title. The command `\captiondelim{`⟨*delim*⟩`}` can be used to change the delimeter. For example, to have an en-dash instead `\captiondelim{--␣}` will do the trick. Notice that no space is put between the delimeter and the title unless it is specified in the ⟨*delim*⟩ parameter. The package initially specifies `\captiondelim{: }` to give the normal delimeter.

`\captionnamefont` The ⟨*font*⟩ specified by `\captionnamefont{`⟨*font*⟩`}` is used for typesetting the caption name; that is, the first part of the caption upto and including the delimeter (e.g., the portion 'Table 3:'). ⟨*font*⟩ can be any kind of font specification and/or command and/or text. This first part of the caption is treated like: `{<font> Table 3; }`, so font declarations, not font text-style commands, are needed, like `\captionnamefont{\Large\sffamily}` to specify a large sans-serif font. The package initially specifies `\captionnamefont{}` to give the normal font.

`\captiontitlefont` Similarly, the ⟨*font*⟩ specified by `\captiontitlefont{`⟨*font*⟩`}` is used for typesetting the title text of a caption. For example, `\captiontitlefont{\itshape}` for an italic title text. The package initially specifies `\captiontitlefont{}` to give the normal font.

`\captionstyle` By default the name and title of a caption are typeset as a block (non-indented) paragraph. `\captionstyle{`⟨*style*⟩`}` can be used to alter this. Sensible values for ⟨*style*⟩ are: `\centering`, `\raggedleft` or `\raggedright` for styles corresponding to these declarations. The `\centerlastline` style gives a block paragraph but with the last line centered. The package initially specifies `\captionstyle{}` to give the normal block paragraph style.

`\hangcaption` `\indentcaption` `\normalcaption` The command `\hangcaption` will cause captions to be typeset with the second and later lines of a multiline caption title indented by the width of the caption name. The command `\indentcaption{`⟨*length*⟩`}` will indent title lines after the first by ⟨*length*⟩. These commands are independent of the `\captionstyle{...}`. Note that a caption will not be simultaneously hung and indented. The `\normalcaption` command undoes any previous `\hangcaption` or `\indentcaption` command. The package initially specifies `\normalcaption` to give the normal non-indented paragraph style.

`\changecaptionwidth` `\normalcaptionwidth` `\captionwidth` Issuing the command `\changecaptionwidth` will cause the captions to be typeset within a total width ⟨*length*⟩ as specified by `\captionwidth{`⟨*length*⟩`}`. Issuing the command `\normalcaptionwidth` will cause captions to be typeset as normal full width captions. The package initially specifies `\normalcaptionwidth` and `\captionwidth{\linewidth}` to give the normal width. If a caption is being set

Table 1
REDESIGNED TABLE CAPTION STYLE

three	III
five	V
eight	VIII

within the side captioned environments from the sidecap package [NiGa98] then it must be a `\normalcaptionwidth` caption.

`\precaption` `\postcaption` The commands `\precaption{`⟨*pretext*⟩`}` and `\postcaption{`⟨*posttext*⟩`}` specify ⟨*pretext*⟩ and ⟨*posttext*⟩ that will be processed at the start and end of a caption. For example

```
\precaption{\rule{\linewidth}{0.4pt}\par}
\postcaption{\rule{\linewidth}{0.4pt}}
```

will draw a horizontal line above and below the captions. The package initially specifies `\precaption{}` and `\postcaption{}` to give the normal appearance.

If any of the above commands are used in a float, or other, environment their effect is limited to the environment. If they are used in the preamble or the main text, their effect persists until replaced by a similar command with a different parameter value. The commands do not affect the apperance of the title in any **List of...**.

`\\` The normal LaTeX command `\\[`⟨*length*⟩`]` can be used within the caption text to start a new line. Remember that `\\` is a fragile command, so if it is used within text that will be added to a **List of...** it must be protected. As examples:

```
\caption{Title with a \protect\\ new line in both the body and List of}
\caption[List of entry with no new line]{Title with a \\ new line}
\caption[List of entry with a \protect\\ new line]{Title text}
```

Effectively, a caption is typeset as though it were:

```
\precaption
{\captionnamefont NAME NUMBER \captiondelim}
{\captionstyle\captiontitlefont THE TITLE}
\postcaption
```

Replacing the above commands by their defaults leads to the simple format:
`{NAME NUMBER: }{THE TITLE}`

As well as using the styling commands to make simple changes to the captioning style more noticeable modifications can also be made. To change the captioning style so that the name and title are typeset in a sans font it is sufficient to do:

```
\captionnamefont{\sffamily}
\captiontitlefont{\sffamily}
```

A more obvious change in styling is shown in table 1, which was coded as:

```
\begin{table}
```

```
\centering
\captionnamefont{\sffamily}
\captiondelim{}
\captionstyle{\\}
\captiontitlefont{\scshape}
\setlength{\belowcaptionskip}{10pt}
\caption{Redesigned table caption style} \label{tab:style}
\begin{tabular}{lr} \hline
 ...
\end{table}
```

This leads to the approximate caption format (processed within `\centering`):
`{\sffamily NAME NUMBER}{\\ \scshape THE TITLE}`
Note that the newline command (`\\`) cannot be put in the first part of the format (i.e., the `{\sffamily NAME NUMBER}`); it has to go into the second part, which is why it is specified via `\captionstyle{\\}` and not `\captiondelim{\\}`.

If a mixture of captioning styles will be used you may want to define a special caption command for each non-standard style. For example for the style of the caption in table 1:

```
\newcommand{\mycaption}[2][\@empty]{%
  \captionnamefont{\sffamily\hfill}
  \captiondelim{\hfill}
  \captionstyle{\centerlastline\\}
  \captiontitlefont{\scshape}
  \setlength{\belowcaptionskip}{10pt}
  \ifx\@empty#1 \caption{#2}\else \caption[#1]{#2}}
```

NOTE: Any code that involves the `@` sign must be either in a package (`.sty`) file or enclosed between a `\makeatletter` ... `\makeatother` pairing.

The code for the table 1 example can now be written as:

```
\begin{table}
\centering
\mycaption{Redesigned table caption style} \label{tab:style}
\begin{tabular}{lr} \hline
 ...
\end{table}
```

Note that in the code for `\mycaption` I have added two `\hfill` commands and `\centerlastline` compared with the original specification. It turned out that the original definitions worked for a single line caption but not for a multiline caption. The additional commands makes it work in both cases, forcing the name to be centered as well as the last line of a multiline title, thus giving a balanced appearence.

Table 2: A multi-part table

just a single line	1

Table 2: Continued

just a single line	2

2.3 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 2 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 is intended to be used to put an anonymous caption into a float environment, but may be used anywhere.

For example, the following code was used to produce the two-line table 3. The `\legend` command can be used within a float independently of any `\caption`

Table 2: Concluded

just a single line	3

Table 3: Another table

A legendary table	5
with two lines	6

The legend

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.

If you want the legend text to be included in the **List of...** use the `\addcontentsline` command in conjunction with the `\legend`. For example:

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

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

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

```
\begin{table}
\centering
\captiontitlefont{\sffamily}
```

Legendary table

An anonymous table	5
with two lines	6

Table: *Named legendary table*

seven	VII
eight	VIII

```
\legend{Legendary table}
\addcontentsline{lot}{table}{Legendary table (toc 1)}
\addcontentsline{lot}{table}{\protect\numberline{}Legendary table (toc 2)}
\begin{tabular}{lc} \hline
 An anonymous 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` `\belowlegendskip` 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.

`\namedlegend` 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 **List of...** 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 may use `\tablename` if called within a `table` environment). The following code is the source of the *Named legendary table*.

```
\begin{table}
\centering
\captionnamefont{\sffamily}
\captiontitlefont{\itshape}
\namedlegend{Named legendary table}
\begin{tabular}{lr} \hline
seven & VII \\
eight & VIII \\ \hline
\end{tabular}
\end{table}
```

`\fleg@type` The macro `\fleg@type{`⟨*name*⟩`}`, 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 defines the ⟨*name*⟩ to be used as the name in `\namedlegend`. Two defaults are provided, namely:

```
\newcommand{\fleg@table}{\tablename}
\newcommand{\fleg@figure}{\figurename}
```

which may be altered via `\renewcommand` if desired (put between a `\makeatletter` and `\makeatother` pair if done in the document).

`\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` `\renewfixedcaption` `\providefixedcaption`

The `\legend` command produces a plain, unnumbered heading. It can also be useful sometimes to have named and numbered captions outside a floating 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
  \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).

2.4 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.

`\bitwonumcaption` `\bionenumcaption`

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

`\bitwonumcaption[`⟨*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:

```
\bitwonumcaption{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:

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

EXAMPLE FIGURE WITH BITWONUMCAPTION

Figure 1: Long
Bild 1: Lang

EXAMPLE FIGURE WITH BIONENUMCAPTION

Figure 2: Long English
Bild 2: Lang Deutsch

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 `\bitwonumcaption`. 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 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` and was produced by the following code:

```
\begin{figure}
\centering
EXAMPLE FIGURE WITH A RULED BICAPTION
\precaption{\rule{\linewidth}{0.4pt}\par}
\midbicaption{\precaption{}\postcaption{\rule{\linewidth}{0.4pt}}}
\bicaption[fig:bi2]{Short English}{Longingly}{Bild}{Langlauf}
\end{figure}
```

`\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

EXAMPLE FIGURE WITH A RULED BICAPTION

Figure 3: Longingly
Bild 3: Langlauf

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}`

`\midbicaption` The bilingual captions are implemented by calling `\caption` twice, once for each language. The command `\midbicaption{`⟨*midtext*⟩`}`, which is similar to the `\precaption` and `\postcaption` commands, is executed just before calling the second `\caption`. Among other things, this can be used to modify the style of the second caption with respect to the first. For example, if there is normally a line above and below normal captions, it is probably undesirable to have a double line in the middle of a bilingual caption. So, for bilingual captions the following may be done within the float before the caption:

```
\precaption{\rule{\linewidth}{0.4pt}\par}
\postcaption{}
\midbicaption{\precaption{}\postcaption{\rule{\linewidth}{0.4pt}}}
```

This sets a line before the first of the two captions, then the `\midbicaption{...}` nulls the pre-caption line and adds a post-caption line for the second caption. The package initially specifies `\midbicaption{}`.

2.4.1 Bilingual captions with longtable

If ccaption and longtable are both being used, the longtable package must be loaded before the ccaption package as the ccaption package makes some changes to lontable's code.

Captions in a `longtable` work slightly differently than in other floats. This necessitates special versions of the bilingual caption commands for use in a `longtable`. These are similar to the commands described earlier but they do not take the optional ⟨*label*⟩ argument. If you need a `\label` put it in the second argument (⟨*long-1*⟩).

`\longbitwonumcaption` This corresponds to the `\bitwonumcaption` command and takes 5 arguments as:
`\longbitwonumcaption{`⟨*short-1*⟩`}{`⟨*long-1*⟩`}{`⟨*NAME*⟩`}{`⟨*short-2*⟩`}{`⟨*long-2*⟩`}`.
Both captions are numbered and both are put into the List of Tables (LoT).

`\longbionenumcaption` This corresponds to the `\bionenumcaption` command and takes 5 arguments as:
`\longbionenumcaption{`⟨*short-1*⟩`}{`⟨*long-1*⟩`}{`⟨*NAME*⟩`}{`⟨*short-2*⟩`}{`⟨*long-2*⟩`}`.
Both captions are numbered. Both are put into the ToC but only first is numbered there.

\longbicaption This corresponds to the \bicaption command and takes 4 arguments as: \longbicaption{⟨*short-1*⟩}{⟨*long-1*⟩}{⟨*NAME*⟩}{⟨*long-2*⟩}.
The first caption is numbered and put into the ToC. The second is neither numbered nor put into the ToC.

\midbicaption This is not used by the \longbi... caption commands; the style of both captions is the same. The spacing after a longtable caption, though, is controlled by the value of \belowcaptionskip.

2.5 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 desireable 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 \contsubbottom The command \contsubtop[⟨*list-entry*⟩][⟨*subcaption*⟩]{⟨*text*⟩} will continue the sub-caption numbering scheme across (continued) floats, putting the ⟨*subcaption*⟩ at the top of the ⟨*text*⟩. If both optional arguments are supplied, ⟨*list-entry*⟩ will be the entry in the List of... and ⟨*subcaption*⟩ will be used as the text for the subcaption. 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 \subbottom The command \subtop[⟨*list-entry*⟩][⟨*subcaption*⟩]{⟨*text*⟩} is in addition to the subfigure package commands \subfigure and \subtable. It puts the ⟨*subcaption*⟩ at the top of the ⟨*text*⟩, and similarly \subbottom[⟨*subcaption*⟩]{⟨*text*⟩} puts ⟨*subcaption*⟩ at the bottom of the ⟨*text*⟩.

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 LaTeX distribution, you may find that you have either version 2.0 or a later version of the subfigure package. If you have version 2.0, then call the ccaption package as:
`\usepackage[subfigure20]{ccaption}`, otherwise as:
`\usepackage[subfigure]{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 the current version. They also apply to version 2.0 except that the `\...top` and `\...bottom` commands do *not* take the first optional argument as: `\...top[`⟨*subcaption*⟩`]{`⟨*text*⟩`}`.

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 the current version 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.

2.6 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{contfigure}{%
   \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.

2.7 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 seperate 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.

`\newfloatlist` The command `\newfloatlist[`⟨*within*⟩`]{`⟨*fenv*⟩`}{`⟨*ext*⟩`}{`⟨*listname*⟩`}{`⟨*capname*⟩`}` creates both a new kind of floating environment called ⟨*fenv*⟩ and a new kind of 'List of' for ⟨*fenv*⟩; the title of this new listing is ⟨*listname*⟩. A caption within the environment will be written out to a file with extension ⟨*ext*⟩. The caption, if present, will start with ⟨*capname*⟩. 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 LaTeX uses `\listfigurename` to store the 'List of Figures' text and `\figurename` to store the 'Figure' text):

```
\newfloatlist{figure}{lof}{\listfigurename}{\figurename}
```

and the command `\listoffigure` (generated by `\newfloatlist`) would typeset the List of Figures.

The optional ⟨*within*⟩ 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:

```
\newfloatlist[chapter]{figure}{lof}{\listfigurename}{\figurename}
```

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

The `\newfloatlist` command generates several new commands that you can use for styling the new listing, similar to the facilities given by the tocloft package [Wil01]; for more detailed information you may wish to read the tocloft documentation. For ease of explanation, assume that the command was called as `\newfloatlist{X}{Z}{flist}{fcap}`, so that `X` corresponds to the name of the new environment ⟨*fenv*⟩ and `Z` corresponds to the file extension ⟨*ext*⟩. The following float environment and commands are then created.

`X` The new float environment is called `X`, and can be used as either `\begin{X}` or `\begin{X*}`, with the matching `\end{X}` or `\end{X*}`.

`\listofX` `\listofX` is similar to `\listoffigures`, etc., in that it typesets the new listing, and heads the list with the value of `flist`.

`Zdepth` The `Zdepth` counter is analogous to the standard `tocdepth` counter in that it specifies that entries in the new listing should not be typeset if their numbering level is greater than `Zdepth`. The default definition is `\setcounter{Zdepth}{1}`. To have a subfloat of `Z` appear in the listing do `\setcounter{Zdepth}{2}`.

`\cftmarkZ` This macro sets the appearance of the running heads on the new listing pages. The default definition gives the same appearance as for the LoF or LoT.

`\cftbeforeZtitleskip` `\cftafterZtitleskip` The lengths `\cftbeforeZtitleskip` and `\cftafterZtitleskip` control the vertical spacing before and after the title of the new listing. By default they are set to give the normal spacing, but you can change them with `\setlength` if you wish.

`\cftZtitlefont` `\cftafterZtitle` The code for typesetting the title of the new listing looks roughly like this

```
\vspace*{\cftbeforeZtitleskip}
{\cftZtitlefont flist}{\cftafterZtitle}
\cftmarkZ
\vskip \cftafterZtitleskip
```

The default definition of `\cftZtitlefont` is for a bold font. If, for example, you would prefer the title to be in a large italic font and set flushright you could:
`\renewcommand{\cftZtitlefont}{\hfill\Large\itshape}`
By default `\cftafterZtitle` is defined to do nothing; you can change it to serve your own purposes. For example:
`\renewcommand{\cftafterZtitle}{\thispagestyle{empty}}`
will set the page style of the first page of the new listing to be `empty`.

`\newfloatentry` The command `\newfloatentry[`⟨*within*⟩`]{`⟨*counter*⟩`}{`⟨*ext*⟩`}{`⟨*level-1*⟩`}` is used internally by `\newfloatlist` to generate a new counter for the new float environment and to generate the typesetting code for entries in the new listing. The required ⟨*counter*⟩ argument is the name for a new counter. If the optional ⟨*within*⟩ argument is used, the counter ⟨*counter*⟩ will be reset each time the counter ⟨*within*⟩ is changed. These first two arguments have the same effect as calling `\newcounter{`⟨*counter*⟩`}[`⟨*within*⟩`]`. The ⟨*ext*⟩ argument is the extension for the file holding the entries, and ⟨*level-1*⟩ is one less than the 'level' of the entry. Continuing the figure example,

`\newfloatlist[chapter]{figure}{lof}{\listfigurename}{\figurename}` will internally call `\newfloatentry[chapter]{figure}{lof}{0}`.

`\newfloatentry` generates a set of commands in addition to those directly generated by `\newfloatlist`. Assuming, as above, that we had `\newfloatlist{X}{Z}{flist}{fcap}` then we will also have `\newfloatentry{X}{Z}{0}`. This generates the following.

`X` The counter `X` matches the environment `X`. This counter is used for numbering captions. Remember that it will be reset according to the ⟨*within*⟩ argument.

`\theX` The command `\theX` prints the value of the `X` counter. It is initially defined so that it prints arabic numerals. If the optional ⟨*within*⟩ argument is used, `\theX` is defined as `\renewcommand{\theX}{\thewithin.\arabic{X}}` otherwise as `\renewcommand{\theX}{\arabic{X}}`.

`\cftbeforeXskip` This length controls the vertical space above a caption entry in the listing. It can be changed by using `\setlength`.

`\cftXindent` `\cftXnumwidth` The indentation of a caption entry in the listing from the left margin is given by the length `\cftXindent`, and the space for the caption number is set by the length `\cftXnumwidth`. These may be changed via `\setlength` or by `\setnewfloatindents`. The default values for these depend on the value of the ⟨*level-1*⟩ argument. A value of zero for this sets the defaults to the figure and table values. A value of one sets them to the defaults for subfigure and subtable values.

The code for typesetting a caption entry is roughly like:

```
{\cftXfont {\cftXpresnum SNUM\cftXaftersnum\hfil} \cftXaftersnumb TITLE}%
           {\cftXleader}{\cftXpagefont PAGE}\cftXafterpnum\par
```

where `SNUM` is the caption number, `TITLE` is the caption text, and `PAGE` is the page number. The other commands are described below.

`\cftXfont` This controls the appearance of the number and title. By default it is defined to use the normal font but it can be changed with `\renewcommand`.

`\cftXpresnum` `\cftXaftersnum` `\cftXaftersnumnb` The caption number is typeset in a box of width `\cftXnumwidth`. Within the box, `\cftXpresnum` is first called, then the number is typeset, then `\cftXaftersnum` is called and finally there is a `\hfil` to make the box contents flush left. After the number box is typeset `\cftXaftersnumb` is called and then the caption text is typeset. By default these three macros are defined to do nothing, but `\renewcommand` can be used to make them do something interesting.

`\cftXleader` `\cftXdotsep` `\cftXleader` defines the leader between the text and the page number; it can be changed by `\renewcommand`. By default it produces a dotted leader with `\cftXdotsep` space between the dots. It default definition is `\newcommand{\cftXdotsep}{4.5}` which gives a 4.5mu (math units) seperation. In spite of it appearing to be a length, changes to `\cftXdotsep` must be made by `\renewcommand`.

`\cftXpagefont` `\cftXafterpnum` `\cftXpagefont` specifies the font to be used for typesetting the page number. By default it is set to the normal font. Finally, `\cftXafterpnum` is called after

setting the page number; by default is does nothing. Both these commands can be changed by `\renewcommand`.

Note that `\newfloatlist` effectively generates all the above commands. Their defaults are set so that the typesetting mimics that for figure and table captions. It is probable that you can ignore all of them, but if you do want to change something the tocloft documentation provides many examples.

`\setnewfloatindents` The command `\setnewfloatindents{`⟨*fenv*⟩`}{`⟨*indent*⟩`}{`⟨*numwidth*⟩`}` sets the ⟨*fenv*⟩'s entry indent to the length ⟨*indent*⟩ and its numwidth to the length ⟨*numwidth*⟩. The ⟨*fenv*⟩ argument is the full name of the (sub)float.

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

```
\usepackage{ccaption}
...
\newcommand{\diagramname}{Diagram}
\newcommand{\listdiagramname}{List of Diagrams}
\newfloatlist{diagram}{dgm}{\listdiagramname}{\diagramname}
\newfixedcaption{\fdiagcaption}{diagram}
\begin{document}
...
\listoffigures
\listfofdiagram
...
\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 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 subfigure option, creates subcaptions (`\subtop` and `\subbottom`, together with their continued forms) for use within the float environment ⟨*fenv*⟩ previously defined via `\newfloatlist[...]{`⟨*fenv*⟩`}{...}`.

The `\newsubfloat` macro internally calls the `\newfloatentry` command and assuming our usual `\newfloatlist{X}{Z}{flist}{fcap}` then `\newsubfloat{X}`

calls
`\newfloatentry[X]{subX}{Z}{1}`
so there is a further set of `\cftsubX...` commands generated for adjusting the typesetting of the subcaption entries. Note that the full name of the entry in the listing is 'sub⟨*fenv*⟩', not just simply '⟨*fenv*⟩'.

`\newfloatpagesoff` The `\newfloatpagesoff{`⟨*fenv*⟩`}` command will turn off page numbering for list entries for ⟨*fenv*⟩. This is probably most likely to be used for switching off page numbers for subfloat entries, in which case it should be called as `\newfloatpagesoff{sub`⟨*fenv*⟩`}`.

`\newfloatpageson` The `\newfloatpageson{`⟨*fenv*⟩`}` command reverses the effect of a corresponding `\newfloatpagesoff{`⟨*fenv*⟩`}`.

`\newfloatenv` `\listfloats` **NOTE:** These two macros were in version 2.7 of the package but were replaced in version 3.0 by the functionally extended `\newfloatlist` and `\listofX` commands, respectively.

There is a limit to the number of List of... listings that (La)TeX can handle. Each kind of listing requires a `\jobname.ext` file and the TeX program has an upper limit on the number of files it can handle. In the most limited circumstance LaTeX requires three files — the `log`, `aux` and `dvi` files. Further files are required for things like a ToC (`toc`) or an index (`idx`). If you try and create too many new listings LaTeX will respond with the error message:

```
No room for a new write
```

If you get such a message the only recourse is to redesign your document.

3 How LaTeX makes captions

This section provides an overview of how LaTeX 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 LaTeX code or you want to make changes to LaTeX's style of captioning.

The LaTeX 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 `ltfloat.dtx`) defines the caption command via
`\def\caption{\refstepcounter\@captype \@dblarg{\@caption\@captype}}`

`\@captype` `\@captype` 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 `ltfloat.dtx`). For a `figure` environment, there is an equivalent to
`\def\@captype{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 responsibilty 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).

`\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` `\belowcaptionskip` 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. In figures, the caption is usually placed below the illustration. The actual space between the bottom of the illustration and the baseline of the first line of the caption is the `\abovecaptionskip` plus the `\parskip` plus the `\baselineskip`. If the illustration is in a `center` environment then additional space will be added by the `\end{center}`; it is usually better to use the `\centering` command rather than the `center` environment.

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

A THOUSAND WORDS...

FIGURE 4: A picture is worth a thousand words

ANOTHER THOUSAND WORDS...

Figure 5 — A different kind of figure caption

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}}
\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. Of course the ccaption package provides the tools that you need to make most, if not all, of any likely caption styles.

3.1 Changing the numbering scheme

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 package[1] which provides the definition of `\@removefromreset`.

You can play with other combinations of `\@addtoreset`, `\@removefromreset`, and `\renewcommand{\the...}{...}` to get the numbering scheme you want.

[1] Available on CTAN in `tex-archive/macros/latex/contrib/supported/carlise`.

3.2 Captions with footnotes

If you want to have a caption with a footnote, think long and hard as to whether this is really essential. It is not normally considered to be good typographic practice, and to rub the point in LaTeX does not make it necessarily easy to do. However, if you (or your publisher) insists, read on.

If it is present, the optional argument to `\caption` is put into the LoF/LoT as appropriate. If the argument is not present, then the text of the required argument is put into the LoF. In the first case, the optional argument is moving, and in the second case the required argument is moving. The `\footnote` command is fragile and must be `\protect`ed (i.e., `\protect\footnote{}`) if it is used in a moving argument. If you don't want the footnote to appear in the LoF, use a footnoteless optional argument and a footnoted required argument.

You will probably be surprised if you just do, for example:

```
\begin{figure}
...
\caption[For LoF]{For figure\footnote{The footnote}}
\end{figure}
```

because (a) the footnote number may be greater than you thought, and (b) the footnote text has vanished. This later is because LaTeX won't typeset footnotes from a float. To get an actual footnote within the float you have to use a minipage, like:

```
\begin{figure}
\begin{minipage}{\linewidth}
...
\caption[For LoF]{For figure\footnote{The footnote}}
\end{minipage}
\end{figure}
```

Now you may find that you get two footnotes for the price of one. Fortunately, if you use the ccaption package *without* the caption2 option, this will not occur.

When using a minipage as above, the footnote text is typeset at the bottom of the minipage (i.e., within the float). If you want the footnote text typeset at the bottom of the page, then you have to use the `\footnotemark` and `\footnotetext` commands like:

```
\begin{figure}
...
\caption[For LoF]{For figure\footnotemark}
\end{figure}
\footnotetext{The footnote}
```

This will typeset the argument of the `\footnotetext` command at the bottom of the page where you called the command. Of course, the figure might have floated

ILLUSTRATION 1 ILLUSTRATION 2

Figure 6 — Float with two illustrations

to a later page, and then it's a matter of some manual fiddling to get everything on the same page, and possibly to get the footnote marks to match correctly with the footnote text.

At this point, you are on your own.

4 Floats

As far as LaTeX is concerned, a float is a box which certain restrictions as to where it can be placed.

4.1 Multiple floats

You can effectively put what you like inside a float box. Normally there is just a single picture or tabular in a float but you can put as many of these as will fit inside a float.

Three typical cases of multiple figures/tables in a single float come to mind:

- Multiple illustrations/tabulars with a single caption.
- Multiple illustrations/tabulars each individually captioned.
- Multiple illustrations/tabulars with one main caption and individual subcaptions.

The subfigure package is designed for the last of these cases; the others do not require a package.

Figure 6 is an example of multiple illustrations in a single float with a single caption. This figure was produced by the following code.

```
\begin{figure}
\centering
\hspace*{\fill} {ILLUSTRATION 1} \hfill {ILLUSTRATION 2} \hspace*{\fill}
\caption{Float with two illustrations} \label{fig:mult1}
\end{figure}
```

The `\hspace*{\fill}` and `\hfill` commands were used to space the two illustrations equally. Of course `\includegraphics` or `tabular` environments could just as well be used instead of the `{ILLUSTRATION N}` text.

The following code produces Figures 7 and 8 which are examples of two seperately captioned illustrations in one float.

```
\begin{figure}
\centering
```

ILLUSTRATION 3

Figure 7 — Illustration 3

ILLUSTRATION 4

Figure 8 — Illustration 4

```
\begin{minipage}{0.4\textwidth}
  \centering
  ILLUSTRATION 3
  \caption{Illustration 3} \label{fig:mult2}
\end{minipage}
\hfill
\begin{minipage}{0.4\textwidth}
  \centering
  ILLUSTRATION 4
  \caption{Illustration 4} \label{fig:mult3}
\end{minipage}
\end{figure}
```

In this case the illustrations (or graphics or tabulars) are put into seperate `minipage` environments within the float, and the captions are also put within the `minipage`s. Note that any required `\label` must also be inside the `minipage`. If you wished, you could add yet another caption after the end of the two `minipage`s.

Keith Reckdahl [Rec97] provides more examples of this kind of thing.

4.2 Where LaTeX puts floats

The general format for a float environment is:
`\begin{float}[`⟨*loc*⟩`] ... \end{float}` or for double column floats:
`\begin{float*}[`⟨*loc*⟩`] ... \end{float*}`
where the optional argument ⟨*loc*⟩, consisting of one or more characters, specifies a location where the float may be placed. Note that the multicol package only supports the starred floats and it will not let you have a single column float. The possible ⟨*loc*⟩ values are one or more of the following:

- `b` *bottom*: at the bottom of a page. This does not apply to double column floats as they may only be placed at the top of a page.
- `h` *here*: if possible exactly where the float environment is defined. It does not apply to double column floats.
- `p` *page*: on a seperate page containing only floats (no text).
- `t` *top*: at the top of a page.
- `!` make an extra effort to place the float at the earliest place specified by the rest of the argument.

The default for ⟨*loc*⟩ is `tbp`, so the float may be placed at the top, or bottom, or on a float-only page; the default works well 95% of the time. Floats of the same kind

are output in definition order, except that a double column float may be output before a later single column float of the same kind, or *vice-versa*[2]. A float is never put on an earlier page than its definition but may be put on the same or later page of its definition. If a float cannot be placed, all suceeding floats will be held up, and LaTeX can store no more than 16 held up floats. A float cannot be placed if it would cause an overfull page, or it otherwise cannot be fitted according the the float parameters. A `\clearpage` or `\cleardoublepage` or `\end{document}` flushes out all unprocessed floats, irrespective of the ⟨*loc*⟩ and float parameters, putting them on float-only pages.

`\suppressfloats` You can use the command `\suppressfloats[`⟨*pos*⟩`]` to suppress floats at a given ⟨*pos*⟩ on the current page. `\suppressfloats[t]` prevents any floats at the top of the page and `\suppressfloats[b]` prevents any floats at the bottom of the page. The simple `\suppressfloats` prevents both top and bottom floats.

The flafter package, which should have come with your LaTeX distribution, provides a means of preventing floats from moving backwards from their definition position in the text. This can be useful to ensure, for example, that a float early in a `\section{}` is not typeset before the section heading.

Table 4 lists the various float parameters and typical default values. All the lengths are rubber lengths, and the actual defaults depend on both the class and its size option.

Given the displayed defaults, the height of a top float must be less than 70% of the textheight and there can be no more than 2 top floats on a text page. Similarly, the height of a bottom float must not exceed 30% of the textheight and there can be no more than 1 bottom float on a text page. There can be no more than 3 floats (top, bottom and here) on the page. At least 20% of a text page with floats must be text. On a float page (one that has no text, only floats) the sum of the heights of the floats must be at least 50% of the textheight. The floats on a float page should be vertically centered.

It can be seen that with the defaults LaTeX might have trouble finding a place for a float. Consider what will happen if a float is a bottom float whose height is 40% of the textheight and this is followed by a float whose height is 90% of the textheight. The first is too large to actually go at the bottom of a text page but too small to go on a float page by itself. The second has to go on a float page but it is too large to share the float page with the first float. LaTeX is stuck!

At this point it is worthwhile to be precise about the effect of a one character ⟨*loc*⟩ argument:

`[b]` means: 'put the float at the bottom of a page with some text above it, and nowhere else'. The float must fit into the `\bottomfraction` space otherwise it and subsequent floats will be held up.

`[h]` means: 'put the float at this point and nowhere else'. The float must fit into the space left on the page otherwise it and subsequent floats will be held up.

[2]This little quirk is fixed by the fixltx2e package, at least for tables and figures. The package is part of a normal LaTeX distribution.

Table 4: Float placement parameters

Parameter	Controls	Default
Counters — change with `\setcounter`		
`topnumber`	max number of floats at top of a page	2
`bottomnumber`	max number of floats at bottom of a page	1
`totalnumber`	max number of floats on a text page	3
`dbltopnumber`	like `topnumber` for double column floats	2
Commands — change with `\renewcommand`		
`\topfraction`	max fraction of page reserved for top floats	0.7
`\bottomfraction`	max fraction of page reserved for bottom floats	0.3
`\textfraction`	min fraction of page that must have text	0.2
`\dbltopfraction`	like `\topfraction` for double column floats	0.7
`\floatpagefraction`	min fraction of a float page that must have float(s)	0.5
`\dblfloatpagefraction`	like `\floatpagefraction` for double column floats	0.5
Text page lengths — change with `\setlength`		
`\floatsep`	vertical space between floats	12pt
`\textfloatsep`	vertical space between a top (bottom) float and suceeding (preceeding) text	20pt
`\intextsep`	vertical space above and below an `h` float	12pt
`\dblfloatsep`	like `\floatsep` for double column floats	12pt
`\dbltextfloatsep`	like `\textfloatsep` for double column floats	20pt
Float page lengths — change with `\setlength`		
`\@fptop`	space at the top of the page	`0pt plus 1fil`
`\@fpsep`	space between floats	`8pt plus 2fil`
`\@fpbot`	space at the bottom of the page	`0pt plus 1fil`
`\@dblfptop`	like `\@fptop` for double column floats	`0pt plus 1fil`
`\@dblfpsep`	like `\@fpsep` for double column floats	`8pt plus 2fil`
`\@dblfpbot`	like `\@fpbot` for double column floats	`0pt plus 1fil`

`[p]` means: 'put the float on a page that has no text but may have other floats on it'. There must be at least '`\floatpagefraction`' worth of floats to go on a float only page before the float will be be output.

`[t]` means: 'put the float at the top of a page with some text below it, and nowhere else'. The float must fit into the `\topfraction` space otherwise it and subsequent floats will be held up.

`[!...]` means: 'ignore the `\...fraction` values for this float'.

You must try and pick a combination from these that will let LaTeX find a place to put your floats. However, you can also can change the float parameters to make it easier to find places to put floats. Some examples are:

- Decrease `\textfraction` to get more 'float' on a text page, but the sum of `\textfraction` and `\topfraction` and the sum of `\textfraction` and `\bottomfraction` should not exceed 1, otherwise the placement algorithm falls apart. A minimum value for `\textfraction` is about 0.10 — a page with less than 10% text looks better with no text at all, just floats.
- Both `\topfraction` and `\bottomfraction` can be increased, and it does not matter if their sum exceeds 1.0. A good typographic style is that floats are encouraged to go at the top of a page, and a better balance is achieved if the float space on a page is larger at the top than the bottom.
- Making `\floatpagefraction` too small might have the effect of a float page just having one small float. However, to make sure that a float page never has more than one float on it, do:
  `\renewcommand{\floatpagefraction}{0.01}`
  `\setlength{\@fpsep}{\textheight}`
- Setting `\@fptop` to `0pt`, `\@fpsep` to `8pt` and `\@fpbot` to `0pt plus 1fil` will force floats on a float page to start at the top of the page.

If you are experimenting, a reasonable starting position is:

```
\setcounter{topnumber}{3}
\setcounter{bottomnumber}{2}
\setcounter{totalnumber}{4}
\renewcommand{\topfraction}{0.85}
\renewcommand{\bottomfraction}{0.5}
\renewcommand{\textfraction}{0.15}
\renewcommand{\floatpagefraction}{0.7}
```

and similarly for double column floats if you will have any.

One of LaTeX's little quirks is that on a text page, the 'height' of a float is its actual height plus `\textfloatsep` or `\floatsep`, while on a float page the 'height' is the actual height. This means that when using the default ⟨*loc*⟩ of `[tbp]` at least one of the text page float fractions (`\topfraction` and/or `\bottomfraction`) must be larger than the `\floatpagefraction` by an amount sufficient to take account of the maximum text page seperation value.

5 The package code

1 ⟨*usc⟩

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

Note (2001/08/03): Older versions of the amsmath package did odd things with `\@tempa`, `\@tempb` and `\@tempc`. I have replaced any original use of these by `\@conttempa`, etc.

Do the options first.

`\if@contsubfigxx` `\if@contsubfigxxi` `\if@contsubfig` These three `\if...` are used to remember if the subfigure20 or subfigure option has been given.

```
2 \newif\if@contsubfigxx
3   \@contsubfigxxfalse
4 \newif\if@contsubfigxxi
5   \@contsubfigxxifalse
6 \newif\if@contsubfig
7   \@contsubfigfalse
8 \DeclareOption{subfigure20}{\@contsubfigxxtrue\@contsubfigxxifalse\@contsubfigtrue}
9 \DeclareOption{subfigure21}{\@contsubfigxxfalse\@contsubfigxxitrue\@contsubfigtrue
10  \PackageWarningNoLine{ccaption}{%
11      The subfigure21 option is deprecated.\MessageBreak
12      Try and use the subfigure option instead}}
13 \DeclareOption{subfigure}{\@contsubfigxxfalse\@contsubfigxxitrue\@contsubfigtrue}
```

`\if@contcapoption` This `\if...` is used to remember if the caption2 option has been given

```
14 \newif\if@contcapoption
15   \@contcapoptionfalse
16 \DeclareOption{caption2}{\@contcapoptiontrue}
```

`\if@conttitleopt` This `\if...` is used to remember if the titles option has been given

```
17 \newif\if@conttitleopt
18   \@conttitleoptfalse
19 \DeclareOption{titles}{\@conttitleopttrue}
```

`\ProcessOptions` Now process the options.

```
20
21 \ProcessOptions\relax
22
```

5.1 Caption styling

The caption styling[3] is accomplished by redefining the `\@makecaption` command. First, though, define and initialise the user-level commands.

The styling is only defined if the caption2 option is *not* given. But first we have to declare some new `\if` commands before testing the option.

[3] Thanks to Donald Arseneau and Arash Esbatil for their perceptive comments on early versions of the styling code.

\if@contcw
\if@conthang
\if@contindent

For use when checking caption width and captioning styles styles.

```
\newif\if@contcw
\newif\if@conthang
\newif\if@contindent

```

Issue a warning if the caption2 option has been used.

```
\if@contcapoption
  \PackageWarningNoLine{ccaption)}%
    {You have used the caption2 option.\MessageBreak
     The ccaption styling commands\MessageBreak
     are unavailable to you}
\else

```

\captiondelim
\@contdelim

For the caption delimeter.

```
\newcommand{\captiondelim}[1]{\def\@contdelim{#1}}
\captiondelim{: }

```

\captionnamefont
\@contnfont

The font for the caption name.

```
\newcommand{\captionnamefont}[1]{\def\@contnfont{#1}}
\captionnamefont{}

```

\captiontitlefont
\@conttfont

The font for the caption title.

```
\newcommand{\captiontitlefont}[1]{\def\@conttfont{#1}}
\captiontitlefont{}

```

\flushleftright
\centerlastline

These are in addition to the \centering, \raggedleft and \raggedright declarations for paragraphing. \flushleftright sets the skips to TeX's normal (block) paragraphing values, while \centerlastline sets the skips to give a centered last line in a block paragraph.

```
\newcommand{\flushleftright}{%
  \leftskip\z@ \rightskip\z@
  \parfillskip=\z@ plus 1fil}
\newcommand{\centerlastline}{%
  \leftskip=\z@ plus 1fil
  \rightskip=\z@ plus -1fil
  \parfillskip=\z@ plus 2fil}

```

\captionstyle
\@contcstyle

The paragraphing style for the caption.

```
\newcommand{\captionstyle}[1]{\def\@contcstyle{#1}}
\captionstyle{}

```

\@contcwidth
\captionwidth
\changecaptionwidth
\normalcaptionwidth

The macros for dealing with the caption width.

```
\newlength{\@contcwidth}
\newcommand{\captionwidth}[1]{\setlength{\@contcwidth}{#1}}
\captionwidth{\linewidth}
\newcommand{\changecaptionwidth}{\@contcwtrue}
\newcommand{\normalcaptionwidth}{\@contcwfalse}
\normalcaptionwidth

```

\@contindw
\hangcaption
\indentcaption
\normalcaption

The macros for hanging and indented captions.

```
\newlength{\@contindw}
\newcommand{\hangcaption}{\@conthangtrue\@contindentfalse}
\newcommand{\indentcaption}[1]{\setlength{\@contindw}{#1}%
  \@conthangfalse\@contindenttrue}
\newcommand{\normalcaption}{\@conthangfalse\@contindentfalse}
\normalcaption

```

\precaption
\@contpre
\postcaption
\@contpost
\midbicaption
\@contmidbi

The macros for the pre- and post-caption text/commands, and for the mid-caption command for bilingual captions.

```
\newcommand{\precaption}[1]{\def\@contpre{#1}}
\precaption{}
\newcommand{\postcaption}[1]{\def\@contpost{#1}}
\postcaption{}
\newcommand{\midbicaption}[1]{\def\@contmidbi{#1}}
\midbicaption{}

```

\@makecaption

This is a reimplementation of the kernel `\@makecaption` command. As well as including the caption typesetting commands it enables captions that include forced newlines (e.g., by `\\`).

The first part is due to Donald Arseneau[4] from postings to the CTT newsgroup and Email discussions. The `\topskip` strut is used whenever the caption is the first part of the float. This means, among other things, that if a caption comes at the top of a page, then the first line of the caption will be aligned with the normal first line of a page. The `\abovecaptionskip` is only used when there is something above the caption in the current float.

```
\long\def\@makecaption#1#2{\let\@conttempa\relax
  \ifdim\prevdepth>-99\p@ \vskip\abovecaptionskip
  \else \def\@conttempa{\vbox to\topskip{}}\fi
```

\@contfnote
\@contfmark

The caption title will be typeset twice, firstly to measure its width and secondly to actually typeset it. To avoid problems caused by a footnote in the caption getting processed twice, we temporarily disable the expected relevant commands.

```
  \let\@contfnote\footnote \renewcommand{\footnote}[2][]{}
  \let\@contfmark\footnotemark \renewcommand{\footnotemark}[1][]{}
  \let\@contlabel\label \renewcommand{\label}[1]{}
```

[4] Email: `asnd@triumf.ca`

Now measure the width of the total caption, not forgetting to take account of the font specifications, and then restore the footnoting.

```
\sbox\@tempboxa{\@contnfont #1\@contdelim \@conttfont #2}
\let\footnote\@contfnote
\let\footnotemark\@contfmark
\let\label\@contlabel
```

If the caption is less than one line, then the whole caption needs to be centered on the page (otherwise the short caption may be typeset flushleft).

```
\ifdim\wd\@tempboxa<\linewidth \centering \fi
\if@contcw
```

For typesetting at anything other than the normal width, put the caption into a `\parbox` of the specified width. This must be centered.

```
    \centering
    \parbox{\@contcwidth}{%
  \fi
  \if@conthang
```

For a hanging caption we have to measure the width of the caption name, then typeset the whole caption in a hanging paragraph.

```
    \sbox\@tempboxa{\@contnfont #1\@contdelim}
    \@contpre%
    {\@contnfont #1\@contdelim}\@conttempa
    {\@contcstyle\hangindent=\wd\@tempboxa\hangafter=\@ne\@conttfont #2\par}
  \else
    \if@contindent
```

An indented caption is similar, except the amount of indentation is kept in `\@contindw`.

```
      \@contpre%
      {\@contnfont #1\@contdelim}\@conttempa
      {\@contcstyle\hangindent=\@contindw\hangafter=\@ne\@conttfont #2\par}
    \else
```

For the normal style, just typeset the caption.

```
      \@contpre%
      {\@contnfont #1\@contdelim}\@conttempa
      {\@contcstyle\@conttfont #2\par}
    \fi
  \fi
```

Finish off the typesetting by processing the post-text, and if not using the normal width then close off the `\parbox`, and lastly put in some vertical space.

```
  \@contpost
  \if@contcw
    \par
    }  % end of parbox
  \fi
  \vskip\belowcaptionskip}

```

This finishes off the non caption2 option.

```
\fi  % end of test (\if@contcapoption) on caption2 option
```

5.2 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 too much with numbering or list of ... entries.

```
\newcommand{\contcaption}{%
  \addtocounter{\@captype}{\m@ne}%
  \refstepcounter{\@captype}%
  \@contcaption\@captype}
```

`\@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 LaTeX `\@caption` command.

```
\long\def\@contcaption#1#2{%
  \par
  \begingroup
     \@parboxrestore
     \if@minipage
       \@setminipage
     \fi
     \normalsize
     \@makecaption{\csname fnum@#1\endcsname}{\ignorespaces #2}\par
  \endgroup}
```

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

```
\newlength{\abovelegendskip}
\setlength{\abovelegendskip}{0.5\baselineskip}
\newlength{\belowlegendskip}
\setlength{\belowlegendskip}{\abovelegendskip}
```

`\legend` The command is called as `\legend{`⟨*text*⟩`}`. It is intended to be used in a float environment for an 'anonymous' caption, but can be used anywhere.

The implementation is similar to the `\caption` command but we have to eliminate printing of a delimeter.

```
\newcommand{\legend}[1]{%
  \par
  \begingroup
     \@parboxrestore
```

```
    \if@minipage
      \@setminipage
    \fi
    \normalsize
    \captiondelim{\mbox{}}
    \@makecaption{}{\ignorespaces #1}\par
  \endgroup}
```

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

```
\newcommand{\namedlegend}{\@dblarg{\@legend\@captype}}
```

`\@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 typeseting the name of the legend.

```
\long\def\@legend#1[#2]#3{%
  \par
  \csname flegtoc@#1\endcsname{#2}%
  \begingroup
    \@parboxrestore
    \if@minipage
      \@setminipage
    \fi
    \normalsize
    \@makecaption{\csname fleg@#1\endcsname}{\ignorespaces #3}\par
  \endgroup}
```

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

```
\newcommand{\flegtoc@table}[1]{}
\newcommand{\flegtoc@figure}[1]{}
```

`\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.

```
\newcommand{\fleg@table}{\tablename}
\newcommand{\fleg@figure}{\figurename}
```

5.3 Non-float captions

`\newfixedcaption` `\renewfixedcaption` `\providefixedcaption` These commands are defined in terms of their `\...command` counterparts. Call as `\...fixedcaption[⟨capcommand⟩]{⟨command⟩}{⟨env⟩}`

```
\newcommand{\newfixedcaption}[3][\caption]{%
  \newcommand{#2}{\def\@captype{#3}#1}}
\newcommand{\renewfixedcaption}[3][\caption]{%
  \renewcommand{#2}{\def\@captype{#3}#1}}
\newcommand{\providefixedcaption}[3][\caption]{%
  \providecommand{#2}{\def\@captype{#3}#1}}

```

5.4 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).

```
\begingroup
\catcode`\Q=3
\long\gdef\@if@contemptyarg#1{\@xif@contmt#1QQ\@secondoftwo\@firstoftwo\@nil}
\long\gdef\@xif@contmt#1#2Q#3#4#5\@nil{#4}
\endgroup

```

`\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.

```
\newcommand{\bitwonumcaption}[6][\@empty]{%
  \begingroup
```

Check if the first language argument is vacuous, then call the normal `\caption` for language 1.

```
  \@if@contemptyarg{#2}{\caption{#3}}{\caption[#2]{#3}}
```

Do the optional labeling.

```
  \ifx\@empty#1\else
    \label{#1}
  \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.

```
  \setlength{\abovecaptionskip}{0pt}
```

```
\setlength{\belowcaptionskip}{0pt}
\edef\@conttempc{#4}
\expandafter \renewcommand \csname \@captype name\endcsname{\@conttempc}
\addtocounter{\@captype}{-1}
```

Now repeat for the second language caption.

```
\@contmidbi
\@if@contemptyarg{#5}{\caption{#6}}{\caption[#5]{#6}}
\endgroup}

```

`\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.

```
\newcommand{\bionenumcaption}[6][\@empty]{%
  \begingroup
```

Check if the first language argument is vacuous, then call the normal `\caption` for language 1.

```
  \@if@contemptyarg{#2}{\caption{#3}}{\caption[#2]{#3}}
```

Do the optional labeling.

```
  \ifx\@empty#1\else
    \label{#1}
  \fi
```

Do the between captions code.

```
  \setlength{\abovecaptionskip}{0pt}
  \setlength{\belowcaptionskip}{0pt}
  \edef\@conttempc{#4}
  \expandafter \renewcommand \csname \@captype name\endcsname{\@conttempc}
```

Use a continuation caption for the second language, not forgetting to add the appropriate unnumbered text to the List.

```
  \@contmidbi
  \contcaption{#6}
  \@if@contemptyarg{#5}{%
    \addcontentsline{\csname ext@\@captype\endcsname}{\@captype}%
      {\protect\numberline{}{\ignorespaces #6}}}{%
    \addcontentsline{\csname ext@\@captype\endcsname}{\@captype}%
      {\protect\numberline{}{\ignorespaces #5}}}
  \endgroup}

```

`\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.

```
\newcommand{\bicaption}[5][\@empty]{%
  \begingroup
```

Check if the first language argument is vacuous, then call the normal `\caption` for language 1.

```
  \@if@contemptyarg{#2}{\caption{#3}}{\caption[#2]{#3}}
```

Do the optional labeling.

```
\ifx\@empty#1\else
  \label{#1}
\fi
```

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

```
\setlength{\abovecaptionskip}{0pt}
\setlength{\belowcaptionskip}{0pt}
\edef\@conttempc{#4}
\expandafter \renewcommand \csname \@captype name\endcsname{\@conttempc}
\@contmidbi
\contcaption{#5}
\endgroup}

```

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

```
\newcommand{\bicontcaption}[3]{%
  \begingroup
```

Call `\contcaption` for language 1.

```
  \contcaption{#1}
```

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

```
  \setlength{\abovecaptionskip}{0pt}
  \setlength{\belowcaptionskip}{0pt}
  \edef\@conttempc{#2}
  \expandafter \renewcommand \csname \@captype name\endcsname{\@conttempc}
  \@contmidbi
  \contcaption{#3}
  \endgroup}

```

5.5 The code for the longtable package

`\LT@makecaption` This is defined in the longtable package and sets a caption essentially as a centered multicolumn entry in the table. To utilize ccaption's font settings it has to be modified. N.B. that `#1` is either `\@firstofone` or `\@gobble` so we need double braces to protect the font change.

```
\def\LT@makecaption#1#2#3{%
  \LT@mcol\LT@cols c{\hb@xt@ \z@{\hss\parbox[t]\LTcapwidth{%
    \sbox\@tempboxa{#1{{\@contnfont #2\@contdelim}}\@conttfont #3}%
    \ifdim\wd\@tempboxa>\hsize
      #1{{\@contnfont #2\@contdelim}}\@conttfont #3%
    \else
      \hb@xt@ \hsize{\hfil\box\@tempboxa\hfil}%
    \fi
    \endgraf\vskip\belowcaptionskip}%
  \hss}}}
```

\longbitwonumcaption A version of \bitwonumcaption for use in a longtable.

```
\newcommand*{\longbitwonumcaption}[5]{%
  \@if@contemptyarg{#1}{\caption{#2}}{\caption[#1]{#2}}%
  \global\let\@cont@oldtablename\tablename
  \gdef\tablename{#3}
  \\
  \@if@contemptyarg{#4}{\caption{#5}}{\caption[#4]{#5}}%
  \global\let\tablename\@cont@oldtablename}
```

\@cont@LT@nonumintoc
\@cont@oldLT@c@ption We need a special version of longtable's \LT@c@ption that does not put a number in the ToC.

```
\def\@cont@LT@nonumintoc#1[#2]#3{%
  \LT@makecaption#1\fnum@table{#3}%
  \def\@tempa{#2}%
  \ifx\@tempa\@empty\else
    {\let\\\space
      \addcontentsline{lot}{table}{\protect\numberline{}{#2}}}%
  \fi}
\let\@cont@oldLT@c@ption\LT@c@ption
```

\longbionenumcaption A version of \bionenumcaption for use in a longtable.

```
\newcommand*{\longbionenumcaption}[5]{%
  \@if@contemptyarg{#1}{\caption{#2}}{\caption[#1]{#2}}%
  \global\let\@cont@oldtablename\tablename
  \gdef\tablename{#3}
  \global\let\LT@c@ption\@cont@LT@nonumintoc
  \\
  \@if@contemptyarg{#4}{\caption{#5}}{\caption[#4]{#5}}%
  \global\let\tablename\@cont@oldtablename
  \global\let\LT@c@ption\@cont@oldLT@c@ption}
```

\longbicaption A version of \bicaption for use in a longtable.

```
\newcommand*{\longbicaption}[4]{%
  \@if@contemptyarg{#1}{\caption{#2}}{\caption[#1]{#2}}%
  \\
  \caption*{{\normalfont\@contnfont #3\@contdelim} #4}}
```

5.6 The subfigure options

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

```
\if@contsubfig
  \newcounter{@contsubnum}
  \newcommand{\@contkeep}{\setcounter{@contsubnum}{\value{sub\@captype}}}
  \newcommand{\@contset}{\setcounter{sub\@captype}{\value{@contsubnum}}}
  \newcommand{\subconcluded}{\setcounter{sub\@captype}{0}}
  \let\subfigold@contcaption\@contcaption
```

`\toclevel@subtable` `\toclevel@subfigure` These are needed if the hyperref package is loaded as well as subfigures.

```
  \providecommand{\toclevel@subtable}{1}
  \providecommand{\toclevel@subfigure}{1}
\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.)

```
\newif\if@contmaincaption
  \@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.

```
\newif\if@contbotsub
  \@contbotsubtrue

```

5.6.1 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.

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

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

```
  \let\@contoldc\caption
  \renewcommand{\caption}{\@contmaincaptiontrue\@contoldc}
  \let\@contoldcont\contcaption
  \renewcommand{\contcaption}{\@contmaincaptiontrue\@contoldcont}
  \let\@contoldf\@float
  \renewcommand{\@float}[1]{\@contmaincaptionfalse
                \@ifundefined{c@sub#1}{}{\csname c@sub#1\endcsname = 0\relax}
                \@contoldf{#1}}
  \let\@contoldff\@dbflt
  \renewcommand{\@dbflt}[1]{\@contmaincaptionfalse
                \@ifundefined{c@sub#1}{}{\csname c@sub#1\endcsname = 0\relax}
                \@contoldff{#1}}

```

\@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.

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

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

```
    \vtop{%
      \vbox{\vskip\subfigtopskip
            \box\@tempboxa}%
      \ifx \@empty#2\relax \else
        \vskip\subfigcapskip
        \@subcaption{#1}{#2}%
      \fi
      \vskip\subfigbottomskip}%
  \else
```

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

```
    \vtop{%
      \ifx \@empty#2\relax \else
        \vskip\subfigcapskip
        \begingroup\@subcaption{#1}{#2}\endgroup%
      \fi
      \vbox{\vskip\subfigtopskip
            \box\@tempboxa}%
      \vskip\subfigbottomskip}%
  \fi
  \egroup}

```

\@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.

```
\renewcommand{\@subcaption}[2]{%
  \begingroup
    \let\label\@gobble
    \let\protect\string      % SDC mod
    \if@contmaincaption
      \addcontentsline{\csname ext@#1\endcsname}{#1}%
        {\protect\numberline{\csname p@#1\endcsname\csname the#1\endcsname}%
        {\ignorespaces #2}}%
```

```
        \gdef\@subfigcaptionlist{}
      \else
        \xdef\@subfigcaptionlist{%
        \@subfigcaptionlist,%
%%        {\string\numberline {\@currentlabel}%   % SDC mod
        {\protect\numberline {\@currentlabel}%   % SDC mod
         \noexpand{\ignorespaces #2}}}%
      \fi
    \endgroup
    \@nameuse{@make#1caption}{\@nameuse{@the#1}}{#2}}

```

\subfigure
\subtable

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.

```
  \let\subfigure\subbottom
  \let\subtable\subtop

\fi
}
```

The end of the \AtBeginDocument code for subfigure20.

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

```
\if@contsubfigxx
```

\subbottom
\@contsubbody

\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.

```
  \newcommand{\subbottom}{%
    \@contbotsubtrue
    \@contsubbody}

  \newcommand{\@contsubbody}{%
    \bgroup
    \if@contmaincaption\else
      \advance\csname c@\@captype\endcsname\@ne
    \fi
    \refstepcounter{sub\@captype}\@contkeep%
    \leavevmode
    \@ifnextchar [%
      {\@subfloat{sub\@captype}}
      {\@subfloat{sub\@captype}[\@empty]}}

```

\contsubbottom
\subbody@cont

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.

```
\newcommand{\contsubbottom}{%
  \@contbotsubtrue
  \subbody@cont}

\newcommand{\subbody@cont}{%
  \bgroup
  \@contset
  \refstepcounter{sub\@captype}\@contkeep%
  \leavevmode
  \@ifnextchar [%
    {\@subfloat{sub\@captype}}
    {\@subfloat{sub\@captype}[\@empty]}}

```

\subtop

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

```
\newcommand{\subtop}{%
  \@contbotsubfalse
  \@contsubbody}

```

\contsubtop

The continued version of \subtop.

```
\newcommand{\contsubtop}{%
  \@contbotsubfalse
  \subbody@cont}

```

\@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.

```
\long\def\@contcaption#1#2{%
  \subfigold@contcaption{#1}{#2}%
  \@for \@conttempa:=\@subfigcaptionlist \do {%
    \ifx\@empty\@conttempa\relax \else
      \addcontentsline
        {\@nameuse{ext@sub#1}}%
        {sub#1}%
        {\@conttempa}%
     \fi}%
  \gdef\@subfigcaptionlist{}}

```

\contsubtable
\contsubfigure

Aliases for \contsubtop and \contsubbottom, respectively.

```
\let\contsubtable\contsubtop
\let\contsubfigure\contsubbottom

```

The end of the subfigure20 option code.

```
\fi
```

This is the end of the version 2.0 code.

5.6.2 Option subfigure21

`\caption` `\contcaption` `\@float` `\@dbflt`

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

```
\if@contsubfigxxi
  \let\@contoldc\caption
  \renewcommand{\caption}{\@contmaincaptiontrue\@contoldc}
  \let\@contoldcont\contcaption
  \renewcommand{\contcaption}{\@contmaincaptiontrue\@contoldcont}
  \let\@contoldf\@float
  \renewcommand{\@float}[1]{\@contmaincaptionfalse
                \@ifundefined{c@sub#1}{}{\csname c@sub#1\endcsname = 0\relax}
                \@contoldf{#1}}
  \let\@contoldff\@dbflt
  \renewcommand{\@dbflt}[1]{\@contmaincaptionfalse
                \@ifundefined{c@sub#1}{}{\csname c@sub#1\endcsname = 0\relax}
                \@contoldff{#1}}
\fi
```

`\@contsubfloat` This is a version of the subfigure `\subfigure` command. The revised version stores the subcounter.

```
\newcommand{\@contsubfloat}{%
  \bgroup
  \let\subfig@oldlabel=\label
  \let\label=\sub@label
  \refstepcounter{sub\@captype}\@contkeep%   % <- change here
  \@ifnextchar [%
    {\@@cont@subfloat}%
    {\@@cont@subfloat[\@empty]}}
```

`\@@contsubfloat` This is a revised version of the subfigure `\@subfigure` command (just the called macronames are changed).

```
\def\@@contsubfloat[#1]{%
  \@ifnextchar [%
    {\@@@contsubfloat{sub\@captype}[#1]}%
    {\@@@contsubfloat{sub\@captype}[\@empty #1][#1]}}
```

`\@@@contsubfloat` This is a modified version of the subfigure `\@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 a parameter `#1`.

```
\long\def\@@@contsubfloat#1[#2][#3]#4{%
  \@tempcnta=\@ne
  \ifsf@tight
    \if@minipage
      \@tempcnta=\z@
    \else
      \ifdim\lastskip=\z@
        \@tempcnta=\@ne
      \else
        \@tempcnta=\tw@
      \fi
    \fi
  \fi
  \if@contbotsub
    \def\subfig@top{\subfigtopskip}%
    \def\subfig@bottom{\subfigbottomskip}%
  \else
    \def\subfig@top{\subfigbottomskip}%
    \def\subfig@bottom{\subfigtopskip}%
  \fi
  \setbox\@tempboxa \hbox{#4}%
  \@tempdima=\wd\@tempboxa
  \vtop\bgroup
    \vbox\bgroup
    \ifcase\@tempcnta
      \@minipagefalse
    \or
      \vspace{\subfig@top}
    \or
      \ifdim \lastskip=\z@ \else
        \@tempskipb\subfig@top\relax\@xaddvskip
      \fi
    \fi
    \if@contbotsub
      \box\@tempboxa\egroup
      \ifx \@empty#3\relax \else
        \vskip\subfigcapskip
        \@subcaption{#1}{#2}{#3}%
      \fi
    \else
      \ifx\@empty#3\relax \else
        \@subcaption{#1}{#2}{#3}%
        \vskip\subfigcapskip
        \vskip\subfigcaptopadj
      \fi\egroup
      \box\@tempboxa
```

```
    \fi
    \vspace{\subfig@bottom}
  \egroup
\egroup}
```

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

```
\let\cont@subfig@oldcaption\@caption
```

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`.

```
\newcommand{\doxxi@contcaption}{%
  \long\def\@contcaption##1##2{%
    \if@contbotsub
      \@listsubcaptions{##1}%
      \subfigold@contcaption{##1}{##2}
    \else
      \subfigold@contcaption{##1}{##2}
      \@listsubcaptions{##1}%
    \fi}
}
```

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

```
%%%\if@contsubfigxxi
```

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

```
  \newcommand{\subbottom}{%
    \@contbotsubtrue
    \@contsubbody}

  \newcommand{\@contsubbody}{%
    \bgroup
    \let\subfig@oldlabel=\label
    \let\label=\sub@label
    \if@contmaincaption\else
      \advance\csname c@\@captype\endcsname\@ne
    \fi
    \refstepcounter{sub\@captype}\@contkeep%
    \leavevmode
    \@ifnextchar [%
      {\@@contsubfloat}%
```

```
521       {\@@contsubfloat[\@empty]}}
522
```

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

```
523   \newcommand{\contsubbottom}{%
524     \@contbotsubtrue
525     \subbody@cont}
526
527   \newcommand{\subbody@cont}{%
528     \bgroup
529     \let\subfig@oldlabel=\label
530     \let\label=\sub@label
531     \@contset
532     \refstepcounter{sub\@captype}\@contkeep%
533     \leavevmode
534     \@ifnextchar [%
535       {\@@contsubfloat}%
536       {\@@contsubfloat[\@empty]}}
537
```

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

```
538   \newcommand{\subtop}{%
539     \@contbotsubfalse
540     \@contsubbody}
541
542   \newcommand{\contsubtop}{%
543     \@contbotsubfalse
544     \subbody@cont}
545
```

`\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.

```
546   \newcommand{\contsubfigure}{%
547     \bgroup
548     \let\subfig@oldlabel=\label
549     \let\label=\sub@label
550     \@contset
551     \refstepcounter{sub\@captype}\@contkeep%
552     \@ifnextchar [%
553       {\@@contsubfloat}%
554       {\@@contsubfloat[\@empty]}}
555
```

`\@contsf` `\@contst` These are versions of the `\subfigure` and `\subtable` macros written using the ccaption style.

```
556 \newcommand{\@contsf}{\@contbotsubtrue%
557   \ifsubfiguretopcap\@contbotsubfalse\fi%
```

```
  \@contsubbody}
\newcommand{\@contst}{\@contbotsubtrue%
  \ifsubtabletopcap\@contbotsubfalse\fi%
  \@contsubbody}
```

Now these can be used if appropriate within the `\AtBeginDocument` code. But first call for the new version of `\@contcaption`.

```
\if@contsubfigxxi

  \doxxi@contcaption

  \AtBeginDocument{%
    \let\@subfloat\@@@contsubfloat
    \let\@subfigure\@@contsubfloat
    \let\subfigure\@contsf
    \let\subtable\@contst
    \let\contsubfigure\contsubbottom
    \let\contsubtable\contsubtop
    \long\def\@caption#1[#2]#3{%
      \cont@subfig@oldcaption{#1}[{#2}]{#3}}
}

```

The end of the subfigure21 option code.

```
\fi

```

5.7 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`.
- `\toclevel@fenv` Holding a bookmark level.

Note that the `\fleg...` macros are only required for the ccaption package, and `\toclevel@fenv` is only required if the hyperref package is being used. 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.

```
\newcounter{newflo@tctr}
\@ifundefined{c@figure}{\setcounter{newflo@tctr}{1}}{
  \@ifundefined{c@table}{\setcounter{newflo@tctr}{2}}{
    \setcounter{newflo@tctr}{4}}}

```

\cftdot These macros are also provided by the tocloft package, but we need them in any \cftdotsep event.

\cftdotfill
\@cfttocstart
\@cfttocfinish

```
\providecommand{\cftdot}{.}
\providecommand{\cftdotsep}{4.5}
\providecommand{\cftdotfill}[1]{%
  \leaders\hbox{$\m@th\mkern #1 mu \hbox{\cftdot}\mkern #1 mu$}\hfill}
\providecommand{\@cfttocstart}{%
  \@ifundefined{chapter}{}{%
    \if@twocolumn
      \@restonecoltrue\onecolumn
    \else
      \@restonecolfalse
    \fi}}
\providecommand{\@cfttocfinish}{%
  \@ifundefined{chapter}{}{\if@restonecol\twocolumn\fi}}

```

\newfloatentry `\newfloatentry[⟨within⟩]{⟨counter⟩}{⟨ext⟩}{⟨level-1⟩}` generates the commands for typesetting a caption in a float and a caption in a listing.

```
\newcommand{\newfloatentry}[4][\@empty]{%

```

\c@X Create the new counter. An error if it exists.

\theX

```
  \@ifundefined{c@#2}{%
    \ifx \@empty#1\relax
      \newcounter{#2}
    \else
      \@ifundefined{c@#1}{\PackageWarning{ccaption}%
                           {#1 has no counter for use as a 'within'}
        \newcounter{#2}}%
      {\newcounter{#2}[#1]%
       \expandafter\edef\csname the#2\endcsname{%
         \expandafter\noexpand\csname the#1\endcsname.\noexpand\arabic{#2}}}
    \fi
    \setcounter{#2}{0}
  }
  {\PackageError{ccaption}{#2 has been previously defined}{\@eha}}

```

That finishes off the error checking, rest is defined in any event

`\l@X` `\l@X{title}{page}` typesets the entry in the listing, but only if the `Zdepth` is greater than ⟨*level-1*⟩.

```
\@namedef{l@#2}##1##2{%
  \ifnum \@nameuse{c@#3depth} > #4\relax
  \vskip \@nameuse{cftbefore#2skip}
  {\leftskip \@nameuse{cft#2indent}\relax
   \rightskip \@tocrmarg
   \parfillskip -\rightskip
   \parindent \@nameuse{cft#2indent}\relax\@afterindenttrue
   \interlinepenalty\@M
   \leavevmode
   \@tempdima \@nameuse{cft#2numwidth}\relax
   \expandafter\let\expandafter\@cftbsnum\csname cft#2presnum\endcsname
   \expandafter\let\expandafter\@cftasnum\csname cft#2aftersnum\endcsname
   \expandafter\let\expandafter\@cftasnumb\csname cft#2aftersnumb\endcsname
   \advance\leftskip\@tempdima \null\nobreak\hskip -\leftskip
   {\@nameuse{cft#2font}##1}\nobreak
   \@nameuse{cft#2fillnum}{##2}}
\fi
  }  % end of \l@#2
```

Now define all the layout commands used by `\l@X`. The default values for these correspond to those for figure and table entries.

`\cftbeforeXskip`

```
\expandafter\newlength\csname cftbefore#2skip\endcsname
  \setlength{\@nameuse{cftbefore#2skip}}{\z@ \@plus .2\p@}
```

`\cftXindent` `\cftXnumwidth`

```
\expandafter\newlength\csname cft#2indent\endcsname
\expandafter\newlength\csname cft#2numwidth\endcsname
```

Set the default values for the indent and numwidth depending on the entry's level. A level of 1 (⟨*level-1*⟩ = 0) corresponds to a figure.

```
\ifcase #4\relax  % 0
  \setlength{\@nameuse{cft#2indent}}{1.5em}
  \setlength{\@nameuse{cft#2numwidth}}{2.3em}
\or               % 1
  \setlength{\@nameuse{cft#2indent}}{3.8em}
  \setlength{\@nameuse{cft#2numwidth}}{3.2em}
\or               % 2
  \setlength{\@nameuse{cft#2indent}}{7.0em}
  \setlength{\@nameuse{cft#2numwidth}}{4.1em}
\or               % 3
  \setlength{\@nameuse{cft#2indent}}{10.0em}
  \setlength{\@nameuse{cft#2numwidth}}{5.0em}
\else             % anything else
  \setlength{\@nameuse{cft#2indent}}{1.5em}
  \setlength{\@nameuse{cft#2numwidth}}{2.3em}
```

```
\fi
```

\cftXfont \cftXpresnum \cftXaftersnum \cftXaftersnumb \cftXdotsep \cftXleader \cftXpagefont \cftXafterpnum

And the rest of the commands

```
\@namedef{cft#2font}{\normalfont}
\@namedef{cft#2presnum}{}
\@namedef{cft#2aftersnum}{}
\@namedef{cft#2aftersnumb}{}
\@namedef{cft#2dotsep}{\cftdotsep}
\@namedef{cft#2leader}{\normalfont\cftdotfill{\@nameuse{cft#2dotsep}}}
\@namedef{cft#2pagefont}{\normalfont}
\@namedef{cft#2afterpnum}{}
```

\cftXfillnum

This typesets the leader and the page number.

```
\@namedef{cft#2fillnum}##1{%
  {\@nameuse{cft#2leader}}\nobreak
  \hb@xt@\@pnumwidth{\hfil\@nameuse{cft#2pagefont}##1}%
  \@nameuse{cft#2afterpnum}\par}
```

\toclevel@X

This is required for the hyperref package.

```
\@namedef{toclevel@#2}{#4}
```

The end of `\newfloatentry`

```
} % end \newfloatentry

```

\newfloatlist

`\newfloatlist[`⟨*within*⟩`]{`⟨*fenv*⟩`}{`⟨*ext*⟩`}{`⟨*listname*⟩`}{`⟨*capname*⟩`}` creates the commands for a new float environment ⟨*fenv*⟩ (aka `X`) and a new List of for ⟨*fenv*⟩, using ⟨*ext*⟩ (aka `Z`) as the file extension.

```
\newcommand{\newfloatlist}[5][\@empty]{%
```

Call `\newfloatentry[within]{X}{Z}{0}` to set up for typesetting the entry.

```
  \ifx \@empty#1\relax
    \newfloatentry{#2}{#3}{0}
  \else
    \newfloatentry[#1]{#2}{#3}{0}
  \fi
```

\ftype@X

Define the float type, set it to the float counter, and double the counter afterwards.

```
\@namedef{ftype@#2}{\value{newflo@tctr}}
\addtocounter{newflo@tctr}{\value{newflo@tctr}}
```

\ext@X Zdepth

Define `\ext@X` for the file extension and set the new `Zdepth` depth counter to 1.

```
\@namedef{ext@#2}{#3}  % file extension
\newcounter{#3depth}
\setcounter{#3depth}{1}

```

\cftmarkZ \cftmarkZ specifies the marks for the page headings for the new listing.

```
\@namedef{cftmark#3}{%
  \@mkboth{\MakeUppercase{#4}}{\MakeUppercase{#4}}
```

\listofX \listofX typesets the listing.

```
\if@conttitleopt
```

For the titles option, basically copy the code from the standard \tableofcontents command definition.

```
  \@namedef{listof#2}{%
    \@cfttocstart
    \@ifundefined{chapter}{\section*{#4}}{\chapter*{#4}}
    \@nameuse{cftmark#3}
    \@starttoc{#3}%
    \@cfttocfinish}
\else
```

Otherwise, provide a fully parameterised heading style.

```
  \@namedef{listof#2}{%
    \@cfttocstart
    \par
    \begingroup
      \parindent\z@ \parskip\z@
      \@nameuse{@cftmake#3title}
      \@starttoc{#3}%
    \endgroup
    \@cfttocfinish}
\fi
```

\@cftmakeZtitle \@cftmakeZtitle typeset the title heading for the liusting.

```
  \@namedef{@cftmake#3title}{%
    \@ifundefined{chapter}{%
      \vspace{\@nameuse{cftbefore#3titleskip}}}{%
      \vspace*{\@nameuse{cftbefore#3titleskip}}}
    \interlinepenalty\@M
    {\@nameuse{cft#3titlefont}#4}{\@nameuse{cftafter#3title}}
    \@nameuse{cftmark#3}
    \par\nobreak
    \vskip \@nameuse{cftafter#3titleskip}
    \@afterheading}
```

\cftbeforeZtitleskip \cftafterZtitleskip \cftZtitlefont \cftafterZtitle Define the lengths and commands for controlling the title heading layout. The values depend on whether the document is chaptered or not.

```
  \expandafter\newlength\csname cftbefore#3titleskip\endcsname
  \expandafter\newlength\csname cftafter#3titleskip\endcsname
  \@ifundefined{chapter}{%
     \setlength{\@nameuse{cftbefore#3titleskip}}{3.5ex \@plus 1ex \@minus .2ex}
```

```
718       \setlength{\@nameuse{cftafter#3titleskip}}{2.3ex \@plus .2ex}
719       \@namedef{cft#3titlefont}{\normalfont\Large\bfseries}
720     }{%
721       \setlength{\@nameuse{cftbefore#3titleskip}}{50pt}
722       \setlength{\@nameuse{cftafter#3titleskip}}{40pt}
723       \@namedef{cft#3titlefont}{\normalfont\Huge\bfseries}
724     }
725     \@namedef{cftafter#3title}{}
726
```

`\fps@X` `\fps@X` is the default float placement specification, `\fnum@X` typesets the caption
`\fnum@X` name and number, and `\flegtoc@X` is for named legends.
`\flegtoc`

```
727   \@namedef{fps@#2}{tbp}                      % position
728   \@namedef{fnum@#2}{#5~\@nameuse{the#2}}     % caption naming
729   \@namedef{flegtoc@#2}##1{}                  % named legend
730
```

`X` Finally define the new float environment, in both normal and starred forms.
`X*`

```
731   \newenvironment{#2}{\@float{#2}}{\end@float}
732   \newenvironment{#2*}{\@dblfloat{#2}}{\end@dblfloat}
```

This ends the definition of `\newfloatlist`.

```
733 } % end \newlistof
734
```

`\newfloatenv` Up to version 2.7 of the package the command `\newfloatenv[`⟨*within*⟩`]{`⟨*fenv*⟩`}{`⟨*ext*⟩`}{`⟨*capname*⟩`}` created a new float environment. It was replaced in later versions by `\newfloatlist`. Print a warning message if it is used.

```
735 \newcommand{\newfloatenv}[4][\@empty]{%
736   \PackageError{ccaption}{\protect\newfloatenv\space has been replaced
737                           by\MessageBreak
738                          \protect\newfloatlist}{\@eha}
739 }
740
```

`\listfloats` Up to version 2.7 the `\listfloats{`⟨*fenv*⟩`}{`⟨*heading*⟩`}` command produced a list of floats for ⟨*fenv*⟩. It was replaced in later versions by the generated command `\listoffenv`. Print an error message.

```
741 \newcommand{\listfloats}[2]{%
742   \PackageError{ccaption}{\protect\listfloats{#1}{...} has been
743                           replaced by\MessageBreak
744                           \protect\listof #1}{\@eha}
745 }
746
```

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.
- `\@@thesubfenv` for alternative label reference.
- `\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.
- `\toclevel@subfenv` for hyperref bookmarks

`\newsubfloat` `\newsubfloat{⟨fenv⟩}` creates the commands for a new subfloat for ⟨*fenv*⟩ (aka X).

```
\newcommand{\newsubfloat}[1]{%
```

Call `\newfloatentry[X]{subX}{extX}{1}` to get most of the work done.

```
  \newfloatentry[#1]{sub#1}{\@nameuse{ext@#1}}{1}
```

`\ext@subX` And now for the rest of the commands for subcaptions.

`\thesubX`

`\@thesubX`

`\@@thesubX`

`\p@subX`

`\@makesubXcaption`

```
  \@namedef{ext@sub#1}{\csname ext@#1\endcsname}
  \@namedef{thesub#1}{(\alph{sub#1})}
  \@namedef{@thesub#1}{{\subcaplabelfont\@nameuse{thesub#1}}\space}
  \@namedef{@@thesub#1}{\@nameuse{thesub#1}}
  \@namedef{p@sub#1}{\csname the#1\endcsname}
  \@namedef{@makesub#1caption}{\@makesubfigurecaption}
}

```

`\newfloatpagesoff` `\newfloatpagesoff{⟨fenv⟩}` switches off page numbers in the listing for entries of type ⟨*fenv*⟩. It does this by redefining the `\cftXfillnum` command.

```
\DeclareRobustCommand{\newfloatpagesoff}[1]{
  \@namedef{cft#1fillnum}##1{%
    \parfillskip=\z@ plus1fil\@nameuse{cft#1afterpnum}\par}}

```

`\newfloatpageson` `\newfloatpageson{⟨fenv⟩}` switches on page numbers in the listing for entries of type ⟨*fenv*⟩. It does this by redefining the `\cftXfillnum` command to its default specification.

```
\DeclareRobustCommand{\newfloatpageson}[1]{
  \@namedef{cft#1fillnum}##1{%
    {\@nameuse{cft#1leader}}\nobreak
    \hb@xt@\@pnumwidth{\hfil\@nameuse{cft#1pagefont}##1}%
    \@nameuse{cft#1afterpnum}\par}}

```

`\setnewfloatindents` `\setnewfloatindents{`⟨*fenv*⟩`}{`⟨*indent*⟩`}{`⟨*width*⟩`}` sets the indent and numwidth for the float entry ⟨*fenv*⟩.

```
\newcommand{\setnewfloatindents}[3]{%
  \setlength{\@nameuse{cft#1indent}}{#2}
  \setlength{\@nameuse{cft#1numwidth}}{#3}
}
```

The end of this package.

⟨/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 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.

For checking if an optional argument is present I used code along the lines:

```
\newcommand{\com}[4][\@empty]{...
\ifx \@empty#1\else %argument present
```

Unfortunately I was not consistent, as Benjamin Bayart found[5] when he used an optional argument that started with a double character, like `\bicaption[ccapt3]{...`, which caused nasty things to happen. In these cases I had coded:

```
\ifx #1\@empty\else %argument present
```

I really should have known better as this results in TRUE with `apt3` being left dangling (and typeset).

References

[Coc95] Steven Douglas Cochran. *The subfigure package*. February 2002. (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`)

[NiGa98] Rolf Niepraschk and Hubert Gäßlein. *The sidecap package*. June 1998. (Available from CTAN as file `sidecap.dtx`)

[Rec97] Keith Reckdahl. *Using Imported Graphics in LaTeX2e*. December 1997. (Available from CTAN as file `info/epslatex.ps` or `info/epslatex.pdf`)

[Som95] Harald Axel Sommerfeldt. *The caption package*. October 1995. (Available from CTAN as file `caption2.dtx`)

[Wil96] Peter R. Wilson. *LaTeX for standards: The LaTeX package files user manual*. NIST Report NISTIR, June 1996.

[Wil01] Peter R. Wilson. *The tocloft package*. March 2001. (Available from CTAN as file `tocloft.dtx`)

[5] Email to me on 2005/03/29.

Index

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

Change History

v11

General: Output character table to class and package files only . . . 1

v2.1

General: Added legends 1

v2.2

General: Added fixed captions and new floats 1

v2.2a

General: Improved documentation 1

v2.3

General: Added namedlegend command 1

v2.3a

General: Fixed uppercasing bug. Changed UpperCase to MakeUppercase 1

v2.4

General: Added subfigure v2.0 package support 1

v2.5

General: Added subfigure v2.1 package support 1

Addition of subfigure20 section in code portion 40

v2.6

General: Added bilingual captions 1

\bicaption: New \bicaption command 37

\bicontcaption: New \bicontcaption command 38

\bionenumcaption: New \bionenumcaption command 37

\bitwonumcaption: New \bitwonumcaption command 36