# Quick floats in LaTeX [*]

Joseph Wright [†]

Released 2007/09/20

**Abstract**

The `trivfloat` package provides a quick method for defining new float types in LaTeX. A single command sets up a new float in the same style as the LaTeX kernel `figure` and `table` float types.

## Contents

## 1 Introduction

The LaTeX kernel provides the `figure` and `table` floating environment, but does not provide an easy method for defining new float types. This problem is addressed by the `float` package, which provides an array of commands to create new float types. However, the one command missing is a quick one to set up a new float type with no customisation. This is addressed by the `trivfloat` package.

## 2 Usage

`\trivfloat` The package itself provides only a single command to the user, `\trivfloat`. This takes a single argument `{`⟨*name*⟩`}`, which will be the name of the new floating environment. The new environment can be used as normal; the new floats should behave exactly the same as `figure` and `table` environments. The `trivfloat` package works hard to ensure that the newly-produced floats behave exactly like pre-defined ones. The exact behaviour of the new floats is dependant on the document class used: `trivfloat` knows about the standard classes, `memoir` and KOMA-Script. The `\trivfloat` command should only be used in the preamble of the document; LaTeX will complain if you use it later. New floats can also

[*] This file describes version v1.3b, last revised 2007/09/20.

[†] E-mail: joseph.wright@morningstar2.co.uk

be generated by passing the name of the desired float type to the package as an option. In this way several new floats can be produced in on go. The new floats will respect the normal position modifiers, `t`, `b`, `h`, `p` and (if `memoir` is not in use) `H` for *really* here.

The `trivfloat` package defines a new environment ⟨*float*⟩ for each new float, which should be used in the same way as a figure or table environment. The command `\`⟨*`float`*⟩`name` is provided for the naming of float captions. In analogy to `\figurename` and `\tablename`, the default value is "*Float*." Secondly, the `\listof`⟨*`float`*⟩`s` command is defined. This will produce a list of the new float types, in the same way as `\listoffigures` gives a figure listing. The title text for this is stored in `\list`⟨*`float`*⟩`name`, and defaults to "List of *Float*s." Both can be redefined either by the user or using the `babel` `\addto` method if needed.[1] Notice that the capitalisation of the float name for the naming commands only alters the first letter of the name given.

An example of the use of the package would be to create a new float type for charts, by having `\trivfloat{chart}` in the preamble. In the body of the document, one can then use `\begin{chart}` ... `\end{chart}` to have a floating chart. This will result in `\chartname` having the value "Chart," and `\listchartname` taking the value "List of Charts." A listing of all of the charts in the document would be obtained using `\listofcharts` (needing two LaTeX runs as normal).

## 3 Known issues

Some of the issues known to the package author are quite complicated, and most users will probably never encounter them. They are marked with a "dangerous bend" sign, and should be skipped by inexperienced TeX users.

The name passed to `\trivfloat` should contain only the letters A–Z and a–z. This ensures that there are no strange errors generated by TeX. Correctly handling non-English words is not possible using an automated system, and so after defining a new float type the macros `\`⟨*`float`*⟩`name` and `\list`⟨*`float`*⟩`name` should be corrected to give the desired names. If you *really* need to use other characters, read the "dangerous bend."

The `babel` system provides support for a wide range of language-specific strings. It also makes changes to the typography of documents. The changes made by `babel` can be in almost any area, and so floats created with `trivfloat` may not act like the standard ones once `babel` is loaded. For example, the `french` option for babel alters the internal function used to make float captions. Users should search through the ⟨*`language`*⟩`.ldf` file for the languages they use for "figure" and "lof" to see what changes `babel` makes to the standard floats. The user can then correct the behaviour of the new floats as needed.

The float-forming mechanism *will* accept characters which are not regarded by TeX as "letters" as it uses `\csname` ... `\endcsname` to generate the new float type. However, very odd and hard to locate errors may occur if this is done. With some document classes (*e.g.* `article`) all seems to work, but other classes (*e.g.* `book`) cause TeX to complain. As a result, all characters passed to `\trivfloat` should be have a catcode of

[1]For a recent discussion of controlling `babel` see: *Enjoying babel*, E. Gregorio, *TUGboat*, **2007**, *28*, 247–255.

11. If you attempt to use character that is not a single character (say for example \ss for ß) things will go badly wrong. Stick to letters.

# 4 Implementation

## 4.1 Setup code

The first part of the package is concerned with basic identification and loading support packages.

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{trivfloat}
  [2007/09/20 v1.3b Quick floats in LaTeX]
```

memoir has its own \newfloat mechanism, and so float is used only if memoir is not loaded.

```
\@ifclassloaded{memoir}
  {}
  {\RequirePackage{float}}
```

## 4.2 Internal macros

\tfl@floatcount A new counter is needed to track how many floats have been generated by trivfloat.

```
\newcount\tfl@floatcount%
```

\tfl@genext \tfl@ext Each float type needs a list to store entries when generating a contents listing. Rather than try complex things with the initial characters of the float name, an arbitrary file extension is used. The extensions lof, log and lot are taken by default, so we don't use them. The chemscheme package uses los, so this is left alone as well. Also notice that if people try to make too many new float types, then extension xxx will be used. This is very unlikely, but odd things might happen so we warn them.

```
\def\tfl@genext{%
  \def\tfl@ext{%
    \ifcase\tfl@floatcount%
```

Zero is not a possibility for \tfl@floatcount is not a possibility, so this is skipped over.

```
    \or % 1
      loa%
    \or % 2
      lob%
    \or % 3
      loc%
    \or % 4
      lod%
    \or % 5
      loe%
    \or % 6
      loh%
    \or % 7
      loi%
    \or % 8
      loj%
    \or % 9
      lok%
    \or % 10
      lol%
    \or % 11
      lom%
    \or % 12
      lon%
    \or % 13
      loo%
    \or % 15
      loq%
    \or % 16
      lor%
    \or % 17
      lou%
    \or % 18
      lov%
    \or % 19
      low%
    \or % 20
      lox%
    \or % 21
      loy%
    \or % 22
      loz%
    \else % 23 or more
      \PackageWarning{trivfloat}%
        {I've run out of extensions \MessageBreak%
         I'm using \jobname.xxx to list all future floats}%
      xxx%
    \fi%
  }%
}%
```

`\tfl@uppercase` To get the correct kerning, a customised version of `\MakeUppercase` is needed. Normally you are not supposed to use `\reserved@a` and `\reserved@b`, but the code here is a very minor alteration of the LaTeX kernel code, so these macros are retained. The use of `\DeclareRobustCommand` here is essential.

```
\DeclareRobustCommand\tfl@uppercase[1]{%
  {%
    \def\i{I}\def\j{J}%
    \def\reserved@a##1##2{\let##1##2\reserved@a}%
    \expandafter\reserved@a\@uclclist\reserved@b{\reserved@b\@gobble}%
    \protected@edef\reserved@a{\uppercase{#1}}%
```

The change from the kernel version comes here.

```
  \expandafter}\reserved@a
}%
```

`\tfl@upperfirst` As suggested by Ulrich Diez, this is an improved method for making case changes than the previous version (v1.2). Notice that it does mean that the user is responsible for the case of everything except the first letter.

```
\def\tfl@upperfirst#1{%
  \protect\@gobble\noexpand\tfl@uppercase#1%
}
```

`\tfl@chapter` `\tfl@orig@chapter` `\tfl@chapter@hook`

A method to add to the `\@chapter` macro is needed. This is not compatible with `\g@addto@macro` directly, so a more complex method is used by providing a suitable hook.

```
\ifx\chapter\@undefined\else
  \def\tfl@chapter@hook{\relax}
  \let\tfl@orig@chapter\@chapter
  \def\tfl@chapter{\tfl@chapter@hook\tfl@orig@chapter}
  \let\@chapter\tfl@chapter
\fi
```

## 4.3 User space macros

`\trivfloat`

The only user-facing macro we need is `\trivfloat`. This does the work of setting up the new float type. Thanks to Heiko Oberdiek for the method to expand arguments correctly: needed so that `\tfl@ext` has the correct value.

```
\newcommand*{\trivfloat}[1]{%
  \AtBeginDocument{%
    \advance\tfl@floatcount\@ne%
    \tfl@genext%
    \PackageInfo{trivfloat}%
      {Listing all ``#1'' floats in \jobname.\tfl@ext}%
```

The first step for forming a new float type is to create the `\`⟨*float*⟩`name` and `\list`⟨*float*⟩`name` macros.

```
    \expandafter\newcommand\expandafter*\expandafter%
      {\csname #1name\endcsname}
      {\noexpand\tfl@upperfirst#1}
    \expandafter\newcommand\expandafter*\expandafter%
      {\csname list#1name\endcsname}
        {List of \noexpand\tfl@upperfirst#1s}
    \begingroup%
    \edef\x{\endgroup%
```

The `memoir`-dependent code now occurs. The case when `memoir` is in use is handled first.

```
      \@ifclassloaded{memoir}
        {\noexpand\newfloat[chapter]{#1}{\tfl@ext}
          {\csname #1name\endcsname}
        \noexpand\newlistof{listof#1s}{\tfl@ext}
          {\csname list#1name\endcsname}
```

The following deals with the vertical space in list of float tables that occurs between each chapter.

```
        \noexpand\addtodef{\noexpand\insertchapterspace}{}%
          {\noexpand\addtocontents%
            {\tfl@ext}
            {\noexpand\protect\noexpand\addvspace{10pt}}}
        \noexpand\newlistentry[chapter]{#1}
          {\tfl@ext}{0}}
```

The creation of new floats needs to know if the document class is using chapters, when `memoir` is not in use.

```
      {\ifx\chapter\@undefined
        \noexpand\newfloat{#1}{tbp}{\tfl@ext}
      \else
        \noexpand\newfloat{#1}{tbp}{\tfl@ext}[chapter]
      \fi
```

The naming of floats for captions and generation of a list of floats is now handled.

```
      \noexpand\floatname{#1}{\csname #1name\endcsname}%
      \noexpand\newcommand{\csname listof#1s\endcsname}%
        {\noexpand\listof{#1}{\csname list#1name\endcsname}}%
      \ifx\chapter\@undefined\else
        \noexpand\g@addto@macro{\noexpand\tfl@chapter@hook}
          {\noexpand\addtocontents%
            {\tfl@ext}
            {\noexpand\protect\noexpand\addvspace{10pt}}}
      \fi}
    }%
  \x%
```

Except for KOMA-Script, all of the supported document classes need some further hacks to get behaviour equivalent to the normal float types. None of these need to have access to the `\tfl@ext`, so are safe outside the `\edef` above. `memoir` does various low-level things, which are reproduced first. Here, `\@ifclassloaded` can be used as `\iftfl@memoir` leads to `\if` nesting errors.

```
  \@ifclassloaded{memoir}
    {\expandafter\renewcommand\expandafter{\csname the#1\endcsname}%
      {\thechapter.\expandafter\@arabic\csname c@#1\endcsname}
    \addtodef{\@smemfront}{}{\counterwithout{#1}{chapter}}
    \addtodef{\@smemmain}{}{%
      \ifartopt\else
        \counterwithin{#1}{chapter}
      \fi}
    \addtodef{\backmatter}{}{%
      \ifartopt\else
        \counterwithout{#1}{chapter}
        \setcounter{#1}{0}
      \fi}
    \ifartopt
      \counterwithout{#1}{chapter}
    \fi
    \cftsetindents{#1}{0em}{2.3em}}
```

The standard document classes do not print the chapter number in captions if it is not greater than zero.

```
  {\ifx\chapter\@undefined\else
    \ifx\KOMAScriptVersion\@undefined
      \expandafter\renewcommand\expandafter{\csname the#1\endcsname}%
        {\ifnum\c@chapter>\z@ \thechapter.\fi
          \expandafter\@arabic\csname c@#1\endcsname}
  \fi\fi}
 }
}%
```

The necessary patching can only be done in the preamble, so `\trivfloat` is banned elsewhere.

```
\@onlypreamble\trivfloat
```

### 4.4 Package options

Any package options given are assumed to be new float types. Every option is therefore passed to `\trivfloat`.

```
\DeclareOption*{\expandafter\trivfloat\expandafter{\CurrentOption}}
\ProcessOptions
```

## 5 Change History

v1.0
- General: Initial public release . . . . . 1

v1.1
- General: Floats can be generated as package options . . . . . . . . . . . . 1

v1.2
- General: Added `memoir` support . . 1
  - Improved emulation of default floats . . . . . . . . . . . . . . . . . . . . 1
- `\tfl@genext`: Removed `los` from list of possible extensions . . . . . 3
- `\tfl@upperfirst`: Made names change case correctly . . . . . . . . . 4
- `\trivfloat`: Added `memoir`-specific code . . . . . . . . . . . . . . . 5
  - Added chapter detection . . . . . . 5
  - Create "name" commands rather than using text directly . . . . . . . 5
  - Only in preamble . . . . . . . . . . . . 6
  - Removed auxiliary macro `\tfl@list` . . . . . . . . . . . . . . . 6

v1.3
- `\tfl@upperfirst`: New macro . . 4
  - Replaces `\tfl@genname` and `\tfl@gennamea` . . . . . . . . . . . 4
- `\trivfloat`: Removed redundant numeral-based code . . . . . . . . . 5
  - Use `\newcommand*` for "name" commands . . . . . . . . . . . . . . . . 5

v1.3a
- `\tfl@uppercase`: New macro . . . 4
- `\tfl@upperfirst`: Uses modified version of `\MakeUppercase` . . 4

v1.3b
- General: Fixed error with `hyperref` . . . . . . . . . . . . . . . . 1
  - Licence changed from GPL to LPPL . . . . . . . . . . . . . . . . . . . . . 1

## 6 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.