The currfile Package

Martin Scharrer
martin@scharrer-online.de

http://www.ctan.org/pkg/currfile

Version v0.5 – 2011/09/18

Abstract

This small package provides the file name and path information of the current input file as LaTeX macros. It properly supports file name with multiple dots and the `\input@path` feature used by some packages like `import`.

1 Usage

`\currfiledir`
`\currfilebase`
`\currfileext`
`\currfilename`
`\currfilepath`

The directory, base (name without extension), extension (without dot), name (=base+'.'+ext) and path (=dir+name) of the current file are provided by these macros. This means that the macros returns the file information of the file they are used in. All macros are fully expanded, i.e. only hold text and not further macros. They are also "sanitized" to ensure that all characters, especially special ones like '_', are taken verbatim. However this special characters might not be displayed correctly in all fonts. A good font is texttype (`\ttfamily`, `\texttt{...}`), but other fonts can be used using the `url` package, e.g.: `\urlstyle{rm}\expandafter\nolinkurl\expandafter{\currfilename}`.

Special care is taken to keep the file information of `\included` files till the final `\clearpage` command, so that page header and footer of the last page will hold the correct data.

Since v0.2 all files are are taken into account, i.e. files read using `\input`, `\include`, `\InputIfFileExists`, `\usepackage`, `\RequirePackage` and even `\LoadClass` and similar macros. Before v0.2 only `\input` or `\include` and the main file were taken into account.

This package uses the `filehook` package written by the same author. See there for possible incompatibilities with classes or other packages.

More detailed information can be found in the implementation section 4 if required.

```
\ifcurrfiledir{⟨text⟩}{⟨true⟩}{⟨false⟩}
\ifcurrfilebase{⟨text⟩}{⟨true⟩}{⟨false⟩}
\ifcurrfileext{⟨text⟩}{⟨true⟩}{⟨false⟩}
\ifcurrfilename{⟨text⟩}{⟨true⟩}{⟨false⟩}
\ifcurrfilepath{⟨text⟩}{⟨true⟩}{⟨false⟩}
```

New in v0.4 from 2011/01/09

This if-macros allow the comparison of ⟨*text*⟩ with the current file directory, base, extension, name and path, respectively. The ⟨*text*⟩ is fully expanded and sanitized for the comparison. Example: `\ifcurrfileext{cfg}{I'm in a config file}{No config file!}`

```
\ifcurrfile{⟨currfile macro or text⟩}{⟨text⟩}{⟨true⟩}{⟨false⟩}
```

Compares the given ⟨*currfile macro or text*⟩ with ⟨*text*⟩. Both are taken as file name parts and are fully expanded and sanitized before the comparison. This general macro is a little slower then the specialised macros above but might be useful to compare different file names/paths where non of the two is the current file. Note that the all comparisons are done insensitive to the catcodes of the texts, which is what users want. Different comparision macros (`\ifx`, `ifthenelse`) might not do this.

Package Options

The package provides two options mainext and maindir which can be used to provide the extension and directory of the main file. This is required if the above macros should be used for the main file itself and if this does has a file extension other than '`.tex`' (e.g. a `.dtx` file) or is not located in the current directory. To provide support for the macros defined by the `fink` package (see section 3) a `fink` option exists.

2 Usage inside file hooks

This package uses the 'EveryFile' hooks of the `filehook` package to update its macros. Special care is taken to do this in a way so that the macros can be used safely inside other hook code, including other 'EveryFile' hooks. Please note that the 'AtEndOfPackageFile' and 'AtEndOfClassFile' hooks are executed after 'AtEndOfEveryFile' and therefore the `currfile` macros will hold the values of the parent file, not of that package or class file.

3 Compatibility with the `fink` package

The `fink` package (*fi*le *n*ame *k*eeper) provides a similar functionality. It has inspired this package in several points (e.g. package options). However, it does not exclude package and other preamble files and does not take care to change the filename *after* the `\clearpage` of `\include`. The author of `fink` is now discontinuing it in favour of this package. Existing documents which use `fink` should either rename the related macros as shown by Table 1 or use the `fink` option of `currfile` which defines the `fink` macros to use the `currfile` ones.

Because both packages do basically the same thing, especially patch the same macros, there are incompatible and should not be loaded at the same time. In consent with the `fink` package author this package will undo most of the `fink` code if it was already loaded or prevent it from being loaded afterwards.

Table 1: Conversion from `fink` package to `currfile`.

fink	currfile	Example Result
\finkdir	\currfiledir	
\finkbase	\currfilebase	currfile
\finkext	\currfileext	dtx
\finkfile	\currfilename	currfile.dtx
\finkpath	\currfilepath	currfile.dtx

4 Implementation

4.1 Package header

```
\NeedsTeXFormat{LaTeX2e}[1999/12/01]
\ProvidesPackage{currfile}[%
%<!DATE>
%<!VERSION>
%<*DRIVER>
    2099/01/01 develop
%</DRIVER>
    ydoc package to describe macros, environments, ⟋
       options etc.]
```

4.2 Options

```
\RequirePackage{kvoptions}
\SetupKeyvalOptions{family=currfile,prefix=currfile@}

\@ifpackageloaded{fink}{%
    \DeclareStringOption[\fnk@mainext]{mainext}%
    \DeclareStringOption[\fnk@maindir]{maindir}%
    \DeclareBoolOption[true]{fink}%
    \PackageWarning{currfile}{Deprecated package '⟋
       fink' detected. %
      The 'fink' option will default to 'true'.^^J%
      If set to 'false' no 'fink' macros will be ⟋
         changed but they will stop
      working correctly!}%
}{%
    \DeclareStringOption[tex]{mainext}%
    \DeclareStringOption[\@currdir]{maindir}%
    \DeclareBoolOption[false]{fink}%
}%
\DeclareVoidOption{force}{\PassOptionsToPackage{force⟋
   }{filehook}}
\RequirePackage{filehook}[2011/01/09]
\ProcessKeyvalOptions*\relax

```

```
\begingroup
\xdef\currfile@mainext{\currfile@mainext}%
\xdef\currfile@maindir{\currfile@maindir}%
\def\@tempa{./}%
\ifx\@tempa\currfile@maindir
    \global\let\currfile@maindir\empty
\fi
\endgroup
```

4.3 File Hooks

The `filehook` package is used to execute the macros at the correct places. However it must be loaded before the option processed because the `fink` compatibility code in `filehook-fink` will modify the option list. The internal interface, not the user-interface, is used to make sure that the file names are valid for all other hooks.

```
\filehook@prefixwarg\filehook@every@atbegin{%
  \currfile@push
  \currfile@set{#1}%
}
\filehook@appendwarg\filehook@every@atend{%
  \currfile@pop
}
```

4.4 Set Current Values

`\currfile@set`

Sets the file information which are parsed by LaTeX's `\filename@parse`.

```
\def\currfile@set#1{%
  \begingroup
    \edef\@filef@und{#1}%
    \ifx\input@path\@undefined\else
        \currfile@checkpath
    \fi
    \@onelevel@sanitize\@filef@und
    \let\filename@simple\currfile@parseext
    \let\filename@base\@gobble
    \expandafter\filename@parse\expandafter{\
      @filef@und}%
    \global\let\currfiledir\filename@area
    \global\let\currfilebase\filename@base
    \global\let\currfileext\filename@ext
    \xdef\currfilename{\currfilebase\ifx\currfileext\
      @empty\else.\currfileext\fi}%
    \xdef\currfilepath{\currfiledir\currfilename}%
  \endgroup
```

```
%<debug> \expandafter\gdef\expandafter\dindent\
   expandafter{\dindent\space}%
%<debug> \message{^^JDEBUG: \dindent\empty Entering
   file '\currfilename' ^^J }%
}
```

\currfile@checkpath

This loop is placed in an own macro for efficiency reasons. In the majority of cases it should not be needed and having it as a macro avoids the need to skip over this code as part of a conditional clause for every read file.

```
\def\currfile@checkpath{%
    \openin\@inputcheck\@filef@und\relax
    \ifeof\@inputcheck
        \expandafter\@tfor
        \expandafter\@tempb
        \expandafter:\expandafter=\input@path\do{%
            \openin\@inputcheck\@tempb\@filef@und\
               relax
            \ifeof\@inputcheck\else
                \edef\@filef@und{\@tempb\@filef@und}%
                \@break@tfor
            \fi
        }%
    \fi
    \closein\@inputcheck
}
```

\currfile@parseext

Replacement for `\filename@simple` to allow multiple dots in a filename. This needs `\let\filename@base\@gobble` before it is called.

```
\begingroup
\@makeother{.}
\gdef\currfile@parseext#1.#2\\{%
    \ifx\\#2\\%
        \ifx\filename@base\@gobble
            \def\filename@base{#1}%
            \let\filename@ext\currfile@defaultext%
        \else
            \def\filename@ext{#1}%
        \fi
    \else
        \edef\filename@base{\filename@base.#1}%
        \def\@tempa{\currfile@parseext#2\\}%
        \expandafter\@tempa
    \fi
```

```
}
\endgroup
```

`\currfile@defaultext`

Holds the default extension 'tex' with catcode *other* like `\jobname`.

```
\def\currfile@defaultext{tex}
\@onelevel@sanitize\currfile@defaultext
```

4.5 File Stack

The file information are pushed and popped on a stack to save and restore them when entering and leaving a sub-file, respectively. This is quite similar to the way LaTeX saves file base names and extension as well as the '@' status (letter or other) for package and class files.

`\currfile@push`

```
\def\currfile@push{%
  \xdef\currfile@stack{%
    {\currfiledir}%
    {\currfilebase}%
    {\currfileext}%
    \currfile@stack
  }%
}
```

`\currfile@pop`

```
\def\currfile@pop{%
%<debug> \message{^^JDEBUG: \dindent\empty Leaving ⟋
    file '\currfilename' ^^J }%
  \ifx\currfile@stack\empty
    \PackageWarning{currfile}{File stack underflow!}%
    \global\let\currfile@stack\currfile@stackinit
  \fi
  \expandafter\currfile@pop@\currfile@stack\relax
  \relax\relax\relax
%<debug> \message{^^JDEBUG: \dindent\empty Restoring ⟋
    file '\currfilename' ^^J }%
}
```

`\currfile@pop@`

```
\def\currfile@pop@#1#2#3#4\relax{%
  \gdef\currfiledir{#1}%
  \gdef\currfilebase{#2}%
  \gdef\currfileext{#3}%
  \xdef\currfilename{\currfilebase\ifx\currfileext\empty\else.\currfileext\fi}%
  \xdef\currfilepath{\currfiledir\currfilename}%
  \gdef\currfile@stack{#4}%
%<debug> \expandafter\expandafter\expandafter\gdef
%<debug> \expandafter\expandafter\expandafter\dindent
%<debug> \expandafter\expandafter\expandafter{\expandafter\@gobble\dindent}%
}
```

`\currfile@stack`

`\currfile@stackinit`

Place `\jobname` values on stack and use this as init value.

```
\def\currfile@stack{}
\currfile@set{\currfile@maindir\jobname.\currfile@mainext}
\currfile@push
\let\currfile@stackinit\currfile@stack
```

4.6 If Macros

`\ifcurrfilename`

```
\newcommand*\ifcurrfilename{\begingroup\currfile@if\currfilename}
```

`\ifcurrfilebase`

```
\newcommand*\ifcurrfilebase{\begingroup\currfile@if\currfilebase}
```

\ifcurrfileext

```
\newcommand*\ifcurrfileext{\begingroup\currfile@if\currfileext}
```

\ifcurrfiledir

```
\newcommand*\ifcurrfiledir{\begingroup\currfile@if\currfiledir}
```

\ifcurrfilepath

```
\newcommand*\ifcurrfilepath{\begingroup\currfile@if\currfilepath}
```

\ifcurrfile

#1: currfile macro or text

Expands and sanitizes the first argument and then calls the internal if-macro with the result.

```
\newcommand*\ifcurrfile[1]{%
    \begingroup
    \edef\@tempb{#1}%
    \@onelevel@sanitize\@tempb
    \currfile@if\@tempb
}
```

\currfile@if

#1: currfile macro to compare

#2: compare text

Expands the text and sanitize it to ensure correct neutral catcodes. Then it temp macro is compared to the given currfile macro.

```
\def\currfile@if#1#2{%
    \edef\@tempa{#2}%
    \@onelevel@sanitize\@tempa
    \ifx\@tempa#1%
        \endgroup
        \expandafter\@firstoftwo
    \else
        \endgroup
        \expandafter\@secondoftwo
    \fi
}
```

4.7 Fink Macros

The `fink` option defines all `fink` package macros to use the ones provided by this package. If the `fink` package was loaded beforehand the restoration of these macros must be avoided at the end of this file (`finks` `\InputIfFileExists` was then used to load this package). If the package was not loaded its version is set to a dummy value and its options to this package options. If `fink` is attempted to be loaded later it will trigger an package option clash if different option are used. Otherwise it will be taken as already loaded and not loaded "again".

```
\ifcurrfile@fink
    \def\finkfile{\currfilename}%
    \def\finkdir{\currfiledir}%
    \def\finkpath{\currfilepath}%
    \def\finkbase{\currfilebase}%
    \def\finkext{\currfileext}%
    \@ifpackageloaded{fink}{%
        \def\fink@restore#1{}%
    }{%
        \@namedef{ver@fink.sty}{2011/01/09}%
        \expandafter\edef\csname opt@fink.sty\endcsname{%
         maindir=\currfile@maindir,mainext=\currfile@mainext
        }%
    }%
\else
    \@ifpackageloaded{fink}{}{%
        \AtBeginOfPackageFile{fink}{%
            \PackageError{currfile}{The 'fink' package is now deprecated. %
             Load 'currfile' with the 'fink' option or see the upgrade guide in the manual}{}%
        }%
    }%
\fi
```