# The currfile Package

Martin Scharrer
martin.scharrer@web.de

Version v1.0 – 2024/03/14

CTAN: https://www.ctan.org/pkg/currfile

VC: https://github.com/MartinScharrer/currfile

**Abstract**

This small package provides the file name and path information of the current input file as LaTeX macros. It properly supports file names with multiple dots and the `\input@path` feature used by some packages like `import`. Optionally also the absolute location of current input file can be provided if a special compiler option is enabled.

## Contents

## 1 Direct filename support in LaTeX 2020/10/01

With the LaTeX release 2020/10/01 the core now provides own filehooks and access to the current filename and path. This package still uses the authors `filehook` package to collect the filenames but there the core implementation is now used internally. Due to this users could see changes in the output in some special cases.

For new documents users might want to use the new core macros `\CurrentFile` and `\CurrentFilePath` discribed in `ltfilehook-doc` directly and use this package only if the advanced features – like parent or absolute file names - are required. Note that `\CurrentFile` is empty for the main document which filename can be accessed usually over `\jobname`.

## 2 Important notice

This package relies heavily on the package `filehook` by the same author and installs at-begin and at-end hooks for all files. The provided macros for the file names *will only work for files loaded* after *the package*. This means only for files which are getting opened afterwards and where not yet opened, as the at-begin hook is required to run. The macros will therefore not work correctly in a package which loaded `currfile`, but for all further loaded packages.

The options abspath and realmainfile may be used to improve the behavior in such cases.

However, it is simply recommended to load `currfile` as early as possible. It can even be loaded before the class using `\RequirePackage{currfile}`.

## 3 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}`. Note that the directory separator is always ‘/’ even under MS Windows.

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.

If required more detailed information can be found in the implementation section (compile this manual with `\AlsoImplementation`).

```
currfiledepth
```

New in v0.7 from 2012/05/15

This LaTeX counter provides the nesting depth of the current input file. For the main file name it has a value of 0. Inside a sub-file of the main file it has a value of 1, while in a sub-file to that file it has a value of 2, etc. Like all LaTeX counter it can be typeset using `\arabic{currfiledepth}` and its numeric value can be accessed using `\value{currfiledepth}`.

```
\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.

## 3.1 Package Options

The package provides the following options:

The string options mainext and maindir can be used to provide the extension (without the dot) 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. The mainext is by default '`tex`' and the maindir is the empty string. See also the related `realmainfile` option.

To provide support for the macros defined by the `fink` package (see section 8) a boolean fink option exists. If the `fink` package is loaded before `currfile` this option is automatically enabled to provide compatibility. The mainext and maindir options are then automatically set to the values of the identical options of `fink`. Note that `fink` is now officially declared deprecated by its author in favour of `currfile`.

New in v0.6 from 2012/05/06

Using the boolean abspath option the support for absolute directories and paths is enabled. This loads the sub-package `currfile-abspath`. This requires the compiler option '`-recorder`' to be used. See section 4 for more details. This option will also enabled the related option realmainfile if it isn't explicitly used.

New in v0.6 from 2012/05/06

If the boolean realmainfile option is enabled the real main file path is determined using the sub-package `currfile-abspath`. This option is very useful if the jobname is set to something else then the main file base name, e.g. using the '`-jobname`' compiler option. Note that this option requires the correct file extension being set using mainext (without the dot). If the main file has the default extension '`.tex`' then no extra efford is required. For '`.dtx`' files `mainext=dtx` must also be used, otherwise the correct file name can't be determined. This option requires the compiler option '`-recorder`' to be used in order to work correctly. See section 4 for more details.

New in v0.7 from 2012/05/15

Using the boolean parent option additional macros are set for the parent file of the current file. See section 5 for more details.

# 4 Absolute paths

New in v0.6 from 2012/05/06

If required also the absolute path and directory of the current input file can be provided. This feature is implemented by the sub-package `currfile-abspath` which is automatically loaded by the abspath option. The absolute path information are read from the '`\jobname.fls`' file produced by the '`-recorder`' compiler option (i.e. compile with '`pdflatex -recorder file.tex`'). Without this compiler option a warning is produced and the related macros stay empty. Note that TeX Live under Linux and Windows is able to read the currently produced '`.fls`' file while with MiKTeX the file of the last run is still active. The means that the information is not available with MiKTeX in the very first compiler run and every time the '`.fls`' file is deleted. Also, changes in the file structure are taking two compiler runs in order to be noticed. With TeX Live the '`.fls`' file and its information is always current.

## 4.1 Additional `currfile` macros for absolute dir and path

Using the abspath package option of `currfile` the following macros with absolute directories and paths are enabled.

```
\currfileabsdir
\currfileabspath
```

Hold the absolute directory and path (directory plus filename) of the current input file, respectively. Both macros will always be empty if the '`.fls`' file is not available.

```
\ifcurrfileabsdir{⟨text⟩}{⟨true⟩}{⟨false⟩}
\ifcurrfileabspath{⟨text⟩}{⟨true⟩}{⟨false⟩}
```

These if-macros allow the comparison of ⟨*text*⟩ with the current absolute file directory and path, respectively. The ⟨*text*⟩ is fully expanded and sanitized for the comparison.

## 4.2 Stand-alone usage of `currfile-abspath`

The sub-package `currfile-abspath` can also be used on its own and provides the following lower-level macros:

```
\getpwd
\thepwd
```

The parent working directory (PWD) is read by `\getpwd` from the '`.fls`' file and stored in `\thepwd`. All characters of the directory will have catcode 12 (other) except spaces which still have catcode 10 (space). If the PWD could not be determined, because there was not '`.fls`' file, then '`\thepwd`' will be empty. Note that

`\getmainfile` and `\getabspath` both also set `\thepwd` while reading the '`.fls`' file.

```
\getmainfile
\themainfile
```

With `\getmainfile` the name of the main file is read from the '`.fls`' file and stored in `\themainfile`. All characters of the file name will have catcode 12 (other) except spaces which still have catcode 10 (space). If the main file name could not be determined, because there was not '`.fls`' file, then '`\themainfile` ' will be empty.

Here it is assumed that the first read file with the '`.tex`' extension is the main file. There is currently no user interface for this sub-package to change this extension (to e.g. '`.dtx`'). However, this can be done using the mainext option of the main package `currfile`.

```
\getabspath{⟨file name⟩}
\getabspath{⟨dir/file name⟩}
\theabspath
\theabsdir
```

With `\getabspath{`*⟨file name⟩*`}` the absolute path of the given file is read from the '`.fls`' file and stored in `\theabspath`. The directory component (path without file name) is stored in `\theabsdir`. The directory always ends in a '`/`' (even under MS Windows), so that a file name can be appended directly. All characters of both macros will have catcode 12 (other) except spaces which still have catcode 10 (space). Note that *⟨file name⟩* must include the full extension but can include a directory component (e.g. '`subdir/file.tex`'). If a directory component is given it will not be part of `\theabsdir`. For example `\getabspath{subdir/file.tex}` will return `\theabspath` as '`/absolute/path/subdir/file.tex`' and `\theabsdir` as '`/absolute/path/`'. This differs from `\currfileabsdir` which will always includes all directory elements, which is done by appending `\currfiledir` after `\theabsdir`.

The given file must be directly accessible by TeX, i.e. from the current directory or using the `TEXMF` tree, otherwise it the absolute location can not be determined. If the file can't be found in the '`.fls`' file the macros `\theabspath` and `\theabsdir` are empty.

## 5 Parent file macros

New in v0.7 from 2012/05/15

If the parent package option is enabled the following macros are provided. They are no direct `\if...` macros for these but the general `\ifcurrfile` macro can be used with them.

```
\parentfiledir
\parentfilebase
\parentfileext
\parentfilename
\parentfilepath
```

These macros are analog to the `\currfile...` macros but for the parent file of the current file. They are all empty for the main file which does not have a parent file.

```
\parentfileabsdir
\parentfileabspath
```

These macros are provided if the abspath option is used in addition to parent. They are also empty for the main file.

## 6 Access all parent files

New in v0.7 from 2012/05/15

If the parents package option is enabled the following macros are provided. Note that this option does not enable the parent option.

```
\currfilegetparents
```

This macro locally defines file name macros for all parent files of the current input file: The macros are named like the ones of the parent option, but also include a trailing roman number to indicate the parent file level, starting with ‘i‘ for the first parent file:

```
\parentfilediri       \parentfiledirii       ...
\parentfilebasei      \parentfilebaseii      ...
\parentfileexti       \parentfileextii       ...
\parentfilenamei      \parentfilenameii      ...
\parentfilepathi      \parentfilepathii      ...
\parentfileabsdiri    \parentfileabsdirii    ...
\parentfileabspathi   \parentfileabspathii   ...
```

Note that these macros are only defined in the current group and not globally like all other file name macros. The number of the highest parent file is given by the `currfiledepth` counter.

For example, if `\currfilegetparents` is used in a sub-sub-sub-file the `currfiledepth` counter is 3 and `\parentfilenameiii` is the name of the main document file.

## 7 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.

## 8 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` |