
Devoir Documentation

Release 1.0.0

Louis Paternault

Aug 18, 2023

CONTENTS

1	Rationale	3
2	Download and install	5
3	Usage	7
3.1	Positional Arguments	7
3.2	Named Arguments	7
4	Configuration	9
4.1	Configuration file	9
4.2	String formatting	9
4.3	Templates	10
5	Indices and tables	11

`devoir` is a tool aimed at quickly setting up a working environment to edit a file.

RATIONALE

When editing a LaTeX file, I want the file being edited with `vim`, the compiled file displayed using a pdf viewer, and latex being run whenever something changes, using `latexmk`. But wait, there is more.

- I often start a LaTeX document by copying an existing one, as a template.
- The pdf file may or may not exist when I start working: if I have already been working on this file before, the pdf file exists; if not, it does not exist, and my pdf viewer won't start on a non-existing file.

This program aims to automate all this process. I built it to process LaTeX files, but it should work with other files too.

DOWNLOAD AND INSTALL

See the [main project page](#) for instructions, and [changelog](#).

USAGE

Prepare working environment to edit a file.

```
usage: devoir [-h] [--version] [-n] [template] file
```

3.1 Positional Arguments

template	Template to use: if set, this file is copied as FILE before editing it
file	File to edit.

3.2 Named Arguments

--version	show program's version number and exit
-n, --dry-run	Don't actually run any command; just print them. Default: False

CONFIGURATION

4.1 Configuration file

Configuration files are placed in directory `~/.devoir/ftplugins`. When calling `devoir`, the config file corresponding to the extension of the file argument is loaded: for instance, calling `devoir foo.tex` would load configuration file `~/.devoir/ftplugins/tex.cfg`.

Here is an example.

```
[config]

cwd = {dirname}

[process]

pre = test -e {basename}.pdf || cp {configdir}/templates/pdf {basename}.pdf
cmd1 = evince {basename}.pdf &
cmd2 = screen $EDITOR {basename}.tex
cmd3 = screen latexmk -pvc {basename}
```

The following options are available:

- Section `config`
 - `cwd`: commands are called from this directory.
- Section `process`

The values of keys are commands to be run, in a shell. Keys are meaningless: you can use them to label your commands.

4.2 String formatting

All values of the configuration files are formatted with the following dictionary:

- `basename`: base name of the edited file (that is, file without its directory).
- `dirname`: absolute directory name of the edited file.
- `filename`: filename, as passed to `devoir`.
- `configdir`: path of the configuration file used.

4.3 Templates

When editing a file (e.g. `devoir foo.tex`), a file `foo.tex` is created before being edited:

- if a `template` argument was provided, it is used;
- otherwise, if a `~/.devoir/templates/EXTENSION` file exists, it is used (e.g. `~/.devoir/templates/tex`);
- otherwise, create an empty file.

Note that this is compatible with *vim* in two ways:

- When editing an empty file, vim still loads the corresponding (vim) template if necessary.
- As templates are identified by their extension, having `~/.devoir/templates` be a symbolic link to `~/.vim/templates/` should work in many cases.

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`