

---

# Flycheck

*Release 27*

October 06, 2016

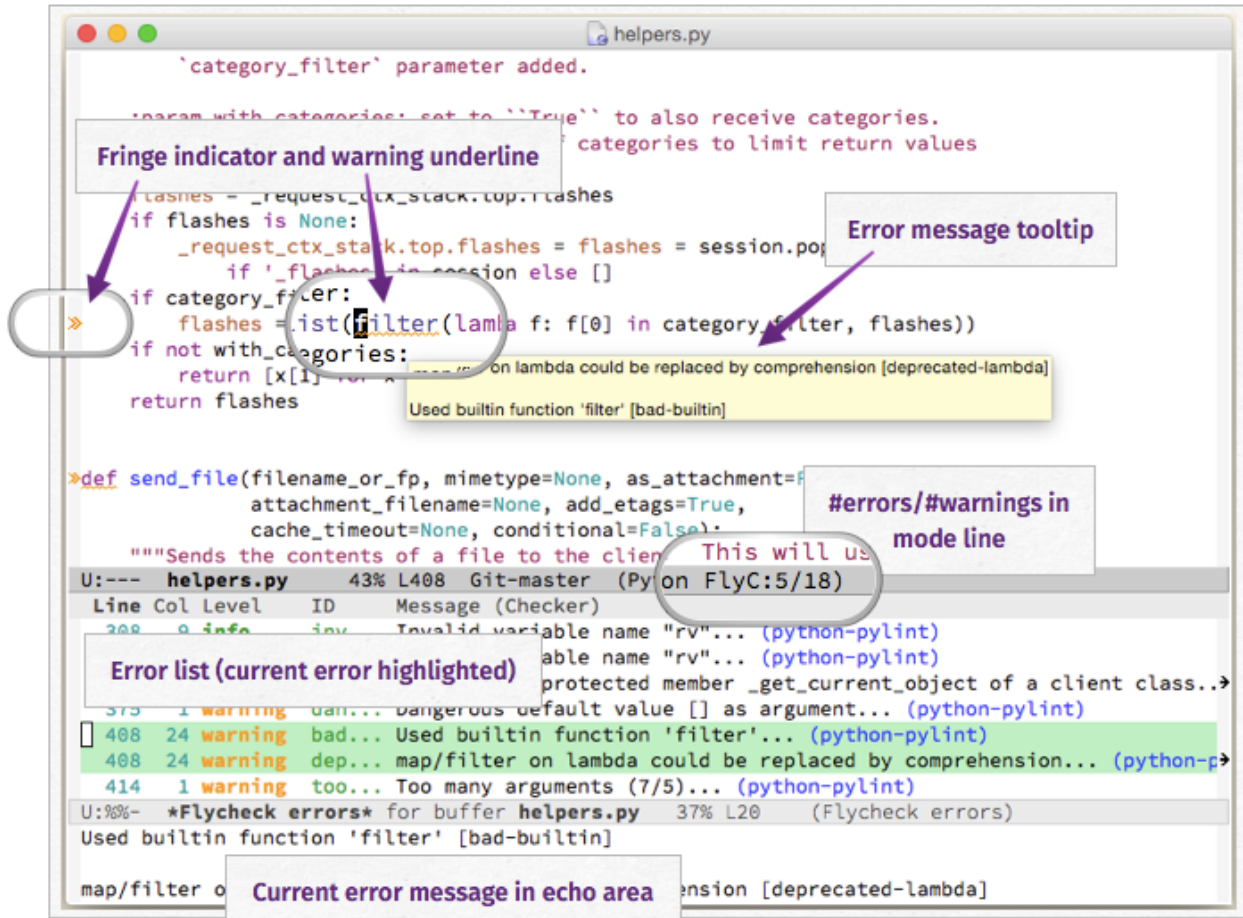


<b>1</b>	<b>Try out</b>	<b>3</b>
<b>2</b>	<b>The User Guide</b>	<b>5</b>
2.1	Installation . . . . .	5
2.2	Quickstart . . . . .	6
2.3	Syntax checks in buffers . . . . .	7
2.4	Syntax checkers . . . . .	11
2.5	Error reports in buffers . . . . .	13
2.6	Flycheck versus Flymake . . . . .	15
<b>3</b>	<b>The Developer Guide</b>	<b>25</b>
<b>4</b>	<b>The Community Guide</b>	<b>27</b>
4.1	Flycheck Code of Conduct . . . . .	27
4.2	Recommended extensions . . . . .	28
4.3	Get help . . . . .	29
4.4	People . . . . .	30
<b>5</b>	<b>The Contributor Guide</b>	<b>35</b>
5.1	Contributor's Guide . . . . .	35
5.2	Maintainer's Guide . . . . .	38
<b>6</b>	<b>Indices and Tables</b>	<b>43</b>
6.1	Supported Languages . . . . .	43
6.2	Glossary . . . . .	57
6.3	Changes . . . . .	57
<b>7</b>	<b>Licensing</b>	<b>61</b>
7.1	Flycheck licenses . . . . .	61
<b>8</b>	<b>TODO</b>	<b>89</b>



**Flycheck** is a modern on-the-fly syntax checking extension for GNU Emacs, intended as replacement for the older Flymake extension which is part of GNU Emacs. For a detailed comparison to Flymake see [Flycheck versus Flymake](#).

It uses various syntax checking and linting tools to *automatically check the contents of buffers* while you type, and reports warnings and errors directly in the buffer, or in an optional error list:



Out of the box Flycheck supports over *40 different programming languages* with more than 80 different syntax checking tools, and comes with a simple interface to define new syntax checkers.

Many *3rd party extensions* provide new syntax checkers and other features like alternative error displays or mode line indicators.



---

## Try out

---

Flycheck needs GNU Emacs 24.3 or newer, and works best on Unix systems. **Windows users**, please be aware that Flycheck does not support Windows officially, although it should mostly work fine on Windows. See [Windows support](#) and watch out for [known Windows issues](#)!

To try Flycheck in your Emacs session install some *syntax checker tools* and type the following in your `*scratch*` buffer and run `M-x eval-buffer`:

```
(require 'package)
(add-to-list 'package-archives
  '("melpa" . "http://stable.melpa.org/packages/") t)
(package-initialize)

(package-install 'flycheck)

(global-flycheck-mode)
```

For a permanent installation of Flycheck follow the [Installation](#) instructions. For a gentle introduction into Flycheck features go through [Quickstart](#) guide.





---

## The User Guide

---

The User Guide provides installation and usage help for Flycheck. It starts with installation instructions and a quick start tutorial and then focuses on an in-depth references of all parts of Flycheck.

We are currently in the process of converting the old Texinfo manual to Sphinx. Meanwhile you can read a simple HTML version of the old manual at [flycheck.html](#).

---

### Todo

Port the Texinfo manual

---

## 2.1 Installation

This document gives you detailed instructions and information about installing Flycheck.

### 2.1.1 Prerequisites

Flycheck needs GNU Emacs 24.3 and works best on Unix-like systems like Linux or OS X. It does not support older releases of GNU Emacs or other flavours of Emacs (e.g. XEmacs, Aquamacs, etc.).

### Windows support

**Flycheck does not explicitly support Windows**, but tries to maintain Windows compatibility and should generally work fine on Windows, too. However, we can neither answer questions about Windows nor fix bugs that only occur on Windows without the help of active Windows users. Please watch out for [known Windows issues](#).

### Syntax checking tools

Flycheck does not check buffers itself but relies on *external* programs to check buffers. These programs must be installed separately. Please take a look at the [list of supported languages](#) to find out what tools are required for a particular language.

Many of these programs are available in the package repositories of Linux distributions or in [Homebrew](#) for OS X. Others can be installed with standard package managers such as Rubygems, NPM, Cabal, etc.

### 2.1.2 Package installation

We recommend to install Flycheck with Emacs' built-in package manager. Flycheck is available in the popular [MELPA](#) archive which provides up to date snapshots of Flycheck's development state. The sibling repository [MELPA Stable](#) serves tagged releases of Flycheck instead. We advise to use MELPA if you are fine with weekly or even daily updates. If you would prefer longer time between releases use MELPA Stable instead.

Unfortunately neither of these repositories are available in Emacs by default. You must explicitly add them to `package-archives`, by adding the following to your *init file*:

```
(require 'package)
(add-to-list 'package-archives
  '("melpa" . "https://melpa.org/packages/") t)
(package-initialize)
```

This adds MELPA; for MELPA Stable replace `https://melpa.org` with `https://stable.melpa.org`. If you do not know where your init file is inspect the value of `user-init-file` with `C-h v user-init-file`.

Once the repository is set up you can install Flycheck from Emacs' package menu at `M-x list-packages`, or directly with `M-x package-install RET flycheck`.

#### use-package

You may want to take a look at [use-package](#) which provides simple syntax to declare and configure packages in your init file. In addition to the Github README the article [My Emacs configuration with use-package](#) has more information about `use-package`. Specifically it allows to automatically install missing packages from package archive when Emacs starts.

Add the following form to your init file to setup Flycheck with `use-package`:

```
(use-package flycheck
  :ensure t
  :init (global-flycheck-mode))
```

Then press `C-M-x` with point somewhere in this form to install and enable Flycheck for the current Emacs session.

### 2.1.3 Alternative installation methods

Some users prefer to install Flycheck via other methods such as `el-get`, Git submodules, etc.

We do **not** support any of these methods, and advise against any alternative installation method. We do not consider it a bug if Flycheck works when installed as above but not with a different installation method.

**Warning:** If you install Flycheck in any way other than *our official packages* you do so **at your own risk**.

Please beware of breakage and understand that while we do not actively work against alternative installation methods we will not make compromises to support alternative installation methods. We will close issues reported for alternative installation if we fail to reproduce them with a proper installation of Flycheck.

## 2.2 Quickstart

This page gives a quick introduction into Flycheck and an overview of its most important features. Before you start here please make sure that Flycheck is *installed*.

### 2.2.1 Enable Flycheck

Now add the following code to your *init file* to permanently enable syntax checking with Flycheck:

```
(add-hook 'after-init-hook #'global-flycheck-mode)
```

### 2.2.2 Install syntax checker programs

Now you need to install syntax checking programs for the languages you'd like to use Flycheck with. The *list of supported languages* tells you which languages Flycheck supports and what programs it uses.

For instance, you can install [Pylint](#) for Python and [ESLint](#) for Javascript:

```
$ pip install pylint
$ npm install eslint
```

### 2.2.3 Check syntax in a buffer

Now you are ready to use Flycheck in a Python or Javascript buffer. Visit a Python or Javascript file and check whether your Flycheck setup is complete with `C-c ! v`.

If everything is green Flycheck will now start to check the buffer on the fly while you are editing. Whenever you make a mistake that the eslint or Pylint catch Flycheck will highlight the corresponding place in the buffer with an error underline whose color reflects the severity of the issue. Additionally Flycheck will put a symbol into the fringe for affected lines and show the total number of errors and warnings in the buffer in the mode line.

### 2.2.4 Navigate and list errors

With `C-c ! n` and `C-c ! p` you can now jump back and forth between erroneous places. If you keep on such a place for a little while Flycheck will show the corresponding error message in the each area. Likewise, if you hover such a place with the mouse cursor Flycheck will show the error message in a tooltip.

Press `C-c ! l` to pop up a list of all errors in the current buffer. This list automatically updates itself when you fix errors or introduce new ones, and follows the currently selected buffer. If the error list is selected you can type `n` and `p` to move up and down between errors and jump to their corresponding location in the buffer.

### 2.2.5 More features

All Flycheck commands are available in the Emacs Menu at *Tools -> Syntax checking*:

The same menu also pops up when you click on the mode line lighter:

## 2.3 Syntax checks in buffers

This document explains how and when Flycheck checks buffers.

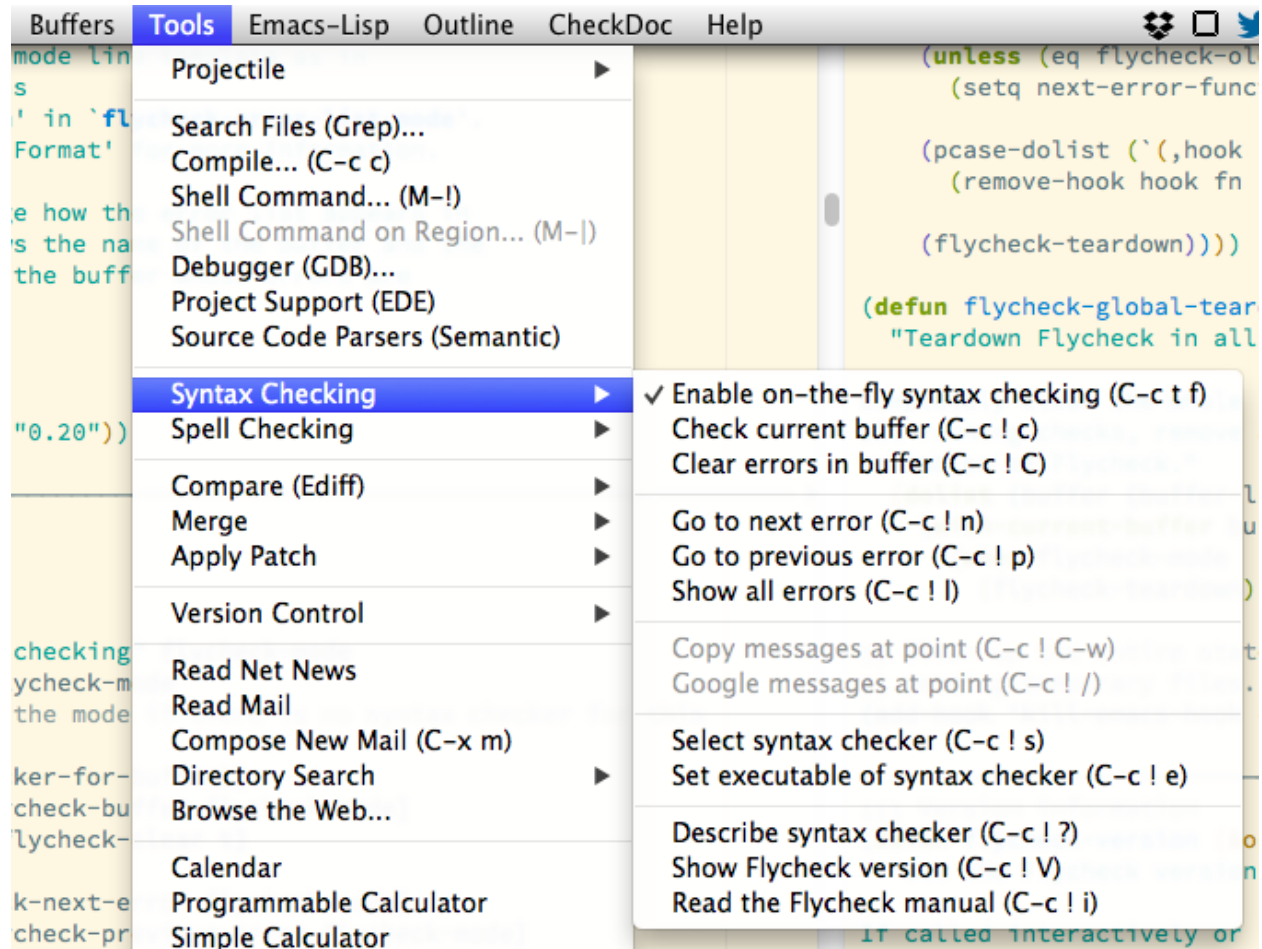


Fig. 2.1: The menu of Flycheck, showing all available Flycheck commands

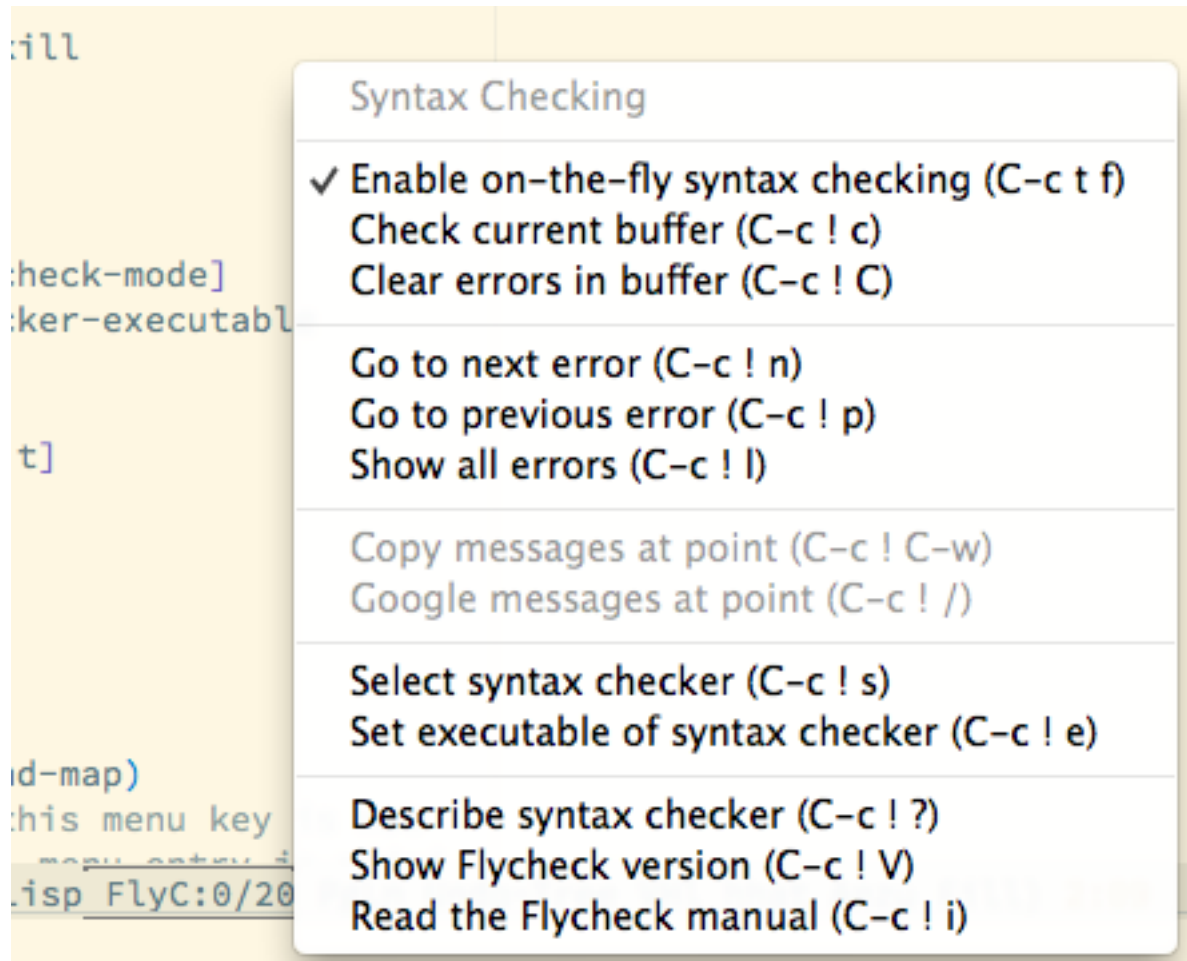


Fig. 2.2: The mode line menu of Flycheck

### 2.3.1 Minor modes

Flycheck provides two Emacs minor modes for automatic syntax checking: *Flycheck Mode* to enable syntax checking in the current buffer, and *Global Flycheck Mode* to enable syntax checking in all buffers whenever possible.

#### Minor Mode **Flycheck Mode**

Enable *automatic syntax checking* in the current buffer.

#### Minor Mode **Global Flycheck Mode**

Enable *Flycheck Mode* in all buffers where syntax checking is possible.

---

**Note:** This mode does not enable *Flycheck Mode* in remote files (via TRAMP) and encrypted files. Checking remote files may be very slow depending on the network connections, and checking encrypted files would leak confidential data to temporary files and subprocesses.

You can manually enable *Flycheck Mode* in these buffers nonetheless, but we do *not* recommend this for said reasons.

---

Add the following to your *init file* to enable syntax checking permanently:

```
(add-hook 'after-init-hook #'global-flycheck-mode)
```

You can exclude specific major modes from syntax checking with *flycheck-global-modes*:

#### User option **flycheck-global-modes**

Major modes for which *Global Flycheck Mode* turns on *Flycheck Mode*:

**t (the default)** Turn *Flycheck Mode* on for all major modes.

**(foo-mode ...)** Turn *Flycheck Mode* on for all major modes in this list, i.e. whenever the value of `major-mode` is contained in this list.

**(not foo-mode ...)** Turn *Flycheck Mode* on for all major nodes *not* in this list, i.e. whenever the value of `major-mode` is *not* contained in this list.

---

**Note:** *Global Flycheck Mode* never turns on *Flycheck Mode* in major modes whose `mode-class` property is `special`, regardless of the value of this option. Syntax checking simply makes no sense in special buffers which are typically intended for non-interactive display rather than editing.

---

**See also:**

*Major Mode Conventions*(*elisp*) Information about major modes, and modes marked as special.

### 2.3.2 Automatic syntax checks

By default *Flycheck Mode* automatically checks a buffer whenever

- it is enabled,
- the buffer is saved,
- a new line is inserted,
- or a short time after the last change was made in a buffer.

You can customise this behaviour with *flycheck-check-syntax-automatically*:

**User option `flycheck-check-syntax-automatically`**

A list of events which trigger a syntax check in the current buffer:

**save** Check the buffer immediately after it was saved.

**new-line** Check the buffer immediately after a new line was inserted.

**idle-change** Check the buffer a short time after the last change. The delay is customisable with `flycheck-idle-change-delay`:

**User option `flycheck-idle-change-delay`**

Seconds to wait after the last change to the buffer before starting a syntax check.

**mode-enabled** Check the buffer immediately after *Flycheck Mode* was enabled.

For instance with the following setting *Flycheck Mode* will only check the buffer when it was saved:

```
(setq flycheck-check-syntax-automatically '(mode-enabled save))
```

## 2.3.3 Manual checks

You can also start a syntax check explicitly with `C-c ! c`:

**C-c ! c**

**M-x flycheck-buffer**

Check syntax in the current buffer.

## 2.3.4 Debugging

To make sure that syntax checking works correctly verify your setup:

**C-c ! v**

**M-x flycheck-verify-setup**

Show a buffer with information about your *Flycheck Mode* setup for the current buffer.

Lists all syntax checkers available for the current buffer, and potential issues with their setup.

## 2.4 Syntax checkers

This document explains how Flycheck selects external tools to check a buffer.

Flycheck does not check buffers on its own. Instead it delegates this task to external *syntax checkers* which are external programs or services that receive the contents of the current buffer and return a list of errors in the buffer, together with metadata that tells Flycheck how to run the program, how to pass buffer contents to it, and how to extract errors.

**See also:**

*Supported Languages* A complete list of all syntax checkers included in Flycheck

Like everything else in Emacs syntax checkers have online documentation which you can access with `C-c ! ?`:

**C-c ! ?**

**M-x flycheck-describe-checker**

Prompt for the name of a syntax checker and pop up a Help buffer with its documentation.

The documentation includes the name of the program or service used, a list of major modes the checker supports and a list of all options for this syntax checker.

### 2.4.1 Automatic syntax checker selection

Normally Flycheck automatically selects the best syntax checkers for the current buffer from *flycheck-checkers* whenever it needs to check the buffer:

**User option *flycheck-checkers***

A list of all syntax checkers available for syntax checking.

A syntax checker in this list is a *registered syntax checker*.

Flycheck picks the first syntax checker from this list which exists and supports the current major mode, and runs it over the current buffer. When the checker has finished Flycheck whether it asks for a next syntax checker to run, and if so, runs the next syntax checker, and so on, until there is no more syntax checker for the current buffer. This process repeats whenever Flycheck needs to check the buffer according to *flycheck-check-syntax-automatically*.

For instance, the first syntax checker for Emacs Lisp is *emacs-lisp* which checks Emacs Lisp with Emacs' own byte compiler. This syntax checker asks for *emacs-lisp-checkdoc* to run next, which checks for stylistic issues in Emacs Lisp docstrings. Thus Flycheck will first run the byte compiler and then checkdoc in an Emacs Lisp buffer.

### 2.4.2 Manual syntax checker selection

Alternatively you can tell Flycheck explicitly which syntax checker to start with in the current buffer:

**C-c ! s**

**M-x *flycheck-select-checker***

Prompt for a syntax checker and use this syntax checker as the first syntax checker for the current buffer.

Flycheck may still run further syntax checkers from *flycheck-checkers* if the selected syntax checker asks for it.

Flycheck will use the selected syntax checker as “entry point” for syntax checks in the current buffer, just as if it had selected this syntax checker automatically. It will automatically run further syntax checkers from *flycheck-checkers* if the selected syntax checker asks for it.

Under the hood **C-c ! s** sets *flycheck-checker*:

**Variable *flycheck-checker***

The name of a syntax checker to use for the current buffer.

If *nil* (the default) let Flycheck *automatically select* the best syntax checker from *flycheck-checkers*.

If set to a syntax checker Flycheck will use this syntax checker as the first one in the current buffer, and run subsequent syntax checkers just as if it had selected this one automatically.

If the syntax checker in this variable does not work in the current buffer signal an error.

This variable is buffer-local.

We recommend to set *flycheck-checker* via directory local variables to enforce a specific syntax checker for a project. For instance, Flycheck usually prefers *javascript-eslint* for Javascript buffers, but if your project uses *javascript-jshint* instead you can tell Flycheck to use *javascript-jshint* for all Javascript buffers of your project with the following command in the top-level directory of your project: **M-x add-dir-local-variable RET js-mode RET flycheck-checker RET javascript-jshint**. A new buffer pops up that shows the newly created entry in the directory variables. Save this buffer and kill it. From now on Flycheck will check all Javascript files of this project with JSHint.

**See also:**

*Locals(emacs)* General information about local variables.

*Directory Variables(emacs)* Information about directory variables.



To go back to automatic selection either set `flycheck-checker` to nil or type `C-u C-c ! s`:

**C-u C-c ! s**

**C-u M-x flycheck-select-checker**

Remove any selected syntax checker and let Flycheck again *select a syntax checker automatically*.

### 2.4.3 Disabled syntax checkers

Even if you *select a checker manually* Flycheck may still use a syntax checker that you'd not like to use. To completely opt out from a specific syntax checker disable it:

**C-c ! x**

**M-x flycheck-disable-checker**

Prompt for a syntax checker to disable in the current buffer.

For instance if you do not care for documentation conventions of Emacs Lisp you can opt out from `emacs-lisp-checkdoc` which checks your code against these conventions with `C-c ! x emacs-lisp-checkdoc`. After the next check all checkdoc warnings will be gone from the buffer.

Internally this command changes the buffer-local `flycheck-disabled-checkers`:

**User option flycheck-disabled-checkers**

A list of disabled syntax checkers. Flycheck will *never* use disabled syntax checkers to check a buffer.

This option is buffer-local. You can customise this variable with `M-x customize-variable RET flycheck-disabled-checkers` or set the default value in your *init file* to permanently disable specific syntax checkers. For instance:

```
(setq-default flycheck-disabled-checkers '(c/c++-clang))
```

will permanently disable `c/c++-clang` in all buffers.

You can also disable syntax checkers per project with directory local variables. For instance type `M-x add-dir-local-variable RET emacs-lisp-mode RET flycheck-disabled-checkers RET emacs-lisp-checkdoc` in your *user emacs directory* to disable `emacs-lisp-checkdoc` for all Emacs Lisp files in your personal configuration.

**See also:**

*Locals(emacs)* General information about local variables.

*Directory Variables(emacs)* Information about directory variables.

To enable a disabled checker again, remove it from `flycheck-disabled-checkers` or use `C-u C-c ! x`:

**C-u C-c ! x**

**C-u M-x flycheck-disable-checker**

Prompt for a disabled syntax checker to enable again in the current buffer.

## 2.5 Error reports in buffers

This document explains how Flycheck shows results of syntax checks in the current buffer.

When a syntax check in the current buffer has finished Flycheck reports the results of the check in the current buffer in two ways:

- Highlight errors, warnings, etc. directly in the buffer according to `flycheck-highlighting-mode`.
- Indicate errors, warnings, etc. in the fringe according to `flycheck-indication-mode`.

## 2.5.1 Error levels

All errors that syntax checkers report have a *level* which tells you the severity of the error. Flycheck has three built-in levels:

**error** Severe errors like syntax or type errors.

**warning** Potential but not fatal mistakes which you should likely fix nonetheless.

**info** Purely informational messages which inform about notable things in the current buffer, or provide additional help to fix errors or warnings.

Each error level has a distinct highlighting and colour which helps you to identify the severity of each error right in the buffer.

## 2.5.2 Error highlights

Flycheck highlights errors directly in the buffer according to *flycheck-highlighting-mode*:

**User option `flycheck-highlighting-mode`**

How Flycheck highlights errors and warnings in the buffer:

**`nil`** Do not highlight anything at all.

**`lines`** Highlight the whole line and discard any information about the column.

**`columns`** Highlight the column of the error if any, otherwise like `lines`.

**`symbols`** Highlight the entire symbol around the error column if any, otherwise like `columns`. This is this default.

**`sexps`** Highlight the entire expression around the error column if any, otherwise like `columns`.

**Warning:** In some major modes `sexps` is *very* slow, because discovering expression boundaries efficiently is hard.  
The built-in `python-mode` is known to suffer from this issue.  
Be careful when enabling this mode.

The highlights use the following faces depending on the error level:

**Face `flycheck-error`**

**Face `flycheck-warning`**

**Face `flycheck-info`**

The highlighting face for `error`, `warning` and `info` levels respectively.

## 2.5.3 Fringe icons

In GUI frames Flycheck also adds icons to the fringe—the left or right border of an Emacs window—to help you identify erroneous lines quickly:

**User option `flycheck-indication-mode`**

How Flycheck indicates errors and warnings in the buffer fringes:

**`left-fringe` or `right-fringe`** Use the left or right fringe respectively.

**`nil`** Do not indicate errors and warnings in the fringe.

**Face `flycheck-fringe-error`**

**Face `flycheck-fringe-warning`**

**Face `flycheck-fringe-info`**

The icon faces for `error`, `warning` and `info` levels respectively.

## 2.5.4 Error thresholds

To avoid flooding a buffers with excessive highlighting, cluttering the appearance and slowing down Emacs, Flycheck takes precautions against syntax checkers that report a large number of errors exceeding `flycheck-checker-error-threshold`:

**User option `flycheck-checker-error-threshold`**

The maximum number of errors a syntax checker is allowed to report.

If a syntax checker reports more errors the error information is **discarded**. To not run into the same issue again on the next syntax check the syntax checker is automatically added to `flycheck-disabled-checkers` in this case to disable it for the next syntax check.

## 2.5.5 Clear results

You can explicitly remove all highlighting and indication and all error information from a buffer:

**C-c ! C**

**M-x flycheck-clear**

Clear all reported errors, all highlighting and all indication icons from the current buffer.

**C-u C-c ! C**

**C-u M-x flycheck-clear**

Like **C-c ! C** but also interrupt any syntax check currently running. Use this command if you think that Flycheck is stuck.

## 2.6 Flycheck versus Flymake

This article provides information about Flycheck compares to the *built-in* Flymake mode. It does not consider the improved [Flymake fork](#) or third-party extensions such as [flymake-easy](#) or [flymake-cursor](#), but references them at appropriate places.

We aim for this comparison to be neutral and complete, but do not provide any guarantee for completeness or correctness of the following information. Moreover, we consider Flycheck superior to Flymake in all aspects. As such, you may find this page biased towards Flycheck. Please excuse this as well as any factual mistake or lack of information. Please suggest improvements.

---

**Note:** This comparison was written around the time Emacs 24.5 was released, and only updated infrequently since then. Flycheck has changed and hopefully improved meanwhile, and Flymake may have done so as well. As such parts of this article may be outdated and have become incorrect by now. Likewise screenshots that show particular behaviour of Flycheck or Flymake have aged; the corresponding features of Flycheck and Flymake may look different now, or have gone altogether.

Please report any incorrectness and any inconsistency you find, and feel free to [edit this page](#) and improve it.

---

## 2.6.1 Overview

This table intends to give an overview about the differences and similarities between Flycheck and the default install of Flymake. It is not a direct comparison to third-party extensions such as [flymake-easy](#), [flymake-cursor](#), or forks of Flymake. For a more comprehensive look compared to those extensions, please read the details in the main article and the footnotes.

Please do **not** use this table alone to make your personal judgment. Read the detailed review in the following sections, too, at least with regards to the features you are interested in.

	Flycheck	Flymake
Supports Emacs versions	24.3	22+
Built-in	no <sup>1</sup>	yes
Enables automatically if possible	yes	no
Checks after	save, newline, change	newline, change
Checks in background	yes	yes
Automatic syntax checker selection	By major mode and custom predicates	By file name patterns <sup>2</sup>
Manual syntax checker selection	yes	<b>no</b>
Multiple syntax checkers per buffer	yes	<b>no</b> <sup>3</sup>
Supported languages	>40	~5 <sup>4</sup>
Checking remote files via Tramp	said to work, but not officially supported <sup>5</sup>	partly?
Definition of new syntax checkers	Single declarative function/macro	Function definition and various variables <sup>6</sup>
Functions as syntax checkers	yes	<b>no</b> <sup>7</sup>
Error levels	errors, warnings, informational, custom levels	errors, warnings <sup>8</sup>
Error identifiers	yes	no
Error parsing	Regular expressions, custom parsers for structured formats (XML, JSON, etc.)	Regular expressions
Multiline error messages	yes	<b>no</b>
Error highlighting in buffers	yes	yes
Fringe icons for errors	yes	yes (Emacs 24.1+)
Error message display	Tooltip, echo area, fully customizable	Tooltip only <sup>9</sup>
List of all errors	yes	<b>no</b>
Resource consumption	low	<b>high</b> <sup>10</sup>
Unit tests	all syntax checkers, large parts of internals	<b>none?</b>

<sup>1</sup> Flycheck is **unlikely to ever become part of Emacs**, see [issue 801](#).

<sup>2</sup> The 3rd party library [flymake-easy](#) allows to use syntax checkers per major mode.

<sup>3</sup> Various 3rd party packages thus use custom shell scripts to call multiple syntax checking tools at once.

<sup>4</sup> However, the [Flymake page](#) in the EmacsWiki provides recipes for many other languages, although of varying quality. Furthermore, the popular ELPA archive MELPA provides many packages which add more languages to Flymake. There is also a [Flymake fork](#), which supports more languages out of the box, among other fixes and improvements.

<sup>5</sup> See for instance [this comment](#).

<sup>6</sup> [flymake-easy](#) provides a function to define a new syntax checker, which sets all required variables at once.

<sup>7</sup> The [Flymake fork](#) adds support for info messages.

<sup>8</sup> [flymake-easy](#) **overrides** internal functions of Flymake to add support for multiline error messages.

<sup>9</sup> The 3rd party library [flymake-cursor](#) shows Flymake error messages at point in the echo area.

<sup>10</sup> The third-party [Flymake fork](#) mostly fixes the performance and resource consumption issues in Flymake.

## 2.6.2 Detailed review

### Relation to Emacs

**Flymake** is part of GNU Emacs since GNU Emacs 22. As such, contributions to Flymake are subject to the FSF policies on GNU projects. Most notably, contributors are required to assign their copyright to the FSF by signing a contributor agreement.

**Flycheck** is not part of GNU Emacs, and is **unlikely to ever be** (see [issue 801](#)). However, it is free software as well, and publicly developed on the well-known code hosting platform [Github](#). Contributing to Flycheck does not require a copyright assignments.

### Enabling syntax checking

**Flymake** is not enabled automatically for supported languages. It must be enabled for each mode individually and **carefully**, because it does not deal well with unavailable syntax checker tools. In a GUI frame, it signals errors in GUI dialogs. In a TTY frame, it does not signal any error at all, but instead silently hangs. The same occurs when a syntax checker tool becomes unavailable after Flymake Mode is enabled (for instance, because the underlying tool was uninstalled).

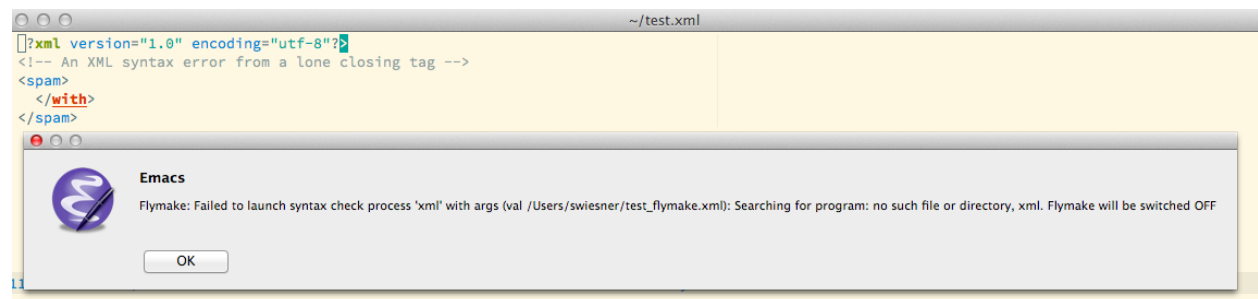


Fig. 2.3: Flymake showing a GUI dialog to inform that a syntax checker tool is not available

The third-party library [flymake-easy](#) provides an alternate way to enable Flymake Mode, which gracefully handles unavailable syntax checkers. It does not check whether the tool still exists before a syntax check, though, and thus does still exposes above behavior when a tool becomes unavailable after the mode was enabled.

**Flycheck** provides a global mode `global-flycheck-mode`, which enables syntax checking in every supported language. If a syntax checking tool is not available Flycheck fails gracefully, does not enable syntax checking, and just indicates the failure in the mode line.

### Syntax checkers

**Flymake** supports Java, Makefiles, Perl, PHP, TeX/LaTeX and XML. Notably, it does *not* support Emacs Lisp. A third-party [Flymake fork](#) supports more languages, though. Furthermore there are many recipes for other languages on the [Flymake page](#) in the EmacsWiki and many extension packages for other languages in the popular ELPA archive [MELPA](#).

**Flycheck** provides support for over 40 languages with over 70 syntax checkers, most of them contributed by the community. Notably, Flycheck does *not* support Java and Makefiles.

### Definition of new syntax checkers

**Flymake** does not provide a single function to define a new syntax checker. Instead, one has to define an “init” function, which returns the command, and add this function to `flymake-allowed-file-name-masks`. Additionally, one has to add the error patterns to `flymake-err-line-patterns`. As such, defining a syntax checker is difficult for users who are not familiar with Emacs Lisp. `flymake-easy` provides an easier way to define new syntax checkers, though.

**Flycheck** provides a single function `flycheck-define-checker` to define a new syntax checker. This function uses a declarative syntax which is easy to understand even for users unfamiliar with Emacs Lisp. In fact most syntax checkers in Flycheck were contributed by the community.

For example, the Perl checker in Flymake is defined as follows:

```
(defun flymake-perl-init ()
  (let* ((temp-file (flymake-init-create-temp-buffer-copy
                     'flymake-create-temp-inplace))
        (local-file (file-relative-name
                      temp-file
                      (file-name-directory buffer-file-name))))
    (list "perl" (list "-wc" local-file)))

(defcustom flymake-allowed-file-name-masks
  '(/; ...
    ("\\.p[ml]\\\\" flymake-perl-init)
    /; ...
  )

(defvar flymake-err-line-patterns
  (append
   '(/; ...
     ;; perl
     ("\\(\\.\\*\\\) at \\([^\n]+\\\) line \\([0-9]+\\\) [,\\.\\n]" 2 3 nil 1)
     /; ...
   )
  /; ...
  )
```

Whereas Flycheck’s definition of the same checker looks like this:

```
(flycheck-define-checker perl
  "A Perl syntax checker using the Perl interpreter.

  See URL `http://www.perl.org'."
  :command ("perl" "-w" "-c" source)
  :error-patterns
  ((error line-start (minimal-match (message))
          " at " (file-name) " line " line
          (or "." (and ", " (zero-or-more not-newline))) line-end))
  :modes (perl-mode cperl-mode))
```

### Functions as syntax checkers

**Flymake** cannot check a buffer with a custom Emacs Lisp function.

**Flycheck** provides the `flycheck-define-generic-checker` function to define a syntax checker based on an arbitrary Emacs Lisp function. Flycheck supports synchronous as well as asynchronous functions, and provides simple callback-based protocol to communicate the status of syntax checks. This allows Flycheck to use persistent

background processes for syntax checking. For instance, `flycheck-ocaml` uses a running `Merlin` process to check OCaml buffers. This is much easier and faster than invoking the OCaml compiler.

### Customization of syntax checkers

**Flymake** does not provide built-in means to customize syntax checkers. Instead, when defining a new syntax checker the user needs to declare customization variables explicitly and explicitly check their value in the `init` function.

**Flycheck** provides built-in functions to add customization variables to syntax checkers and splice the value of these variables into the argument list of a syntax checking tool. Many syntax checkers in Flycheck provide customization variables. For instance, you can customize the enabled warnings for C with `flycheck-clang-warnings`. Flycheck also tries to automatically find configuration files for syntax checkers.

### Executables of syntax checkers

**Flymake** does not provide built-in means to change the executable of a syntax checker.

**Flycheck** implicitly defines a variable to set the path of a syntax checker tool for each defined syntax checker and provides the interactive command `flycheck-set-checker-executable` to change the executable used in a buffer.

### Syntax checker selection

**Flymake** selects syntax checkers based on file name patterns in `flymake-allowed-file-name-masks`. Effectively this duplicates the existing logic Emacs uses to choose the right major mode, but lacks its flexibility and power. For instance, Flymake cannot pick a syntax checker based on the shebang of a file.

**Flycheck** uses the major mode to select a syntax checker. This reuses the existing sophisticated logic Emacs uses to choose and configure major modes. Flycheck can easily select a Python syntax checker for a Python script without file extension, but with proper shebang, simply because Emacs correctly chooses Python Mode for such a file.

### Custom predicates

**Flymake** does not allow for custom predicates to implement more complex logic for syntax checker selection. For instance, Flymake cannot use different syntax checkers for buffer depending on the value of a local variable.

However, `flymake-easy` patches Flymake to allow for custom syntax checkers per buffer. This does not happen automatically though. The user still needs to explicitly register a syntax checker in a major mode hook.

**Flycheck** supports custom predicate function. For instance, Emacs uses a single major mode for various shell script types (e.g. Bash, Zsh, POSIX Shell, etc.), so Flycheck additionally uses a custom predicate to look at the value of the variable `sh-shell` in Sh Mode buffers to determine which shell to use for syntax checking.

### Manual selection

**Flymake** does not provide means to manually select a specific syntax checker, either interactively, or via local variables.

**Flycheck** provides the local variable `flycheck-checker` to explicitly use a specific syntax checker for a buffer and the command `flycheck-select-checker` to set this variable interactively.

### Multiple syntax checkers per buffer

**Flymake** can only use a single syntax checker per buffer. Effectively, the user can only use a single tool to check a buffer, for instance either PHP Mess Detector or PHP CheckStyle. Third party extensions to Flycheck work around this limitation by supplying custom shell scripts to call multiple syntax checking tools at once.

**Flycheck** can easily apply multiple syntax checkers per buffer. For instance, Flycheck will check PHP files with PHP CLI first to find syntax errors, then with PHP MessDetector to additionally find idiomatic and semantic errors, and eventually with PHP CheckStyle to find stylistic errors. The user will see all errors reported by all of these utilities in the buffer.

### Errors

#### Error levels

**Flymake** supports error and warning messages. The pattern of warning messages is *hard-coded* in Emacs 24.3, and only became customizable in upcoming Emacs 24.4. The patterns to parse messages are kept separate from the actual syntax checker.

The third-party [Flymake fork](#) also supports info messages, and makes the pattern of warning messages customizable as well.

**Flycheck** supports error, warning and info messages. The patterns to parse messages of different levels are part of the syntax checker definition, and thus specific to each syntax checker. Flycheck allows to define new error levels for use in custom syntax checkers with `flycheck-define-error-level`.

#### Error identifiers

**Flymake** does not support unique identifiers for different kinds of errors.

**Flycheck** supports unique identifiers for different kinds of errors, if a syntax checker provides these. The identifiers appear in the error list and in error display, and can be copied independently, for instance for use in an inline suppression comment or to search the web for a particular kind of error.

#### Error parsing

**Flymake** parses the output of syntax checker tools with regular expressions only. As it splits the output by lines regardless of the regular expressions, it does not support error messages spanning multiple lines (such as returned by the Emacs Lisp byte compiler or by the Glasgow Haskell Compiler).

[flymake-easy](#) overrides internal Flymake functions to support multiline error messages.

**Flycheck** can use regular expressions as well as custom parsing functions. By means of such functions, it can parse JSON, XML or other structured output formats. Flycheck includes some ready-to-use parsing functions for well-known output formats, such as Checkstyle XML. By parsing structured output format, Flycheck can handle arbitrarily complex error messages. With regular expressions it uses the error patterns to split the output into tokens and thus handles multiline messages just as well.

#### Error message display

In GUI frames, **Flymake** shows error messages in a tool tip, if the user hovers the mouse over an error location. It does not provide means to show error messages in a TTY frame, or with the keyboard only.



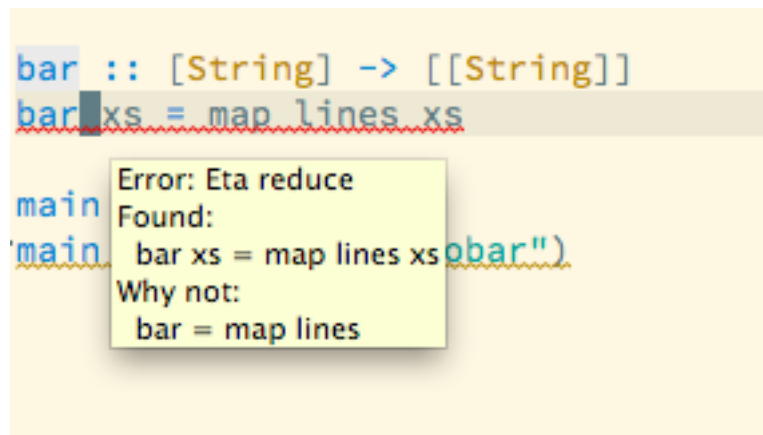


Fig. 2.4: Flymake error message in tooltip

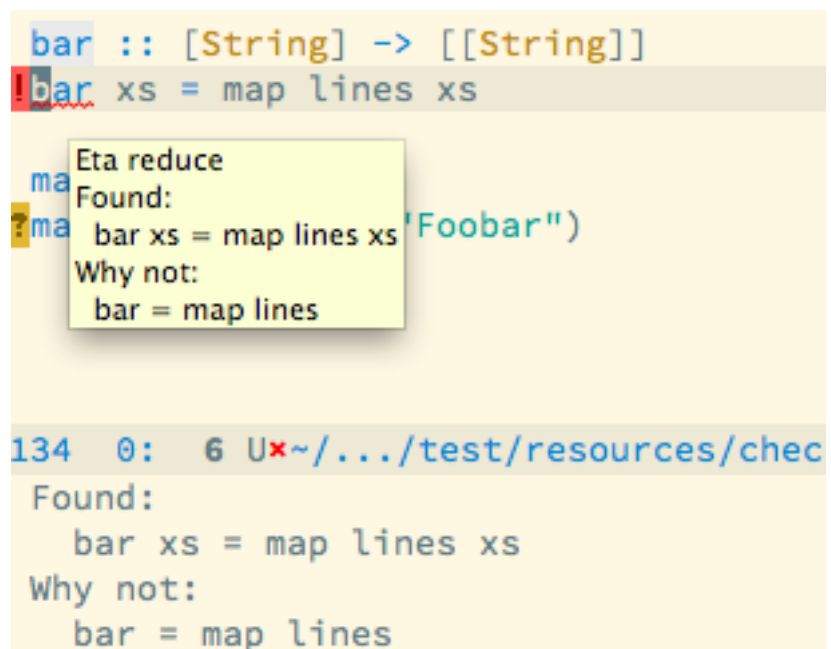


Fig. 2.5: Flycheck error message in tooltip and echo area

The third-party library `flymake-cursor` shows Flymake error messages at point in the echo area, by overriding internal Flymake functions.

**Flycheck** shows error message tool tips as well, but also displays error messages in the echo area, if the point is at an error location. This feature is fully customizable via `flycheck-display-errors-function`.

## Error list

**Flymake** does not provide means to list all errors in the current buffer.

**Flycheck** can list all errors in the current buffer in a separate window. This error list is automatically updated after each syntax check, and follows the focus.

The screenshot shows an Emacs window with a Haskell file `Warnings.hs` open. The file contains the following code:

```
module Haskell.Warnings
where

import Haskell.Internal (bar)

?foo = 10 :: Integer

spam :: [String] -> [[String]]
!spam eggs = map lines eggs

main :: IO ()
?main = (putStrLn bar)
```

Below the code, a table lists the errors found:

Line	Col	Level	Message
6	1	warning	Top-level binding with no type signature: foo :: Integer (haskell-ghc)
9	1	error	Eta reduce

Below the table, the 'Found' and 'Why not' sections are shown:

```
Found:
  spam eggs = map lines eggs
Why not:
  spam = map lines (haskell-hlint)
12 8 warning Redundant bracket
Found:
  (putStrLn bar)
Why not:
  putStrLn bar (haskell-hlint)
```

At the bottom of the window, a status bar shows: `301 0: 1 UR*Flycheck errors* for buffer Warnings.hs All Flycheck errors. hhat`

Fig. 2.6: Listing all errors in the current buffer

## Resource consumption

### Syntax checking

**Flymake** starts a syntax check after every change, regardless of whether the buffer is visible in a window or not. It does not limit the number of concurrent syntax checks. As such, Flymake starts many concurrent syntax checks when many buffers are changed at the same time (e.g. after a VCS revert), which is known to freeze Emacs temporarily.

The third-party [Flymake fork](#) limits the number of concurrent syntax checks. It does not take care to check visible buffers first, though.

**Flycheck** does not conduct syntax checks in buffers which are not visible in any window. Instead it defers syntax checks in such buffers until after the buffer is visible again. Hence, Flycheck does only start as many concurrent syntax checks as there are visible windows in the current Emacs session.

### Checking for changes

**Flymake** uses a *separate* timer (in `flymake-timer`) to periodically check for changes in each buffer. These timers run even if the corresponding buffers do not change. This is known to cause considerable CPU load with many open buffers.

The third-party [Flymake fork](#) uses a single global timer to check for changes. This greatly reduces the CPU load, but still consumes some marginal CPU, even if Emacs is idle and not in use currently.

**Flycheck** does not use timers at all to check for changes. Instead it registers a handler for Emacs' built-in `after-change-functions` hook which is run after changes to the buffer. This handler is only invoked when the buffer actually changed and starts a one-shot timer to delay the syntax check until the editing stopped for a short time, to save resources and avoid checking half-finished editing.

### Unit tests

**Flymake** does not appear to have a test suite at all.

**Flycheck** has unit tests for all built-in syntax checkers, and for large parts of the underlying machinery and API. Contributed syntax checkers are required to have test cases. A subset of the test suite is continuously run on [Travis CI](#).



---

## The Developer Guide

---

The Developer Guide shows how to write syntax checkers for Flycheck and how to extend Flycheck.

---

### Todo

Port the extending section from the Texinfo manual

---



---

## The Community Guide

---

The Community Guide provides information about Flycheck’s ecosystem and community.

### 4.1 Flycheck Code of Conduct

Our Code of Conduct defines the social norms and policies within Flycheck’s community. Whenever you interact with Flycheck or Flycheck developers, whether in our official channels or privately, you’re expected to follow this Code of Conduct.

#### 4.1.1 Conduct

**Contact:** *Any moderator*

- We are committed to providing a friendly, safe and welcoming environment for all, regardless of level of experience, gender, gender identity and expression, sexual orientation, disability, personal appearance, body size, race, ethnicity, age, religion, nationality, or similar personal characteristic.
- Please avoid using overtly sexual nicknames or other nicknames that might detract from a friendly, safe and welcoming environment for all.
- Please be kind and courteous. There’s no need to be mean or rude.
- Please do not curse or use bad words. Foul language will not help us to build a great product.
- Respect that people have differences of opinion and that every design or implementation choice carries a trade-off and numerous costs. There is seldom a right answer.
- Please keep unstructured critique to a minimum. If you have solid ideas you want to experiment with, make a fork and see how it works.
- We will exclude you from interaction if you insult, demean or harass anyone. That is not welcome behaviour. We interpret the term “harassment” as including the definition in the [Citizen Code of Conduct](#); if you have any lack of clarity about what might be included in that concept, please read their definition. In particular, we don’t tolerate behavior that excludes people in socially marginalized groups.
- Private harassment is also unacceptable. No matter who you are, if you feel you have been or are being harassed or made uncomfortable by a community member, please contact a *moderator* immediately. Whether you’re a regular contributor or a newcomer, we care about making this community a safe place for you and we’ve got your back.
- Likewise any spamming, trolling, flaming, baiting or other attention-stealing behaviour is not welcome.

### 4.1.2 Moderation

These are the policies for upholding our community’s standards of conduct in our communication channels, most notably in Flycheck’s Github organisation and in Flycheck’s Gitter channels.

1. Remarks that violate the Flycheck code of conduct, including hateful, hurtful, oppressive, or exclusionary remarks, are not allowed.
2. Remarks that moderators find inappropriate, whether listed in the code of conduct or not, are also not allowed.
3. Moderators will first respond to such remarks with a warning.
4. If the warning is unheeded, the user will be “kicked,” i.e., kicked out of the communication channel to cool off.
5. If the user comes back and continues to make trouble, they will be banned, i.e., indefinitely excluded.
6. Moderators may choose at their discretion to un-ban the user if it was a first offense and they offer the offended party a genuine apology.
7. If a moderator bans someone and you think it was unjustified, please take it up with that moderator, or with a different moderator, **in private**. Complaints about bans in-channel are not allowed.
8. Moderators are held to a higher standard than other community members. If a moderator creates an inappropriate situation, they should expect less leeway than others.

In the Flycheck community we strive to go the extra step to look out for each other. Don’t just aim to be technically unimpeachable, try to be your best self. In particular, avoid flirting with offensive or sensitive issues, particularly if they’re off-topic; this all too often leads to unnecessary fights, hurt feelings, and damaged trust; worse, it can drive people away from the community entirely.

And if someone takes issue with something you said or did, resist the urge to be defensive. Just stop doing what it was they complained about and apologize. Even if you feel you were misinterpreted or unfairly accused, chances are good there was something you could have communicated better — remember that it’s your responsibility to make your fellow Flycheck people comfortable. Everyone wants to get along and we are all here first and foremost because we want to talk about cool technology. You will find that people will be eager to assume good intent and forgive as long as you earn their trust.

—

Adapted from the [Rust Code of Conduct](#).

Copyright (c) 2015 Sebastian Wiesner and Flycheck contributors

Copyright (c) 2014 The Rust Project Developers

## 4.2 Recommended extensions

The Emacs community has produced a number of extensions to Flycheck. This page lists all that we know of and can safely recommend to our users. *Official* extensions are (co-)maintained by the [Flycheck maintainers](#) who will take care to update official extensions in case of breaking changes in Flycheck and work to provide extra API for extensions if needed. If you’d like to make your extension an *official* one and move it into the [Flycheck Github organisation](#) please contact a [maintainer](#).

If you do know extensions not in this list, or would like to see your own extension here, please feel free to [add it](#).

We would like to thank all people who created and contributed to Flycheck extensions for their awesome work. Without your help and support Flycheck would not be what it is today.



### 4.2.1 User interface

- `flycheck-color-mode-line` (*official*) colors the mode line according to the Flycheck status.
- `flycheck-pos-tip` (*official*) shows Flycheck error messages in a graphical popup.
- `liblit/flycheck-status-emoji` adds cute emoji (e.g. for errors) to Flycheck's mode line status.

### 4.2.2 Language integration

- `flycheck-cask` (*official*) makes Flycheck use Cask packages for Emacs Lisp syntax checking in `Cask` projects.
- `flycheck-rust` (*official*) configures Flycheck according to the Cargo settings and layouts of the current Rust project.
- `flycheck-haskell` (*official*) configures Flycheck from the Cabal settings and sandbox in Haskell projects.
- `Wilfred/flycheck-pkg-config` configures Flycheck to use settings from `pkg-config` when checking C/C++.

### 4.2.3 Additional languages and syntax checkers

- `Gnouc/flycheck-checkbashisms` adds a shell script syntax checker using `checkbashisms` which is part of Debian `devscripts` and checks for common Bash constructs in POSIX shell scripts.
- `clojure-emacs/squiggly-clojure` provides syntax checking for Clojure.
- `flycheck-d-unittest` (*official*) adds a Flycheck checker to run unit tests for D programs on the fly.
- `flycheck-google-cpplint` (*official*) adds a syntax checker for Google's C++ style checker.
- `cmarqu/flycheck-hdl-irun` adds a syntax checker for hardware description languages (HDLs) supported by Cadence IES/irun.
- `Sarcasm/flycheck-irony` adds a Flycheck syntax checker for C, C++ and Objective C using `Irony Mode`.
- `purcell/flycheck-ledger` adds a syntax checker for the `Ledger` accounting tool.
- `flycheck-mercury` (*official*) adds a Flycheck syntax checker for the `Mercury` language.
- `flycheck-ocaml` (*official*) adds a syntax checker for OCaml.
- `purcell/flycheck-package` checks Emacs Lisp packages for common problems with package metadata.
- `Wilfred/flycheck-pyflakes` adds a Python syntax checker using `Pyflakes`.

## 4.3 Get help

Please ask questions about Flycheck on [Stack Exchange](#) or in our [Gitter chat](#). We try to answer all questions as fast and as precise as possible.

To report bugs and problems please use our [issue tracker](#). You can follow the progress of your issues on our [Waffle board](#). Please note that we have a special policy for *Windows-only issues*.

Please follow our [Code of Conduct](#) in all these places.

## 4.4 People

### 4.4.1 Teams

#### Maintainers

- **Sebastian Wiesner** ([lunaryorn](#), head maintainer, GPG key 5C42FE98)
- Clément Pit-Claudel ([cpitclaudel](#), maintainer)

We maintain Flycheck and all official extensions within the [Flycheck organisation](#), and set the direction and scope Flycheck. We also accept or decline pull requests and feature proposals, implement changes and fix bugs in Flycheck.

Our GPG keys are used to sign commits on Github and to sign release tags for Flycheck.

Mention with `@flycheck/maintainers`.

#### Moderators

Our moderators help uphold our [Flycheck Code of Conduct](#). Currently, we do not have a dedicated moderation team; all our *Maintainers* also serve as moderators in our Github organisation and in our official communication channels.

Mention with `@flycheck/moderators`.

---

**Note:** If you'd like to help out with moderation, please contact a maintainer.

---

#### Language teams

These teams provide support for particular languages in Flycheck.

##### Go

- Dominik Honnef ([dominikh](#))

Mention with `@flycheck/go`.

##### Javascript

- Sasa Jovanic ([Simplify](#))

Mention with `@flycheck/javascript`.

##### TypeScript

- Sasa Jovanic ([Simplify](#))

Mention with `@flycheck/typescript`.

### 4.4.2 Acknowledgements

We would also like to thank the following people and projects:

- Bozhidar Batsov ([bbatsov](#)) for his valuable feedback and his constant support and endorsement of Flycheck from the very beginning. Notably he added Flycheck to his popular [Prelude](#) project at a very early stage and thus brought Flycheck to many new users.
- Magnar Sveen ([magnars](#)) for his [dash.el](#) and [s.el](#) libraries, which support considerable parts of Flycheck internals, and greatly helped to overcome the Sebastian's initial aversion to Emacs Lisp.
- Martin Grenfell ([scroolouse](#)) for the Vim syntax checking extension [Syntastic](#) which saved Sebastian's life back when he was using Vim, and served as inspiration for Flycheck and many of its syntax checkers.
- Matthias Güdemann ([mgudemann](#)), for his invaluable work on Flycheck's logo.
- Pavel Kobayakov for his work on GNU Flymake, which is a great work on its own, despite its flaws and weaknesses.
- Simon Carter ([bbbscarter](#)), for his patient in-depth testing of automatic syntax checking, and his very constructive feedback.
- Steve Purcell ([purcell](#)) for his valuable feedback, the fruitful discussions and his important ideas about the shape and design of Flycheck, and his indispensable and dedicated work on MELPA, which drives the continuous distribution of Flycheck to its users.
- Sylvain Benner ([sylv20bnr](#)) for the awesomeness that is [Spacemacs](#).

### 4.4.3 Contributors

The following people—listed in alphabetical order—contributed substantial code to Flycheck:

- Alain Kalker ([ackalker](#))
- Alex Reed ([acr4](#))
- Atila Neves ([atilaneves](#))
- Bozhidar Batsov ([bbatsov](#))
- Clément Pit-Claudé ([cpitclaudel](#), maintainer)
- Cristian Capdevila ([capdevc](#))
- Damon Haley ([dhaley](#))
- David Caldwell ([caldwell](#))
- David Holm ([dholm](#))
- Derek Chen-Becker ([dchenbecker](#))
- Derek Harland ([donkopotamus](#))
- Dominik Honnef ([dominikh](#))
- Doug MacEachern ([dougmac](#))
- Drew Wells ([drewwells](#))
- Erik Hetzner ([egh](#))
- Fanael Linithien ([Fanael](#))
- Gereon Frey ([gfrey](#))

- Gulshan Singh ([gsingh93](#))
- Iain Beeston ([iainbeeston](#))
- Jackson Ray Hamilton ([jacksonrayhamilton](#))
- Jim Hester ([jimhester](#))
- Jimmy Yuen Ho Wong ([wyuenho](#))
- Krzysztof Witkowski ([kwitek](#))
- Lee Adams ([leeaustinadams](#))
- Lorenzo Villani ([lvillani](#))
- Magnar Sveen ([magnars](#))
- Malyshev Artem ([profit404](#))
- Marcin Antczak ([marcinant](#))
- Marcus Majewski ([hekto](#))
- Marian Schubert ([maio](#))
- Mario Rodas ([marsam](#))
- Mark Hellewell ([markhellewell](#))
- Mark Karpov ([mrkkrp](#))
- Matthew Curry ([strawhatguy](#))
- Matthias Dahl ([BinaryKhaos](#))
- Michael Pankov ([mkpankov](#))
- Michael Alan Dorman ([mdorman](#))
- Miro Bezjak ([mbezjak](#))
- Mitch Tishmack ([mitchty](#))
- Moritz Bunkus ([mbunkus](#))
- Omair Majid ([omajid](#))
- Per Nordlöw ([nordlow](#))
- Peter Eisentraut ([petere](#))
- Philipp Stephani ([phst](#))
- Peter Vasil ([ptrv](#))
- Robert Dallas Gray ([rdallasgray](#))
- Robert O'Connor ([robbyoconnor](#))
- Robert Zaremba ([robert-zaremba](#))
- Sasa Jovanic ([Simplify](#))
- Sean Gillespie ([swgillespie](#))
- Sean Salmon ([phatcabbage](#))
- Sebastian Beyer ([sebastianbeyer](#))
- Sebastian Wiesner ([lunaryorn](#), founder, head maintainer)

- Stephen Lewis ([stephenjlewis](#))
- Steve Purcell ([purcell](#))
- Sven Keidel ([svenkeidel](#))
- Sylvain Benner ([sylv20bnr](#))
- Sylvain Rousseau ([thisirs](#))
- Syohei Yoshida ([syohex](#))
- Ted Zlatanov ([tzz](#))
- Tom Jakubowski ([tomjakubowski](#))
- Tomoya Tanjo ([tom-tan](#))
- Victor Deryagin ([vderyagin](#))
- Vlatko Basic ([vlatkoB](#))
- William Cummings ([wcummings](#))
- William Xu ([xwl](#))
- Yannick Roehlly ([yannick1974](#))
- Yasuyuki Oka ([yasuyk](#))
- Zhuo Yuan ([yzprofile](#))
- Łukasz Jędrzejewski ([jedrz](#))

For a complete list of all code contributors see the [Contributor Graph](#) or `git shortlog --summary`.



---

## The Contributor Guide

---

The Contributor Guide explains how to contribute to Flycheck.

### 5.1 Contributor's Guide

Thank you very much for your interest in contributing to Flycheck! We'd like to warmly welcome you in the Flycheck community, and hope that you enjoy your time with us!

There are many ways to contribute to Flycheck, and we appreciate all of them. We hope that this document helps you to contribute. If you have questions, please ask on our [issue tracker](#) or in our [Gitter chatroom](#).

For a gentle start please take a look at all the things we [need your help with](#) and look for [beginner-friendly tasks](#).

Please note that all contributors are expected to follow our *Code of Conduct*.

#### 5.1.1 Bug reports

Bugs are a sad reality in software, but we strive to have as few as possible in Flycheck. Please liberally report any bugs you find. If you are not sure whether something is a bug or not, please report anyway.

If you have the chance and time please [search existing issues](#), as it's possible that someone else already reported your issue. Of course, this doesn't always work, and sometimes it's very hard to know what to search for, so this is absolutely optional. We definitely don't mind duplicates, please report liberally.

To open an issue simply fill out the [issue form](#). To help us fix the issue, include as much information as possible. When in doubt, better include too much than too little. Here's a list of facts that are important:

- What you did, and what you expected to happen instead
- Whether and how you were able to [reproduce the issue in emacs -Q](#)
- Your Flycheck setup from `M-x flycheck-verify-setup`
- Your operating system
- Your Emacs version from `M-x emacs-version`
- Your Flycheck version from `M-x flycheck-version`

## Windows-only issues

As Flycheck does not support Windows officially we generally do *not* attempt to fix issues that only occur on Windows. We will move all Windows-only issues to the [list of open Windows issues](#), and leave them to Windows users and developers.

We welcome anyone who wants to fix open Windows issues, and we will merge pull requests for improved Windows compatibility. If you know Windows and Emacs, please take a look at the list of open Windows issues and try to fix any of these.

### 5.1.2 Feature requests

To request a new feature please open a new issue through our [issue form](#).

A feature request needs to find a core developer or maintainer who adopts and implements it. Otherwise we will move the issue to the [S-needs your love](#) column of our [Waffle board](#) where issues sit that wait for a pull request from the community.

### 5.1.3 The Build system

Flycheck provides a Makefile with some convenient targets to compile and test Flycheck. The Makefile requires [Cask](#), the Emacs Lisp dependency manager. Run `make help` to see a list of all available targets. Some common ones are:

- `make init` initialises the project by installing local Emacs Lisp dependencies.
- `make compile` compiles Flycheck and its libraries to byte code.
- `make specs` runs all [Buttercup](#) specs for Flycheck. Set **PATTERN** to run only specs matching a specific regular expression, e.g. `make PATTERN='^Mode Line' specs` to run only tests for the mode line.
- `make test` runs all ERT unit tests for Flycheck. We are phasing ERT out in favour of Buttercup; no new ERT unit tests will be added and this target will eventually be removed.
- `make integ` runs all integration tests for Flycheck syntax checkers. These tests are very dependent on the checker programs and their versions; expect failures when running this target. Set **SELECTOR** to run only tests matching a specific ERT selector, e.g. `make SELECTOR='(language haskell)' integ` to run only integration tests for Haskell. `make LANGUAGE=haskell integ` is a shortcut for this.

### 5.1.4 Pull requests

Pull Requests are the primary mechanism to submit your own changes to Flycheck. Github provides great documentation about [Pull Requests](#).

Please make your pull requests against the `master` branch.

Use `make specs test` to test your pull request locally. When making changes to syntax checkers of a specific language, it's also a good idea to run `make LANGUAGE=language integ` and check whether the tests for the particular language still work. A successful `make integ` is by no means mandatory for pull requests, though, we will test your changes, too.

All pull requests are reviewed by a [maintainer](#). Feel free to mention individual developers (e.g. @lunaryorn) to request a review from a specific person, or @flycheck/maintainers if you have general questions or if your pull request was waiting for review too long.



Additionally, all pull requests go through automated tests on [Travis CI](#) which check code style, run unit tests, etc. After the pull request was reviewed and if all tests passed a maintainer will eventually cherry-pick or merge your changes and close the pull request.

## Commit guidelines

The art of writing good commit messages is a wide subject. This model commit message illustrates our style:

```
Fix a foo bug

The first line is the summary, 50 characters or less. Write in the
imperative and in present tense: "Fix bug", not "fixed bug" or "fixes
bug".

After the summary more paragraphs with detailed explanations may follow,
wrapped at 72 characters. Separate multiple paragraphs by blank lines.

You may use simple formatting like *emphasis* or underline, but keep
it to a minimum. Commit messages are not in Markdown :)

Commit messages may reference issues by number, like this: See GH-42.
Please use `GH-` to prefix issue numbers. You may also close issues
like this: Fixes GH-42 and closes GH-42.
```

[Git Commit](#) and [Magit](#) provide Emacs mode for Git commit messages, which helps you to comply to these guidelines.

## 5.1.5 Writing documentation

Documentation improvements are very welcome. Flycheck's manual is written in [reStructuredText](#) and built with [Sphinx](#). The source of the manual resides in the `doc/` directory.

You need Python 3.4 or newer to install [Sphinx](#) for Flycheck's documentation. On OS X it is recommended that you use [Homebrew](#) to install the latest Python version with `brew install python3`. On Linux you should be able to obtain Python 3.4 from the package manager of your distribution.

With Python 3 installed change into the `doc/` directory and run `make init` to install Sphinx and related tools required for Flycheck's documentation. We recommend that you use [virtualenv](#) to avoid a global installation of Python modules. `make init` will warn you if you do not.

When editing documentation run `make html-auto` to view the results of your edits. This target runs a local webserver at <http://localhost:8000> which serves the HTML documentation and watches the documentation sources for changes to rebuild automatically. When you finished your edits it is a good idea to run `make linkcheck` to verify all links in the documentation. Note that this target can take a while especially when run on a clean build.

Run `make help` to see a list of all available Make targets for the documentation.

Documentation pull requests work in the same way as other pull requests. To find documentation issues sort by the [A-documentation](#) label.

## 5.1.6 Issue management

We manage all issues and pull requests on our [Waffle board](#). The board has six columns which correspond to S- labels on Github:

- The *Backlog* (no S label) holds all incoming issues. Pull requests waiting for review sit here, as well as bugs that were reported or stories and tasks that are not ready to work on yet.

- *Community* (`S-needs your love` label) issues are those that we will not work on ourselves. These issues need pull requests from the community to be solved. Look at this column to find spots to contribute to.
- *Blocked* (`S-blocked` label) issues are waiting for something, like a change in an upstream project or a feedback from another developer. A `B-` label may provide additional clue why the issue is blocked. Blocked issues may also appear in the backlog, but in this column we actively seek to remove the blockers and move the issue to *Ready*.
- In *Ready* (`S-ready` label) we keep issues that we are ready to work on. This includes bugs which we can reproduce and fix, and pull requests that were reviewed and are ready to be merged now. Look at this column to see what's coming next to Flycheck.
- When we start to work on an issue it moves into *In Progress* (`S-in progress` label).
- Open pull requests, whether from contributors or core developers, start in the `S-to review` column for review by maintainers. Once review is complete we will either merge the pull request and thus move it to *Done*, or move the issue back to `S-in progress` if the pull request still needs work.
- Eventually issues move into *Done* when they are closed.

In addition to these columns which reflect the basic issue workflow we also use a variety of labels to group issues:

- Yellow, **A**-prefixed labels describes the area of Flycheck the issue belongs to.
- Orange, **B**-prefixed labels gives reasons why an issue is blocked.
- Green, **E**-prefixed labels denotes the level of experience necessary to address an issue.
- Blue, **K**-prefixed labels tells the kind of an issue, i.e. whether it's a bug, a feature request, etc.
- Grey, **R**-prefixed labels inform about the resolution of an issue.

### 5.1.7 Out of tree contributions

There are many ways that you can contribute to Flycheck that go beyond this repository.

Answer questions in our [Gitter channel](#) or on [StackExchange](#).

Participate in Flycheck discussions in other Emacs communities and help users with troubles.

Write *extensions for Flycheck*.

---

This contributing guide is heavily inspired by [Rust's excellent contributing information](#).

## 5.2 Maintainer's Guide

### 5.2.1 Git workflow

Our Git workflow is simple:

- The `master` branch is always shippable.
- Every feature and every non-trivial change goes through a pull request.

GitHub calls this the “GitHub Flow” and has a very nice [visual guide](#) for this model.

## Branch rules

Our workflow implies a couple of rules about which branches to push code to:

- Please do not commit directly to `master` unless it's a trivial change, a safe refactoring, a small bug or spelling fix, etc. If in doubt please use a separate branch and open a [pull request](#).
- Please commit new features, larger changes and refactorings and updates to documentation to separate branches and open a pull request for review and discussion.

---

**Important:** When creating a new branch please use a *descriptive name* to communicate the purpose of the branch to other developers and maintainers. `fix-bug-42` is not a great name, but `42-fix-void-function-error-in-error-list` is.

If your branch addresses a specific Github issue please name your branch *issue-description*, where *issue* is the number of the Github issue *without* any prefix and *description* is the description of the branch. This convention helps us to link branches to issues and has the added bonus of automatically moving issues into “In progress” on our [Waffle board](#).

---

We do not enforce these rules to give you the freedom to ignore them when need be, like in the case of a very urgent but non-trivial bug fix. But please do try to follow these rules most of the time as they help us to maintain a high code quality in `master`.

For [maintainers](#) these rules are relaxed: They may commit to any branch at any time. Nonetheless we also recommend that maintainers open pull requests for discussion.

## Pull requests

---

### Todo

Explain how to review and merge pull requests

---

## Signatures for commits and tags

We sign all release tags as part of our [Release process](#). Thus you need a GPG key pair for Git. Github provides a great guide which helps you to [generate a key](#) and to [tell Git about your key](#). Please also [add your key](#) to your Github account.

We also recommend that you sign all your commits with your key. Again, Github provides a good guide to [sign commits](#).

**See also:**

**Signing Your Work** For more information about signing commits and tags take a look at the section in the Git manual.

## 5.2.2 Tooling and Services

In addition to [Github](#) where we host code and do code reviews we use a bit of extra tooling and some 3rd party services for Flycheck:

- [ReadTheDocs](#) hosts <http://www.flycheck.org> and automatically rebuilds it on every change. It works mostly automatically and requires little configuration.
- [Travis CI](#) runs our tests after every push and for every pull request. It's configured through `.travis.yml`.

All *maintainers* have administrative access to these services so in case of an issue just contact them.

### 5.2.3 Maintenance scripts

Administrative processes are tedious and time-consuming, so we try to automate as much as possible. The `maint/` directory contains many scripts for this purpose. `make -C maint/ help` provides an overview over all administrative tasks.

Most of these scripts require Python 3.5 and additional Python libraries. On OS X it is recommended that you use [Homebrew](#) to install the latest Python version with `brew install python3`. On Linux you should be able to obtain Python 3.5 from the package manager of your distribution.

To install all required libraries run `make -C maint init`. We recommend that you use [virtualenv](#) to avoid a global installation of Python modules. `make init` will warn you if you do not.

### 5.2.4 Versioning and releases

We use a single continuously increasing version number for Flycheck. Breaking changes may occur at any point.

Please feel free to make a release whenever you think it's appropriate. It's generally a good idea to release when

- you fixed an important bug that affects many users,
- there are a couple of new syntax checkers available,
- there's a major new feature in `master`,
- etc.

In doubt just make a release. We aim to release early and frequently. If anything breaks anything we can just publish another release afterwards.

#### Release process

First, check that

1. you are on `master`,
2. your working directory is clean, i.e. has no uncommitted changes or untracked files,
3. all commits are pushed,
4. and Travis CI passes for the latest commit on `master`.

If all is good a new release is as simple as

```
$ make -C maint release
```

This runs the release script in `maint/release.py`. If any of the above requirements isn't met the release script will signal an error and abort.

The release script bumps the version number, commits and tags a new release, and pushes it to Github.

---

**Note:** The tag is *signed*; you must configure Git for *signing commits and tags* before you make a release the first time. After pushing the new release to Github, the script bumps the version number again, to the next snapshot, and commits the changes again.

---

When the script has completed, please announce the new release in the public [Gitter](#) channel, and wherever else you see fit.



---

## Indices and Tables

---

- *Supported Languages*
- Glossary
- Changes
- genindex
- search

### 6.1 Supported Languages

This document lists all programming and markup languages which Flycheck supports.

---

**Note:** Extensions may provide support for additional languages or add deeper integration with existing languages.

Take a look at the *list of extensions* to see what the community can offer to you.

---

Each language has one or more syntax checkers whose names follow a convention of *language-tool*. All syntax checkers are listed in the order they would be applied to a buffer, with all available options. For more information about a syntax checker open Emacs and use **flycheck-describe-checker** to view the docstring of the syntax checker. Likewise, you may use **describe-variable** to read the complete docstring of any option.

#### 6.1.1 Ada

##### **ada-gnat**

Check ADA syntax and types with **GNAT**.

##### **User option flycheck-gnat-args**

A list of additional options.

##### **User option flycheck-gnat-include-path**

A list of include directories. Relative paths are relative to the path of the buffer being checked.

##### **User option flycheck-gnat-language-standard**

The language standard to use as string.

##### **User option flycheck-gnat-warnings**

A list of additional warnings to enable. Each item is the name of a warning category to enable.

### 6.1.2 AsciiDoc

#### **asciidoc**

Check [AsciiDoc](#) with the standard AsciiDoc processor.

### 6.1.3 C/C++

Flycheck checks C and C++ with either *c/c++-clang* or *c/c++-gcc*, and then with *c/c++-cppcheck*.

#### **c/c++-clang**

#### **c/c++-gcc**

Check C/C++ for syntax and type errors with [Clang](#) or [GCC](#) respectively.

**User option flycheck-clang-args**

**User option flycheck-gcc-args**

A list of additional arguments for *c/c++-clang* and *c/c++-gcc* respectively.

**User option flycheck-clang-blocks**

Whether to enable blocks in *c/c++-clang*.

**User option flycheck-clang-definitions**

**User option flycheck-gcc-definitions**

A list of additional preprocessor definitions for *c/c++-clang* and *c/c++-gcc* respectively.

**User option flycheck-clang-include-path**

**User option flycheck-gcc-include-path**

A list of include directories for *c/c++-clang* and *c/c++-gcc* respectively, relative to the file being checked.

**User option flycheck-clang-includes**

**User option flycheck-gcc-includes**

A list of additional include files for *c/c++-clang* and *c/c++-gcc* respectively, relative to the file being checked.

**User option flycheck-clang-language-standard**

**User option flycheck-gcc-language-standard**

The language standard to use in *c/c++-clang* and *c/c++-gcc* respectively as string, via the `-std` option.

**User option flycheck-clang-ms-extensions**

Whether to enable Microsoft extensions to C/C++ in *c/c++-clang*.

**User option flycheck-clang-no-exceptions**

**User option flycheck-gcc-no-exceptions**

Whether to disable exceptions in *c/c++-clang* and *c/c++-gcc* respectively.

**User option flycheck-clang-no-rtti**

**User option flycheck-gcc-no-rtti**

Whether to disable RTTI in *c/c++-clang* and *c/c++-gcc* respectively, via `-fno-rtti`.

**User option flycheck-clang-standard-library**

The name of the standard library to use for *c/c++-clang*, as string.

**User option flycheck-gcc-openmp**

Whether to enable OpenMP in *c/c++-gcc*.

**User option flycheck-clang-pedantic**

**User option flycheck-gcc-pedantic**

Whether to warn about language extensions in *c/c++-clang* and *c/c++-gcc* respectively.



**User option flycheck-clang-pedantic-errors**

**User option flycheck-gcc-pedantic-errors**

Whether to error on language extensions in *c/c++-clang* and *c/c++-gcc* respectively.

**User option flycheck-clang-warnings**

**User option flycheck-gcc-warnings**

A list of additional warnings to enable in *c/c++-clang* and *c/c++-gcc* respectively. Each item is the name of a warning or warning category for `-W`.

**c/c++-cppcheck**

Check C/C++ for semantic and stylistic issues with *cppcheck*.

**User option flycheck-cppcheck-checks**

A list of enabled checks. Each item is the name of a check for the `--enable` option.

**User option flycheck-cppcheck-inconclusive**

Whether to enable inconclusive checks. These checks may yield more false positives than normal checks.

**User option flycheck-cppcheck-include-path**

A list of include directories. Relative paths are relative to the file being checked.

**User option flycheck-cppcheck-language-standard**

The C or C++ language standard to use via `--std=`.

## 6.1.4 CFEngine

**cfengine**

Check syntax with *CFEngine*.

## 6.1.5 Chef

**chef-foodcritic**

Check style in Chef recipes with *foodcritic*.

**User option flycheck-foodcritic-tags**

A list of tags to select.

## 6.1.6 Coffeescript

Flycheck checks Coffeescript syntax with *coffee* and then lints with *coffee-coffeelint*.

**coffee**

Check syntax with the *Coffeescript* compiler.

**coffee-coffeelint**

Lint with *Coffeelint*.

**User option flycheck-coffeelintrc**

Configuration file for this syntax checker. See flycheck-config-files.

## 6.1.7 Coq

**coq**

Check and proof with the standard *Coq* compiler.

## 6.1.8 CSS

### **css-csslint**

Check syntax and style with [CSSLint](#).

## 6.1.9 D

### **d-dmd**

Check syntax and types with ([DMD](#)).

**User option flycheck-dmd-include-path**

A list of include directories.

**User option flycheck-dmd-args**

A list of additional arguments.

See also:

[flycheck-d-unittest](#) Flycheck extension which provides a syntax checker to run D unittests on the fly and report the results with Flycheck.

## 6.1.10 Emacs Lisp

Flycheck checks Emacs Lisp with [emacs-lisp](#) and then with [emacs-lisp-checkdoc](#).

### **emacs-lisp**

Check syntax with the built-in byte compiler.

**User option flycheck-emacs-lisp-load-path**

The load path as list of strings. Relative directories are expanded against the default-directory of the buffer being checked.

**User option flycheck-emacs-lisp-initialize-packages**

Whether to initialize Emacs' package manager with `package-initialize` before checking the buffer. If set to `auto` (the default), only initialize the package managers when checking files under `user-emacs-directory`.

**User option flycheck-emacs-lisp-package-user-dir**

The package directory as string. Has no effect if [flycheck-emacs-lisp-initialize-packages](#) is nil.

### **emacs-lisp-checkdoc**

Check Emacs Lisp documentation conventions with `checkdoc`.

See also:

[Documentation Tips\(\*elisp\*\)](#) Information about documentation conventions for Emacs Lisp.

[purcell/flycheck-package](#) Flycheck extension which adds a syntax checker to check for violation of Emacs Lisp library headers and packaging conventions.

[Library Headers\(\*elisp\*\)](#) Information about library headers for Emacs Lisp files.

## 6.1.11 Erlang

### **erlang**

Check Erlang with the standard [Erlang](#) compiler.

**User option flycheck-erlang-include-path**

A list of include directories.

**User option flycheck-erlang-library-path**

A list of library directories.

## 6.1.12 ERuby

**eruby-erubis**

Check ERuby with [erubis](#).

## 6.1.13 Fortran

**fortran-gfortran**

Check Fortran syntax and type with [GFortran](#).

**User option flycheck-gfortran-args**

A list of additional arguments.

**User option flycheck-gfortran-include-path**

A list of include directories. Relative paths are relative to the file being checked.

**User option flycheck-gfortran-language-standard**

The language standard to use via the `-std` option.

**User option flycheck-gfortran-layout**

The source code layout to use. Set to `free` or `fixed` for free or fixed layout respectively, or `nil` (the default) to let GFortran automatically determine the layout.

**User option flycheck-gfortran-warnings**

A list of warnings enabled via the `-W` option.

## 6.1.14 Go

Flycheck checks Go with the following checkers:

1. *go-gofmt*
2. *go-golint*
3. *go-vet*
4. *go-build* or *go-test*
5. *go-errcheck*
6. *go-unconvert*

**go-gofmt**

Check Go syntax with [gofmt](#).

**go-golint**

Check Go code style with [Golint](#).

**go-vet**

Check Go for suspicious code with [vet](#).

**User option flycheck-go-vet-print-functions**

A list of print-like functions to check calls for format string problems.

**User option `flycheck-go-vet-shadow`**

Whether to check for shadowed variables, in Go 1.6 or newer.

**`go-build`**

Check syntax and type with the [Go compiler](#).

**User option `flycheck-go-build-install-deps`**

Whether to install dependencies while checking.

**User option `flycheck-go-build-tags`**

A list of build tags.

**`go-test`**

Check syntax and types of Go tests with the [Go compiler](#).

**`go-errcheck`**

Check for unhandled error returns in Go with [errcheck](#).

**`go-unconvert`**

Check for unnecessary type conversions with [unconvert](#).

### 6.1.15 Groovy

**`groovy`**

Check syntax using the [Groovy compiler](#).

### 6.1.16 Haml

**`haml`**

Check syntax with the [Haml compiler](#).

### 6.1.17 Handlebars

**`handlebars`**

Check syntax with the [Handlebars compiler](#).

### 6.1.18 Haskell

Flycheck checks Haskell with [haskell-stack-ghc](#) (in Stack projects) or [haskell-ghc](#), and then with [haskell-hlint](#).

**See also:**

**`flycheck-haskell`** Flycheck extension to configure Flycheck's Haskell checkers from the metadata, with support for Cabal sandboxes.

**`flycheck-hdevtools`** Flycheck extension which adds an alternative syntax checker for GHC using [hdevtools](#).

**`haskell-stack-ghc`**

**`haskell-ghc`**

Check syntax and type [GHC](#). In [Stack](#) projects invoke GHC through Stack to bring package dependencies from Stack in.

**User option `flycheck-ghc-args`**

A list of additional arguments.

**User option flycheck-ghc-no-user-package-database**

Whether to disable the user package database (only for *haskell-ghc*).

**User option flycheck-ghc-stack-use-nix**

Whether to enable Nix support for Stack (only for *haskell-stack-ghc*).

**User option flycheck-ghc-package-databases**

A list of additional package databases for GHC (only for *haskell-ghc*). Each item points to a directory containing a package directory, via `-package-db`.

**User option flycheck-ghc-search-path**

A list of module directories, via `-i`.

**User option flycheck-ghc-language-extensions**

A list of language extensions, via `-X`.

**haskell-hlint**

Lint with *hlint*.

**User option flycheck-hlint-args**

A list of additional arguments.

**User option flycheck-hlint-language-extensions**

A list of language extensions to enable.

**User option flycheck-hlint-ignore-rules**

A list of rules to ignore.

**User option flycheck-hlint-hint-packages**

A list of additional hint packages to include.

**User option flycheck-hlintrc**

Configuration file for this syntax checker. See flycheck-config-files.

## 6.1.19 HTML

**html-tidy**

Check HTML syntax and style with *Tidy HTML5*.

**User option flycheck-tidyrc**

Configuration file for this syntax checker. See flycheck-config-files.

## 6.1.20 Jade

**jade**

Check syntax using the *Jade* compiler.

## 6.1.21 Javascript

Flycheck checks Javascript with one of *javascript-eslint*, *javascript-jshint* or *javascript-gjslint*, and then with *javascript-jscs*.

Alternatively *javascript-standard* is used instead all of the former ones.

**javascript-eslint**

Check syntax and lint with *ESLint*.

**User option `flycheck-eslint-rulesdir`**

A directory with custom rules.

**User option `flycheck-eslintrc`**

Configuration file for this syntax checker. See `flycheck-config-files`.

**`javascript-jshint`**

Check syntax and lint with [JSHint](#).

**User option `flycheck-jshint-extract-javascript`**

Whether to extract Javascript from HTML before linting.

**User option `flycheck-jshintrc`**

Configuration file for this syntax checker. See `flycheck-config-files`.

**`javascript-gjslint`**

Lint with [Closure Linter](#).

**User option `flycheck-gjslintrc`**

Configuration file for this syntax checker. See `flycheck-config-files`.

**`javascript-jscs`**

Check code style with [JSCS](#).

**User option `flycheck-jscsrc`**

Configuration file for this syntax checker. See `flycheck-config-files`.

**`javascript-standard`**

Check syntax and code style with [Standard](#) or [Semistandard](#).

## 6.1.22 JSON

Flycheck checks JSON with `json-jsonlint` or `json-python-json`.

**`json-jsonlint`**

Check JSON with [jsonlint](#).

**`json-python-json`**

Check JSON with Python's built-in `json` module.

## 6.1.23 Less

**`less`**

Check syntax with the [Less](#) compiler.

## 6.1.24 Lua

Flycheck checks Lua with `luacheck`, falling back to `lua`.

**`luacheck`**

Check syntax and lint with [Luacheck](#).

**User option `flycheck-luacheckrc`**

Configuration file for this syntax checker. See `flycheck-config-files`.

**`lua`**

Check syntax with the [Lua compiler](#).

### 6.1.25 Markdown

#### **markdown-mdl**

Check Markdown with [markdownlint](#).

**User option flycheck-markdown-mdl-rules**

A list of enabled rules.

**User option flycheck-markdown-mdl-tags**

A list of enabled rule tags.

**User option flycheck-markdown-mdl-style**

Configuration file for this syntax checker. See flycheck-config-files.

### 6.1.26 Perl

Flycheck checks Perl with [perl](#) and [perl-perlcritic](#).

#### **perl**

Check syntax with the [Perl](#) interpreter.

**User option flycheck-perl-include-path**

A list of include directories, relative to the file being checked.

#### **perl-perlcritic**

Lint and check style with [Perl::Critic](#).

**User option flycheck-perlcritic-severity**

The severity level as integer for the `--severity`.

**User option flycheck-perlcriticrc**

Configuration file for this syntax checker. See flycheck-config-files.

### 6.1.27 PHP

Flycheck checks PHP with [php](#), [php-phpmd](#) and [php-phpcs](#).

#### **php**

Check syntax with [PHP CLI](#)

#### **php-phpmd**

Lint with [PHP Mess Detector](#).

**User option flycheck-phpmd-rulesets**

A list of rule sets. Each item is either the name of a default rule set, or the path to a custom rule set file.

#### **php-phpcs**

Check style with [PHP Code Sniffer](#).

Needs PHP Code Sniffer 2.6 or newer.

**User option flycheck-phpcs-standard**

The coding standard, either as name of a built-in standard, or as path to a standard specification.

### 6.1.28 Processing

#### **processing**

Check syntax using the [Processing](#) compiler.

### 6.1.29 Puppet

Flycheck checks Puppet with *puppet-parser* and lints with *puppet-lint*.

#### **puppet-parser**

Check syntax with the [Puppet](#) compiler.

#### **puppet-lint**

Link with [Puppet Lint](#).

#### **User option flycheck-puppet-lint-disabled-checks**

A list of checks to disable.

#### **User option flycheck-puppet-lint-rc**

Configuration file for this syntax checker. See flycheck-config-files.

### 6.1.30 Python

Flycheck checks Python with *python-flake8* or *python-pylint*, and falls back to *python-pycompile* if neither of those is available.

See also:

[flycheck-pyflakes](#) Flycheck extension which adds a syntax checker using [Pyflakes](#).

#### **python-flake8**

Check syntax and lint with [flake8](#).

#### **User option flycheck-flake8-error-level-alist**

An alist mapping Flake8 error IDs to Flycheck error levels.

#### **User option flycheck-flake8-maximum-complexity**

The maximum McCabe complexity allowed for methods.

#### **User option flycheck-flake8-maximum-line-length**

The maximum length of lines.

#### **User option flycheck-flake8rc**

Configuration file for this syntax checker. See flycheck-config-files.

#### **python-pylint**

Check syntax and lint with [Pylint](#).

#### **User option flycheck-pylint-use-symbolic-id**

Whether to report symbolic (e.g. `no-name-in-module`) or numeric (e.g. `E0611`) message identifiers.

#### **User option flycheck-pylintrc**

Configuration file for this syntax checker. See flycheck-config-files.

#### **python-pycompile**

Check syntax with Python's byte compiler (see [py\\_compile](#)).

### 6.1.31 R

#### **r-lintr**

Check syntax and lint with [lintr](#).

#### **User option flycheck-lintr-caching**

Whether to enable caching in `lintr`. On by default; it is not recommended to disable caching unless it causes actual problems.



**User option `flycheck-lintr-linters`**

Linters to use as a string with an R expression which selects the linters to use.

### 6.1.32 Racket

**racket**

Check syntax with `raco expand` from the `compiler-lib` package.

### 6.1.33 RPM Spec

**rpm-rpmlint**

Lint with `rpmlint`.

### 6.1.34 reStructuredText

Flycheck checks reStructuredText with `rst-sphinx` in Sphinx projects and with `rst` otherwise.

**rst-sphinx**

Check documents with Sphinx.

**User option `flycheck-sphinx-warn-on-missing-references`**

Whether to emit warnings for all missing references.

**rst**

Check documents with `docutils`.

### 6.1.35 Ruby

Flycheck checks Ruby with `ruby-rubocop` and `ruby-rubylint`, falling back to `ruby` or `ruby-jruby` for basic syntax checking if those are not available.

**ruby-rubocop**

Check syntax and lint with RuboCop.

**User option `flycheck-rubocop-lint-only`**

Whether to suppress warnings about style issues, via the `--lint` option.

**User option `flycheck-rubocoprc`**

Configuration file for this syntax checker. See flycheck-config-files.

**ruby-rubylint**

Check syntax and lint with `ruby-lint`.

**User option `flycheck-rubylintrc`**

Configuration file for this syntax checker. See flycheck-config-files.

**ruby**

Check syntax with the Ruby interpreter.

**ruby-jruby**

Check syntax with the JRuby interpreter.

### 6.1.36 Rust

Flycheck checks [Rust](#) with *rust-cargo* in Cargo projects, or *rust* otherwise.

**rust-cargo****rust**

Check syntax and types with the [Rust](#) compiler. In a [Cargo](#) project the compiler is invoked through `cargo rustc` to take Cargo dependencies into account.

See also:

**flycheck-rust** Flycheck extension to configure Rust syntax checkers according to the current [Cargo](#) project.

**User option flycheck-rust-args**

A list of additional arguments.

**User option flycheck-rust-check-tests**

Whether to check test code in Rust.

**User option flycheck-rust-crate-root**

A path to the crate root for the current buffer, or nil if the current buffer is a crate by itself.

*rust-cargo* ignores this option as the crate root is given by Cargo.

**User option flycheck-rust-crate-type**

The type of the crate to check, as string for the `--crate-type` option.

**User option flycheck-rust-library-path**

A list of additional library directories. Relative paths are relative to the buffer being checked.

### 6.1.37 Sass

**sass**

Check syntax with the [Sass](#) compiler.

**User option flycheck-sass-compass**

Whether to enable the Compass CSS framework via `--compass`.

### 6.1.38 Scala

Flycheck checks Scala with *scala* and *scala-scalastyle*.

**scala**

Check syntax and types with the [Scala](#) compiler.

---

**Note:** This syntax checker is fairly primitive. For a better Scala experience we recommend [Enzyme](#).

---

**scala-scalastyle**

Check style with [Scalastyle](#).

**User option flycheck-scalastylerc**

Configuration file for this syntax checker. See flycheck-config-files.

---

**Important:** A configuration file is mandatory for this syntax checker. If *flycheck-scalastylerc* is not set or the configuration file not found this syntax checker will not be applied.

---

### 6.1.39 SCSS

Flycheck checks SCSS with `scss-lint`, falling back to `scss`.

#### **scss-lint**

Check syntax and lint with [SCSS-Lint](#).

#### **User option flycheck-scss-lintrc**

Configuration file for this syntax checker. See flycheck-config-files.

#### **scss**

Check syntax with the [SCSS compiler](#).

#### **User option flycheck-scss-compass**

Whether to enable the Compass CSS framework with `--compass`.

### 6.1.40 Shell scripting languages

Flycheck checks various shell scripting languages:

- Bash with `sh-bash` and `sh-shellcheck`
- POSIX shell (i.e. `/bin/sh`) with `sh-posix-dash` or `sh-posix-bash`
- Zsh with `sh-zsh`

#### **sh-bash**

Check [Bash](#) syntax.

#### **sh-posix-dash**

Check POSIX shell syntax with [Dash](#).

#### **sh-posix-bash**

Check POSIX shell syntax with [Bash](#).

#### **sh-zsh**

Check [Zsh](#) syntax.

#### **sh-shellcheck**

Lint Bash and POSIX shell with [ShellCheck](#).

#### **User option flycheck-shellcheck-excluded-warnings**

A list of excluded warnings.

### 6.1.41 Slim

#### **slim**

Check Slim using the [Slim compiler](#).

### 6.1.42 SQL

#### **sql-sqlint**

Check SQL syntax with [Sqlint](#).

### 6.1.43 TeX/LaTeX

Flycheck checks TeX and LaTeX with either *tex-chktex* or *tex-lacheck*.

#### **tex-chktex**

Check style with [ChkTeX](#).

#### **User option flycheck-chktexrc**

Configuration file for this syntax checker. See flycheck-config-files.

#### **tex-lacheck**

Check style with [Lacheck](#).

### 6.1.44 Texinfo

#### **texinfo**

Check syntax with **makeinfo** from [Texinfo](#).

### 6.1.45 TypeScript

#### **typescript-tslint**

Check syntax and style with [TSLint](#).

#### **User option flycheck-typescript-tslint-config**

Configuration file for this syntax checker. See flycheck-config-files.

---

**Important:** A configuration file is mandatory for this syntax checker. If *flycheck-typescript-tslint-config* is not set or the configuration file not found this syntax checker will not be used.

---

#### **User option flycheck-typescript-tslint-rulesdir**

Additional rules directory, for user created rules.

### 6.1.46 Verilog

#### **verilog-verilator**

Check syntax with [Verilator](#).

#### **User option flycheck-verilator-include-path**

A list of include directories. Relative paths are relative to the file being checked.

### 6.1.47 XML

Flycheck checks XML with *xml-xmlstarlet* or *xml-xmllint*.

#### **xml-xmlstarlet**

Check syntax with [XMLStarlet](#).

#### **xml-xmllint**

Check syntax with **xmllint** from [Libxml2](#).

## 6.1.48 YAML

Flycheck checks YAML with `yaml-jsyaml` or `yaml-ruby`.

### **yaml-jsyaml**

Check syntax with `js-yaml`.

### **yaml-ruby**

Check syntax with Ruby's YAML parser.

## 6.2 Glossary

The glossary explains most of the special terms we use in this documentation. some of these are originally explained in the [Emacs manual](#) or the [Emacs Lisp reference](#), but we reproduce them here for convenience.

**init file, user init file** Your main Emacs configuration file. It's typically located in your *user emacs directory* at `$HOME/.emacs.d/init.el`. Emacs also looks at `$HOME/.emacs`, but this location is not recommended anymore. To find out the actual path to your init file of your Emacs session inspect the value of the variable `user-init-file` with `C-h v user-init-file`. You can visit it directly with `M-: (find-file user-init-file)`.

**See also:**

*Init File(emacs)* More information about the init file.

*Init File(elisp)* Programming interface for the init file.

**user emacs directory** The directory for all Emacs related files of the current user, at `~/.emacs.d/`. Many Emacs packages create data files in this directory, and it holds the recommended location for the *init file* at `~/.emacs.d/init.el`.

**registered syntax checker** A syntax checker in *flycheck-checkers*. Flycheck will only use these syntax checkers when checking buffers automatically.

## 6.3 Changes

### 6.3.1 27 (May 08, 2016)

- **Breaking changes**
  - Require PHP Code Sniffer 2.6 or newer for `php-phpcs` [GH-921]
- **New syntax checkers:**
  - Go with `go-unconvert` [GH-905]
  - Markdown with `mdl` [GH-839] [GH-916]
  - TypeScript with `tslint` [GH-947] [GH-949]
- **Improvements:**
  - Pass checkdoc settings from Emacs to *emacs-lisp-checkdoc* [GH-741] [GH-937]
- **Bug fixes:**
  - Fix parsing of syntax errors in triple-quoted strings for `python-pycompile` [GH-948]
  - Correctly handle rules based on the current file name in `php-phpcs` [GH-921]

### 6.3.2 26 (Apr 27, 2016)

Flycheck now has a [Code of Conduct](#) which defines the acceptable behaviour and the moderation guidelines for the Flycheck community. [\[GH-819\]](#)

Flycheck also provides a [Gitter channel](#) now for questions and discussions about development. [\[GH-820\]](#)

The native Texinfo manual is again replaced with a [Sphinx](#) based documentation. We hope that this change makes the manual easier to edit and to maintain and more welcoming for new contributors. The downside is that we can not longer include a Info manual in Flycheck's MELPA packages.

From this release onward Flycheck will use a single continuously increasing version number. Breaking changes may occur at any point.

- **Breaking changes:**

- Remove `flycheck-copy-messages-as-kill`, obsolete since Flycheck 0.22
- Remove `flycheck-perlcritic-verbosity`, obsolete since Flycheck 0.22
- Replace `flycheck-completion-system` with `flycheck-completing-read-function` [\[GH-870\]](#)
- JSON syntax checkers now require `json-mode` and do not check in Javascript Mode anymore
- Prefer `eslint` over `jshint` for Javascript
- Obsolete `flycheck-info` in favour of the new `flycheck-manual` command

- New syntax checkers:

- Processing [\[GH-793\]](#) [\[GH-812\]](#)
- Racket [\[GH-799\]](#) [\[GH-873\]](#)

- New features:

- Add `flycheck-puppet-lint-rc` to customise the location of the puppetlint configuration file [\[GH-846\]](#)
- Add `flycheck-puppet-lint-disabled-checks` to disable specific checks of puppetlint [\[GH-824\]](#)
- New library `flycheck-buttercup` to support writing [Buttercup](#) specs for Flycheck
- Add `flycheck-perlcriticrc` to set a configuration file for Perl::Critic [\[GH-851\]](#)
- Add `flycheck-jshint-extract-javascript` to extract Javascript from HTML [\[GH-825\]](#)
- Add `flycheck-cppcheck-language-standard` to set the language standard for cppcheck [\[GH-862\]](#)
- Add `flycheck-mode-line-prefix` to customise the prefix of Flycheck's mode line lighter [\[GH-879\]](#) [\[GH-880\]](#)
- Add `flycheck-go-vet-shadow` to check for shadowed variables with `go vet` [\[GH-765\]](#) [\[GH-897\]](#)
- Add `flycheck-ghc-stack-use-nix` to enable Nix support for Stack GHC [\[GH-913\]](#)

- Improvements:

- Map error IDs from flake8-pep257 to Flycheck error levels
- Explicitly display errors at point with `C-c ! h` [\[GH-834\]](#)
- Merge message and checker columns in the error list to remove redundant ellipsis [\[GH-828\]](#)
- Indicate disabled checkers in verification buffers [\[GH-749\]](#)

- Do not enable Flycheck Mode in `fundamental-mode` buffers [GH-883]
- Write `go test` output to a temporary files [GH-887]
- Check whether `lintr` is actually installed [GH-911]
- Bug fixes:
  - Fix folding of C/C++ errors from included files [GH-783]
  - Fix verification of SCSS-Lint checkstyle reporter
  - Don't fall back to `rust` if `rust-cargo` should be used [GH-817]
  - Don't change current buffer when closing the error message buffer [GH-648]
  - Never display error message buffer in current window [GH-822]
  - Work around a caching issue in Rubocop [GH-844]
  - Fix checkdoc failure with some Emacs Lisp syntax [GH-833] [GH-845] [GH-898]
  - Correctly parse Haskell module name with exports right after the module name [GH-848]
  - Don't hang when sending buffers to node.js processes on Windows [GH-794][GH-850]
  - Parse suggestions from `hlint` [GH-874]
  - Go `errcheck` handles multiple `$GOPATH` entries correctly now [GH-580][GH-906]
  - Properly handle Go build failing in a directory with multiple packages [GH-676] [GH-904]
  - Make `cppcheck` recognise C++ header files [GH-909]
  - Don't run `phpcs` on empty buffers [GH-907]





---

## Licensing

---

Flycheck is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

Flycheck is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

See *GNU General Public License 3* for a copy of the GNU General Public License.

Permission is granted to copy, distribute and/or modify the Flycheck documentation under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. See *GNU Free Documentation License* for a copy of the GNU Free Documentation License.

Alternatively, you may copy, distribute and/or modify the Flycheck documentation under the terms of the Creative Commons Attribution-ShareAlike 4.0 International Public License. See *Creative Commons Attribution-ShareAlike 4.0 International* for a copy of the license.

Permission is granted to copy, distribute and/or modify the Flycheck logo under the terms of the Creative Commons Attribution-ShareAlike 4.0 International Public License. See *Creative Commons Attribution-ShareAlike 4.0 International* for a copy of the license.

## 7.1 Flycheck licenses

### 7.1.1 GNU General Public License 3

<p style="text-align: center;">GNU GENERAL PUBLIC LICENSE Version 3, 29 June 2007</p> <p>Copyright (C) 2007 Free Software Foundation, Inc. &lt;<a href="http://fsf.org/">http://fsf.org/</a>&gt; Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.</p> <p style="text-align: center;">Preamble</p> <p>The GNU General Public License is a free, copyleft license for software and other kinds of works.</p> <p>The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to</p>
---

share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

#### TERMS AND CONDITIONS

##### 0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

#### 1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

## 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

## 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's

users, your or third parties' legal rights to forbid circumvention of technological measures.

#### 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

#### 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

#### 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License,

in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent

the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

#### 7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the

terms of sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

#### 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the



violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

#### 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

#### 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

#### 11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a

consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

## 12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

## 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

## 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

## 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF

ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year>  <name of author>
```

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year>  <name of author>
```

This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.  
This is free software, and you are welcome to redistribute it  
under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>.

## 7.1.2 GNU Free Documentation License

GNU Free Documentation License  
Version 1.3, 3 November 2008

Copyright (C) 2000, 2001, 2002, 2007, 2008 Free Software Foundation, Inc.  
<<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

### 0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

### 1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or

processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

## 2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

## 3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated

as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

#### 4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and



publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

## 5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the

Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

## 6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

## 7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

## 8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

## 9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

## 10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of

following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

## 11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

## ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (c)  YEAR  YOUR NAME.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.3
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
A copy of the license is included in the section entitled "GNU
Free Documentation License".
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

### 7.1.3 Creative Commons Attribution-ShareAlike 4.0 International

Attribution-ShareAlike 4.0 International

=====

Creative Commons Corporation ("Creative Commons") is not a law firm and does not provide legal services or legal advice. Distribution of Creative Commons public licenses does not create a lawyer-client or other relationship. Creative Commons makes its licenses and related information available on an "as-is" basis. Creative Commons gives no warranties regarding its licenses, any material licensed under their terms and conditions, or any related information. Creative Commons disclaims all liability for damages resulting from their use to the fullest extent possible.

Using Creative Commons Public Licenses

Creative Commons public licenses provide a standard set of terms and conditions that creators and other rights holders may use to share original works of authorship and other material subject to copyright and certain other rights specified in the public license below. The following considerations are for informational purposes only, are not exhaustive, and do not form part of our licenses.

Considerations for licensors: Our public licenses are intended for use by those authorized to give the public permission to use material in ways otherwise restricted by copyright and certain other rights. Our licenses are irrevocable. Licensors should read and understand the terms and conditions of the license they choose before applying it. Licensors should also secure all rights necessary before applying our licenses so that the public can reuse the material as expected. Licensors should clearly mark any material not subject to the license. This includes other CC-licensed material, or material used under an exception or limitation to copyright. More considerations for licensors:

[wiki.creativecommons.org/Considerations\\_for\\_licensors](http://wiki.creativecommons.org/Considerations_for_licensors)

Considerations for the public: By using one of our public licenses, a licensor grants the public permission to use the licensed material under specified terms and conditions. If the licensor's permission is not necessary for any reason—for example, because of any applicable exception or limitation to copyright—then that use is not regulated by the license. Our licenses grant only permissions under copyright and certain other rights that a licensor has authority to grant. Use of

the licensed material may still be restricted for other reasons, including because others have copyright or other rights in the material. A licensor may make special requests, such as asking that all changes be marked or described. Although not required by our licenses, you are encouraged to respect those requests where reasonable. More considerations for the public:

[wiki.creativecommons.org/Considerations\\_for\\_licensees](http://wiki.creativecommons.org/Considerations_for_licensees)

=====

## Creative Commons Attribution-ShareAlike 4.0 International Public License

By exercising the Licensed Rights (defined below), You accept and agree to be bound by the terms and conditions of this Creative Commons Attribution-ShareAlike 4.0 International Public License ("Public License"). To the extent this Public License may be interpreted as a contract, You are granted the Licensed Rights in consideration of Your acceptance of these terms and conditions, and the Licensor grants You such rights in consideration of benefits the Licensor receives from making the Licensed Material available under these terms and conditions.

### Section 1 -- Definitions.

- a. Adapted Material means material subject to Copyright and Similar Rights that is derived from or based upon the Licensed Material and in which the Licensed Material is translated, altered, arranged, transformed, or otherwise modified in a manner requiring permission under the Copyright and Similar Rights held by the Licensor. For purposes of this Public License, where the Licensed Material is a musical work, performance, or sound recording, Adapted Material is always produced where the Licensed Material is synched in timed relation with a moving image.
- b. Adapter's License means the license You apply to Your Copyright and Similar Rights in Your contributions to Adapted Material in accordance with the terms and conditions of this Public License.
- c. BY-SA Compatible License means a license listed at [creativecommons.org/compatiblelicenses](http://creativecommons.org/compatiblelicenses), approved by Creative Commons as essentially the equivalent of this Public License.
- d. Copyright and Similar Rights means copyright and/or similar rights closely related to copyright including, without limitation, performance, broadcast, sound recording, and Sui Generis Database Rights, without regard to how the rights are labeled or categorized. For purposes of this Public License, the rights specified in Section 2(b)(1)-(2) are not Copyright and Similar Rights.
- e. Effective Technological Measures means those measures that, in the absence of proper authority, may not be circumvented under laws fulfilling obligations under Article 11 of the WIPO Copyright Treaty adopted on December 20, 1996, and/or similar international agreements.

- f. Exceptions and Limitations means fair use, fair dealing, and/or any other exception or limitation to Copyright and Similar Rights that applies to Your use of the Licensed Material.
- g. License Elements means the license attributes listed in the name of a Creative Commons Public License. The License Elements of this Public License are Attribution and ShareAlike.
- h. Licensed Material means the artistic or literary work, database, or other material to which the Licensor applied this Public License.
- i. Licensed Rights means the rights granted to You subject to the terms and conditions of this Public License, which are limited to all Copyright and Similar Rights that apply to Your use of the Licensed Material and that the Licensor has authority to license.
- j. Licensor means the individual(s) or entity(ies) granting rights under this Public License.
- k. Share means to provide material to the public by any means or process that requires permission under the Licensed Rights, such as reproduction, public display, public performance, distribution, dissemination, communication, or importation, and to make material available to the public including in ways that members of the public may access the material from a place and at a time individually chosen by them.
- l. Sui Generis Database Rights means rights other than copyright resulting from Directive 96/9/EC of the European Parliament and of the Council of 11 March 1996 on the legal protection of databases, as amended and/or succeeded, as well as other essentially equivalent rights anywhere in the world.
- m. You means the individual or entity exercising the Licensed Rights under this Public License. Your has a corresponding meaning.

## Section 2 -- Scope.

### a. License grant.

1. Subject to the terms and conditions of this Public License, the Licensor hereby grants You a worldwide, royalty-free, non-sublicensable, non-exclusive, irrevocable license to exercise the Licensed Rights in the Licensed Material to:
  - a. reproduce and Share the Licensed Material, in whole or in part; and
  - b. produce, reproduce, and Share Adapted Material.
2. Exceptions and Limitations. For the avoidance of doubt, where Exceptions and Limitations apply to Your use, this Public License does not apply, and You do not need to comply with its terms and conditions.
3. Term. The term of this Public License is specified in Section

6(a).

4. Media and formats; technical modifications allowed. The Licensor authorizes You to exercise the Licensed Rights in all media and formats whether now known or hereafter created, and to make technical modifications necessary to do so. The Licensor waives and/or agrees not to assert any right or authority to forbid You from making technical modifications necessary to exercise the Licensed Rights, including technical modifications necessary to circumvent Effective Technological Measures. For purposes of this Public License, simply making modifications authorized by this Section 2(a)(4) never produces Adapted Material.
5. Downstream recipients.
  - a. Offer from the Licensor -- Licensed Material. Every recipient of the Licensed Material automatically receives an offer from the Licensor to exercise the Licensed Rights under the terms and conditions of this Public License.
  - b. Additional offer from the Licensor -- Adapted Material. Every recipient of Adapted Material from You automatically receives an offer from the Licensor to exercise the Licensed Rights in the Adapted Material under the conditions of the Adapter's License You apply.
  - c. No downstream restrictions. You may not offer or impose any additional or different terms or conditions on, or apply any Effective Technological Measures to, the Licensed Material if doing so restricts exercise of the Licensed Rights by any recipient of the Licensed Material.
6. No endorsement. Nothing in this Public License constitutes or may be construed as permission to assert or imply that You are, or that Your use of the Licensed Material is, connected with, or sponsored, endorsed, or granted official status by, the Licensor or others designated to receive attribution as provided in Section 3(a)(1)(A)(i).

b. Other rights.

1. Moral rights, such as the right of integrity, are not licensed under this Public License, nor are publicity, privacy, and/or other similar personality rights; however, to the extent possible, the Licensor waives and/or agrees not to assert any such rights held by the Licensor to the limited extent necessary to allow You to exercise the Licensed Rights, but not otherwise.
2. Patent and trademark rights are not licensed under this Public License.
3. To the extent possible, the Licensor waives any right to collect royalties from You for the exercise of the Licensed Rights, whether directly or through a collecting society



under any voluntary or waivable statutory or compulsory licensing scheme. In all other cases the Licensor expressly reserves any right to collect such royalties.

### Section 3 -- License Conditions.

Your exercise of the Licensed Rights is expressly made subject to the following conditions.

#### a. Attribution.

1. If You Share the Licensed Material (including in modified form), You must:
  - a. retain the following if it is supplied by the Licensor with the Licensed Material:
    - i. identification of the creator(s) of the Licensed Material and any others designated to receive attribution, in any reasonable manner requested by the Licensor (including by pseudonym if designated);
    - ii. a copyright notice;
    - iii. a notice that refers to this Public License;
    - iv. a notice that refers to the disclaimer of warranties;
    - v. a URI or hyperlink to the Licensed Material to the extent reasonably practicable;
  - b. indicate if You modified the Licensed Material and retain an indication of any previous modifications; and
  - c. indicate the Licensed Material is licensed under this Public License, and include the text of, or the URI or hyperlink to, this Public License.
2. You may satisfy the conditions in Section 3(a)(1) in any reasonable manner based on the medium, means, and context in which You Share the Licensed Material. For example, it may be reasonable to satisfy the conditions by providing a URI or hyperlink to a resource that includes the required information.
3. If requested by the Licensor, You must remove any of the information required by Section 3(a)(1)(A) to the extent reasonably practicable.

#### b. ShareAlike.

In addition to the conditions in Section 3(a), if You Share Adapted Material You produce, the following conditions also apply.

1. The Adapter's License You apply must be a Creative Commons

license with the same License Elements, this version or later, or a BY-SA Compatible License.

2. You must include the text of, or the URI or hyperlink to, the Adapter's License You apply. You may satisfy this condition in any reasonable manner based on the medium, means, and context in which You Share Adapted Material.
3. You may not offer or impose any additional or different terms or conditions on, or apply any Effective Technological Measures to, Adapted Material that restrict exercise of the rights granted under the Adapter's License You apply.

#### Section 4 -- Sui Generis Database Rights.

Where the Licensed Rights include Sui Generis Database Rights that apply to Your use of the Licensed Material:

- a. for the avoidance of doubt, Section 2(a)(1) grants You the right to extract, reuse, reproduce, and Share all or a substantial portion of the contents of the database;
- b. if You include all or a substantial portion of the database contents in a database in which You have Sui Generis Database Rights, then the database in which You have Sui Generis Database Rights (but not its individual contents) is Adapted Material, including for purposes of Section 3(b); and
- c. You must comply with the conditions in Section 3(a) if You Share all or a substantial portion of the contents of the database.

For the avoidance of doubt, this Section 4 supplements and does not replace Your obligations under this Public License where the Licensed Rights include other Copyright and Similar Rights.

#### Section 5 -- Disclaimer of Warranties and Limitation of Liability.

- a. UNLESS OTHERWISE SEPARATELY UNDERTAKEN BY THE LICENSOR, TO THE EXTENT POSSIBLE, THE LICENSOR OFFERS THE LICENSED MATERIAL AS-IS AND AS-AVAILABLE, AND MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND CONCERNING THE LICENSED MATERIAL, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHER. THIS INCLUDES, WITHOUT LIMITATION, WARRANTIES OF TITLE, MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, ABSENCE OF LATENT OR OTHER DEFECTS, ACCURACY, OR THE PRESENCE OR ABSENCE OF ERRORS, WHETHER OR NOT KNOWN OR DISCOVERABLE. WHERE DISCLAIMERS OF WARRANTIES ARE NOT ALLOWED IN FULL OR IN PART, THIS DISCLAIMER MAY NOT APPLY TO YOU.
- b. TO THE EXTENT POSSIBLE, IN NO EVENT WILL THE LICENSOR BE LIABLE TO YOU ON ANY LEGAL THEORY (INCLUDING, WITHOUT LIMITATION, NEGLIGENCE) OR OTHERWISE FOR ANY DIRECT, SPECIAL, INDIRECT, INCIDENTAL, CONSEQUENTIAL, PUNITIVE, EXEMPLARY, OR OTHER LOSSES, COSTS, EXPENSES, OR DAMAGES ARISING OUT OF THIS PUBLIC LICENSE OR USE OF THE LICENSED MATERIAL, EVEN IF THE LICENSOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH LOSSES, COSTS, EXPENSES, OR DAMAGES. WHERE A LIMITATION OF LIABILITY IS NOT ALLOWED IN FULL OR

IN PART, THIS LIMITATION MAY NOT APPLY TO YOU.

- c. The disclaimer of warranties and limitation of liability provided above shall be interpreted in a manner that, to the extent possible, most closely approximates an absolute disclaimer and waiver of all liability.

#### Section 6 -- Term and Termination.

- a. This Public License applies for the term of the Copyright and Similar Rights licensed here. However, if You fail to comply with this Public License, then Your rights under this Public License terminate automatically.
- b. Where Your right to use the Licensed Material has terminated under Section 6(a), it reinstates:
  - 1. automatically as of the date the violation is cured, provided it is cured within 30 days of Your discovery of the violation; or
  - 2. upon express reinstatement by the Licensor.

For the avoidance of doubt, this Section 6(b) does not affect any right the Licensor may have to seek remedies for Your violations of this Public License.

- c. For the avoidance of doubt, the Licensor may also offer the Licensed Material under separate terms or conditions or stop distributing the Licensed Material at any time; however, doing so will not terminate this Public License.
- d. Sections 1, 5, 6, 7, and 8 survive termination of this Public License.

#### Section 7 -- Other Terms and Conditions.

- a. The Licensor shall not be bound by any additional or different terms or conditions communicated by You unless expressly agreed.
- b. Any arrangements, understandings, or agreements regarding the Licensed Material not stated herein are separate from and independent of the terms and conditions of this Public License.

#### Section 8 -- Interpretation.

- a. For the avoidance of doubt, this Public License does not, and shall not be interpreted to, reduce, limit, restrict, or impose conditions on any use of the Licensed Material that could lawfully be made without permission under this Public License.
- b. To the extent possible, if any provision of this Public License is deemed unenforceable, it shall be automatically reformed to the minimum extent necessary to make it enforceable. If the provision cannot be reformed, it shall be severed from this Public License

without affecting the enforceability of the remaining terms and conditions.

- c. No term or condition of this Public License will be waived and no failure to comply consented to unless expressly agreed to by the Licensor.
- d. Nothing in this Public License constitutes or may be interpreted as a limitation upon, or waiver of, any privileges and immunities that apply to the Licensor or You, including from the legal processes of any jurisdiction or authority.

=====

Creative Commons is not a party to its public licenses. Notwithstanding, Creative Commons may elect to apply one of its public licenses to material it publishes and in those instances will be considered the "Licensor." The text of the Creative Commons public licenses is dedicated to the public domain under the CC0 Public Domain Dedication. Except for the limited purpose of indicating that material is shared under a Creative Commons public license or as otherwise permitted by the Creative Commons policies published at [creativecommons.org/policies](http://creativecommons.org/policies), Creative Commons does not authorize the use of the trademark "Creative Commons" or any other trademark or logo of Creative Commons without its prior written consent including, without limitation, in connection with any unauthorized modifications to any of its public licenses or any other arrangements, understandings, or agreements concerning use of licensed material. For the avoidance of doubt, this paragraph does not form part of the public licenses.

Creative Commons may be contacted at [creativecommons.org](http://creativecommons.org).

---

### TODO

---

---

#### **Todo**

Explain how to review and merge pull requests

---

(The original entry is located in `/home/docs/checkouts/readthedocs.org/user_builds/flycheck/checkouts/27/doc/contributor/maintaining.r`, line 67.)

---

#### **Todo**

Port the Texinfo manual

---

(The original entry is located in `/home/docs/checkouts/readthedocs.org/user_builds/flycheck/checkouts/27/doc/index.rst`, line 70.)

---

#### **Todo**

Port the extending section from the Texinfo manual

---

(The original entry is located in `/home/docs/checkouts/readthedocs.org/user_builds/flycheck/checkouts/27/doc/index.rst`, line 87.)

---



## A

Ada  
  language, 43  
ada-gnat  
  Syntax checker, 43  
AsciiDoc  
  language, 43  
asciidoc  
  Syntax checker, 44

## C

C  
  language, 44  
C++  
  language, 44  
C-c  
  ?  
    key binding, 11  
  C  
    key binding, 15  
  c  
    key binding, 11  
  s  
    key binding, 12  
  v  
    key binding, 11  
  x  
    key binding, 13  
C-u C-c  
  C  
    key binding, 15  
  s  
    key binding, 13  
  x  
    key binding, 13  
C-u M-x flycheck-clear  
  key binding, 15  
C-u M-x flycheck-disable-checker  
  key binding, 13  
C-u M-x flycheck-select-checker

  key binding, 13  
c/c++-clang  
  Syntax checker, 44  
c/c++-cppcheck  
  Syntax checker, 45  
c/c++-gcc  
  Syntax checker, 44  
CFEngine  
  language, 45  
cfengine  
  Syntax checker, 45  
Chef  
  language, 45  
chef-foodcritic  
  Syntax checker, 45  
coffee  
  Syntax checker, 45  
coffee-coffeelint  
  Syntax checker, 45  
Coffeescript  
  language, 45  
Configuration file  
  flycheck-chktextrc, 56  
  flycheck-coffeelintrc, 45  
  flycheck-eslintrc, 50  
  flycheck-flake8rc, 52  
  flycheck-gjslintrc, 50  
  flycheck-hlintrc, 49  
  flycheck-jscsrc, 50  
  flycheck-jshintrc, 50  
  flycheck-luacheckrc, 50  
  flycheck-markdown-mdl-style, 51  
  flycheck-perlcriticrc, 51  
  flycheck-puppet-lint-rc, 52  
  flycheck-pylintrc, 52  
  flycheck-rubocoprc, 53  
  flycheck-rubylintrc, 53  
  flycheck-scalastylerc, 54  
  flycheck-scss-lintrc, 55  
  flycheck-tidyr, 49  
  flycheck-typescript-tslint-config, 56

- Coq
  - language, 45
- coq
  - Syntax checker, 45
- CSS
  - language, 45
- css-csslint
  - Syntax checker, 46
- D
- D
  - language, 46
- d-dmd
  - Syntax checker, 46
- E
- Emacs Lisp
  - language, 46
- emacs-lisp
  - Syntax checker, 46
- emacs-lisp-checkdoc
  - Syntax checker, 46
- Erlang
  - language, 46
- erlang
  - Syntax checker, 46
- ERuby
  - language, 47
- eruby-erubis
  - Syntax checker, 47
- F
- Face
  - flycheck-error, 14
  - flycheck-fringe-error, 14
  - flycheck-fringe-info, 14
  - flycheck-fringe-warning, 14
  - flycheck-info, 14
  - flycheck-warning, 14
- Flycheck Mode
  - Minor Mode, 10
- flycheck-buffer
  - Interactive command, 11
- flycheck-check-syntax-automatically
  - User option, 10
- flycheck-checker
  - Variable, 12
- flycheck-checker-error-threshold
  - User option, 15
- flycheck-checkers
  - User option, 12
- flycheck-chktexrc
  - User option, 56
- flycheck-clang-args
  - User option, 44
- flycheck-clang-blocks
  - User option, 44
- flycheck-clang-definitions
  - User option, 44
- flycheck-clang-include-path
  - User option, 44
- flycheck-clang-includes
  - User option, 44
- flycheck-clang-language-standard
  - User option, 44
- flycheck-clang-ms-extensions
  - User option, 44
- flycheck-clang-no-exceptions
  - User option, 44
- flycheck-clang-no-rtti
  - User option, 44
- flycheck-clang-pedantic
  - User option, 44
- flycheck-clang-pedantic-errors
  - User option, 44
- flycheck-clang-standard-library
  - User option, 44
- flycheck-clang-warnings
  - User option, 45
- flycheck-clear
  - Interactive command, 15
- flycheck-coffeelintrc
  - User option, 45
- flycheck-cppcheck-checks
  - User option, 45
- flycheck-cppcheck-include-path
  - User option, 45
- flycheck-cppcheck-inconclusive
  - User option, 45
- flycheck-cppcheck-language-standard
  - User option, 45
- flycheck-describe-checker
  - Interactive command, 11
- flycheck-disable-checker
  - Interactive command, 13
- flycheck-disabled-checkers
  - User option, 13
- flycheck-dmd-args
  - User option, 46
- flycheck-dmd-include-path
  - User option, 46
- flycheck-emacs-lisp-initialize-packages
  - User option, 46
- flycheck-emacs-lisp-load-path
  - User option, 46
- flycheck-emacs-lisp-package-user-dir
  - User option, 46
- flycheck-erlang-include-path



- User option, 46
- flycheck-erlang-library-path
  - User option, 47
- flycheck-error
  - Face, 14
- flycheck-eslint-rulesdir
  - User option, 49
- flycheck-eslintrc
  - User option, 50
- flycheck-flake8-error-level-alist
  - User option, 52
- flycheck-flake8-maximum-complexity
  - User option, 52
- flycheck-flake8-maximum-line-length
  - User option, 52
- flycheck-flake8rc
  - User option, 52
- flycheck-foodcritic-tags
  - User option, 45
- flycheck-fringe-error
  - Face, 14
- flycheck-fringe-info
  - Face, 14
- flycheck-fringe-warning
  - Face, 14
- flycheck-gcc-args
  - User option, 44
- flycheck-gcc-definitions
  - User option, 44
- flycheck-gcc-include-path
  - User option, 44
- flycheck-gcc-includes
  - User option, 44
- flycheck-gcc-language-standard
  - User option, 44
- flycheck-gcc-no-exceptions
  - User option, 44
- flycheck-gcc-no-rtti
  - User option, 44
- flycheck-gcc-openmp
  - User option, 44
- flycheck-gcc-pedantic
  - User option, 44
- flycheck-gcc-pedantic-errors
  - User option, 44
- flycheck-gcc-warnings
  - User option, 45
- flycheck-gfortran-args
  - User option, 47
- flycheck-gfortran-include-path
  - User option, 47
- flycheck-gfortran-language-standard
  - User option, 47
- flycheck-gfortran-layout
  - User option, 47
- flycheck-gfortran-warnings
  - User option, 47
- flycheck-ghc-args
  - User option, 48
- flycheck-ghc-language-extensions
  - User option, 49
- flycheck-ghc-no-user-package-database
  - User option, 48
- flycheck-ghc-package-databases
  - User option, 49
- flycheck-ghc-search-path
  - User option, 49
- flycheck-ghc-stack-use-nix
  - User option, 49
- flycheck-gjslintc
  - User option, 50
- flycheck-global-modes
  - User option, 10
- flycheck-gnat-args
  - User option, 43
- flycheck-gnat-include-path
  - User option, 43
- flycheck-gnat-language-standard
  - User option, 43
- flycheck-gnat-warnings
  - User option, 43
- flycheck-go-build-install-deps
  - User option, 48
- flycheck-go-build-tags
  - User option, 48
- flycheck-go-vet-print-functions
  - User option, 47
- flycheck-go-vet-shadow
  - User option, 47
- flycheck-highlighting-mode
  - User option, 14
- flycheck-hlint-args
  - User option, 49
- flycheck-hlint-hint-packages
  - User option, 49
- flycheck-hlint-ignore-rules
  - User option, 49
- flycheck-hlint-language-extensions
  - User option, 49
- flycheck-hlintc
  - User option, 49
- flycheck-idle-change-delay
  - User option, 11
- flycheck-indication-mode
  - User option, 14
- flycheck-info
  - Face, 14
- flycheck-jscsrc

- User option, 50
- flycheck-jshint-extract-javascript
  - User option, 50
- flycheck-jshintrc
  - User option, 50
- flycheck-lintr-caching
  - User option, 52
- flycheck-lintr-linters
  - User option, 52
- flycheck-luacheckrc
  - User option, 50
- flycheck-markdown-mdl-rules
  - User option, 51
- flycheck-markdown-mdl-style
  - User option, 51
- flycheck-markdown-mdl-tags
  - User option, 51
- flycheck-perl-include-path
  - User option, 51
- flycheck-perlcrritic-severity
  - User option, 51
- flycheck-perlcrriticrc
  - User option, 51
- flycheck-phpcs-standard
  - User option, 51
- flycheck-phpmd-rulesets
  - User option, 51
- flycheck-puppet-lint-disabled-checks
  - User option, 52
- flycheck-puppet-lint-rc
  - User option, 52
- flycheck-pylint-use-symbolic-id
  - User option, 52
- flycheck-pylintrc
  - User option, 52
- flycheck-rubocop-lint-only
  - User option, 53
- flycheck-rubocoprc
  - User option, 53
- flycheck-rubylintrc
  - User option, 53
- flycheck-rust-args
  - User option, 54
- flycheck-rust-check-tests
  - User option, 54
- flycheck-rust-crate-root
  - User option, 54
- flycheck-rust-crate-type
  - User option, 54
- flycheck-rust-library-path
  - User option, 54
- flycheck-sass-compass
  - User option, 54
- flycheck-scalastylerc

- User option, 54
- flycheck-scss-compass
  - User option, 55
- flycheck-scss-lintrc
  - User option, 55
- flycheck-select-checker
  - Interactive command, 12
- flycheck-shellcheck-excluded-warnings
  - User option, 55
- flycheck-sphinx-warn-on-missing-references
  - User option, 53
- flycheck-tidyrc
  - User option, 49
- flycheck-typescript-tslint-config
  - User option, 56
- flycheck-typescript-tslint-rulesdir
  - User option, 56
- flycheck-verify-setup
  - Interactive command, 11
- flycheck-verilator-include-path
  - User option, 56
- flycheck-warning
  - Face, 14
- Fortran
  - language, 47
- fortran-gfortran
  - Syntax checker, 47

## G

- Global Flycheck Mode
  - Minor Mode, 10
- Go
  - language, 47
- go-build
  - Syntax checker, 48
- go-errcheck
  - Syntax checker, 48
- go-gofmt
  - Syntax checker, 47
- go-golint
  - Syntax checker, 47
- go-test
  - Syntax checker, 48
- go-unconvert
  - Syntax checker, 48
- go-vet
  - Syntax checker, 47
- Groovy
  - language, 48
- groovy
  - Syntax checker, 48

## H

- Haml

- language, 48
- haml
  - Syntax checker, 48
- Handlebars
  - language, 48
- handlebars
  - Syntax checker, 48
- Haskell
  - language, 48
- haskell-ghc
  - Syntax checker, 48
- haskell-hlint
  - Syntax checker, 49
- haskell-stack-ghc
  - Syntax checker, 48
- HTML
  - language, 49
- html-tidy
  - Syntax checker, 49

## I

- init file, 57
- Interactive command
  - flycheck-buffer, 11
  - flycheck-clear, 15
  - flycheck-describe-checker, 11
  - flycheck-disable-checker, 13
  - flycheck-select-checker, 12
  - flycheck-verify-setup, 11

## J

- Jade
  - language, 49
- jade
  - Syntax checker, 49
- Javascript
  - language, 49
- javascript-eslint
  - Syntax checker, 49
- javascript-gjslint
  - Syntax checker, 50
- javascript-jscs
  - Syntax checker, 50
- javascript-jshint
  - Syntax checker, 50
- javascript-standard
  - Syntax checker, 50
- JSON
  - language, 50
- json-jsonlint
  - Syntax checker, 50
- json-python-json
  - Syntax checker, 50

## K

- key binding
  - C-c
    - ?, 11
    - C, 15
    - c, 11
    - s, 12
    - v, 11
    - x, 13
  - C-u C-c
    - C, 15
    - s, 13
    - x, 13
  - C-u M-x flycheck-clear, 15
  - C-u M-x flycheck-disable-checker, 13
  - C-u M-x flycheck-select-checker, 13

## L

- language
  - Ada, 43
  - AsciiDoc, 43
  - C, 44
  - C++, 44
  - CFEngine, 45
  - Chef, 45
  - Coffeescript, 45
  - Coq, 45
  - CSS, 45
  - D, 46
  - Emacs Lisp, 46
  - Erlang, 46
  - ERuby, 47
  - Fortran, 47
  - Go, 47
  - Groovy, 48
  - Haml, 48
  - Handlebars, 48
  - Haskell, 48
  - HTML, 49
  - Jade, 49
  - Javascript, 49
  - JSON, 50
  - Less, 50
  - Lua, 50
  - Markdown, 50
  - Perl, 51
  - PHP, 51
  - Processing, 51
  - Puppet, 51
  - Python, 52
  - R, 52
  - Racket, 53
  - reStructuredText, 53
  - RPM Spec, 53

- Ruby, 53
- Rust, 53
- Sass, 54
- Scala, 54
- SCSS, 54
- Shell scripting languages, 55
- Slim, 55
- SQL, 55
- TeX/LaTeX, 55
- Texinfo, 56
- TypeScript, 56
- Verilog, 56
- XML, 56
- YAML, 56
- Less
  - language, 50
- less
  - Syntax checker, 50
- Lua
  - language, 50
- lua
  - Syntax checker, 50
- luacheck
  - Syntax checker, 50

## M

- Markdown
  - language, 50
- markdown-mdl
  - Syntax checker, 51
- Minor Mode
  - Flycheck Mode, 10
  - Global Flycheck Mode, 10

## P

- Perl
  - language, 51
- perl
  - Syntax checker, 51
- perl-perlcritic
  - Syntax checker, 51
- PHP
  - language, 51
- php
  - Syntax checker, 51
- php-phpcs
  - Syntax checker, 51
- php-phpmd
  - Syntax checker, 51
- Processing
  - language, 51
- processing
  - Syntax checker, 51
- Puppet

- language, 51
- puppet-lint
  - Syntax checker, 52
- puppet-parser
  - Syntax checker, 52
- Python
  - language, 52
- python-flake8
  - Syntax checker, 52
- python-pycompile
  - Syntax checker, 52
- python-pylint
  - Syntax checker, 52

## R

- R
  - language, 52
- r-lintr
  - Syntax checker, 52
- Racket
  - language, 53
- racket
  - Syntax checker, 53
- registered syntax checker, 57
- reStructuredText
  - language, 53
- RPM Spec
  - language, 53
- rpm-rpmlint
  - Syntax checker, 53
- rst
  - Syntax checker, 53
- rst-sphinx
  - Syntax checker, 53
- Ruby
  - language, 53
- ruby
  - Syntax checker, 53
- ruby-jruby
  - Syntax checker, 53
- ruby-rubocop
  - Syntax checker, 53
- ruby-rubylint
  - Syntax checker, 53
- Rust
  - language, 53
- rust
  - Syntax checker, 54
- rust-cargo
  - Syntax checker, 54

## S

- Sass
  - language, 54

- sass
  - Syntax checker, 54
- Scala
  - language, 54
- scala
  - Syntax checker, 54
- scala-scalastyle
  - Syntax checker, 54
- SCSS
  - language, 54
- scss
  - Syntax checker, 55
- scss-lint
  - Syntax checker, 55
- sh-bash
  - Syntax checker, 55
- sh-posix-bash
  - Syntax checker, 55
- sh-posix-dash
  - Syntax checker, 55
- sh-shellcheck
  - Syntax checker, 55
- sh-zsh
  - Syntax checker, 55
- Shell scripting languages
  - language, 55
- Slim
  - language, 55
- slim
  - Syntax checker, 55
- SQL
  - language, 55
- sql-sqlint
  - Syntax checker, 55
- Syntax checker
  - ada-gnat, 43
  - asciidoc, 44
  - c/c++-clang, 44
  - c/c++-cppcheck, 45
  - c/c++-gcc, 44
  - cfengine, 45
  - chef-foodcritic, 45
  - coffee, 45
  - coffee-coffeelint, 45
  - coq, 45
  - css-csslint, 46
  - d-dmd, 46
  - emacs-lisp, 46
  - emacs-lisp-checkdoc, 46
  - erlang, 46
  - eruby-erubis, 47
  - fortran-gfortran, 47
  - go-build, 48
  - go-errcheck, 48
  - go-gofmt, 47
  - go-golint, 47
  - go-test, 48
  - go-unconvert, 48
  - go-vet, 47
  - groovy, 48
  - haml, 48
  - handlebars, 48
  - haskell-ghc, 48
  - haskell-hlint, 49
  - haskell-stack-ghc, 48
  - html-tidy, 49
  - jade, 49
  - javascript-eslint, 49
  - javascript-gjslint, 50
  - javascript-jscs, 50
  - javascript-jshint, 50
  - javascript-standard, 50
  - json-jsonlint, 50
  - json-python-json, 50
  - less, 50
  - lua, 50
  - luacheck, 50
  - markdown-mdl, 51
  - perl, 51
  - perl-perlcritic, 51
  - php, 51
  - php-phpcs, 51
  - php-phpmd, 51
  - processing, 51
  - puppet-lint, 52
  - puppet-parser, 52
  - python-flake8, 52
  - python-pycompile, 52
  - python-pylint, 52
  - r-lintr, 52
  - racket, 53
  - rpm-rpmlint, 53
  - rst, 53
  - rst-sphinx, 53
  - ruby, 53
  - ruby-jruby, 53
  - ruby-rubocop, 53
  - ruby-rubylint, 53
  - rust, 54
  - rust-cargo, 54
  - sass, 54
  - scala, 54
  - scala-scalastyle, 54
  - scss, 55
  - scss-lint, 55
  - sh-bash, 55
  - sh-posix-bash, 55
  - sh-posix-dash, 55

- sh-shellcheck, 55
- sh-zsh, 55
- slim, 55
- sql-sqlint, 55
- tex-chktex, 56
- tex-lacheck, 56
- texinfo, 56
- typescript-tslint, 56
- verilog-verilator, 56
- xml-xmllint, 56
- xml-xmlstarlet, 56
- yaml-jsyaml, 57
- yaml-ruby, 57

## T

- tex-chktex
  - Syntax checker, 56
- tex-lacheck
  - Syntax checker, 56
- TeX/LaTeX
  - language, 55
- Texinfo
  - language, 56
- texinfo
  - Syntax checker, 56
- TypeScript
  - language, 56
- typescript-tslint
  - Syntax checker, 56

## U

- user emacs directory, 57
- user init file, 57
- User option
  - flycheck-check-syntax-automatically, 10
  - flycheck-checker-error-threshold, 15
  - flycheck-checkers, 12
  - flycheck-chktexrc, 56
  - flycheck-clang-args, 44
  - flycheck-clang-blocks, 44
  - flycheck-clang-definitions, 44
  - flycheck-clang-include-path, 44
  - flycheck-clang-includes, 44
  - flycheck-clang-language-standard, 44
  - flycheck-clang-ms-extensions, 44
  - flycheck-clang-no-exceptions, 44
  - flycheck-clang-no-rtti, 44
  - flycheck-clang-pedantic, 44
  - flycheck-clang-pedantic-errors, 44
  - flycheck-clang-standard-library, 44
  - flycheck-clang-warnings, 45
  - flycheck-coffeelint, 45
  - flycheck-cppcheck-checks, 45
  - flycheck-cppcheck-include-path, 45

- flycheck-cppcheck-inconclusive, 45
- flycheck-cppcheck-language-standard, 45
- flycheck-disabled-checkers, 13
- flycheck-dmd-args, 46
- flycheck-dmd-include-path, 46
- flycheck-emacs-lisp-initialize-packages, 46
- flycheck-emacs-lisp-load-path, 46
- flycheck-emacs-lisp-package-user-dir, 46
- flycheck-erlang-include-path, 46
- flycheck-erlang-library-path, 47
- flycheck-eslint-rulesdir, 49
- flycheck-eslintrc, 50
- flycheck-flake8-error-level-alist, 52
- flycheck-flake8-maximum-complexity, 52
- flycheck-flake8-maximum-line-length, 52
- flycheck-flake8rc, 52
- flycheck-foodcritic-tags, 45
- flycheck-gcc-args, 44
- flycheck-gcc-definitions, 44
- flycheck-gcc-include-path, 44
- flycheck-gcc-includes, 44
- flycheck-gcc-language-standard, 44
- flycheck-gcc-no-exceptions, 44
- flycheck-gcc-no-rtti, 44
- flycheck-gcc-openmp, 44
- flycheck-gcc-pedantic, 44
- flycheck-gcc-pedantic-errors, 44
- flycheck-gcc-warnings, 45
- flycheck-gfortran-args, 47
- flycheck-gfortran-include-path, 47
- flycheck-gfortran-language-standard, 47
- flycheck-gfortran-layout, 47
- flycheck-gfortran-warnings, 47
- flycheck-ghc-args, 48
- flycheck-ghc-language-extensions, 49
- flycheck-ghc-no-user-package-database, 48
- flycheck-ghc-package-databases, 49
- flycheck-ghc-search-path, 49
- flycheck-ghc-stack-use-nix, 49
- flycheck-gjslint, 50
- flycheck-global-modes, 10
- flycheck-gnat-args, 43
- flycheck-gnat-include-path, 43
- flycheck-gnat-language-standard, 43
- flycheck-gnat-warnings, 43
- flycheck-go-build-install-deps, 48
- flycheck-go-build-tags, 48
- flycheck-go-vet-print-functions, 47
- flycheck-go-vet-shadow, 47
- flycheck-highlighting-mode, 14
- flycheck-hlint-args, 49
- flycheck-hlint-hint-packages, 49
- flycheck-hlint-ignore-rules, 49
- flycheck-hlint-language-extensions, 49

- flycheck-hlintrc, 49
- flycheck-idle-change-delay, 11
- flycheck-indication-mode, 14
- flycheck-jscsrc, 50
- flycheck-jshint-extract-javascript, 50
- flycheck-jshinttrc, 50
- flycheck-lintr-caching, 52
- flycheck-lintr-linters, 52
- flycheck-luacheckrc, 50
- flycheck-markdown-mdl-rules, 51
- flycheck-markdown-mdl-style, 51
- flycheck-markdown-mdl-tags, 51
- flycheck-perl-include-path, 51
- flycheck-perlcritic-severity, 51
- flycheck-perlcriticrc, 51
- flycheck-phpcs-standard, 51
- flycheck-phpmd-rulesets, 51
- flycheck-puppet-lint-disabled-checks, 52
- flycheck-puppet-lint-rc, 52
- flycheck-pylint-use-symbolic-id, 52
- flycheck-pylintrc, 52
- flycheck-rubocop-lint-only, 53
- flycheck-rubocoprc, 53
- flycheck-rubylintrc, 53
- flycheck-rust-args, 54
- flycheck-rust-check-tests, 54
- flycheck-rust-crate-root, 54
- flycheck-rust-crate-type, 54
- flycheck-rust-library-path, 54
- flycheck-sass-compass, 54
- flycheck-scalastylerc, 54
- flycheck-scss-compass, 55
- flycheck-scss-lintrc, 55
- flycheck-shellcheck-excluded-warnings, 55
- flycheck-sphinx-warn-on-missing-references, 53
- flycheck-tidyr, 49
- flycheck-typescript-tslint-config, 56
- flycheck-typescript-tslint-rulesdir, 56
- flycheck-verilator-include-path, 56

## V

Variable

- flycheck-checker, 12

Verilog

- language, 56

verilog-verilator

- Syntax checker, 56

## X

XML

- language, 56

xml-xmllint

- Syntax checker, 56

xml-xmlstarlet

Syntax checker, 56

## Y

YAML

- language, 56

yaml-jsyaml

- Syntax checker, 57

yaml-ruby

- Syntax checker, 57