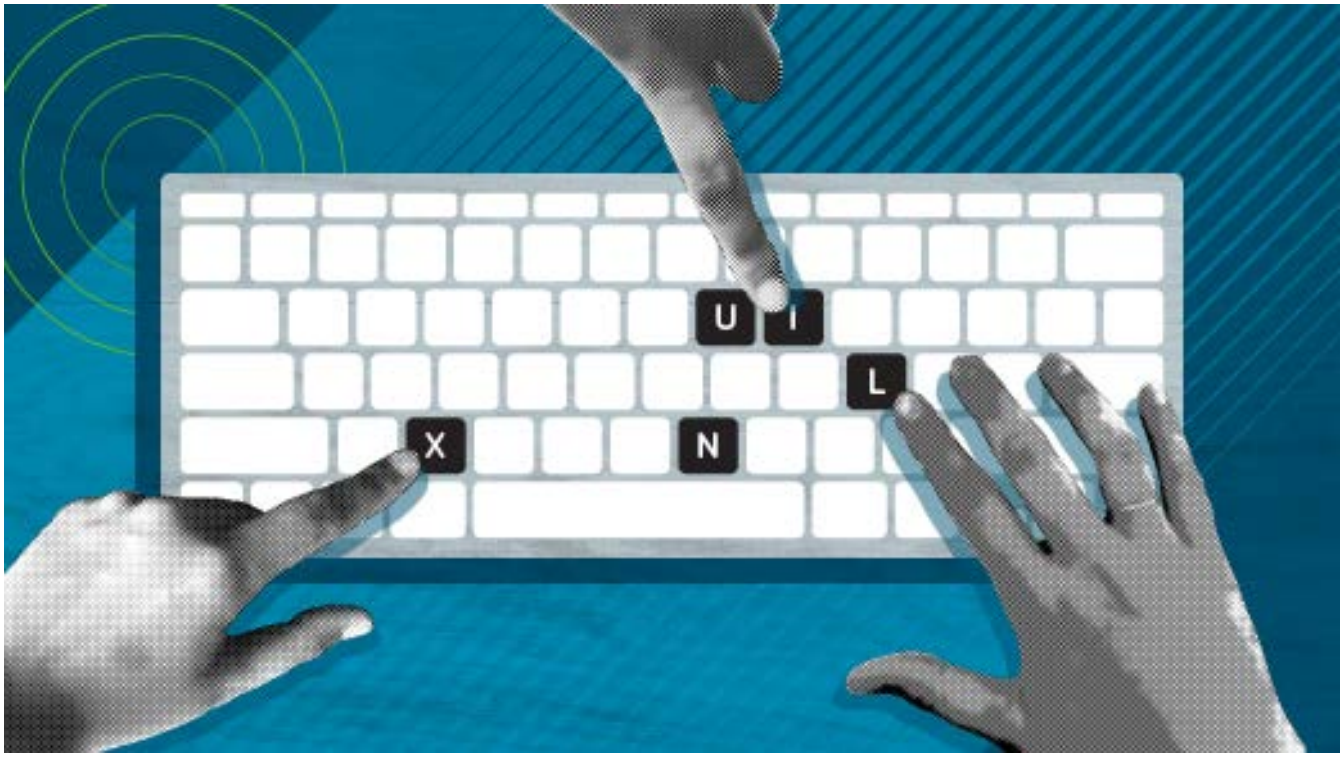
5 Git configurations I make on Linux

This is a simple guide to quickly get started working with Git and a few of its many configuration options.

By Alan Formy-Duval (Correspondent)

September 22, 2022 | 0 Comments | 5 min read



Setting up Git on Linux is simple, but here are the five things I do to get the perfect configuration:

1. Create global configuration
2. Set default name
3. Set default email address
4. Set default branch name

5. Set default editor

I manage my code, shell scripts, and documentation versioning using Git. This means that for each new project I start, the first step is to create a directory for its content and make it into a Git repository:

```
$ mkdir newproject
$ cd newproject
$ git init
```

More on Git

What is Git?

Git cheat sheet

Markdown cheat sheet

New Git articles

There are certain general settings that I always want. Not many, but enough that I don't want to have to repeat the configuration each time. I like to take advantage of the *global* configuration capability of Git.

Git offers the `git config` command for manual configuration but this is a lot of work with certain caveats. For example, a common item to set is your email address. You can set it by running `git config user.email` followed by your email address. However, this will only take effect if you are in an existing Git directory:

```
$ git config user.email alan@opensource.com
fatal: not in a git directory
```

Plus, when this command is run within a Git repository, it only configures that specific one. The process must be repeated for new repositories. I can avoid this repetition by setting it globally. The *--global* option will instruct Git to write the

email address to the global configuration file; `~/.gitconfig`, even creating it if necessary:

> Remember, the `tilde (~)` character represents your home directory. In my case that is /home/alan.

```
$ git config --global user.email alan@opensource.com
$ cat ~/.gitconfig
[user]
        email = alan@opensource.com
```

The downside here is if you have a large list of preferred settings, you will have a lot of commands to enter. This is time-consuming and prone to human error. Git provides an even more efficient and convenient way to directly edit your global configuration file—that is the first item on my list!

1. Create global configuration

If you have just started using Git, you may not have this file at all. Not to worry, let's skip the searching and get started. Just use the `--edit` option:

```
$ git config --global --edit
```

If no file is found, Git will generate one with the following content and open it in your shell environment's default editor:

```
# This is Git's per-user configuration file.
[user]
# Please adapt and uncomment the following lines:
#       name = Alan
#       email = alan@hopper
~
~
~
"~/.gitconfig" 5L, 155B
  1,1           All
```

Now that we have opened the editor and Git has created the global configuration file behind the scenes, we can continue with the rest of the settings.

2. Set default name

Name is the first directive in the file, so let's start with that. The command line to set mine is `git config --global user.name "Alan Formy-Duval"`. Instead of running this command, just edit the *name* directive in the configuration file:

```
name = Alan Formy-Duval
```

3. Set default email address

The email address is the second directive, so let's update it. By default, Git uses your system-provided name and email address. If this is incorrect or you prefer something different, you can specify it in the configuration file. In fact, if you have not configured them, Git will let you know with a friendly message the first time you commit:

```
Committer: Alan <alan@hopper>
Your name and email address were configured automatically
based
on your username and hostname. Please check that they are
accurate....
```

The command line to set mine is `git config --global user.email "alan@opensource.com"`. Instead, edit the *email* directive in the configuration file and provide your preferred address:

```
email = alan@opensource.com
```

The last two settings that I like to set are the default branch name and the default editor. These directives will need to be added while you are still in the editor.

4. Set default branch name

There is currently a trend to move away from the usage of the word *master* as the default branch name. As a matter of fact, Git will highlight this trend with a friendly message upon initialization of a new repository:

```
$ git init
hint: Using 'master' as the name for the initial branch.
This default branch name
hint: is subject to change. To configure the initial
branch name to use in all
hint: of your new repositories, which will suppress this
warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
```

This directive, named *defaultBranch*, needs to be located in a new section named *init*. It is now generally accepted that many coders use the word *main* for their default branch. This is what I like to use. Add this section followed by the directive to the configuration:

More Linux resources

Linux commands cheat sheet

Advanced Linux commands cheat sheet

Free online course: RHEL technical overview

Linux networking cheat sheet

SELinux cheat sheet

Linux common commands cheat sheet

What are Linux containers?

Our latest Linux articles

```
[init]
            defaultBranch = main
```

5. Set default editor

The fifth setting that I like to set is the default editor. This refers to the editor that Git will present for typing your commit message each time you commit changes to your repository. Everyone has a preference whether it is nano, emacs, vi, or something else. I'm happy with vi. So, to set your editor, add a *core* section that includes the *editor* directive:

```
[core]
            editor = vi
```

That's the last one. Exit the editor. Git saves the global configuration file in your home directory. If you run the editing command again, you will see all of the content. Notice that the configuration file is a plain text file, so it can also be viewed using text tools such as the `cat` command. This is how mine appears:

```
$ cat ~/.gitconfig
[user]
        email = alan@opensource.com
        name = Alan Formy-Duval
[core]
        editor = vi
[init]
        defaultBranch = main
```

This is a simple guide to quickly get started working with Git and a few of its many configuration options. There are many other articles on Git here at Opensource.com, as well as our downloadable Git cheat sheet.

Tags: LINUX GIT

Alan Formy-Duval

Alan has 20 years of IT experience, mostly in the Government and Financial sectors. He started