CC Primer: Compute Canada for geoscientists in a rush

Ricardo Barros Lourenco

Last revision: 2022-02-04

Contents

Chapter 1

Front Matter

1.1 Copyright

1.1.1 How to cite this work

Please use this BibTeX fragment:

```
@book{BarrosLourenco2022,
  title     = "CC Primer: Compute Canada for geoscientists in a rush",
  author    = "Barros Lourenco, Ricardo",
  year      = 2022,
  doi       = "10.5281/zenodo.5937906"
  note      = {\url{https://ricardobarroslourenco.github.io/CCprimer/}(visited yyyy-mm-dd)}
}
```

Obs.: Note that the DOI is of a previous version (10.5281/zenodo.5937906) when comapred to the badge displayed on this website, because it relates to all versions of this document, rather than a especific one.

1.2 Cronology

2022-01-31 - Beginning of v0.1. (This version is quite unstable, and versioning during this stage will be limited).

Chapter 2

Version Control Systems

Version Control Systems Wikipedia contributors [2002] (VCS - also named as *Revision Control*, *Source Code Control*, *Source Code Management*) are Computer Systems traditionally used to manage the complexities of software development, majorly Computer Systems involving multiple software modules, and large development teams.

Such systems help to organize such development, especially when collaboration of multiple developers is done, and several times people are doing code changes in a same, or close, part of the code, which may break it, or even generate unexpected results that would not be traceable.

2.1 VCS: Local repository vs. Remote repository

Common VCS's (ex.: CVS, SVN and Git) always are client-server systems.

The VCS client is not merely a means to access a central repository hosted at the VCS server, but actually is part of running checks and balances that are necessary when adding code to the central repository. A situation that demands that is when multiple users are doing a change in a same piece of code, and checking it only at the central repository may complicate more a situation already complicated (simultaneous contribution). Therefore, it hosts, locally a local copy of the codebase, which reflects a view of that codebase in time (more preciselly when that local user retrieved a copy from the central repository for editing).

Once the changes are done, the user saves a local copy of these at the local repository, and if these meet the criteria for changes (and this may vary a lot among systems), the user is able to persist this change at the local repository,

as a new version of the codebase. This process is known as *commit* (in this case, a local one).

2.2 Git

Git is a contemporary VCS that is used as the backbone of operations run on GitHub. So, when planning to operate with GitHub, it is almost certain that you will need to install a Git client on your machine.

We can say *almost* because GitHub provides an own client, GitHub Desktop which provides a Graphical User Interface(GUI) to a git client.

We will not cover such GUI usage since on Compute Canada you will not have access to GitHub Desktop on their machines, but only to a regular Git client (that you can use to access either GitHub, or another Git compliant server such as a private repository built with GitLab).

2.2.1 Installing a git client on your machine

2.2.1.1 Ubuntu (or any other Debian-based environment)

First update your Operating System (OS) repository indexes:

```
$ sudo apt-get update
```

Then proceed to install:

```
$ sudo apt-get install git
```

Note: If you use Ubuntu, git is already provided as a base repository on this distribution. If you use another linux, and this does not work for you, please open an issue on this project and use the label *requests*, and I will try to solve and include here for future reference.

2.2.1.2 Apple

Apple is an OS that not use repositories by default, and git is not provided in the main set of software. So you would have three main options to install git:

- Install Xcode (*Preferred*): Apple has a software development kit (SDK) named XCode which provides git among several other software for MacOS and iOS software development. The advantage of using XCode´s git distribution is that it comes adjusted for your OS and is supported by Apple, whcih avoids conflicts and security issues.

  - To install XCode (and git by consequence), open the App Store, and search XCode as a software developed by Apple, and install it. To test, open the Terminal App, and run *git*. You should receive an output of basic description of commands available on git.

- Install Homebrew, and then install Git (if you are a Homebrew user, or uses a lot of *.nix software not supported by Apple it is useful):
  - Download and install homebrew from their website;
  - Install git as:

```
$ brew install git
```

  - In the terminal, run *git* and you should receive an output of basic description of commands available on git.

- Install a standalone version of git (*not recommended*): You can download and install from git´s website a standalone client (I will not be covering this approach, because git will not be updated via this kind a install, posing a security risk and install this at your own risk - and effort!)

2.2.1.3 Windows

Since Windows works with standalone installations, the install follows as this:

- Go to the Git website and download the latest client (download the standalone version, unless you have severe storage limitations on your machine);
- Once downloaded, run the installer with admin permissions and follow the default installation;
- When installed, open PowerShell (PS) (or the command prompt, if you do not have PS installed), and run *git* and hit enter. You should receive an output of basic description of commands available on git.

2.2.2 Setting up your local git client

Once you have your git client up and running, you need to setup the access credentials to connect to a git repository, such as GitHub or GitLab for example.

To do so, run on your bash terminal (or command-line if on windows):

```
$ git config -- global user.name "Your full name"
```

Note: On quotes you need to specify a name (depending on the environment, it should be your full name, or an alias, such as employee number, for example).

This command has a global scope, so all users on your machine would have the same setup.

Then you need to specify an e-mail address (if on GitHub, it is preferred that is linked to your account):

```
$ git config -- global user.email "Your e-mail address"
```

Then check, if all values were set:

```
$ git config --list
```

You should see an output that reflects those previously set name and e-mail values.

2.3 GitHub

2.3.1 Creating an account

2.3.2 Insert GitHub credentials on your local Git

2.3.3 Creating a repository

2.3.4 Retrieving code from a repository

2.3.5 Submitting code to a repository

2.3.6 Conciliating *"conflicts"*

2.3.6.1 Git Merge

2.3.6.2 Git Rebase

2.3.7 Contributing to open-source projects

2.3.7.1 Creating a pull request

Bibliography

Wikipedia contributors. Version control — Wikipedia, the free encyclopedia, 2002. URL https://en.wikipedia.org/wiki/Version_control. [Online; accessed 01-February-2022].