Git & GitHub

In this paper I'm gonna address the need and importance of an VCS(version control system) for a developer. And in the end I'm gonna present the Git tool and one of the most popular Hubs for it, GitHub.

What is Git ?
Git is a version control system that runs on most platforms (Linux, Windows, OS X etc...).It's a distributed system (it does not have a central database, everybody is free to make their own Hub, or store there version there own data locally). It is released under GNU GPL license that allows everybody to use the software in commercial and private use with only requirements being that you need to disclose the source, include a copy of the license in your code, and inform about the majore changes you made (if you did).

It was made by Linus Torvalds (the guy who made the Linux kernel) for the Linux Project .Since then it was used for a lot of projects, like : Android, Arch Linux, Debian, Eclipse, Fedora, Perl, Ruby on Rail, Mint, Gnome, Gimp, and many more.

Why use it ?
There are some versions control system, but the advance of Git is that it's free and open source, being made by the guys who believes in software freedom that was an obvious choice. Some competitors are Mercurial, CSV SVN, and may more.

Having a piece of software that keep track of the changes somebody made in a project is crucial if the size of the project is increasing. With this tracking mechanism we get a variety of tools. Git having a database that mainly stores data, not deleting, once you committed something (in other words decided that you should keep it) you know that you will find it. In fact it's very hard to hide something that you committed. Let's take this scenarios, you have a piece of software that is watched by git with 3 version, let's name them **version1, version2, and version3** if for some reason you wanna go back to **version2** the graph stricture will look something like this :

version1 → version2 → version3→ version2

It will make a symbolic copy of the versions you wanna revert to. This helps with tracking the people who made the change, why, and still allowing to go back to past versions.

Is Git only for programmers ?
The answer is no. People use this types of version control system for many thing, writing documentations, keeping track of changes in a book, or people who design logos and other things use it to keep track of the changes.

It is used in many fields of study.

What do I need to know to use Git?
Usual, not much, but in this paper I'm going to talk about using the Command Line to interact with

Git, but there are many graphical tools.

Why use the command line if there are graphical tools ?
This is a very debated topic. There are two large categories of interfaces. CLI (Command line interface) and GUI (graphical user interface) . A lot of GUI are just graphical wrappers of an CLI. This is because a developer works better with commands for several reasons :

1. It's easy to test, when you work with commands you know that if you have a special command and a given input you can check if the output it's okay. You can have big chains with commands and input and check the output automatically;
2. It gives more flexibility, when you need to configure something you just write 2-3 more letters, in a CLI mode, but in a GUI, you need to hunt down , panels, and check-boxes to get the desired outcome ;
3. It offers the ability of pipelining, you can chain commands and make the output of one be the input of the other, this allows for modularity, instead of having a big program that does everything you have more little programs that work together to obtain the desired behavior. We can take as an example 3 simple linux commands
   - **top** – shows all the process running
   - **grep** – searches for a given pattern
   - **pkill** – kills a process

   We can chain this commands, to get all the processes, search the one we want to stop, and stop it.

But of course not everybody is willing to learn a CLI because even if they have a pattern and if you worked with some the others will be familiar to a center degree, it's a steep learning curve.

Command line prerequisites and quick overview

I'm going to make a brief walk through the commands we need to know .

I. Moving in and out of folders
In a terminal (the place where you write commands) you have the same file structure as in a normal computer.

Biography :

- http://www.howtogeek.com/180167/htg-explains-what-is-github-and-what-do-geeks-use-it-for/
- https://sfconservancy.org/supporter/
- https://git-scm.com/book/en/v2/Getting-Started-Git-Basics
- https://ro.wikipedia.org/wiki/Git
- http://choosealicense.com/