**Introduction**

Firstly, github is a website for hosting projects. With the help of git, projects can be uploaded to the [github](https://github.com/). However, github is used for coordinating multiple people working on a project. It takes note of all the changes made on the project and the date the changes were made. Git is not the only version control system but it is among the most used. It doesn’t require the use of any programming language it involves the use of simple codes which takes a lot of practice to master. This software works on three main operating systems which are Windows, Mac and Linux but it is most compatible with the Linux operating system. This is so because the originator of git is the same as that of Linux. So it has more commands than the others.

**Who uses Git?**

Basically git is for coders. It is for people who create programs. The process of creating programs or software has a lot of aspects. For example, a game it has the graphical aspect, the code aspect, strategy aspect etc. so a single person cannot do that on his own so with the use of git and github different people can collaborate to do that. Then other people can make some changes to the file and send it back to the organization. To cut it short git is used by programmers not only software but all the types.

**Why use Git and the commands it used in a Life Example.**

Git allows a team of people to work together, all using the same files. And it helps the team cope with the confusion that tends to happen when multiple people are editing the same files. There are many ways it can be set up and configured but I’ll explaining only one. For example, when a new employee joins a software building organization, if that organization is on github they will told to download all the files from Github, which is an online server they are all connected to. So he has *his* local version of the files and the other workers that came prior to him their local version even their boss has *his* local version, etc. So when he makes changes to some files, he will go through the following process in git. Firstly, the [git pull](https://git-scm.com/docs/git-pull) command. This pulls the latest changes down from github. After editing the files or creating new ones he runs the [git add](https://git-scm.com/docs/git-add) command. Which adds all of his local changes to git, so that git knows about them. The dot after add specifically means to add *all* the changes he has made; e.g. new files he has added to his local folder or changes he has made to existing files. If he wants, he can add only specific files, e.g. git add <file name.ext>. Then he now writes a comment about the changes he just made with this command: [git commit –m “The comment”](https://git-scm.com/docs/git-commit). Finally, he uploads his changes to the server. To be specific [github](https://github.com/). He does that with the command [git push](https://git-scm.com/docs/git-push). Now, when his colleagues do a git pull command they will get his changes. There are many other commands. But the most useful one is branching which is done with the command [git checkout –b <branch name>](https://git-scm.com/docs/git-branch). Let's say his team is working on a code for an Asteroids game, and he gets the idea for making spinning asteroids. This will involve making some major changes to the existing asteroids code, and he’s a little scared to do that. No worries, he can just make a branch with the above command. To break it down you can type [git branch <branch name>](https://git-scm.com/docs/git-branch) and to switch to a particular branch you type [git checkout <branch name>](https://git-scm.com/docs/git-branch). Now he spends a couple of hours in his branch, doing whatever coding he needs to do, not worrying about messing things up, because he’s in a branch. Let’s say there is a problem with the high scores he can open it on his local computer by pulling it out with the git pull command. When he’s done with fixing it he’ll add it with the command git add and then commit it with the command [git commit –m “the comment”](https://git-scm.com/docs/git-commit). And then push it back with the command git push. The last command I’ll talk about is the [git merge](https://git-scm.com/docs/git-merge) command. This is to merge all commits.