**Git Commands & Terminology**

1. **Repository:** A repository is the most basic element of GitHub. It is basically your folder for the entire project. This folder will contain all you document, project file and each of its revision history. The repository can be public, so multiple collaborators can work on the same project under one repository.



Simple representation of a repository where the initial commit is **C0** and the following commit is **C1**



Visual representation of a project repository with 3 commits.

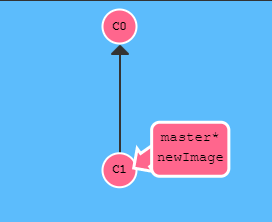
1. **Clone:** A clone is a copy of the repository that is basically saved on your computer instead of on a website’s server. In this copy of your current repository you can edit files in your preferred editor and use Git to keep track of the changes made to the repository without having to being online.



Once the command, “**git clone**” is implemented, it creates a remote repository of a working project

Original git repository with two commits. The initial commit is **C0** and the following commit is **C1**

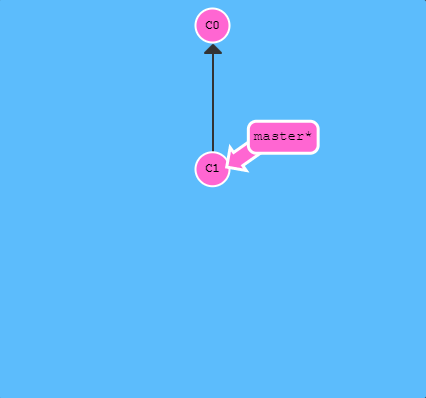
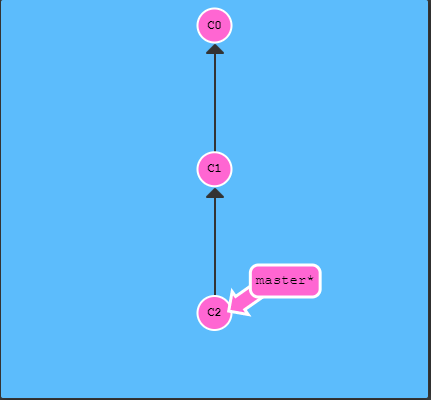
1. **Fork:** A fork is a personal copy of another user’s repository that is basically in your account. This allows you to easily make changes without affecting the original repository. Additionally, this is also attached to the original repository. So, you can essentially submit a pull request to the author/owner of the repository to make update with your new implemented changes.
2. **Branch:** A branch is essentially a parallel version of your repository. This is embedded within the repository but does not affect the primary master branch for the repository. When the changes that needs to be done or updated are completed on the branch, you can merge that branch into the master branch. This will basically publish all the changes to the master branch.



Once the command, “**git branch newImage**” is implemented, a new branch is created which now refers to the commit C1

Original git repository with two commits. The initial commit is **C0** and the following commit is **C1**

1. **Commit:** A commit is an individual change to a file. Every time you save in Git, it essentially created a unique ID for the file. It essentially records a snapshot of all the files in your directory. This allows the user/owner of the repository, to keep track of what changes were made in the file when and by who. (Please see listed example below)

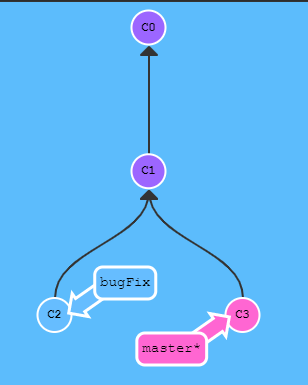
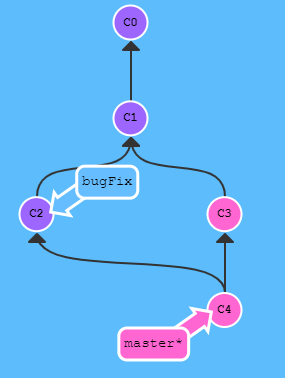


Once the command, **“git commit”** is implemented, it makes changes to the repository. It has a new commit **C2** to the repository and saves its.

Original git repository with two commits. The initial commit is **C0** and the following commit is **C1**

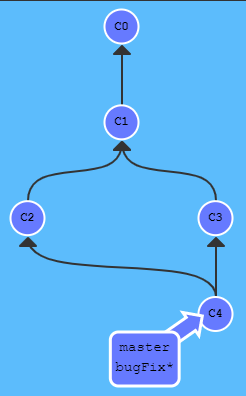
1. **Merge:** Merging essentially takes changes from one branch within the repository and applies them into another. This often takes place a pull request within the repository. This pull request can be done with the GitHub web interface if there are not conflicting changes on the branches.

**Merging bugFix branch into master branch**



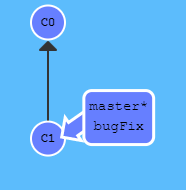
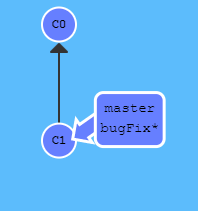
“**git merge bugFix**” command applied

Original repository



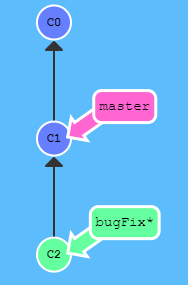
To achieve this step, **“git checkout bugFix”** was applied. This step was performed so, the bugFix branch can be selected. This was than followed by **“git merge master”** command to merge master int bugFix

1. **Checkout:** The checkout command is used when you need to navigate between the created branches in the repository.



**“git checkout bugFix”** command applied to put the asterisk on bugFix from master.

Original repository with two commits. C0 and C1. When command “git branch bugFix” is inserted, it created a new branch on the C1 commit



**“git commit”** command applied to shift bugFix to C2