**Introduction:**

There are many programmers working on the same software development from different corners of world.

Then how to manage the codes? How they will make other understand that what changes they have made? How to commit the codes and maintain different versions? How to merge the codes?

To solve these problems, GIT came into Development world.

GIT is an outstanding Source Code Management [SCM] and Distributed Version Control System.

GIT was created by Linux Torvald, the person who developed Linux kernel.

Obviously, it is an open source tool where every programmer can contribute to build a piece of software from anywhere in the world.

GIT has many features. It may have multiple branches.

A developer can write codes after creating his own branch in local system and merge it with the master branch or other branch of the remote GIT repository.

How to use GIT, the below link can be followed -

<https://opensource.com/article/18/1/step-step-guide-git>

GIT has multiple features to manage the source codes. Today the topic is to discuss about GIT Cherry-Pick which is an important feature and command in GIT.

**git cherry-pick:**

Imagine, a project work is going on to write a script on the history and evolution of cell phones. So, there are many people working on the same project and all are working separately. However, at the end, everyone’s script will be complied together.

Now, member A is writing about Apple phones and suddenly realize it can be better. So, he informed the matter to the other team members who are working on the same project.

Another member X told he is writing script on Android phones and asked member A to have a look.

Then member A looked into teammate’s script and found that some of the portions are same with some changes which are really good.

Therefore, he cherry-picked those changes and pasted in his own script.

This is the same what cherry-pick called in GIT in the matter of software coding industry.

git-cherry-pick is powerful git command and cherry picking is a process to pick up a commit from a branch and apply to some other branch. In simple word, there can be multiple branches where developers commit their codes. Now, one developer supposed to commit his codes in branch A, however he committed the codes in branch B by mistake. That time cherry-pick may switch back the commit to correct branch.

Use below command [in Unix system] to know different options for git-cherry-pick –

$*man git-cherry-pick*

Syntax for cherry-pick command –

*git cherry-pick [–edit] [-n] [-m parent-number] [-x] <commit>*

**When we use git cherry-pick:**

git-cherry-pick is a useful tool however not a best practice for all the time. git-cherry-pick can be used in the following scenarios –

* To make it correct when a commit made in a different branch accidentally
* Preferable traditional merges
* To apply the changes in an existing commit
* Duplicate commit
* Bug fixing

**How git cherry-pick works:**

A defect found in code in the Production environment and a fix needs to be implemented. Once the change implemented and defect is fixed, now it is time to bring this code change back in Development environment so that the defect will not occur again and again in Production environment in future.

The first option is a simply git merge and it is an ideal solution if it works. However, there are other changes done in Production environment also and those can not be bringing back to Development environment while merging. And in that case, cherry-pick is the correct option.

Cherry-pick brings that commit which was made for bug fix only. It does not pick the other commits.

Here is an illustration –

Fig 1 - G and H are Production branch commits. A to F Development branch commits. A problem is found on the Production branch. A fix is developed in H commit which needs to applied in Development branch but commit G is not required to be applied.

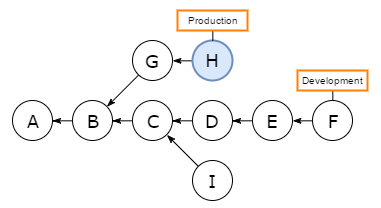
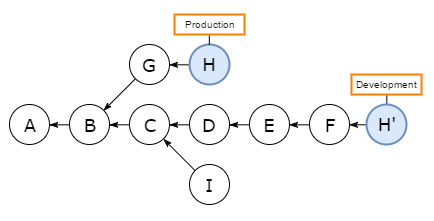
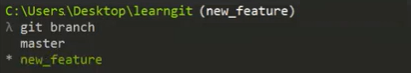


Fig 2 - Now commit H is cherry-picked on Development branch and resulting commit is H’. Commit G changes are not included on Development branch.



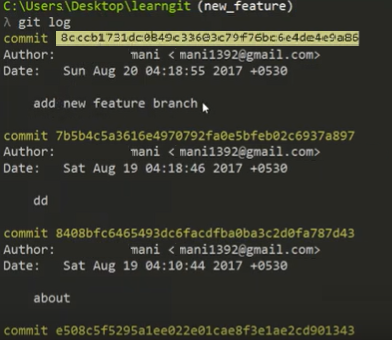
**How to use git cherry-pick with example:**

Suppose we have two branches [master and new\_feature] [command used to see the branch - *git branch*]



We made the commit [Highlighted] in new\_feature branch by mistake. [command used to see the committed logs – *git log*]

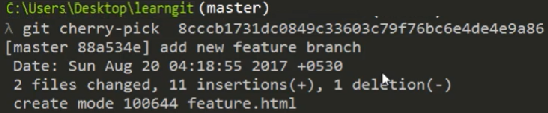
However, it supposed to be in master branch only. First copy the highlighted SHA in a notepad.

****

Now we will use git-cherry-pick command to move this commit to master branch however before that we need to switch to master branch [command used to switch the branch – *git checkout <branchname>*]



[command used – *git cherry-pick <commit id>*] [same SHA should be paste which was copied to notepad earlier with git cherry-pick command]



Now we can see the same commit is available in master branch [command used – *git log*]

For more example of git cherry-pick commands, please see the below link –

<https://git-scm.com/docs/git-cherry-pick>

**Recommendations:**

Three things need to remember while using cherry-pick and working in a team –

* **standardize commit message**

It is better to use standardize commit message and -x if we cherry-pick from a public branch –

*git cherry-pick -x <commit-hash>*

It will avoid merge conflict in future.

* **copy over the notes**

Sometimes some of cherry-pick has notes and when we run cherry-pick, the notes do not get copied. Therefore, it’s better to use –

*git notes copy <from-commit-hash> <to-commit-hash>*

* **cherry-pick multiple commits when they are linear only**

Say, we want to cherry-pick multiple commits like G,H [Fig 1], if they are linear, then only use the below command –

*git cherry-pick G^..H*

**Conclusions in recommended sequences:**

Suppose we want to pick up specific commit from different branch and apply to the current branch, here are steps recommended –

1. Find the commit hash that needs to be cherry-picked first.
2. Go to destination branch.
3. *git cherry-pick -x <commit-hash>*

Resolve the conflicts if they happen. If there are notes in the original commit, those needs to be copied -

git notes copy <original-commit-hash> <new-commit-hash>

Hope you enjoyed the article about git cherry-pick. Thank you.

----------------------------------- XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX --------------------------------------