Experiment No. : 06 Date: Page No.: 13

Name of the Experiment : Git Commands.

Git Commands

GitBranch , git checkout , git pull , git push , git merge.

1. gitbranch (or) create Branch :

-> you can create a new branch with the help of the git branch command.

Syntax: $ git branch <branch name>

git checkout :

-> Git allows you to switch between the branches without making a commit.

-> you can switch between two branches with the 'gitcheckout' command

Syntax: $ git checkout <branch name>

git merge or merge Branch :

-> git allows you to merge the other branch with the currently active branch.

-> you can merge two branches with the help of "git merge" command.

Syntax : $ git merge <branch name>

Experiment No. : 06 Date: Page No.: 14

Name of the Experiment : Git commands

git pull :

-> This command fetches and merges changes on the remote server to your working directory

git push :

-> This command sends the committed changes of master branch to your remote repository.

Experiment No. : 7 Date: Page No.: 15

Name of the Experiment : Branching and merge requests

Edit a File, commit and push. Create a branch. make changes, push and create merge request

Step-1 : initially create a repository and then go for add file option. click on create new file

Step-2 : In this step edit the file and then go for commit changes and select commit directly to the main branch and then commit changes

Step 3 : After clicking commit change then directly go back and then click on new branch as shown below

Step 4 : In this step specify the new branch name as Feature branch as shown in the below figure and click on create new branch.

Step 5 : Then go back the current repository and click on master icon where u can find all branches present click on the feature-branch.

Step 6 : on clicking on the feature-branch u can find the files present in the feature-branch either u can edit the files already present or elese u can add the files to feature branch, Then go for commit changes.

Experiment No. : 7 Date: Page No.: 16

Name of the Experiment : Branching and merge request

Step 7 : on going repository you can find the below figure and there you compare and pull request and click on that option.

Step 8 : upon scroll down you can find the create pull request option where you can click on that option for pull the changes.

Step 9 : In this last step click on merge pull request so that you can merge the files from feature branch to the master branch.

Step 10 : finally the feature-branch is merged with master branch.

Experiment No. : 8 Date: Page No.: 17

Name of the Experiment : Cherry-pick

Cherry-Pick command

step-1 : Opening the git bash and creating a new project named Sample and initialising the repo using the git init command.

step 2 : creating a '---.txt' file using vi command to the project let s say an index file and add it to our Sample project and make a commit and write a commit message before pressing the Enter one can check your commit by git log command easily.

Step 3 : Now assume we have 2 versions, So create 2 different branches by using the git branch command and move to a branch, let say 2 by using git checkout command.

Step 4 : Now Suppose you want to work on Some new feature so creating and adding a new features File lets say feature.txt using vi and add command. respectively as Sho Then commit your changes with a commit message
one can check your commit by git log command as shown be

Experiment No. : 8 Date: Page No.: 18

Name of the Experiment : Cherry-Pick

Step 5 : Now suppose we found a bug in our feature and we came to know that this same bug is also present in our 1 branch
And now we are trying to fix some bug or issue as shown below by adding a fix.txt file let's suppose and add it to to the current branch i.e 2 and committing the required changes.

Step 6 : Now, we have fixed the bug in branch 2, but we also need to add this fix to our branch 1 also, but we don't want to merge this branch 2 into our branch 1 because the work might still be going on the feature.
Thus, in this scenario, we can cherry-pick this particular commit. To do so, just copy the hash value highlighted in the above diagram, them move to branch 1 by using checkout and then use the command cherry-pick and paste the hash we just copied.
As seen clearly fro, then we notice that previously we have only index.txt before doing cherry-picking but now we have the fix.txt file also in our 1st -branch
Now if we try to check git log --oneline, we will be able to see that commit also came in the branch 1

Experiment No. : 9 Date: Page No.: 19

Name of the Experiment : Resolve a merge conflict

Resolving a merge conflict on Github

Step 1 : Initially open the repository which contains the files in master branch.

Step 2 : Then click on the file and do some changes in the file (file.txt)

Step 3 : commit the new changes to new branch and name it as subbranch2 and click on Propose changes button.

Step 4 : Then go back to the sample repository you can see the compare and Pull request from subbran2

Step 5 : Again in this step go to same file (file.txt) and so some changes as below and click on commit changes.

Step 6 : click on commit directly to the master branch and go for commit changes.

Experiment No. : 9 Date: Page No.: 20

Name of the Experiment : Resolve a merge conflict

Step 7 : Then again go back to repository and click on compare and pull request, you can find an conflict as cant automically merge.

Step 8 : Next step is scroll down the page there is an pull request option. Click on pull request

Step 9 : Next after clicking on pull request next window appears as below where u can see the resolve conflicts, click on resolve conflicts.

Step 10 : Next step delete the text enclosed in <<<< and >>>> Part and click on mark as resolved button on the right hand side.
And then go for commit merge button at the right hand side

Step 11 : After commit merge a window will pop up as below with merge pull request click on merge pull request and go for commit merge.