**General Question/ Answer –**

**What is GIT?**

Git is a version control tool

**What is Pull?**

By pull command we can change our local repository as remote

**What is Push?**

By push command we can change our remote repository as local

**What is commit?**

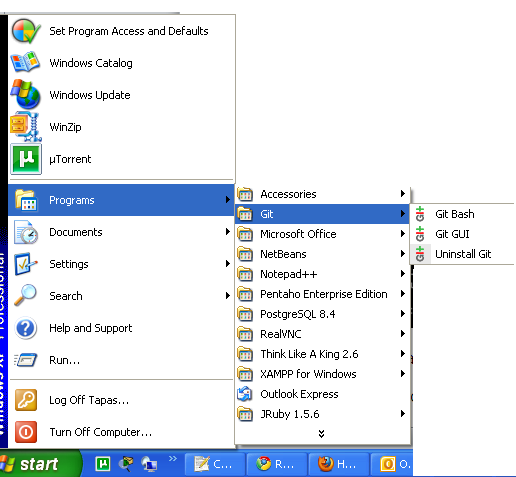
Commit is a comment to know why the changes are made. We have to commit every time, we try to made changes to our remote repository.

**What is Clone?**

By clone command we can create a new repository in our local system as remote repository

**How to check GIT is installed in the system or not?**

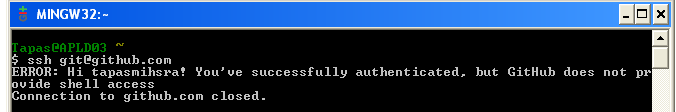
Click on the Start Button>Programs>Git>



**How to test SSH key?**

To test the ssh key we have to provide the following command in Git Bash-

$ ssh git@andolasoft.co.in

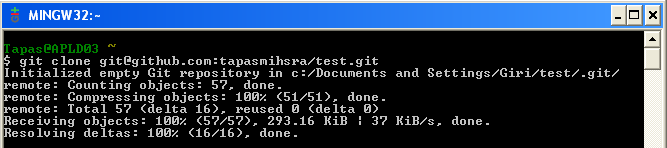


**How to clone a repository?**

“Clone” means to copy the web repository to our local system.

To clone a repository we have to put the following command in Git Bash –

$ git clone <repository URL>



To get the repository URL we have to login to our web repository account/ contact to the Project Lead

**How to view the local repository?**

By default git clones the repository to the following path- C:\Documents and Settings\<your user>\<your project>

If we want to clone the repository to a different place then we have to give the following command in Git Bash and then after we have to clone the repository.

$ cd <desire location>

**How to get updated files from web repository?**

If anyone made changes in your remote repository and you want to reflect the same to your local repository then have to issue the following command

$ cd <repository location>

$ git pull

**How to send updated files to web repository?**

If you made some changes in your local repository and want to reflect the same to your remote then have to issue the following command

$ cd <repository location>

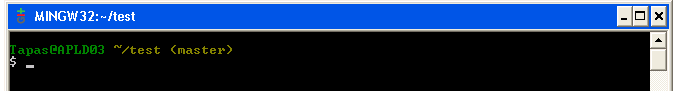
$ git add <file name>/<.”to add all files”>

$ git commit –m ‘<your comment>’

$ git push

**How to know in which branch you are working?**

The branch name is displayed in the Git Bash after the repository name like this

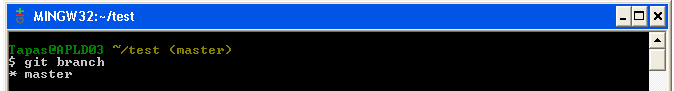


Here we are in <master> branch.

**How to view/list branches?**

To view how many branches are there we have to issue the following command-

$ Git branch -a



Here we have one branch named \*master and this is the default branch of our repository.

**How to switch to a different branch?**

To switch between two branches we have to issue following command-

$ git checkout <branch name>

After giving the command the desired branch will reflect on the Git Bash instead the present branch

Here we can pull/ push our changes to the desired branch

**How to merge two branches?**

Basically we merge two branches to exchange the modified files between them.

To merge two branches we have to give following command

$ cd <repository location>

$ git checkout <branch name “updated” >

$ git pull

$ git checkout <branch name to be merged >

$ git merge <branch name “updated” >

$ git push origin < branch name to be merged >

**Way we should follow during development using GIT:**

1. Clone the repository

git clone <repository URL>

1. Create a new local branch from you production or testing branch

git checkout –b featurebranch

1. Changes should made to the new branch

vi filename

1. Push the changes to remote

git add filename

git commit –m “Changes made for xyz module”

1. Merge the changes to the QA/testing branch

git checkout QA\tesing\staging

git merge featurebranch

git push

1. Delete the new branch

git –d featurebranch

1. Merge the change to Master/live/production branch

git merge QA\tesing\staging

**How to merge a particular commit to a branch using cherry-pick:**

First of all, use git log or the awesome TurtoiseGit tool to see exactly which commit you want to pick. An example:

dd2e86 - 946992 - 9143a9 - a6fd86 - 5a6057 [master]

\

76cada - 62ecb3 - b886a0 [feature]

Let’s say you’ve written some code in commit 62ecb3 of the feature branch that is very important right now. It may contain a bug fix or code that other people need to have access to now. Whatever the reason, you want to have commit 62ecb3 in the master branch right now, but not the other code you’ve written in the feature branch. ~ Here comes git cherry-pick. In this case, 62ecb3 is the cherry and you want to pick it!

git checkout master

git cherry-pick 62ecb3

That’s all. 62ecb3 is now applied to the master branch and commited (as a new commit) in master. cherry-pick behaves just like merge. If git can’t apply the changes (e.g. you get merge conflicts), git leaves you to resolve the conflicts manually and make the commit yourself.

*Cherry picking a range of commits*

In some cases picking one single commit is not enough. You need, let’s say three consecutive commits. cherry-pick is not the right tool for this. rebase is. From the previous example, you’d want commit 76cada and 62ecb3in master.

The flow is to first create a new branch from feature at the last commit you want, in this case 62ecb3.

git checkout -b newbranch 62ecb3

Next up, you rebase the newbranch commit --onto master. The 76cada^ indicates that you want to start from that

git rebase --onto master 76cada^

The result is that commits 76cada through 62ecb3 are applied to master.