**Introduction:**

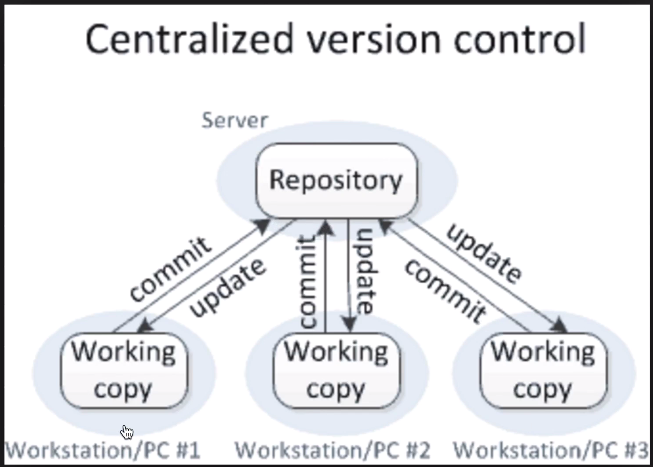
1. What is Git
2. What is Git HUB
3. Is Git related to Git HUB
4. A simple workflow
5. Free end

1.Git - VCS(version Control System): which will maintain details of all the changes made on file which we use. For this we use the VCS . and Git is type of VCS. In simple word this is used to track the changes made on file or folder. The VCS is also use to . **Version control** is a **system** that records changes to a file or set of files over time so that you can recall specific **versions** later. For the examples in this book you will use software **source** code as the files being **version controlled**, though in reality you can do this with nearly any type of file on a computer

There are two type on VCS .  
1. Centralized VCS.   
2. Distributed VCS.

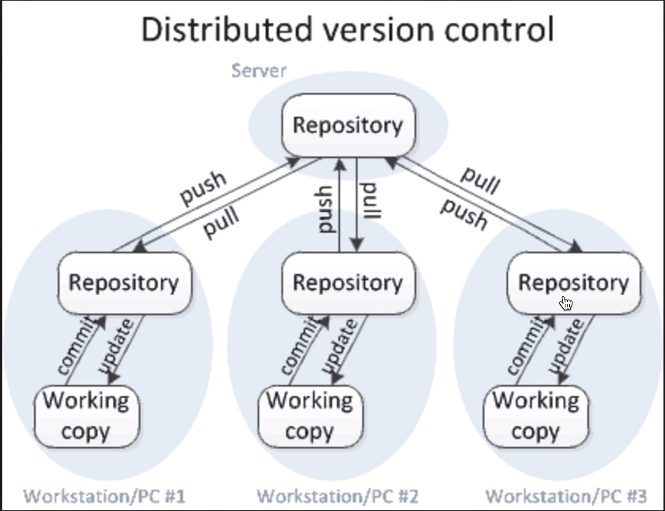
1. Centralized VCS:

Here all the people will be working on a single Centralized Repository as shown in below figure. The drawback of this system is that if anything goes wrong with the centralized repository then it will be very difficult to get the back of the repository. And the people who all are working on the repository have to be online to collaborate their work on Repository.



2. Distributed VCS:

Here the people working on main repository will have a local copy of the main repository on their local systems. The advantage of this system is if anything goes wrong with Main repository it will be very easy to recover the back up from the local repository on the user’s system and no data will be lost. Another main advantage of using distributed VCS is you no need to connected to server all the time. We need to be online only when we need to pull the Main repository work on Local repository and when we need to push the changes to the Main repository. Remaining time we can work offline on local repository. Which is shown in the below fig. The Git is a Distributed VCS type and it is a free end open source



2. **Git HUB**:

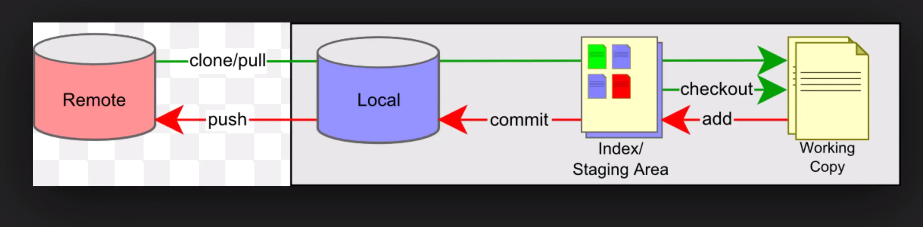
Git HUB is a website which is used to upload your repositories online. Andvantages of git HUB re

1. Provides backup
2. Provides Visual interface to your Repositories
3. Makes collaboration easier

**Git !=Git Hub related**

Git and Git HUB are not related to each other. GIT is VCS and Git HUB is a website. We can use the GIT HUB with any VCS it can be Git or any other VCS. Similarly we can use Git with any other online repository management system such as Git HUB , Bit bucket etc.

**Simple work flow of the Git hub:**



The above fig show the simple work flow of the Git HUB. Here the Main repository is placed on a remote server, we can clone or pull the main repository to our local repository on our local system whenever we want to make changes in main repository. After pulling or cloning we we’ll get all the repositories or branches of the repository. And we can take or check out any branch of the local repository and it will create a working copy to make any changes. Once we pulled the repository on the local system we can go offline to work on Local repository in our system. Once we are done with our changes on working copy we can add this copy to the Index or staging area which shown in the figure. This is still not committed. After adding the working copy to the index area we can execute a ‘commit’ command to commit all the changes to the local repository. To save these changes on the Remote repository we have to push the local repository to the remote server.

**Installing Git and creating environment for Git repository:**

1. Download and install Git
2. Sign up and create account on git HUB
3. Add file or folder to git
4. Track and commit changes
5. Add the repository on git HUB

**Steps to download and install the git on MAC:**

1. Check if the git is already installed. To check this on mac type git –version on terminal
2. Download and install git from <https://git-scm.com/download/win>
3. Signup and create account on git HUB. On <https://github.com/>
4. Add your git HUB email and user name to git by using below commands on your terminal

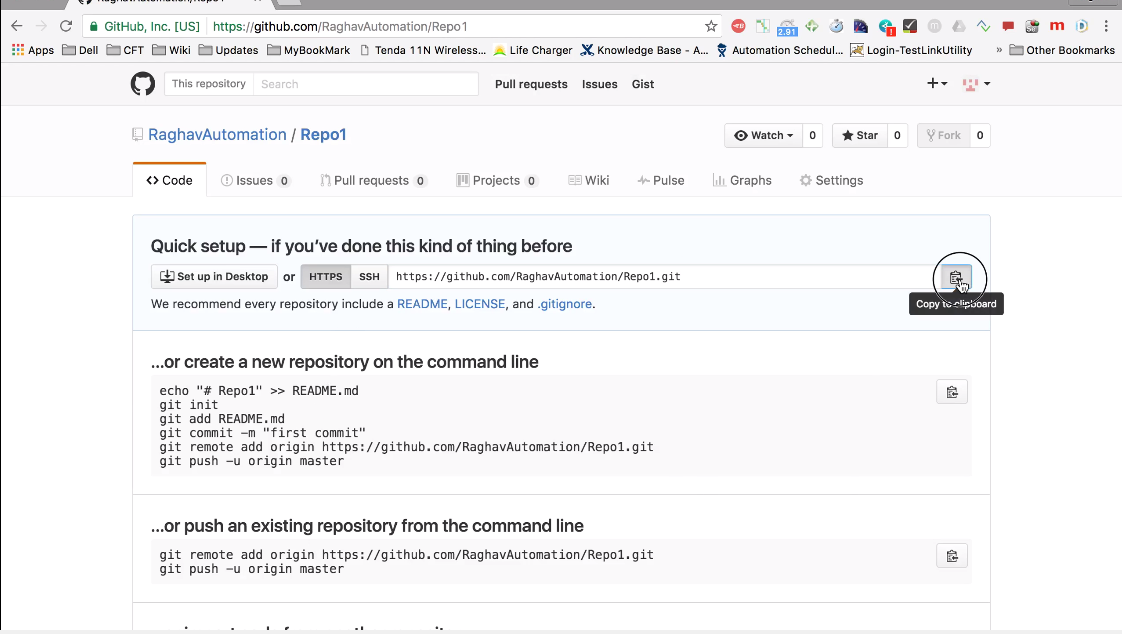
git config –global user.email “[yourGitHub@email.com](mailto:yourGitHub@email.com)” this is an email used for signup on git hub   
git config –global user.name “yourGitHubusername” this is the user name used to sign up on git hub

we are doing this because while committing the changes from the local system it will consider that this is the user who is making the changes on the repository

1. Add file or folders to git to track the changes . go to the location of the folder from the terminal.
2. Commands  
   1. On the terminal go to the location of the folder or project

2. use git init command on terminal  
3. Use default write com.apple.finder AppleShowFiles YES to unhide all the files in the folder

4. Rt. Click on finder and click on re-launch option. We’ll see a git folde inside the folder which we have created  
5. Use git status to get the status of changes made on the folder . Add something to the folder by using ‘touch test1.txt’ and enter which will add the test one text file into the folder. Then use ‘git status’ command to get the status of the folder   
6. To track the file use ‘git add filename.txt’ and then check the status. To add all the text files on the folder use ‘git add \*.txt’. use ‘git add .’ to add all the files on the folder  
7. Use ‘git commit –m” ----“ to commit the changes on the folder . where –m with the string is used for the message displaying purpose.

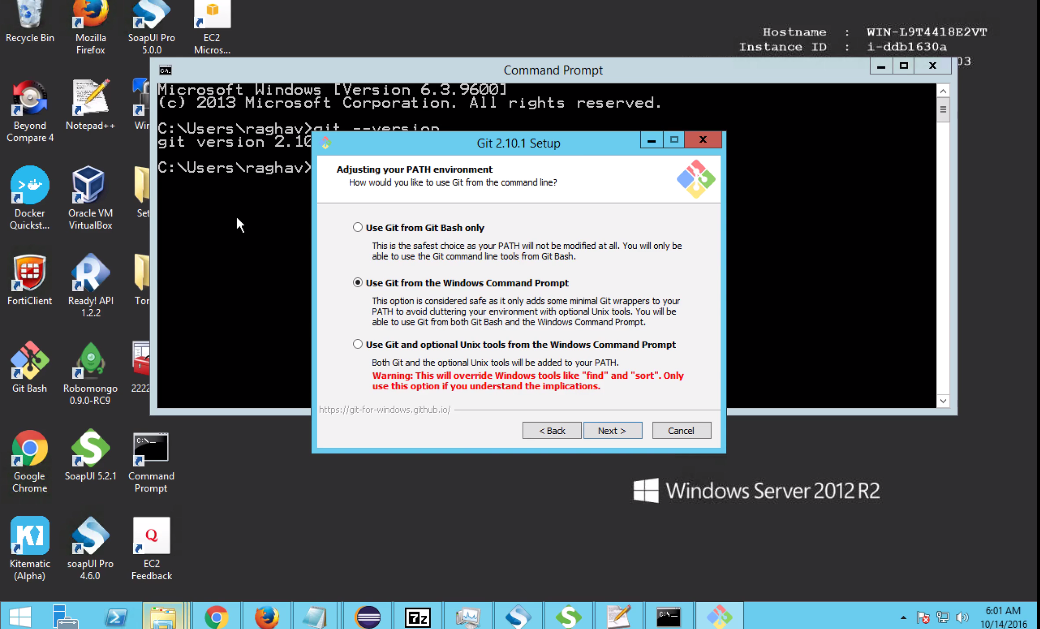
1. Create remote repository and add all files and folders to the repository  
   1. Signing to git hub  
   2. Click on create new repository. It must be public. **Do not select the checkbox “Initialize with readme.txt” while creating new repository it will cause the “error in pushing some reference to the repository” issue.**  
   3. Copy the location of the repository from the git hub which is shown in below pic

4.GO to terminal and type ‘git remote add origin <location of the remote repository from the git hub>’  
5. User git push –u origin master to push the changes on remote repository. Master is a original branch of the repository.  
6. Go to repository on git hub and refresh it . you will see all the changes made on the folder in a repository  
7. User ‘git - -help’ to get all the commands used for git hub with the description

**Steps to download and install the git on windows:**

Better use steps of Mac for windows excluding com.apple.finder AppleShowFiles steps

1. Check if the git is already installed. To check this on mac type git –version on terminal
2. Download and install git from <https://git-scm.com/download/win> . choose the ‘use git from the windows command prompt’ which is shown in the below pic. This will make the user to use all the possible way to use the git commands



The advantage of using git bash instead of command prompt is you can autocomplete the command on git bash and no need to write complete command.

1. Add your project to git. Project can be a folder which holds all the source files of the project.

Commands   
- git config –global user.email “[yourGitHub@email.com](mailto:yourGitHub@email.com)” this is an email used for signup on git hub   
-git config –global user.name “yourGitHubusername” this is the user name used to sign up on git hub  
-git init  
- git status,  
-git add(., \*.\* with extention)  
- git status  
-git commit -m “<your message>”  
- git remote add origin “location of the repository shown in the below pic”  
- git push –u origin master

1. Adding to remote repository
2. Signup and create account on git HUB. On <https://github.com> .
3. Signin to git hub and create a new repository and keep it public

Create remote repository and add all files and folders to the repository

**How To download and install GIT:**1.Go to Below URL  
<https://git-scm.com/>

2. Click on download for windows

Automatically file will get downloaded in exe

3. After successful download install GIT

**Steps to Create repository:**

Every project will have repositories and the code base is maintained in a centralized repository.

1. Go to below URL  
   https://github.com
2. Sign up
3. Sign in
4. Click on new repository
5. Provide project name as a repository name(must be same as the project name)
6. Provide description
7. Choose public repository
8. Select initialize repository and click on create repository(this is done only once in life cycle of the project)

Once repository is created push the framework code into repository.

1. go to any folder rt. click and choose GIT bash

2. use below command the clone of created repository   
 git clone<repository path>  
 **git clone** <https://github.com/Retheshks/cosmic-cubes.git> please ask to your senior to get this link

3. Add newly developed scripts to depository(into the folder created in the drive)

**Steps to add new files:**

1.cd cosmic-cubes(optional)  
 git add –A  
 2.🡪 -A(all Files)  
 3. Before commiting the scrpitr checkout the brach  
 git checkout -b<branch name>  
 git checkout -b ”dev/frameworks”

**Steps to commit the script:**

1. Git config –global user.email [retheshks78@gmail.com](mailto:retheshks78@gmail.com)
2. Git config –global user.name “Retheshks”
3. Git commit -m<message>  
   git commit -m “FrameWork”
4. Push the commited thing to github
5. Git push origin <Branch>  
   git push origin dev/framework  
     
   life cycle of git hub

Clone

Added new files

Create a branch and checkout

push

commit

**review**

Create PR

merge

Steps to create a Pull Request(PR)

1. Go to GITUHB.com and signing
2. Go to repository
3. Click on compare and create pull request
4. Provide description of changes and add previewers
5. Click on pull request button   
   note: every new pull request will have a unique number

Steps to merge pull request

1. Go to repository ad click on pull request tab
2. Choose a pull request and click on it
3. Click on merge branch
4. Click on confirm merge

Automatically the branch is merged to master

Steps to

1. Take a clone of repository
2. Identify the missing page objects for a test
3. Develop missing page objects
4. Import the project to eclips
5. Add testscript   
   git add  
     
   url to get the project from the git hub is as below  
   <https://github.com/Retheshks/cosmic-cubes>

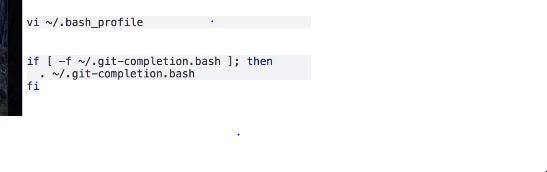
**How to enable git commands auto complete feature on mac OS:**

For autocomplete:

Steps:

1. Go to terminal
2. Put git-completion.bash script to your home dir and use below command



1. Add below command to the terminal. Add following command to .bash\_profile. This tells .bash to run the git-completion bash script if it exists  
   

**Branches and Merging in git hub:**

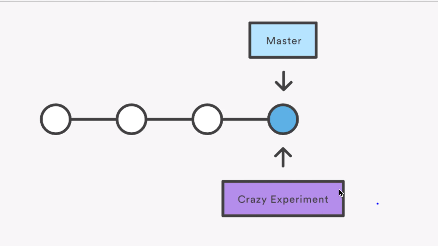
1. What are branches
2. How to create branches
3. How to checkout the branch
4. How to merge the branch to master branch
5. How to delete branch(local and remote)

**Branch:**

When we do some changes our code or adding some new application in our project it is not valid that we can directly make those changes into the main branch or master branch . so to solve this problem we can create our own branches and add those changes into it and after testing and validating the branch and then will merge our branch to the main or master branch. Process of branching and merging on window and MAC are same.

Steps to create branch:

1. Create branch by using the below command

git branch “branch name” 

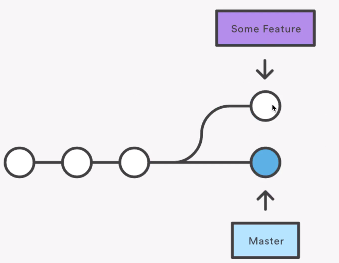
1. Checkout new branch

git checkout “new branch name” .

this will switch the control to new branch from the master branch.

1. Make the changes in the file and add the file to the new branch by using

git add “file name”



1. Commit the changes made on the new file by using

git commit -m ”message”

1. Push the new branch to the repository

git push –u origin “new branch name”

After this go to github.com and refresh the repository you will see 2 branches available in your repository , one is master branch and the other one is the NewBranc h you have created

1. Checkout to the master

git checkout master .

this will switch the control to the maser branch from the new branch.  
you will not see the new file created in master because it has been hided for the master branch and only can be seeing the New Branch created

1. Merge new branch to master branch

git merge “new branch name” .

when you are merging the new branch to the master branch we have to checkout the master branch.

1. After merging go to git hub website and refresh the repository , you will not see the new file because we have not yet pushed the changes to the master branch of our repository
2. Push the master branch to the repository by using the following command

git push -u origin master

**Delete the branch :**

Steps:

1. Use the command below to delete the branch from the local repository

git branch -d “branch name”.

This will only delete the local repository

1. Use the below command to delete the branch from the remote repository

git push origin –delete ”branch name”.

This will delete the branch from the remote repository