Pareen Desai

Assignment 2

2/10/2020

Q U EST IO N S

1. What are the advantages of using Git?

* Keeps track of changes to the code
* Can revert back to old version of the code
* Can test changes to code without losing the original
* It synchronizes code between different people

1. What language is used in Git?

* Ruby, python, haskell

1. What is the meaning of “Index” or “Staging Area” in Git?

* It is the preview of the next commit. The stae between working directory and repository.

1. What is the process for creating a repository in Git?

* git init

git add

git commit

1. What is ‘head’ in Git and how many heads can be created in a repository?

* By default, there is a **head** in every **repository** called master. A **repository can** contain **any** number of **heads**. At **any** given time, one **head** is selected as the “current **head**.”

1. Why do we need branching in Git?

* **Branches** help you organize the workflow more efficiently and rather effortlessly. You can keep working on your **branch** regardless of the work that is happening in other **branches**. Branches are also good when you want one person to work on something without changing the master code.

1. Write a way to create a new branch in Git?

* git checkout -b <branch-name>

1. How do you define a ‘conflict’ in Git?

* A **conflict** arises when two separate branches have made edits to the same line in a file, or when a file has been deleted in one branch but edited in the other.

1. How to resolve a conflict in Git?

* git status gives you where the conflict is and you can chose to fix the problem or accept one of the edits over the other.

1. What is the function of ‘git config’?

* The git config command is a convenience function that is used to set Git configuration values on a global or local project level. The git config command is a convenience function that is used to set Git configuration values on a global or local project level. The git config command is a convenience function that is used to set Git configuration values on a global or local project level.it is a convenience function used to set git configuration values on a global or local project level.

1. What is Git fork?

* **A fork** is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project. Most commonly, **forks** are used to either propose changes to someone else's project or to use someone else's project as a starting point for your own idea.

1. Difference between fork, branch and clone?

* Forking is done on the GitHub Account while Cloning is done using Git. When you fork a repository, you create a copy of the original repository (upstream repository) but the repository remains on your GitHub account. Whereas, when you clone a repository, the repository is copied on to your local machine with the help of Git.
* Forking is a concept while cloning is a process. Forking is just containing a separate copy of the repository and there is no command involved. Cloning is done through the command ‘***git clone***‘ and it is a process of receiving all the code files to the local machine.
* A **fork** is really a Github (not Git) construct to store a **clone** of the repo in your user account. As a **clone**, it will contain all the **branches in the** main repo at the time you made the **fork**. ... These commits are pulled from either my **fork** or my **branch** to the main repo.

1. What's the difference between a "pull request" and a "branch"?

* A **branch is** just a separate version **of** the code. A **pull request is** when someone take the repo, makes their own **branch**, does some changes, then tries to merge that **branch in** (put their changes **in the** other person's code repository).

1. What is the difference between "git pull" and "git fetch"?

* **git fetch** is the command that tells your local git to retrieve the latest meta-data info from the original (yet doesn't do any file transfering. It's more like just checking to see if there are any changes available). **git pull** on the other hand does that AND brings (copy) those changes from the remote repository

1. How to revert previous commit in Git?

* If you want to **revert** the **last commit** just do **git revert** <unwanted **commit** hash> ; then you can push this new **commit**, which undid your **previous commit**. To fix the detached head do **git** checkout <current branch> .

1. Explain the advantages of Forking Workflow

* The main **advantage** of the **Forking Workflow** is that contributions can be integrated without the need for everybody to push to a single central repository. Developers push to their own server-side repositories, and only the project maintainer can push to the official repository.

1. Difference between HEAD, working tree and index, in Git?

* **Working trees**: They are nothing but the files that you are currently **working** on. **HEAD**: **HEAD** is a pointer to the branch or commit that you last checked out, and which will be the parent of a new commit if you make it. ... **Index**: The **git** "**index**" is where you place files you want commit to the **git** repository.

1. How to identify if a certain branch has been merged into master?

* git branch --merged master lists branches merged into master

1. What is the use of a Git clone?

* **Usage**. **git clone** is primarily **used** to point to an existing repo and make a **clone** or copy of that repo at in a new directory, at another location

1. What is Git stash?

* **git stash** temporarily shelves (or stashes) changes you've made to your working copy so you can work on something else, and then come back and re-apply them later on.

1. When should I use "git stash"?

* Use it when you need to work on something else before implementing these changes.

1. What is Git stash drop?

* If you no longer need a particular **stash**, you can delete it with: $ **git stash drop** <stash\_id> .

1. What is Git stash save?

* When you **Git stash** or **Git stash save**, **Git** will actually create a **Git** commit object with some name and then **save** it in your repo

1. What README.MD ? What is its purpose? What does MD stands for?

- A **README** file contains information about other files in a directory or archive of computer software. A form of documentation, it is usually a simple plain text file called READ.ME , **README**. TXT , **README**.**md** (for a text file using markdown markup), **README**. ... The file's name is generally written in uppercase letters.

- **README**.**md** is **used** to generate the html summary you see at the bottom of projects. **Github** has their own flavor of Markdown. Order of Preference: If you have two **files** named **README** and **README**.**md** , the **file** named **README**.**md** is preferred, and it will be **used** to generate **github's** html summary

25. How to create repository from command prompt?

- hub **create -d "<commit name>"**

26. What is the function of ‘git checkout’ in Git?

- The **git checkout** command lets you navigate between the branches created by **git** branch . Checking out a branch updates the files in the working directory to match the version stored in that branch, and it tells **Git** to record all new commits on that branch.

27. How can you bring a new feature in the main branch?

- Create a **new**-**branch** Use a separate **branch** for each **feature** or issue you work on. After creating a **branch**, check it out locally so that any changes you make will be on that **branch**. This checks out a **branch** called **new**-**feature** based on master , and the -b flag tells Git to create the **branch** if it doesn't already exist.

28. What is the function of ‘git rm’?

- The git rm command can be used to remove individual files or a collection of files. The primary function of git rm is to remove tracked files from the Git **index**. Additionally, git rm can be used to remove files from both the staging **index** and the working directory

29. What is the function of ‘git stash apply’?

- git **stash apply** takes a **stashed** change and applies it to your current working tree (also leaving it on the “**stash** stack”). git **stash** branch creates a new branch from the same commit you were on when you **stashed** the changes, and applies the **stashed** changes to that new branch.

30. What is the use of ‘git log’?

- A **Git log** is a running record of commits

31. What is ‘git add’ is used for?

- The **git add** is a command, which adds changes in the working directory to the staging area. With the help of this command, you tell **Git** that you want to **add** updates to a certain file in the next commit. But in order to record changes, you need to run **git** commit too

32. What is 'git diff' is used for?

- **git diff** is a multi-use **Git** command that when executed runs a **diff** function on **Git** data sources. These data sources can be commits, branches, files and more. ... The **git diff** command is often used along with **git** status and **git** log to analyze the current state of a **Git** repo.

33. What is ‘git status’ is used for?

- The **git status** command displays the state of the working directory and the staging area. It lets you see which changes have been staged, which haven't, and which files aren't being tracked by **Git**

34. Can we create multiple branch with one command? no

35. what is the command that is used to delete a branch?

- $ git branch -d yourbranch

$ git branch -D yourbranch (deletes evem though merge not happened)

36. What is another option for merging in git?

- Rebase as an **Alternative** to **Merge**. While **merging** is definitely the easiest and most common way to integrate changes, it's not the only one: "Rebase" is an **alternative** means of integration.

37. How to remove a file from git without removing it from your file system?

- git rm –cached mulogfile.log

38. Use of "git rebase" instead of "git merge"?

- **Git merging** applies all unique commits from branch 1 into branch 2 in one commit with final result. On the other hand, **Git rebasing** gets all unique commits from both branches (1 & 2) and applies them one by one. **Git rebasing** rewrites commit history but don't create an extra commit for **merging**.

39. What is a repository in Git?

- **Git repository** is just a file location where you are storing all the files related to your project.

40. Command used to write a commit message?

-m option

41. What does commit object contain?

- The **commit object contains** the directory tree **object** hash, parent **commit** hash, author, committer, date and message.

42. Write one use-case of Github?

43. Name some alternative of Git?

- GitHub, SVN (Subversion), Bitbucket, Perforce, and **Mercurial** are the most popular alternatives and competitors to Git.

44. What is a gist in Git?

- All **gists** are **git** repositories, so they are automatically versioned, forkable and usable as a **git** repository. A **Gist** is a snippet of code hosted by Github that has all of the benefits of a Github repository, but provides them to you in a more lightweight way

45. What is a gist programming?

- **Gist**: Is an additional feature added to github to allow the sharing of code snippets, notes, to do lists and more. You can save your **Gists** as secret or public. Secret **Gists** are hidden from search engines but visible to anyone you share the url with.

46. Name any two Git repository hosting services which are common?

Gitlab, Bitbucket