Tech Educators

Git fundamentals

Contents:

Introduction to git

Git is a version control system used for tracking changes in computer files and coordinating work on those files among multiple people. It is heavily used in the tech industry and is a necessary skill for a developer.

Git Commands

Let's go through some of the fundamental commands for using git.

git clone

The command 'git clone' is used to create a copy of an existing Git repository. You might use this command when you want to get a copy of a project you'd like to contribute to, or when you want to clone your project from GitHub (or another provider) to your own machine (PC/MacBook/Chromebook etc).

```
Unset
git clone https://github.com/username/repository-name.git
```

Replace *username* with the GitHub user's name and repository-name with the name of the repository you want to clone from GitHub. This will clone the repository into a new directory named *repository-name* in your current location.

git add

After you've made changes to files in a Git repository, you use the command 'git add' to include the changes in the next commit (a snapshot of your project at this point in time). This command stages your changes for committing. You can specify certain files or you can add all changes at once.

To add all files use the below command:

```
Unset
git add .
```

To add only one file, use the command below replacing filename with the name of the file (e.g. index.html)

```
Unset
git add filename
```

git commit

The 'git commit' command is used to save your changes to the local repository. It's like setting a checkpoint in the timeline of your project which you can go back to later if needed. Committing also requires a message which describes the changes you made.

```
Unset
git commit -m 'your commit message here'
```

Writing good, clear and concise commit messages is important and something you will get better at with time. The main part to remember is to describe the changes in simple terms, you can see the changes to the code using GitHub so there is no need to go into detail.

git push

'git push' is used to send your committed changes to a remote repository on GitHub (or another hosting service) so others can see and access them.

```
Unset
git push
```

git status

`git status` is a very handy command when working with Git. It provides crucial information about the current state of your repository. Let's break down a possible output you might see when running `git status`.

Here is an example of terminal output after running `git status` followed by a breakdown of each part:

```
Unset
On branch main

Your branch is up to date with 'origin/main'.


Changes to be committed:

  (use "git restore --staged <file>..." to unstage)

      new file:   test.txt


Untracked files:

  (use "git add <file>..." to include in what will be committed)

      test2.txt
```

On branch main

This tells you that you are currently working on the 'main' branch. A branch is a parallel version of a repository that does not affect the main project.

Your branch is up to date with 'origin/main'.

Here *'origin'* is the default name Git gives to the server where your repository is stored. *'origin/main'* refers to the *'main'* branch on the origin server. When it says your branch is *'up to date'*, it means that your local copy of the branch matches the version on the server.

If you are *'ahead'* of the origin, it means you have committed changes in your local repository that have not been pushed to the server. Being *'behind'* means there are

changes on the server (perhaps committed by other collaborators) that you have not pulled to your local repository yet.

Changes to be committed

These are the changes that you have made and used `git add` to stage, but haven't committed yet. Staging is like putting changes in a queue before finalising them with a commit. In our case, it's saying that a new file `test.txt` has been staged.

Untracked files

These are files that exist in your directory but have not been added to the repository yet. Git does not keep a history of these files. To start tracking changes in these files, you must use `git add`. In this case, `test2.txt` is untracked.

Overall, `git status` gives you a quick overview of all the changes in your repository that are staged, unstaged, and untracked.