Introduction to Github

Importance of Git:

Git is a distributed version control system that is widely used for software development and collaboration. It is a useful tool for developers because it enables them to:

- **Track changes**: Git keeps a record of every change made to the code, so developers can see exactly what has been modified and when. This makes it easy to revert back to a previous version if needed.
- **Collaborate with others**: Git makes it easy for multiple developers to work on the same codebase at the same time. It tracks changes made by different developers and merges them together, ensuring that everyone is working with the most up-to-date version of the code.
- **Keep a history**: Git maintains a complete history of all changes made to the code, so developers can see how the codebase has evolved over time.
- **Make backups**: Git allows developers to create multiple copies of their codebase, which can be useful for making backups or for creating different versions of the code.

To use Git, you'll need to have it installed on your computer. You can download the latest version from the official Git website (https://git-scm.com/).

Github:

GitHub is a web-based platform that provides hosting for software development projects that use the Git version control system. It allows developers to store and manage their code repositories, track changes to the code, and collaborate with other developers on projects.

Developers can create a new repository on GitHub and then use Git to push their local code to the repository on the GitHub server. Other developers can then clone the repository to their own computers, make changes to the code, and push those changes back to the repository.

To use GitHub, you'll need to create an account and set up a profile. Once you have an account, you can create a new repository.

To create a new repository on GitHub, follow these steps:

- Go to the GitHub website (https://github.com/) and log in to your account.
- Click on the plus icon in the top right corner of the page, and then select "New repository" from the drop-down menu.
- Enter a name for your repository in the "Repository name" field.
- Optionally, you can also add a description for your repository in the "Description" field.
- Select whether you want your repository to be public or private.
- Click the "Create repository" button to create your repository.

Once you have created your repository, you can start adding files to it. To do this, you'll need to use Git.

Common git commands:

1. `git init`: Initializes a new Git repository.
2. `git clone`: Makes a copy of an existing Git repository.
3. `git add`: Adds files to the staging area.
4. `git commit`: Saves changes to the local repository.
5. `git push`: Sends changes to a remote repository.

6. `git pull`: Fetches and merges changes from a remote repository.
7. `git branch`: Lists, creates, or deletes branches.
8. `git checkout`: Changes the current branch or restores files.
9. `git merge`: Merges changes from one branch into another.
10. `git stash`: Temporarily saves changes that are not ready to be committed.

Once you have Git installed, you'll need to configure it with your name and email address, which will be associated with your commits:

$ git config --global user.name "Your Name"

$ git config --global user.email "your_email@example.com"

Creating a repository:

git init

```
gupta@LAPTOP-IKNAOILA MINGW64 /c/IIITK/git commands
$ git init
Initialized empty Git repository in C:/IIITK/git commands/.git/
```

git status

```
gupta@LAPTOP-IKNAOILA MINGW64 /c/IIITK/git commands (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        file1.txt
```

git add <file_name>

git add .

git add -p

```
gupta@LAPTOP-IKNAOILA MINGW64 /c/IIITK/git commands (master)
$ git add file1.txt

gupta@LAPTOP-IKNAOILA MINGW64 /c/IIITK/git commands (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   file1.txt
```

git commit -m “Some message”

git commit -a

```
gupta@LAPTOP-IKNAOILA MINGW64 /c/IIITK/git commands (master)
$ git commit -m "File added."
[master (root-commit) 8220f79] File added.
 1 file changed, 1 insertion(+)
 create mode 100644 file1.txt
```

```
$ git commit -a -m "Adding the key of c"
[master (root-commit) 758797a] Adding the key of c
 1 file changed, 2 insertions(+)
 create mode 100644 README.md
```

```
gupta@LAPTOP-IKNAOILA MINGW64 /c/IIITK/git commands (master)
$ git status
On branch master
nothing to commit, working tree clean
```

git branch -M <branch-name>

```
gupta@LAPTOP-IKNAOILA MINGW64 /c/IIITK/git commands (master)
$ git branch -M main
```

git branch -d <branch-name> : For deleting a branch.

git remote add <remote-name> <https:/name-of -the-repository-link>

```
gupta@LAPTOP-IKNAOILA MINGW64 /c/IIITK/git commands (master)
$ git remote add origin https://github.com/mohit-gupta01/tech-article-git.git
```

git push -u <remote-name> <branch-name>

```
gupta@LAPTOP-IKNAOILA MINGW64 /c/IIITK/git commands (main)
$ git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 218 bytes | 218.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/mohit-gupta01/tech-article-git.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

git push --all: This command pushes all the branches to the server repository.

Some other useful Commands:

git clone <https:/name-of-the-repository-link>

```
gupta@LAPTOP-IKNAOILA MINGW64 /c/IIITK/git commands (master)
$ git clone https://github.com/mohit-gupta01/tech-article-git.git
Cloning into 'tech-article-git'...
warning: You appear to have cloned an empty repository.
```

git pull <URL>

```
$ git pull https://github.com/ImDwivedi1/Git-Example
warning: no common commits
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/ImDwivedi1/Git-Example
 * branch            HEAD       -> FETCH_HEAD
fatal: refusing to merge unrelated histories
```

git merge <Branch-name>

```
$ git merge master
Already up to date.
```

git log: To check commit history.

```
gupta@LAPTOP-IKNAOILA MINGW64 /c/IIITK/git commands (main)
$ git log
commit f398de96fe72e775d4630274099cf18c6bb807b0 (HEAD -> main, origin/main)
Author: Mohit <guptamohit01112002@gmail.com>
Date:   Wed Dec 28 10:40:48 2022 +0530

    File changed.

commit 8220f798bdfbbbec1ac763508c265431d552f269
Author: Mohit <guptamohit01112002@gmail.com>
Date:   Wed Dec 28 10:10:19 2022 +0530

    File added.
```

References:

- https://www.javatpoint.com/git-commands
- https://www.freecodecamp.org/news/10-important-git-commands-that-every-developer-should-know/

By: Mohit Gupta