

Common Git Commands

Setup

Set the name and email that will be attached to your commits and tags

```
git config --global user.name "CampusX"
```

```
git config --global user.email "myemail@gmail.com"
```

Starting a Project with Git

Create a local repo (omit <directory> to initialise the current directory as a git repo)

```
git init <directory>
```

Download a remote repo

```
git clone <url>
```

Eg. - git clone <https://github.com/campusx-official/streamlit-basics.git>

Make a Change

Add a file to staging

```
git add <file>
```

Eg. If there is any file named - app.py in you project folder

```
git add app.py
```

Stage all files - . will add all files in the project folder to stage

```
git add .
```

Commit all staged files to git

```
git commit -m "commit message"
```

-m to give any message along with commit

Add all changes made to tracked files & commit

```
git commit -am "commit message"
```

Basic Git Concepts

main: default development branch

origin: default upstream repo

HEAD: current branch

HEAD^: parent of HEAD

HEAD~4: great-great grandparent of HEAD

Branches

List all local branches.

- Add -r flag to show all remote branches.
- Add -a flag for all branches.

```
git branch -r
```

```
git branch -a
```

Create a new branch

```
git branch <new-branch name>
```

Switch to a branch & update the working directory

```
git checkout <branch>
```

Create a new branch and switch to it -> *-b flag*

```
git checkout -b <newbranch>
```

Delete a merged branch

```
git branch -d <branch>
```

Delete a branch, whether merged or not

```
git branch -D <branch>
```

Add a tag to current commit (often used for new version releases)

```
git tag <tag-name>
```

Merging

Merge branch *a* into branch *b*.

```
git checkout b
```

```
git merge a
```

Merge & squash all commits into one new commit

```
git merge --squash a
```

IGNORING PATTERNS

Preventing unintentional staging or committing of files

Save a file with desired patterns, file name as `.gitignore` with either direct string matches or wildcard globs.

Eg. `.gitignore` file content

```
logs/
```

```
*.notes
```

Explaining above file content:

`logs/` -> will ignore all files in `logs` folder

`*.notes` -> will ignore all files with extension `.notes`

Undoing Things

Move (&/or rename) a file & stage move -> `mv` command

```
git mv <existing_path> <new_path>
```

Remove a file from working directory & staging area, then stage the removal – `rm` command

```
git rm <file>
```

Remove from staging area only

```
git rm --cached <file>
```

Review your Repo

List new or modified files not yet committed

```
git status
```

List commit history, with respective IDs

```
git log --oneline
```

Show changes to unstaged files. For changes to staged files, add `--cached` option

```
git diff
```

Show changes between two commits

```
git diff commit1_ID commit2_ID
```

Show all commit logs with indication of any paths that moved

```
git log --stat -M
```

Show the commits that changed file, even across renames

```
git log --follow [file]
```

Synchronising / Updating

Add a remote repo

```
git remote add <alias> <url>
```

View all remote connections. Add `-v` flag to view urls.

```
git remote
```

Remove a connection

```
git remote remove <alias>
```

Rename a connection

```
git remote rename <old> <new>
```

Fetch all branches from remote repo (no merge)

```
git fetch <alias>
```

Fetch a specific branch

```
git fetch <alias> <branch>
```

Fetch the remote repo's copy of the current branch, then merge

```
git pull
```

Move (rebase) your local changes onto the top of new changes made to the remote repo (for clean, linear history)

```
git pull --rebase <alias>
```

Upload local content to remote repo

```
git push <alias>
```

Upload to a branch

```
git push <alias> <branch>
```

EXAMPLE: Pushing new project to git repo

Project file directory

Project/

static
.idea
.gitignore
logs
app.py
data.db
requirements.txt

.gitignore contents

logs/
*.db
.idea

Create a Repository on github:

- Go to [github](#).
- Log in to your account.
- Click the [new repository](#) button in the top-right. You'll have an option there to initialise the repository with a README file, but I don't.
- Click the "Create repository" button.
- You will get a repo url ending with .git, say for example -
<https://github.com/user-name/example.git>

Order of commands :

- Enter [git init](#).
- Enter [git add .](#) to add all the files
- Type [git commit -m "initial commit"](#).
- [git branch -M main](#)
- [git remote add origin https://github.com/user-name/example.git](#)
- [git push -u origin main](#)

All the files from Project directory will get pushed to example repo, leaving files mentioned in .gitignore file.

Happy Learning!!