



Git / GitHub Workshop

Clarusway



Subject: Git Operations

Learning Goals

- Practice using the Git commands.

Introduction

We've covered a lot of Git concepts, but now it's time to put the concepts in to practice. We'll start with Git commands.

Code Along

Part 1 - Create a local repository

1. Open the terminal (Git Bash for Windows user)
- Go to Desktop and create a directory named "my-github" if you do not have already. And, go to "my-github" directory.

- Create another folder named "git-workshop" and go to "git-workshop" directory.

2. Git configuration

- Configure git with our name and email. This is to identify who has done what on git and github.

- Check the setting

3. Create a local repository

- We can do that by running the "init" command.

- Check the if ".git" folder is created.

Part 2 - Create a remote repository

4. Create a remote repository on GitHub

- Go to your GitHub account and create a repository named "git-workshop".
 - Write a description for your repo
 - select Public
 - add a README.MD file

5. Go to terminal

- Check the connected remote repositories. The 'git remote -v' lists all currently configured remote repositories, which at this point is none.

- connect to remote repository

- Verify the new connection

6. Create a file named "file1.txt"

- check the status of the project folder

- store the change in the local repo

7. upload the changes to the remote repo

```
git push -u origin master
```

- check the files on the github repo. (select master branch in GitHub)

Part 3 - Working with branches

8. Create a new remote repo named "git-workshop-2" in GitHub.

9. Clone the remote repo

- go the terminal
- clone the "git-workshop-2"

- check the files in the "git-workshop-2" and see the README.MD and .git file.

10. Create a file named **test.txt**

11. Create a new branch named new-feature-1.

- See branches

- Switch to new-feature-1

- List the files and check the status of the working directory

- Make some changes in the test.txt file, and check the status

- Store the changes to the repo and check the status

- Add another line to test.txt and store it to the local repo.

- Switch the main branch and see the content of the test.txt

- Merge new-feature-1 branch to main branch.

12. Create a new branch named new-feature-2 and switch to it.

- Create a new file named test2.txt, add a line in it and store the changes to repo.

```
vim test2.txt
```

- Switch the main branch again.

- Create a new file test3.txt and send the changes to local repo.

- Open the file named test2.txt, add a line in it and store the changes to repo.

- merge main branch with new-feature-1

13. RESOLVE THE CONFLICT

- edit the file.
- then commit it.

14. Push the local changes to the remote repository

15. Go and check the remote repository, you will see the new files

16. Go to the terminal and delete the branches named front-end and back-end

- List the all branches

😊 **Thanks for Attending** ✍️

