Git / GitHub Workshop-1

Clarusway

Subject: Git Operations

Learning Goals

- Practice using the Git commands.

Introduction

- We've covered some basic 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.

```
mkdir my-github
cd my-github
```

- Create another folder named "git-workshop" in the "my-github" folder and go to "git-workshop" directory.

```
mkdir git-workshop
cd git-workshop
```

2. Git configuration (if you have already done, skip this part)

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

```
git config --global user.name <your_user_name>
git config --global user.email <your_email>
```

- Check the setting

```
git  config --list
```

3. Create a local repository

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

```
git init
```

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

```
ls -a
```

4. Check your default branch name and If your branch name is "master", change it to "main" then check the branch again.

```
git status
git branch -m main
git status
```

5. Create a file named "file1.txt"

```
touch file1.txt
```

- check the status of the project folder

```
git status
```

6. Stage **file1.txt** and check the status of the project folder again.

```
git add file1.txt
git status
```

7. Store it to the local repository.

```
git commit -m "xxxxx"
```

8. Using Vim editor, create a file named **test2.txt**

```
vim file2.txt
```

9. Stage **file2.txt** and check the status of the project folder.

```
git add file2.txt
git status
```

10. Unstage **file2.txt**

```
git rm --cached file2.txt
```

11. check the status of the directory

```
git status
```

12. Store the changes to the local repeository

```
git add .
git commit -m "xxxxxx"
```

13. List the commits

```
git log
```

14. switch to the first commit

```
git checkout 'first commit ID'
```

15. switch to the last commit.

```
git checkout main
```

Part 2 - Working with branches

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

```
touch test.txt
```

17. Create a new branch named "new-feature-1".

```
git branch new-feature-1
```

- See branches

```
git branch (show local branchs)
git branch -r (show remote branchs)
git branch -a (show all local and remote branchs)
```

- Switch to new-feature-1

```
git checkout new-feature-1
```

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

```
ls
git status
```

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

```
vim test.txt
git status
```

- Store the changes to the repo and check the status

```
git add .
git commit -m "added first line"
git  status
```

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

```
vim test.txt
git commit -am "added second line"
```

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

```
git checkout main
cat text.txt
```

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

```
git merge new-feature-1
```

```
cat test.txt
```

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

```
 git checkout -b new-feature-2
```

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

```
vim test2.txt
git add .
git commit -m “created text2.txt”
```

- Switch the main branch again.

```
git checkout main
```

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

```
touch test3.txt
git add .
git commit -m “created text3.txt”
```

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

```
vim test2.txt
git add .
git commit -m “created text2.txt”
```

- merge main branch with new-feature-2

```
git merge new-feature-2
```

19. RESOLVE THE CONFLICT

- edit the file.
- then commit it.

20. Delete the branches named new-feature-1 and new-feature-2

```
git branch -d new-feature-1
git branch -D new-feature-2
```

- List the all branches

```
git branch -a
```

☺ **Thanks for Attending** ✍

Clarusway