GIT & GITHUB Notes

Configure

git config

Params:

Git config --global user.name <name>

Git config --global user.email <email>

Life Cycle of git

Modified → Staging → Commit

Modified:

1. Contains files that are newly added to the folder
2. Files that are updated but not staged

Staging:

1. Contains files that are newly staged
2. Does not add the files that are updated but not explicated staged again
3. A file is need to be staged again and again if modified

Commit:

1. Contains files that are committed

Note:

1. A file that is modified but not staged can not be directly committed
2. If there are two or more files and one is present in modified state and the other in staged state on commit only the staged files will be committed and the modified files will stay in modified

Git add:

It is used to push a file to staged state:

Git add <files>

To add all files to staged state:

Git add .

Git commit:

It is used to commit a file:

Git commit -m “this is the descriptive commit message”

Committing creates Version Control

Revert changes:

Revert changes made in a particular version

Git revert <version_id>

EX. Git revert 7435c62

```
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>My New File</title>
</head>
<body>
    <h1>New Heading of my file before reverting changes</h1>
</body>
</html>
```

Git revert <previous_version_id>

```
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>My New File</title>
</head>
<body>
    <h1>This is my old heading</h1>
</body>
</html>
```

Note: If a version is reverted two/even number of times, you will get back to the original code that was present before reverting

Reset:

Delete all the commit history to a certain point

Git reset <version_id>

This will reset your commit to a previous version but will not change the present file data

```
$ git log --oneline
ba00938 (HEAD -> master) Revert "Revert "Added index title""
401f4b8 Revert "Added index title"
7435c62 Added new heading
d220cf1 updated style
3b4bcf3 Added index title
a08c0b9 Added index and styles fiels
```

After reset

Git reset 7435c62

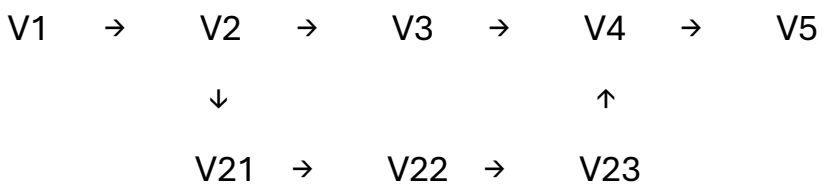
```
tomar@Kanika MINGW64 /d/Dhanuj Tomar/Project
$ git log --oneline
7435c62 (HEAD -> master) Added new heading
d220cf1 updated style
3b4bcf3 Added index title
a08c0b9 Added index and styles fiels
```

Note: to change the file data to a previous version of commit with reset use the –-hard flag

Git reset 7435c62 –-hard

This will delete all the existing commits to a certain point and reset the data in all the files upto this commit.

Branching:



1. To create a branch

   Git branch <branch_name>

2. Switch to new branch

   Git checkout <branch_name>

OR

3. Create and switch to new branch at the same time

   Git checkout -b <branch_name>

4. View all existing branches
   Git branch -a

Merge branch:

Checkout the main branch in which you want to merge the new branch

Use merge command

Ex.

Git checkout master

- In master branch

Git merge feature

Feature branch merged to master branch

Types of Merges:

- Fast Forward: Add new file or data directly to new file
- Recursive: Add new file or

GitHUB:

1. Create a new repo inside github

2. Push project branches to repo (use repo url and branch you want to push)

   Git push <url> <branch_name>

3. (Optional but useful)Give acronym to URL for ease(Generally origin is used as acronym):

   Git remote add <acronym> <URL>

4. Now step 2 can be reused as (if url=origin and branch_name = master)

   Git push origin master

5. Check for existing remote acronym

   Git remote -v

6. Push new Branch

   Git push origin <new_branch>

   This new branch can be agreed upon and requested for merge in github to be added to master branch.

7. Pull request for original branch
   Git pull origin master