**Definition**

Git is software for tracking changes in any set of files, usually used for coordinating work among programmers collaboratively developing source code during software development. Its goals include speed, data integrity, and support for distributed, non-linear workflows.

**Playground**

Please go through the following git learning playground to understand core concepts. It is a request to read through all the material and understand the english that this link has to offer.

<https://learngitbranching.js.org/>

**Scenarios and Self Exercise**

After going through the above playground completely and thoroughly we expect that you have some conceptual understanding of what GIT essentially is. Please go through the following scenarios below to reinforce your understanding of GIT and do self exercise. There are some tricky scenarios that you might encounter during your journey as a software developer and some may be as simple as creating your own git repository.

Before going through the scenarios we expect you to create your github/gitlab/bitbucket account (your choice of cloud), we prefer gitlab.

**IT IS MY HUMBLE REQUEST TO TYPE GIT COMMANDS ON THE TERMINAL RATHER THAN USING A USER INTERFACE. ALL THE EXERCISES ARE TO BE DONE IN ORDER. PLEASE DOCUMENT ALL THE COMMANDS THAT YOU ARE TYPING TO ACHIEVE THIS IN A MARKDOWN DOCUMENT FOR THE BELOW EXERCISES ALONG WITH SCREENSHOTS AND COMMIT THIS FILE IN THE REPOSITORY THAT YOU CREATE IN THE FIRST STEP BELOW AFTER THE END OF ALL THE EXERCISES. THE NAME OF THE MARKDOWN FILE SHOULD BE MY\_SUBMISSION.md.**

**Exercises**

* **Scenario 1:**

Create a personal git repository on gitlab by the name of ***git-exercise*** and create a README.md file via the user interface. (Please go through all the options that gitlab offers while creating the repository). It is our assumption here that you are working with either a **master** or a **main** branch

* **Scenario 2:**

Create an empty folder on your machine with **any name** and add a file in the folder by the name of **my\_name.txt.** Add the following content in this file - My name is <Your Name>. Now commit this file to the repository created in the first exercise - ***git-exercise***. Essentially after this exercise your repository on gitlab should have two files - **my\_name.txt** and **README.md**.

* **Scenario 3:**

**Read a little bit about merge requests/pull requests before going through this exercise.**

Checkout to a new branch named - **exercise/conflict-resolution**. In this branch change the content of **my\_name.txt** to My name used to be <Your Name> and commit and push your changes in this branch. Now go back to your master/main branch and edit the content of my\_name.txt to My name has always been <Your Name>. Commit and push your changes to the master/main branch. Go to Gitlab UI and raise a merge request to merge the **exercise/conflict-resolution** branch to the **master/main** branch. The expected content of my\_name.txt after merging **exercise/conflict-resolution** to **master/main** is:

My name used to be <Your Name>

My name has always been <Your Name>

After successful merge your **exercise/conflict-resolution** branch should be deleted from user interface and bring your local in sync with remote.

* **Scenario 4:**

Checkout to a new branch named **exercise/diff-checker**. Edit the content of **my\_name.txt** to

My name used to be <Your Name>

My name has always been <Your Name>

My name can be <Your Name>

Commit but do not push your changes to this new branch. Check the content differences of **my\_name.txt** between the two branches **master/main** and **exercise/diff-checker**, both ways, and take screenshots of both the outputs. Add these screenshots in the document.

* **Scenario 5:**

Push the branch created in exercise 4 now. Delete the two branches from both local and remote using the command line only.

* **Scenario 6:**

Create a new branch named **exercise/stash-scenario**. Branch it from **master/main** branch. Go back to the master/main branch and add a new commit by editing the file my\_name.txt. Do not push the committed change yet. Following changes go in the **my\_name.txt** file in the earlier commit:

My name used to be <Your Name>

My name has always been <Your Name>

I am about to check git stashing

Checkout to the previously created branch named **exercise/stash-scenario**. Edit the **my\_name.txt** file and add the following lines

My name used to be <Your Name>

My name has always been <Your Name>

I have successfully tested git stashing

Now switch to the **main/master** branch. Please make sure the changes that you just made in the **exercise/stash-scenario** branch are also available in the **master/main** branch when you switch. The final content after this exercise for **my\_name.txt** file should be

My name used to be <Your Name>

My name has always been <Your Name>

I am about to check git stashing

I have successfully tested git stashing

Commit the following changes and push it on your **master/main** repository

* **Scenario 7:**

Checkout to a new branch named **exercise/hooks** from **master/main**. Create a simple git hook that prints “This is the first post-commit hook from <Your Name>” on a post-commit event on the terminal. Raise a simple merge request and merge this hook on **master/main**.

* **Scenario 8:**

Move back to the **main/master** branch before proceeding to this exercise. Create a new repository on github now keep it by the name of ***git-exercise-copy***. Using the same local folder on your machine push all your changes to this new remote github repository. Essentially you have to create a copy of your codebase in a different remote repository.

* **Scenario 9:**

This is the final exercise. Create and push a new tag “1.0.0” for the current **master/main** branch.

Add the document file with all your reasons, commands and snapshots and commit it in the repository.

**Contact**

It is but natural that while going through this you will encounter a lot of challenges. Please reach out to [agoyal@calance.com](mailto:agoyal@calance.com) with all your queries.