**LEARNING GIT THROUGH SESSION NOTES**

Our GIT practice

----------------------

**GIT** is an integral part of **CI/CD** **pipeline** which we are going to implement / establish in this journey of DevOps

**Jenkins** and **Docker** also are the other integral parts

All these 3 tools will make the backbone of DevOps life cycle

Versioning is when we release a software to production/pre-production -> **1.0 -> Release 1.0 <- Snapshot 1.0 <- Snapshot 0.5 <- Snapshot 0.1**

In the meantime if there was a new **branch** (version) created (other than **Master** branch which is default created once the folder is made Git aware by > git init) and we progress with the development of new features OR there is some customization of the old feature released in version 1.0

* 1. -> 1.2 -> 1.3 -> (committed to Git Hub as my repository – main branch is the **master** branch origin ) -> **1.4** (local repository branch **feature** ) -> put the entire local repository of branch "feature" to "master"
* Master branch is synchronized with the **GitHub** (de-centralized/ **DISTRIBUTED**) repository -> once you **push** your development/changes under version 1.4 to your (account) in GitHub repository
* 1.4.1 was unit tested and u r happy to release this into pre-production/testing as version **2.0**

Checkout & Checkin

**Checkout** -> **pull**ing something out of some repository

**Checkin** -> **push**ing something into some repository

**Personal Access Token**

[ use the **Github** top right profile picture click -> settings -> Developer Settings -> Personal Access Tokens -> click on "Personal access token for Github access" -> update the setting for "**Select scopes**" ]

**NOTE**: use **Personal Access Token (PAT)** generated : **ghp\_xxxxxxxxxxxxxxxxxxxxxxxxxxx** when asked for password while doing **push/merge**

**Git on Windows/Mac**

-----------------------------

***GitHub account creation***

Open https://github.com/ in browser and click on signup

Enter your email:

Create a password:

Enter a username:

Would you like to get updates….

Verify your account:

Create Account

***GIT Installation on Windows/MacOS/Linux/Unix***

Download GIT from google : <https://git-scm.com/downloads>

First Step

------------

* Invoke GITBASH (terminal) from the folder (right click and select GitBash )
* $> mkdir <folder-name> [ Create a folder under Windows / Mac ]
* $> cd <folder-name> [ Go to that folder ]
* $> git --version
* $> git status

You will get an fatal error: saying that it is not a git local repository

* $> git init [ you should not use C:/ or C drive which is a s/w repository all the data should be stored under any other drive – you create your Git project on d: or e:…. ]

[ hint: Using 'master' as the name for the initial branch. This default branch name

hint: is subject to change. To configure the initial branch name to use in all

hint: of your new repositories, which will suppress this warning, call:

hint:

hint: git config --global init.defaultBranch <name>

hint:

hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and

hint: 'development'. The just-created branch can be renamed via this command:

hint:

hint: git branch -m <name> ]

* $> git status
* Changing the global configuration for user.name and user.email back to the correct values which will be in sync with Github account creation
  + $> git config --list
  + $> git config --global user.name "<github user-name>"
  + $> git config --global user.email [my-email@yahoo.co.uk](mailto:my-email@yahoo.co.uk)
  + $> git config –list
* $> touch file1.txt
* $> echo "This is the first line of the file 'file1.txt' " > file1.txt [ better to use vi for creating text content in the file ]
* $> git status
* $> git add . [ this will add all files or folders with the changes with mentioning of **.** ]

OR you can use $> git add <filename> [ this will add the particular file or all files or folders with changes with mentioning of . ]

* $> git commit -m "This is my first commit on file1.txt"

**Working directory** becomes clean and all the changes are now committed into **Staging area** (on branch MASTER) ready to be pushed into the GITHUB repository

* $> git log
* $> git config --list

Second step

----------------

* Creating a folder
  + $> mkdir <foldername>
* Going into the folder
  + $> cd <foldername>
* Initiating the folder with git init
  + $> git init
* Checking the hidden folder created by git init as **.git**
  + $> ls -al
* Checking the git branch
  + $> git branch [ no branches as yet as nothing was committed at this stage ]
* Checking all the branches
  + $> git branch --all [ none – reason as above ]
* Creating few files using vi and echo
  + $> vi myfirstfile
  + $> cat myfirstfile
  + $> git status
  + $> echo "This is the second file by name 'mysecondfile' " > mysecondfile
  + $> cat mysecondfile
  + $> vi mysecondfile
  + $> echo "This is the line added/appended to the end of the file 'mysecondfile' " >> mysecondfile
  + $> cat mysecondfile
  + $> git status
* Adding files to Staging area
  + $> git add .
* Committing the changes to the Staging area
  + $> git commit -m "This is commit #1"
* Checking git status and branch
  + $> git status
  + $> git branch (now the master branch is being pointed by the HEAD)
* Creating 4th file by copying the 3rd file.
  + $> vi my3rdfile
  + $> cp my3rdfile my4thfile
  + $> cat my4thfile
* Adding the 3rd file to Staging area but not the 4th file
  + $> git add my3rdfile
* Commit again
  + $> git status
  + $> ls -lrt [ sorted ascending order of listing based on the date and time of the file creation ]
  + $> git commit -m "this is commit#1 for my3rdfile changes"
  + $> git status
* Checking the git environment settings
  + $> git config --list
* Changing the global configuration for user.name and user.email
  + $> git config --global user.name "<github user-name>"
  + $> git config --global user.email [my-email@yahoo.co.uk](mailto:my-email@yahoo.co.uk)
  + $> git config --list
* Check the log and it's variations
  + $> git log
  + $> git log --oneline
  + $> git log --graph --pretty=oneline
* Creating the Github repository under Github account
* Adding the Github remote repository in the local repository environment
  + $> git remote [ no output as remote Github configuration not done yet ]
  + $> git remote -v [ none – reason as above ]
  + $> git remote add origin [https://github.com/<account-name>/<anyrepositoryname>.git](https://github.com/%3caccount-name%3e/%3canyrepositoryname%3e.git)
  + $> git config --list
* Checking remote settings
  + $> git remote
  + $> git remote -v
* Pushing the changes to GitHub repository
  + $> git branch [ to check and confirm which branch the HEAD is pointing to ]
  + $> git push origin master
* Checking the changes if that has been successfully posted in the GitHub repository from the local repo