DevOps is a set of methodology or set of rules

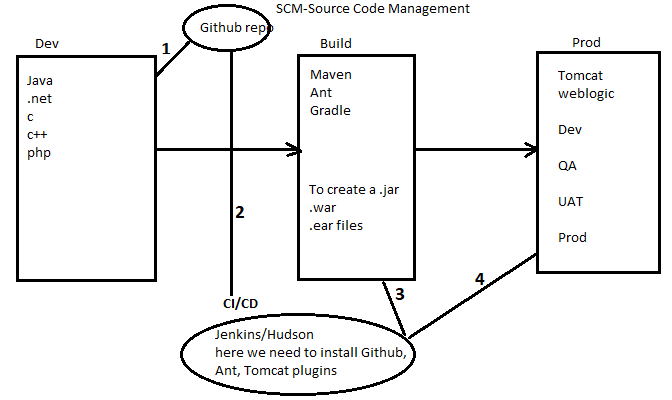
DevOps :- it reduces the gap between the developer’s activities and operational activities.

Devops is the software development method and communication, collaboration and integration between the developer activities and operational activities.

**Why DevOps:**

* It’s a automated process
* Where each and every activity is a predefined with set of rules and regulations
* Like where we have to store the code, when we have to build the code and where we have to deploy the code and how many places we have to deploy the code

**Who involves :** Developers, Build Team, Release Team, System Admins and QA team

.

.java file----->.class file----> .exe file(.war, .ear file)---> deploy this file in linux servers

Step1: code is stored into a github repo

Step2: when the code get updated in the github. Jenkins plugin(respective to tool like tomcat,

weblogic) based tool clone the code from github.

* Configure details like github(url, credentials), maven(path, environment variables),

Tomcat or weblogic or jboss server(url, username, password) in plugin

* Job creation: which git repo, which build tool, which server will configure in job

Step3: now build(test cases, phases) then it creates a executable file(.war, .ear)

Step4: deploy this executable file in the servers(dependency job created process one by one)

Step5: at last it will generate the notification mail

SCM tools: SVN, GIT, Perforce, CVS, TFS

Build tools: Ant, Maven, Gradle, Apache Builder, visual build

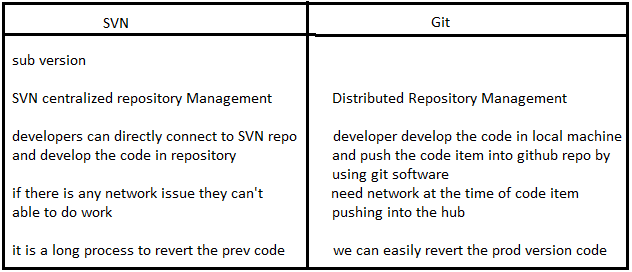
Servers: Tomcat, JBoss, Weblogic, websphere

CI/CD tools: Jenkins, Hudson, Honeypot, Bamboo

Cloud: AWS, IBM cloud, Microsoft Azure, Oracle Cloud

Configuration Management tool: chef, puppet, Ansible

SCM: Source Code Management tool or Version Control tool(SVN, Git(UI, commands))



**How to install Git?**

* To install Git click on this link https://git-scm.com/downloads Download as per your OS and install.
* After installing Git open Git Bash.
* Create a New folder and enter into that dir.
* First we need to initilize the git repository

-> git init

* After that we need to configure user name and email id

-> git config --global user.name " pavan "

-> git config --global user.email " pkumar.datascience@gmail.com "

-> git config --list ( here we can list all configurations )

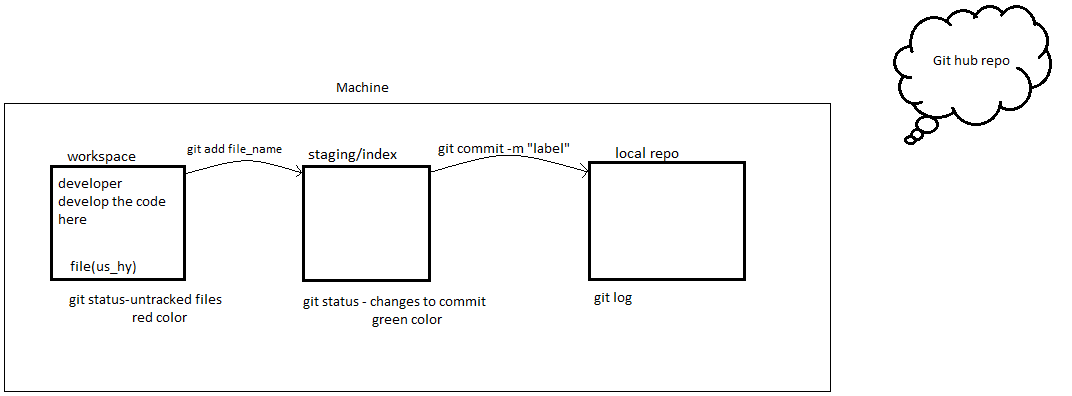
* Create some sample files by using Touch or Vi or Cat commands

EX: Touch file1

vi file2

cat > file3

**Git having three stages:**

****

**git status**

**git init** 🡪 to initialize the git repo(.git will create)

**ls** 🡪tolist the files

**ls –a** 🡪

by default branch is master

git config --global user.name “pavan”

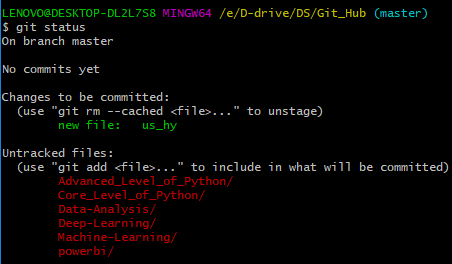
git config --global user.email “pkumar.datascience@gmail.com”

git config --list(to check whether the username and email are configure)

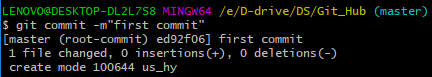
touch file1 🡪 to create empty file

git add us\_hy 🡪 to promote code from workspace to staging area

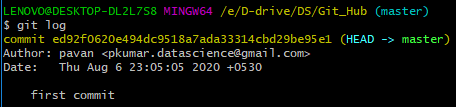
git status 🡪 to check the files



git commit –m”first commit” 🡪 to commit the code into local repo



git log 🡪 to check files in local repo



git add file1 file2 (or) git add . (or) git add \* (or) git add –A

to add multiple files into staging area

git add file1 🡪 to add single file

git commit –m “third commit” f3 🡪 to commit single file

git show <commit\_id > 🡪 to see the files in local repo

**push code from local repo to centralized repo**

create machine-learning/ repo

git clone <https://github.com/pavan-puppala/machine-learning.git>

ls

cd machine\_learning/

ls

ls –la

touch f1 f2 f3

git add .

git status

git commit –m”commit message”

git status

git log

git push 🡪 to push from local repo to centralized repo

git log --oneline 🡪 each commit\_id with 7 characters and label message

git log -2 🡪 two latest logs

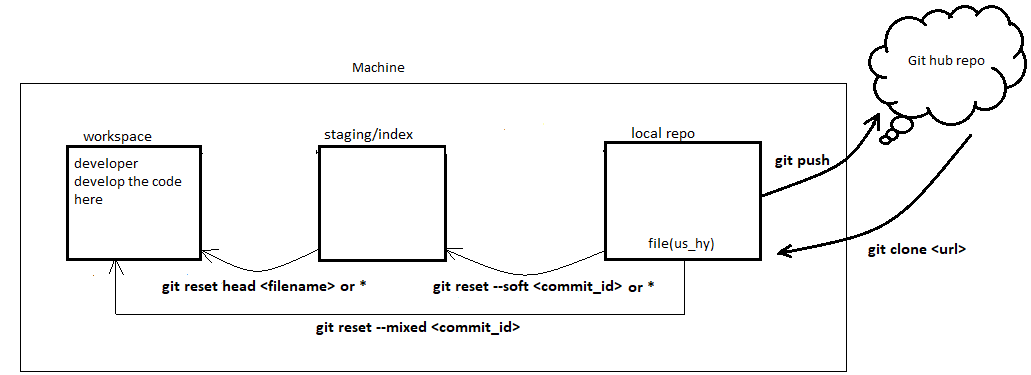
git log --oneline -3 🡪 three latest commit\_id’s with 7 characters

git log --author==pavan 🡪 to check the logs by author name wise

git log --author==pavan -n 🡪 two latest logs by author name wise

git log --since=yy-mm-dd

git log --until=yy-mm-dd

git log --oneline –decorate 🡪 it returns logs and with branch name

Commit\_id1

Commit\_id2

Commit\_id3

Commit\_id4

So if you want to roll back the commit changes of 1 we have to give the commit\_id2 to command

So if you want to roll back the commit changes of 2 we have to give the commit\_id3 to command

So if you want to roll back the commit changes of 3 we have to give the commit\_id4 to command

So if you want to roll back the commit changes of 4 we have to give the readme commit\_id2 to command

git log

git log –oneline

git status

git show <commit\_id>

git reset –soft <commit\_id>

git status

git log –oneline

git reset head <filename>

git status

git log –oneline

git show <commit\_id>

git reset –mixed <commit\_id>

git log –oneline

git status

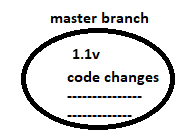
git log –oneline

git reset –mixed <initial commit\_id>

git status

**Branches**

If you want to revert back the code again you have to clone from the centralized repo



To overcome from above line

We need to create another branch then do code modification in master1

