Contents: Location: D:\Notes

- GIT
- Docker

GIT

Linux Command:
> git –version
> cd d:/ ----- Using git bash

General Git Command
> git config --global user.name "Bimal1993"
> git config --global user.email arbimalkumar.1993@gmail.com
> git init
> git status
> git add src / git add –all **---** Can use any add command add all or specific
> git commit -m "First commit"

Create a repo:
Push Local Repository to GitHub:

> git remote add origin https://github.com/Bimal1993/gitdemo.git
> git push --set-upstream origin master

To generate Token:
Settings => Developer Settings => Personal Access Token => Generate New Token => Fillup the form => click Generate token => Copy the generated Token,

Add or make changes to the file:
> git add src
> git commit -m "Second commit"
> git push --set-upstream origin master

> git log //To check git logs

Git Branch:
> git branch developer1
> git branch
developer1
* master

$ git checkout developer1

Git Merge:
$ git branch
* developer1
Master

$ git checkout master
$ git merge developer1
$ git add src
$ git commit -m "new code from branch developer"
$ git push --set-upstream origin master

Delete branch
$ git push --delete origin developer1

Git conflicts:
<<<<<<< HEAD
open an issue
=======
ask your question in IRC.
>>>>>>>

Docker

It include all dependency with version.
> docker -v
https://hub.docker.com/
> docker images
> docker pull bimal1993/docker-demo1
> docker images
> docker run bimal1993/java-docker-app

Create application:
Create a spring boot application

Dockerfile
FROM openjdk:8-jdk-alpine
EXPOSE 8080
ARG JAR_FILE=target/*.war
COPY ${JAR_FILE} app.war
ENTRYPOINT ["java","-jar","app.war"]

Option to build a docker file:
// build images.
20: docker build -f Dockerfile -t bimal1993/docker-demo1 .

//push images
21: docker push bimal1993/docker-demo1

//pull images
22: docker pull bimal1993/docker-demo1
// run image
23: docker run bimal1993/docker-demo1

// to remove image
24: docker rmi -f bimal1993/docker-demo1

1: create docker File.

DockerFile
FROM openjdk:8
ADD target/docker-spring-boot-.jar docker-spring-boot.jar
EXPOSE 8085
ENTRYPOINT ["java" "-jar" "docker-spring-boot.jar"]

$ docker -v
$ docker build -f docker -t docker-spring-boot .
$ docker images
$ docker run -p 8085:8085 docker-spring-boot

2: Docker run on background.
$ docker run -p 500:5000 -d bimal1993/java-docker-app2:firsttry
$ docker logs image-id (to check logs)
$ docker logs -f image-id (following logs)

3: Remove
$ docker images // to check docker images
$ docker container ls -a
$ docker container stop "container-id" //stop container.
$ docker container prune // remove all exited container.
$ docker image rm "image-d" // to remove an image " docker image rm 4bf278546754 7a9513ab1412 "

Plugin:

App.java
```
public class App {
public static void main(String[] args) {
System.out.println("Jenkins, docket automation");
AutomationDocker docker=new AutomationDocker();
System.out.println("Current Status: "+docker.getAutomationStatusInfo());
}
}
```

AutomationDocker.java
```
public class AutomationDocker {
public String getAutomationStatusInfo() {
return "Running";
}
```

}

Dockerfile

```
FROM openjdk:8-jdk-alpine
EXPOSE 8080
ARG JAR_FILE=target/*.jar
COPY ${JAR_FILE} app.jar
ENTRYPOINT ["java","-jar","app.jar"]
```

Steps to configure Genkins:

Sample console output:

Docker hub view:

Local machine view: