

Module 7: Jenkins Pipeline

Demo Document - 1

edureka!

© Brain4ce Education Solutions Pvt. Ltd.

Demo: Pipeline as Code using Jenkinsfile

Problem statement: How to deploy a JAVA application with Jenkins pipeline

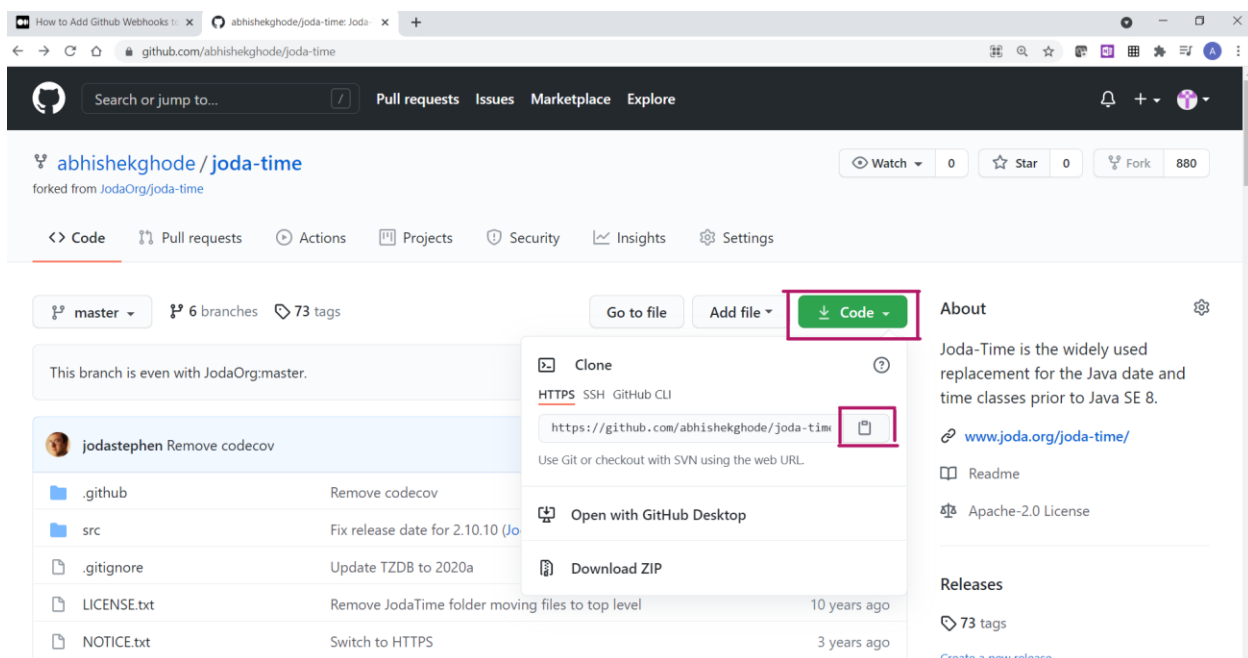
Solution:

Step 1: Adding Jenkinsfile into a git repository

URL to access: github.com

We already have forked joda-time repository in previous lab. Let's clone this repository on local workstation.

Click 'Code' and then copy repository url using copy button.



This forked repository can be cloned on local workstation. We will add a Jenkinsfile. We can commit and push the changes back into this forked repository.

This is called 'Pipeline as Code' which uses continuous delivery pipeline as part of the codebase to be versioned.

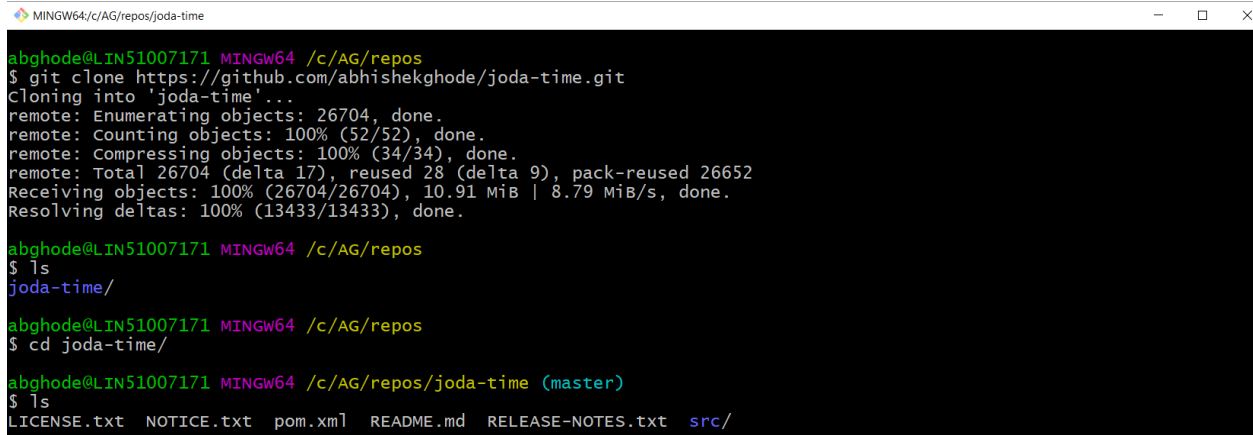
Module 7 – Jenkins Pipeline

Step 2: Clone the github repository on local workstation

Tool to access: git bash (you can use any other git client as well)

Commands to run:

```
git clone https://github.com/abhishekgghode/joda-time.git
ls
cd joda-time
ls
```



```
MINGW64:/c/AG/repos/joda-time
abghode@LIN51007171 MINGW64 /c/AG/repos
$ git clone https://github.com/abhishekgghode/joda-time.git
Cloning into 'joda-time'...
remote: Enumerating objects: 26704, done.
remote: Counting objects: 100% (52/52), done.
remote: Compressing objects: 100% (34/34), done.
remote: Total 26704 (delta 17), reused 28 (delta 9), pack-reused 26652
Receiving objects: 100% (26704/26704), 10.91 MiB | 8.79 MiB/s, done.
Resolving deltas: 100% (13433/13433), done.

abghode@LIN51007171 MINGW64 /c/AG/repos
$ ls
joda-time/

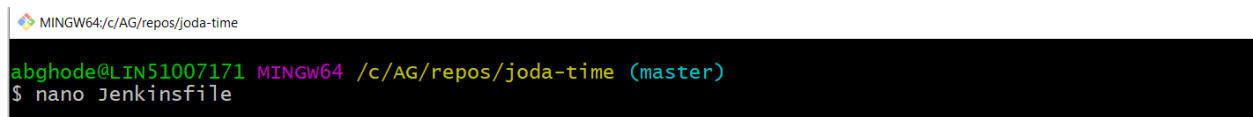
abghode@LIN51007171 MINGW64 /c/AG/repos
$ cd joda-time/

abghode@LIN51007171 MINGW64 /c/AG/repos/joda-time (master)
$ ls
LICENSE.txt  NOTICE.txt  pom.xml  README.md  RELEASE-NOTES.txt  src/
```

These steps will help us to clone and view codebase for joda-time library.

Step 3: Adding a Jenkinsfile into local repository

Commands to access: nano Jenkinsfile (you can use another text editor as well)



```
MINGW64:/c/AG/repos/joda-time
abghode@LIN51007171 MINGW64 /c/AG/repos/joda-time (master)
$ nano Jenkinsfile
```

Module 7 – Jenkins Pipeline

Currently Jenkinsfile is empty. Please add below mentioned content to it.

```
pipeline
{
    agent any

    tools {
        // Install the Maven version "M3" and add to the path
        maven "Maven 3"
    }

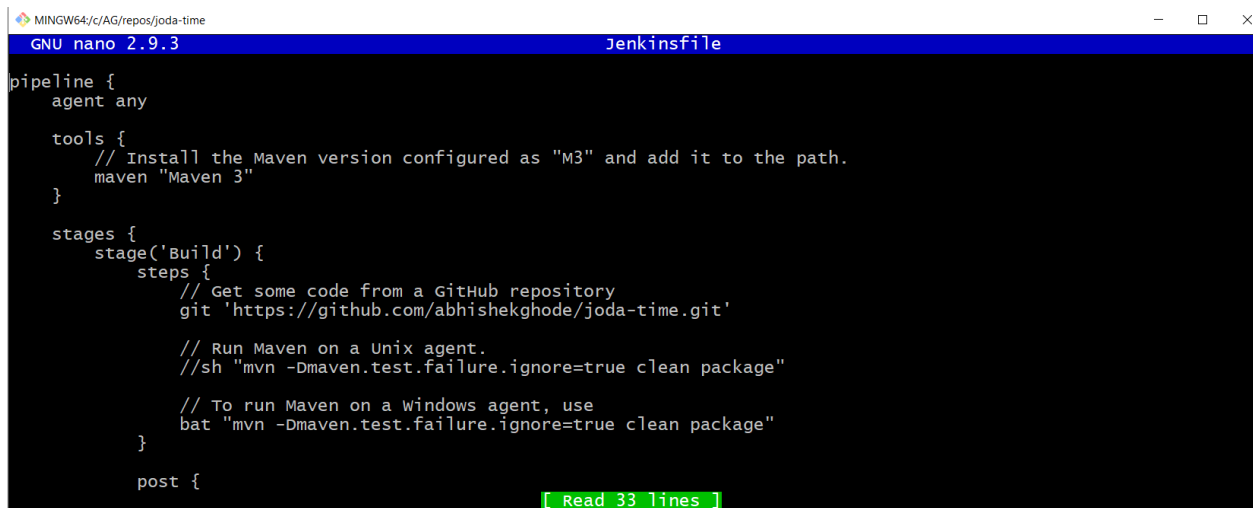
    stages {
        stage('Build') {
            steps {
                // Get some code from a GitHub repository
                git 'https://github.com/abhishekghode/joda-time.git'

                // Run Maven on a Unix agent.
                //sh "mvn -Dmaven.test.failure.ignore=true clean package"

                // To run Maven on a Windows agent
                bat "mvn -Dmaven.test.failure.ignore=true clean package"
            }
        }

        post {
            // Record the test results
            success {
                junit '**/target/surefire-reports/TEST-*.xml'
                //Archive the jar
                archiveArtifacts 'target/*.jar'
            }
        }
    }
}
```

Module 7 – Jenkins Pipeline

A screenshot of a terminal window with a black background. The title bar at the top shows 'MINGW64:/c:/AG/repos/joda-time' and window control buttons. Below the title bar, a blue header bar displays 'GNU nano 2.9.3' and 'jenkinsfile'. The main area contains a Jenkinsfile script in a light blue monospace font. The script defines a pipeline with an 'any' agent, a 'tools' section for Maven, and a 'stages' section with a 'Build' stage containing steps for cloning a GitHub repository and running Maven. A status bar at the bottom right indicates '[Read 33 lines]'.

```
pipeline {
  agent any

  tools {
    // Install the Maven version configured as "M3" and add it to the path.
    maven "Maven 3"
  }

  stages {
    stage('Build') {
      steps {
        // Get some code from a GitHub repository
        git 'https://github.com/abhishekghode/joda-time.git'

        // Run Maven on a Unix agent.
        //sh "mvn -Dmaven.test.failure.ignore=true clean package"

        // To run Maven on a windows agent, use
        bat "mvn -Dmaven.test.failure.ignore=true clean package"
      }
    }
  }

  post {
```

You can save the contents of Jenkinsfile using below keyboard shortcuts if you are using nano text editor.

[ctrl] + x; type y; press [enter]

This declarative pipeline script is similar which we have already used in previous module. Please observe the github repository. Its should be owned by you. Otherwise you may not be able to push changes here.

Step 4: Commit changes locally into git repository and push the changes to github repository.

Commands to run:

- git status
- git add ./Jenkinsfile
- git status

Module 7 – Jenkins Pipeline

```
MINGW64/c/AG/repos/joda-time
abghode@LIN51007171 MINGW64 /c/AG/repos/joda-time (master)
$ git status
On branch master
your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        Jenkinsfile

nothing added to commit but untracked files present (use "git add" to track)
abghode@LIN51007171 MINGW64 /c/AG/repos/joda-time (master)
$ git add ./Jenkinsfile
warning: LF will be replaced by CRLF in Jenkinsfile.
the file will have its original line endings in your working directory.
abghode@LIN51007171 MINGW64 /c/AG/repos/joda-time (master)
$ git status
On branch master
your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        new file:   Jenkinsfile
```

Commands to run: `git commit -m "adding Jenkinsfile"`

`git remote -v`

`git push origin master` (you may need to authenticate using github credentials)

```
MINGW64/c/AG/repos/joda-time
abghode@LIN51007171 MINGW64 /c/AG/repos/joda-time (master)
$ git commit -m "adding Jenkinsfile"
[master c443373e] adding Jenkinsfile
1 file changed, 33 insertions(+)
create mode 100644 Jenkinsfile
abghode@LIN51007171 MINGW64 /c/AG/repos/joda-time (master)
$ git remote -v
origin https://github.com/abhishekghode/joda-time.git (fetch)
origin https://github.com/abhishekghode/joda-time.git (push)
abghode@LIN51007171 MINGW64 /c/AG/repos/joda-time (master)
$ git push origin master
```

A GitHub Login dialog box is displayed over the terminal. It has a title bar 'GitHub Login' and a close button. The main text is 'GitHub Login'. Below it are two input fields: the first contains 'abhishekghode' and the second contains masked characters '*****'. At the bottom, there are two buttons: 'Login' with a checkmark icon and 'Cancel' with an 'X' icon. Below the buttons, there is a link: 'Don't have an account? Sign up' and another link: 'Forgot your password?'.

Module 7 – Jenkins Pipeline

Please ensure that 'git push' command executed successfully.

```
MINGW64/c/AG/repos/joda-time
abghode@LIN51007171 MINGW64 /c/AG/repos/joda-time (master)
$ git commit -m "adding Jenkinsfile"
[master c443373e] adding Jenkinsfile
1 file changed, 33 insertions(+)
create mode 100644 Jenkinsfile

abghode@LIN51007171 MINGW64 /c/AG/repos/joda-time (master)
$ git remote -v
origin https://github.com/abhishekghode/joda-time.git (fetch)
origin https://github.com/abhishekghode/joda-time.git (push)

abghode@LIN51007171 MINGW64 /c/AG/repos/joda-time (master)
$ git push origin master
logon failed, use ctrl+c to cancel basic credential prompt.
Username for 'https://github.com': abhishekghode
counting objects: 3, done.
delta compression using up to 8 threads.
compressing objects: 100% (3/3), done.
writing objects: 100% (3/3), 704 bytes | 704.00 KiB/s, done.
total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/abhishekghode/joda-time.git
27edfffa..c443373e master -> master
```

Step 5: Check Jenkinsfile on github repository

URL to access: <https://github.com/abhishekghode/joda-time>

The screenshot shows the GitHub repository page for `abhishekghode/joda-time`. The repository is on the `master` branch, which is 1 commit ahead of `JodaOrg:master`. The commit history table shows the following entries:

Commit	Message	Time
c443373	adding Jenkinsfile	6 minutes ago
27edffa	Remove JodaTime folder moving files to top level	10 years ago
...

The `Jenkinsfile` commit is highlighted with a red box. The right sidebar shows the repository's README, Apache-2.0 License, and 73 tags.

Now, your github repository is ready to be used for 'pipeline as code' since it has a Jenkinsfile available in the codebase.

Module 7 – Jenkins Pipeline

Step 6: Create a pipeline project on Jenkins

URL to access: <http://localhost:8080/view/all/newJob>

Please ensure to start Jenkins server prior to accessing this url.

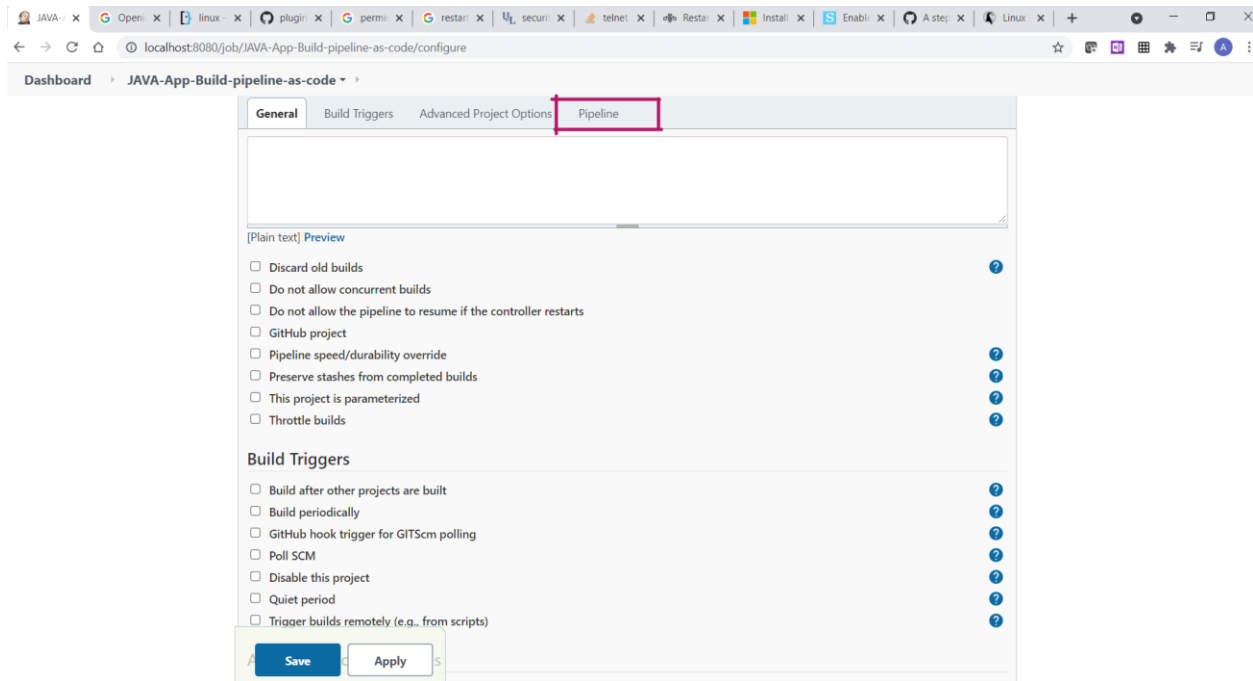
The screenshot shows the Jenkins 'New Job' page. At the top, there's a search bar and a user profile 'admin' with a 'log out' button. Below the header, the 'Enter an item name' section has a text input field containing 'JAVA-App-Build-pipeline-as-code'. Below this, there are four project type options: 'Freestyle project', 'Pipeline' (which is selected and highlighted with a red box), 'Multi-configuration project', and 'Folder'. Each option has a brief description. At the bottom, there's a 'GitHub Organization' section with an 'OK' button.

Step 6: Configuring pipeline project to specify pipeline script

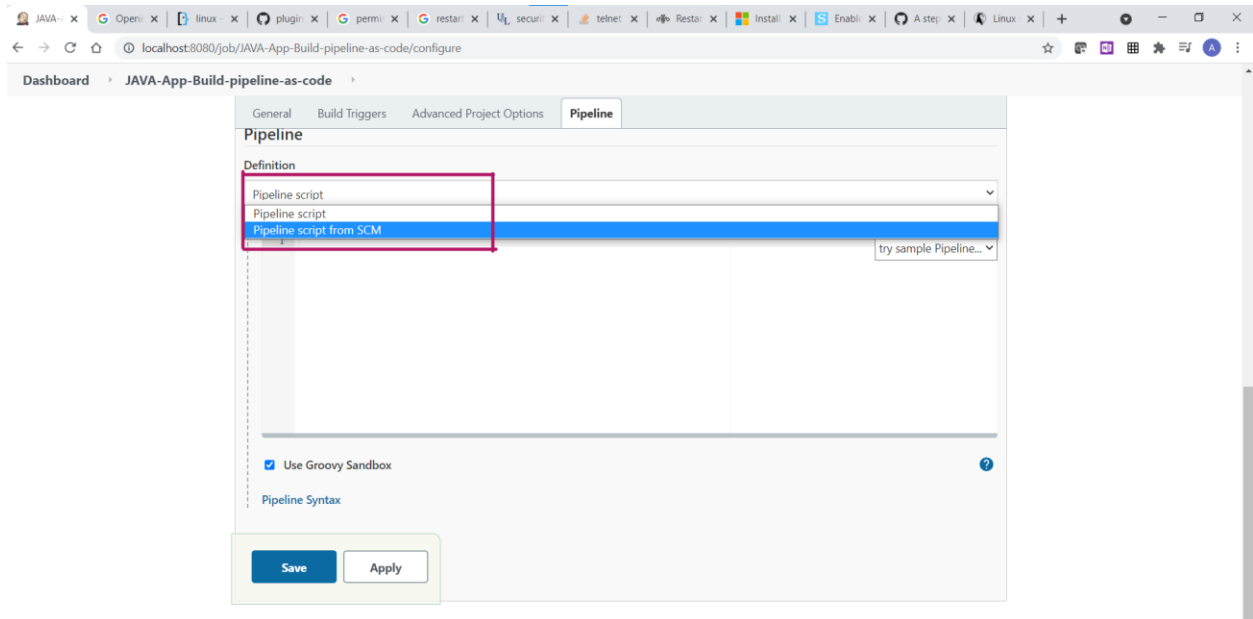
URL to access: <http://localhost:8080/job/JAVA-App-Build-pipeline-as-code/configure>

Click on 'Pipeline'

Module 7 – Jenkins Pipeline



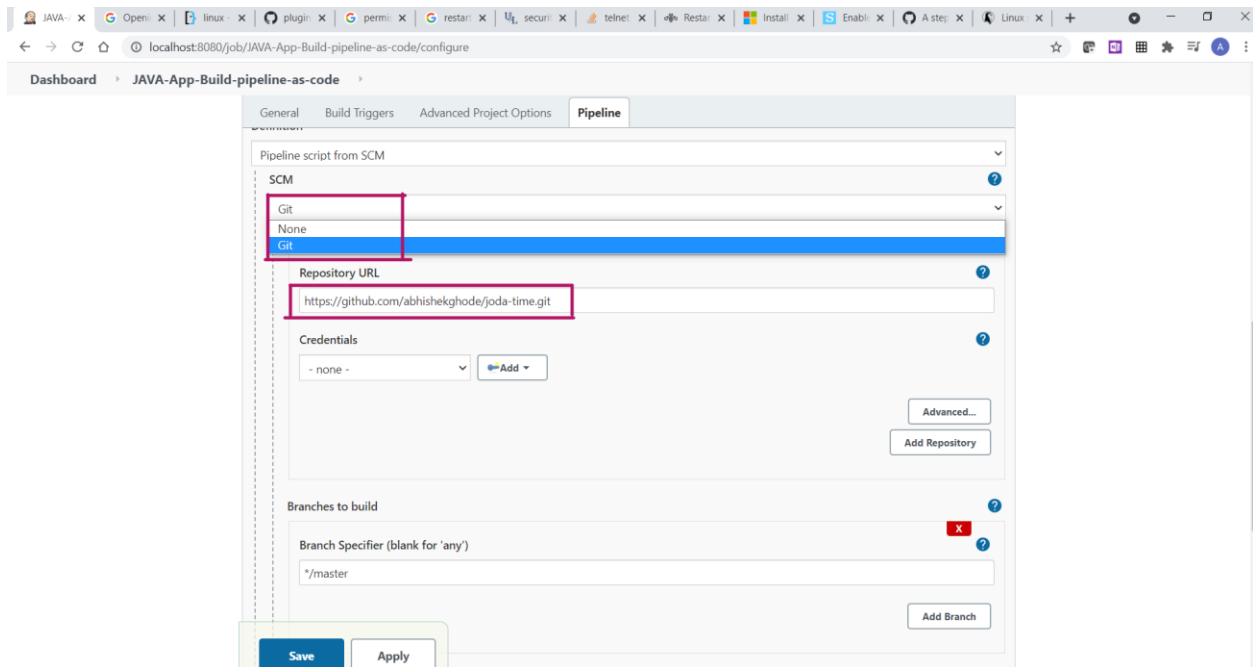
Please select 'Pipeline script from SCM' option from drop-down widget



Module 7 – Jenkins Pipeline

SCM should be selected as 'Git'

Also please enter github repository url. And click on 'Save'



Module 7 – Jenkins Pipeline

Step 7: Build the pipeline project

URL to access: <http://localhost:8080/job/JAVA-App-Build-pipeline-as-code/>

Click on 'Build Now'

