

Practice for Lesson 2: Build and Run Jobs on Jenkins

Practices for Lesson 2

Overview

In these practices, you will build and run a job on Jenkins. Further, schedule a job to execute periodically and also integrate GitHub repository source code with Jenkins job.

Practice 2-3: Integrate GitHub Repository with Jenkins

Overview

In this practice, you will learn to integrate GitHub source code with Jenkins job.

Assumptions

1. You should have completed the Practice of Lesson 2-2
2. User should have created a **GitHub account**.

Tasks

1. Install **Git** in the AWS EC2 instance for **Jenkins** integration.
 - a. Connect to the AWS EC2 instance from putty. Execute the command shown below to install **Git** in the AWS EC2 instance.

```
[ec2-user@ip-172-31-33-131 ~]$ sudo yum install git
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core                               | 3.7 kB      00:00
Resolving Dependencies
--> Running transaction check
--> Package git.x86_64 0:2.23.4-1.amzn2.0.1 will be installed
--> Processing Dependency: perl-Git = 2.23.4-1.amzn2.0.1 for package: git-2.23.4-1.amzn2.0.1.x86_64
--> Processing Dependency: git-core-doc = 2.23.4-1.amzn2.0.1 for package: git-2.23.4-1.amzn2.0.1.x86_64
--> Processing Dependency: git-core = 2.23.4-1.amzn2.0.1 for package: git-2.23.4-1.amzn2.0.1.x86_64
--> Processing Dependency: emacsfilesystem >= 25.3 for package: git-2.23.4-1.amzn2.0.1.x86_64
--> Processing Dependency: perl(Term::ReadKey) for package: git-2.23.4-1.amzn2.0.1.x86_64
--> Processing Dependency: perl(Git::I18N) for package: git-2.23.4-1.amzn2.0.1.x86_64
```

- b. Type “y” when the terminal prompts as shown below to proceed with the installation process.

```
libsecret          x86_64      0.18.5-2.amzn2.0.2
perl-Error         noarch     1:0.17020-2.amzn2
perl-Git           noarch     2.23.4-1.amzn2.0.1
perl-TermReadKey   x86_64     2.30-20.amzn2.0.2

Transaction Summary
-----
Install 1 Package (+7 Dependent packages)

Total download size: 7.9 M
Installed size: 41 M
Is this ok [y/d/N]: y
Downloading packages:
(1/8): emacsfilesystem-25.3-3.amzn2.0.2.noarch.rpm
```

- c. **Git** is successfully installed in the AWS EC2 instance as shown below.

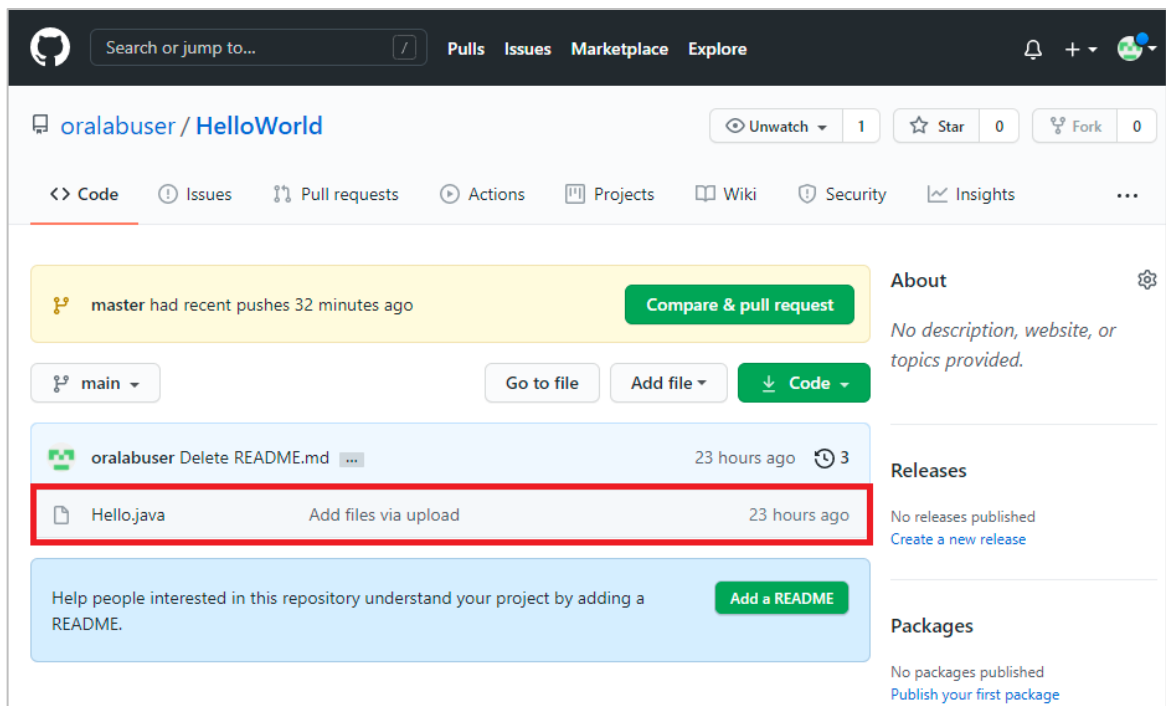
```
Verifying : git-core-2.23.4-1.amzn2.0.1.x86_64
Verifying : 1:perl-Error-0.17020-2.amzn2.noarch
Verifying : 1:emacsfilesystem-25.3-3.amzn2.0.2.noarch

Installed:
git.x86_64 0:2.23.4-1.amzn2.0.1

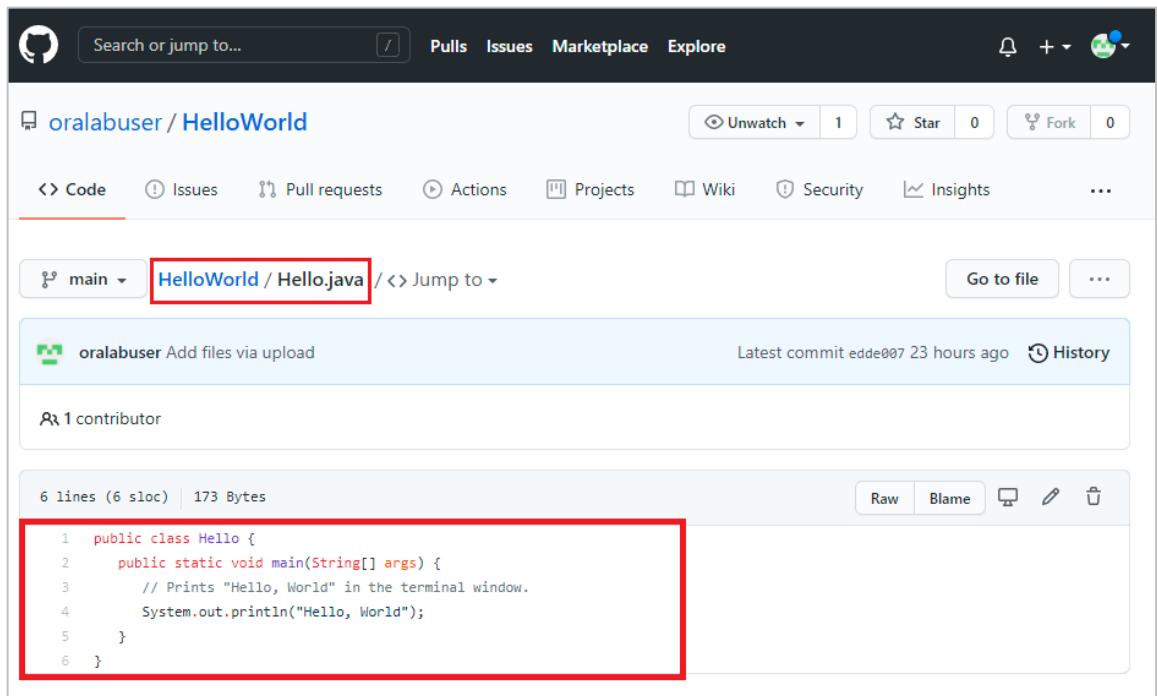
Dependency Installed:
emacsfilesystem.noarch 1:25.3-3.amzn2.0.2
git-core.x86_64 0:2.23.4-1.amzn2.0.1
git-core-doc.noarch 0:2.23.4-1.amzn2.0.1
libsecret.x86_64 0:0.18.5-2.amzn2.0.2
perl-Error.noarch 1:0.17020-2.amzn2
perl-Git.noarch 0:2.23.4-1.amzn2.0.1
perl-TermReadKey.x86_64 0:2.30-20.amzn2.0.2

Complete!
```

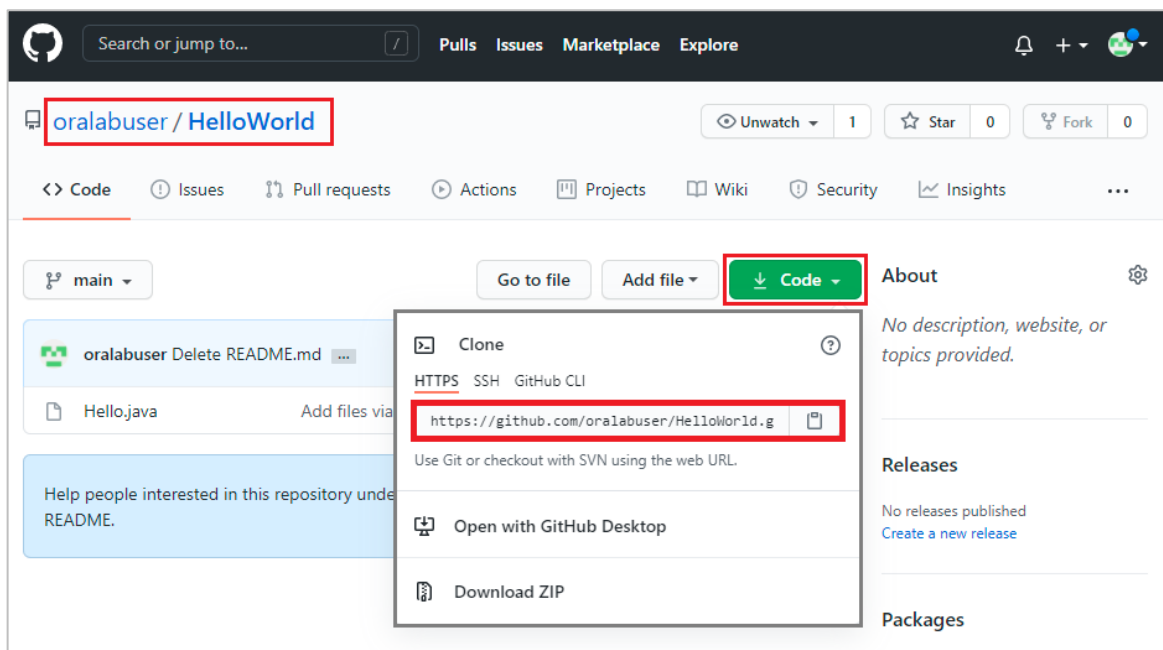
2. To create a basic java program in the **GitHub** repository.
 - a. Create or add a **Hello.java** file in the **GitHub** Repository as shown below.



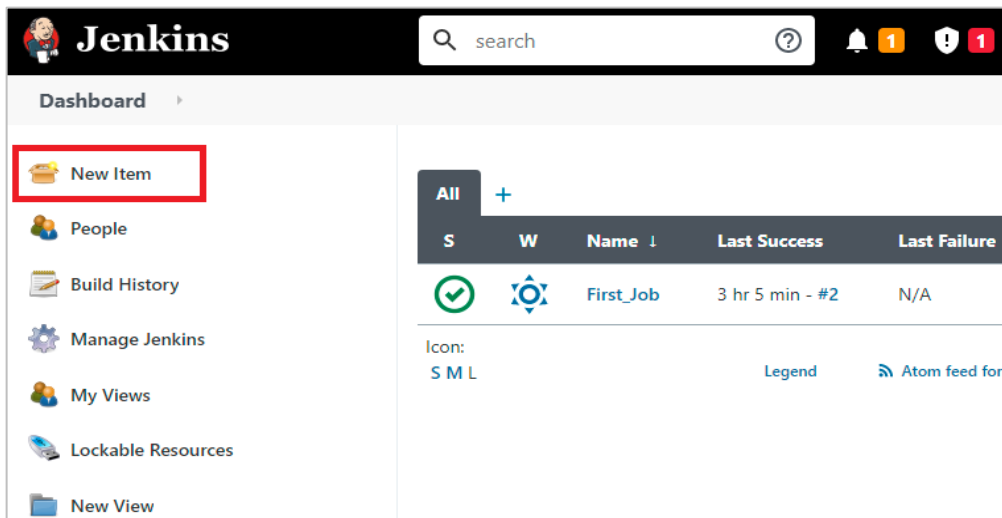
- b. The **Hello.java** file consists of the java program to print the “Hello World” message on successful execution as shown below.



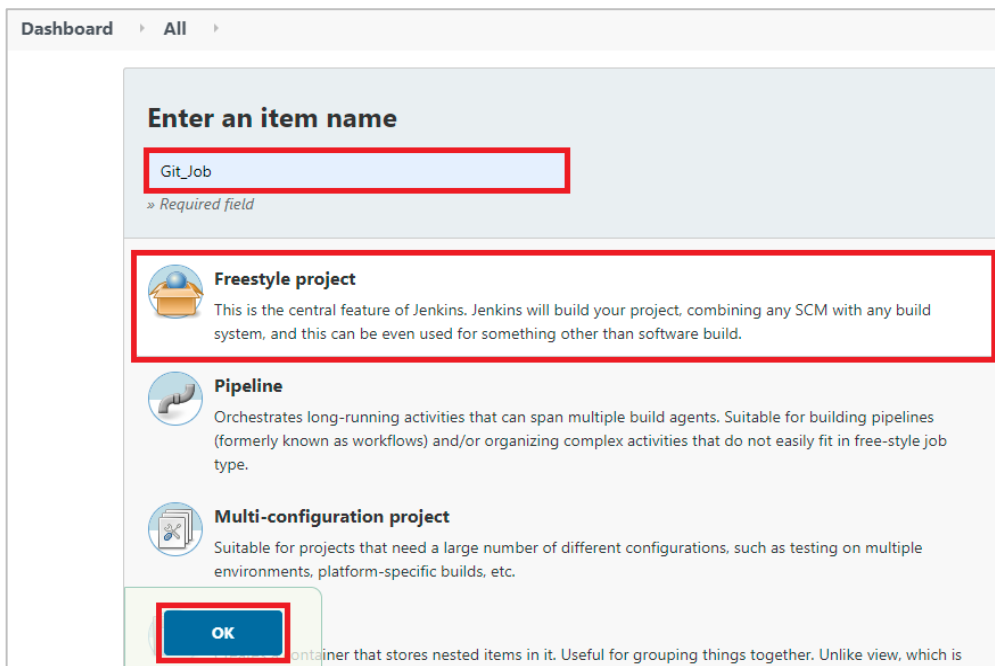
- c. Navigate to the repository page and select **Code**. Copy the **HTTPS** URL link of the repository as shown below.



3. Integrate GitHub source code with Jenkins Job to execute.
 - a. In Jenkins instance Dashboard, navigate to the menu and select **New Item** to create a new job as shown below.



- b. Provide the name for the Job in Jenkins, select **Freestyle project** and click **OK** as shown below.



- c. Scroll down to **Source Code Management** and select **Git** as shown below.

Dashboard > Git_Job >

General **Source Code Management** Build Triggers Build Environment Build Post-build Actions

☐ Disable this project

☐ Execute concurrent builds if necessary

Advanced...

Source Code Management

☐ None

☒ Git

Repositories

Repository URL

Please enter Git repository.

Credentials

▼ Add ▼

- d. Paste the GitHub repository HTTPS URL link in the **Repository URL** as shown below.

Dashboard > Git_Job >

General **Source Code Management** Build Triggers Build Environment Build

Post-build Actions

Source Code Management

☐ None

☒ Git

Repositories

Repository URL

https://github.com/oralabuser/HelloWorld.git

Credentials

- none - ▼ Add ▼

Advanced...

Add Repository

- e. In **Branches to build** provide the specified branch name from the GitHub repository.

General **Source Code Management** Build Triggers Build Environment Build

Post-build Actions

Advanced...
Add Repository

Branches to build ?

Branch Specifier (blank for 'any') X ?

***/main**

Add Branch

Repository browser ?

(Auto) ▾

Additional Behaviours

- f. Scroll down to the **Build** section, click **Add build step** and select **Execute shell** as shown below.

Dashboard ▸ **Second_Job** ▸

General Source Code Management Build Triggers **Build Environment** Build

Post-build Actions

☐ Use secret text(s) or file(s) ?
☐ Abort the build if it's stuck
☐ Add timestamps to the Console Output
☐ Inspect build log for published Gradle build scans
☐ With Ant ?

Build

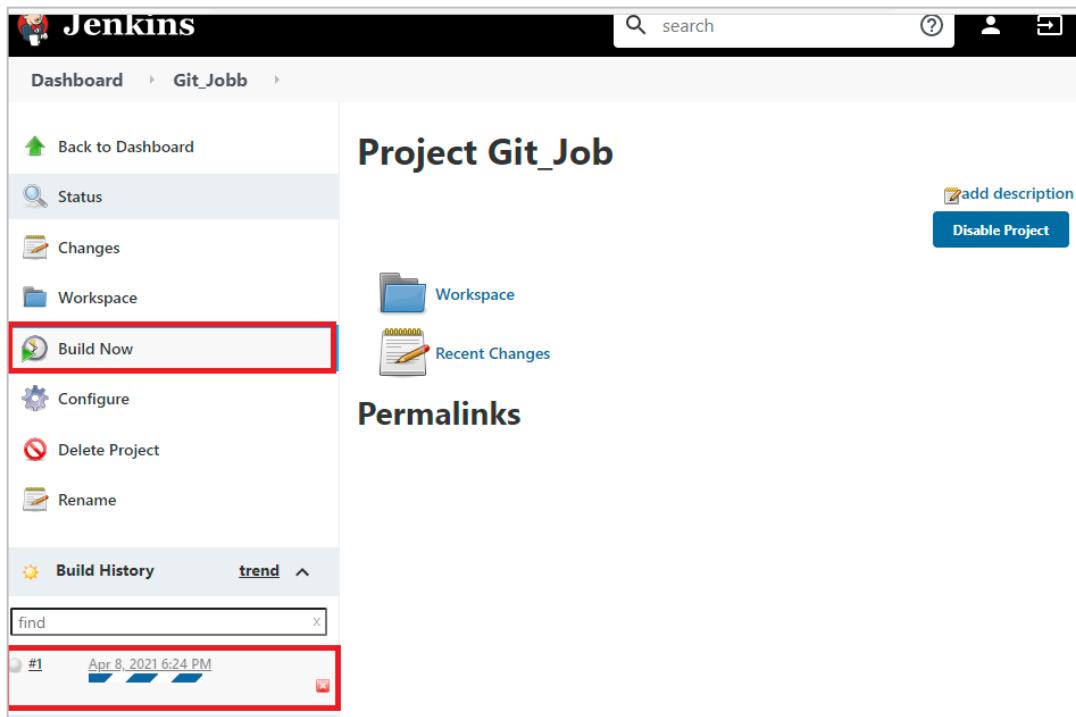
Add build step ▴

- Execute Windows batch command
- Execute shell**
- Invoke Ant
- Invoke Gradle script
- Invoke top-level Maven targets
- Run with timeout
- Set build status to "pending" on GitHub commit

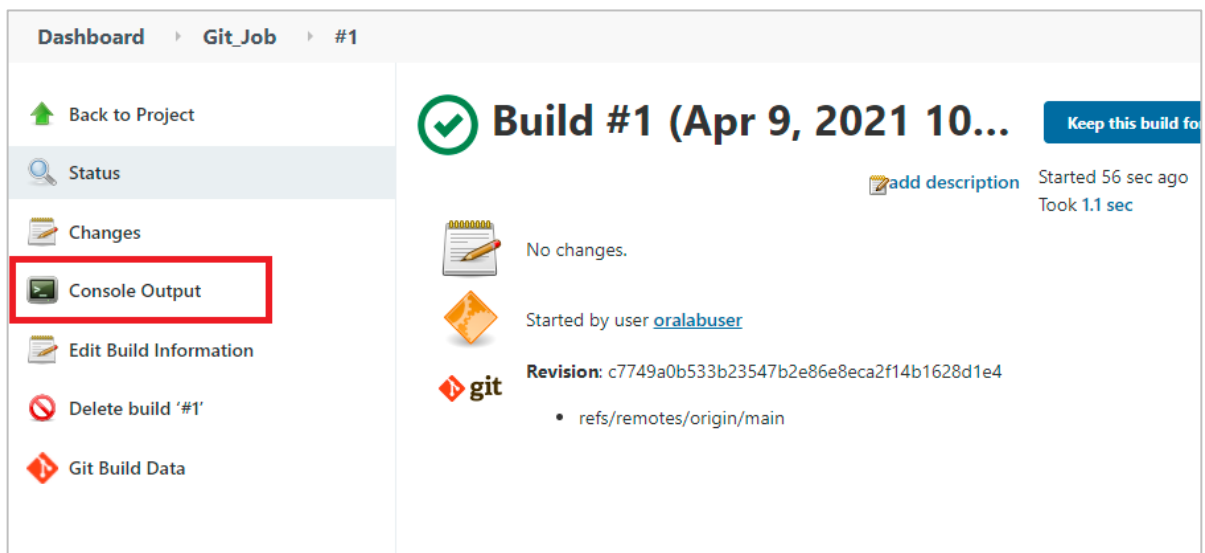
- g. In **Execute shell** provide the command to execute Java program and click **Save** as shown below.

The screenshot shows the Jenkins configuration interface for a build job. At the top, there are tabs for 'General', 'Source Code Management', 'Build Triggers', 'Build Environment', and 'Build'. Below these is a 'Post-build Actions' section. The main area is titled 'Build' and contains a single step named 'Execute shell'. The 'Command' field for this step contains the text: `javac Hello.java` followed by `java Hello` on the next line. A red rectangle highlights this command field. Below the command field is a link that says 'See the list of available environment variables'. To the right of the command field is an 'Advanced...' button. Below the 'Execute shell' step is a button labeled 'Add build step'. Below that is a section for 'Post-build Actions' with a button labeled 'Add post-build action'. At the bottom of the configuration page, there are two buttons: 'Save' and 'Apply'. The 'Save' button is highlighted with a red rectangle.

- h. In the project page, select **Build Now** to build the job in Jenkins and click on the build created under **Build History** as shown below.



- i. In the **Build** page, click **Console Output** to view the output of the Git job.



- j. On successful execution, verify the output which executes the java file in the GitHub repository specified as shown below.

Dashboard > Git_Job > #1

Back to Project

Status

Changes

Console Output

View as plain text

Edit Build Information

Delete build '#1'

Git Build Data

Console Output

Started by user oralabuser

Running as SYSTEM

Building in workspace /var/lib/jenkins/workspace/Git_Job

The recommended git tool is: NONE

No credentials specified

> git rev-parse --is-inside-work-tree # timeout=10

Fetching changes from the remote Git repository

> git config remote.origin.url https://github.com/oralabuser/HelloWorld.git # timeout=10

Fetching upstream changes from https://github.com/oralabuser/HelloWorld.git

> git --version # timeout=10

> git --version # 'git version 2.23.4'

> git fetch --tags --force --progress -- https://github.com/oralabuser/HelloWorld.git +refs/heads/*:refs/remotes/origin/* # timeout=10

> git rev-parse refs/remotes/origin/main^{commit} # timeout=10

Checking out Revision c7749a0b533b23547b2e86e8eca2f14b1628d1e4 (refs/remotes/origin/main)

> git config core.sparsecheckout # timeout=10

> git checkout -f c7749a0b533b23547b2e86e8eca2f14b1628d1e4 # timeout=10

Commit message: "Delete README.md"

First time build. Skipping changelog.

[Git_Job] \$ /bin/sh -xe /tmp/jenkins1121160085960805041.sh

+ javac Hello.java

+ java Hello

Hello, World

Finished: SUCCESS

4. Close the terminal, and logout from the AWS Management console and Jenkins Dashboard.

Copyright © 2021, Proton Expert Systems & Solutions. All rights reserved.

Practice for Lesson 2: Build and Run Jobs on Jenkins

11