



Continuous Integration Demo



3.4: Working Locally with GIT

Locally Working with GIT

Download GIT From

<https://git-scm.com/downloads>

First Repository :

- Create Empty directory on system myrepoone
- Create a repository
git init

```
rvikash@PUNHDCLT58 MINGW32 ~/Desktop
$ cd myrepoone
bash: cd: myrepoone: No such file or directory

rvikash@PUNHDCLT58 MINGW32 ~/Desktop
$ cd C:\myrepoone

rvikash@PUNHDCLT58 MINGW32 /c/myrepoone
$ git init
Initialized empty Git repository in C:/myrepoone/.git/

rvikash@PUNHDCLT58 MINGW32 /c/myrepoone (master)
$ ls -la
./  ../  .git/
```

Checking
repository
created or
not



3.4: Working Locally with GIT

Locally Working with GIT

Setting Name & email Id

- Setting name

`git config --global user.name "Rahul Vikash"`

- Setting email

`$ git config --global user.mail rahul.vikash@capgemini.com`

- Checking configuration

`$ git config --list`

```
rvikash@PUNHDCLT58 MINGW32 /c/myrepoone (master)
$ git config --global user.name "Rahul Vikash"
rvikash@PUNHDCLT58 MINGW32 /c/myrepoone (master)
$ AC
rvikash@PUNHDCLT58 MINGW32 /c/myrepoone (master)
$ git config --global user.mail rahul.vikash@capgemini.com
rvikash@PUNHDCLT58 MINGW32 /c/myrepoone (master)
$ AC
rvikash@PUNHDCLT58 MINGW32 /c/myrepoone (master)
$ git config --list
core.symlinks=false
core.autocrlf=true
core.fscache=true
color.diff=auto
color.status=auto
color.branch=auto
color.interactive=true
pack.packsizeLimit=2g
help.format=html
http.sslCAinfo=C:/Program Files/Git/mingw32/ssl/certs/ca-bundle.crt
diff.astextplain.textconv=astextplain
rebase.autosquash=true
credential.helper=manager
user.name=Rahul Vikash
user.mail=rahul.vikash@capgemini.com
core.repositoryFormatVersion=0
core.filemode=false
core.bare=false
core.logallrefupdates=true
core.symlinks=false
core.ignorecase=true
```



3.4: Working Locally with GIT

Locally Working with GIT

Adding File in repository

MyFirst.txt

Check

\$ ls

Know the status

\$ git status

Add the file in staging

\$ git add MyFirst.txt

Commit in git repository with lock message

\$ git commit -m "Added MyFirst File"

See log

\$ git log



3.4: Working Locally with GIT

Locally Working with GIT

```
rvikash@PUNHDCLT58 MINGW32 /c/myrepoone (master)
$ git commit -m "Added MyFirst File"
[master (root-commit) d06ddf1] Added MyFirst File
Committer: Rahul Vikash <rahul.vikash@capgemini.com>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:
```

```
    git config --global user.name "Your Name"
    git config --global user.email you@example.com
```

After doing this, you may fix the identity used for this commit with:

```
    git commit --amend --reset-author
```

```
1 file changed, 1 insertion(+)
create mode 100644 MyFirst.txt
```



3.4: Working Locally with GIT

Locally Working with GIT-

Adding new data in MyFirst.txt

Now checking status \$git status

```
rvikash@PUNHDCLT58 MINGW32 /c/myrepoone (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   MyFirst.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

Cor

Again check log --\$git log



3.4: Working Locally with GIT

Locally Working with GIT-

To check which line has changed

```
$ git commit -amend
```

Add one more file -MyJava.txt

Unstage the One File

```
$ git reset HEAD MyFirst.txt
```

Unchanged the work done --\$ git checkout -- MyJava.txt

```
rvikash@PUNHDCLT58 MINGW32 /c/myrepoone (master)
$ git checkout -- MyJava.txt

rvikash@PUNHDCLT58 MINGW32 /c/myrepoone (master)
$ git status
on branch master
changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   MyJava.txt
```



3.4: Working Locally with GIT

Locally Working with GIT

Multiple Reset for n number of file ,Goes to last commit

```
$ git reset -- hard
```




3.5: Working Remotely with GIT

Remotely Working with GIT

Cloning the data ----git clone <repo> <directory>

\$ git clone [employee@capgemini.com:myrepoone](mailto:employee@capgemini.com)

git-remote - Manage set of tracked repositories, Manage the set of repositories ("remotes") whose branches you track.

git-push - Update remote refs along with associated objects, Updates remote refs using local refs, while sending objects necessary to complete the given refs.



3.5: Working Remotely with GIT

Remotely Working with GIT

git-fetch - Download objects and refs from another repository

git-pull - Fetch from and integrate with another repository or a local branch

git-merge - Join two or more development histories together



Demo

Demo on GIT –Locally using git init, add, status, log

Demo on GIT- Remotelly using git remote, pull, push, fetch, clone



4.2: Jenkins Introduction

Jenkins Installation

Jenkins is easy to install.

Download Jenkins.war file from the Jenkins site:

- <http://jenkins-ci.org>

Jenkins can be installed in different ways:

- As a standalone application
- Windows Service
- Deploy it on any application server.



4.2: Jenkins Introduction

Jenkins Installation

To start Jenkins as a standalone application execute the below command in command prompt:

- `java -jar jenkins.war -- On Port 8080`
- `java -jar jenkins.war --ajp13Port=-1 --httpPort=8082 -On different port`
- Once Jenkins is started, the Jenkins dash board can be accessed by giving the following link in the browser

<http://localhost:8080/>

- To stop Jenkins, press Ctrl+C


Below are the steps to start Jenkins as a windows service

- First, start Jenkins as a standalone application and access Jenkins dash board.
- Click "Manage Jenkins" link available in Jenkins dash board.
- Select "Installation Directory" for Jenkins and click on Install.
- After installation, Jenkins will always run on portno 8080.



4.3: Creating Job in Jenkins

Jenkins Installation

 **Jenkins**

[?](#) **Rahul Vikash** | [log out](#)

Jenkins [ENABLE AUTO REFRESH](#)

 New Item

 People

 Build History

 Manage Jenkins

 My Views

 Credentials

[add description](#)

Welcome to Jenkins!

Please **create new jobs** to get started.

Build Queue

No builds in the queue.

Build Executor Status

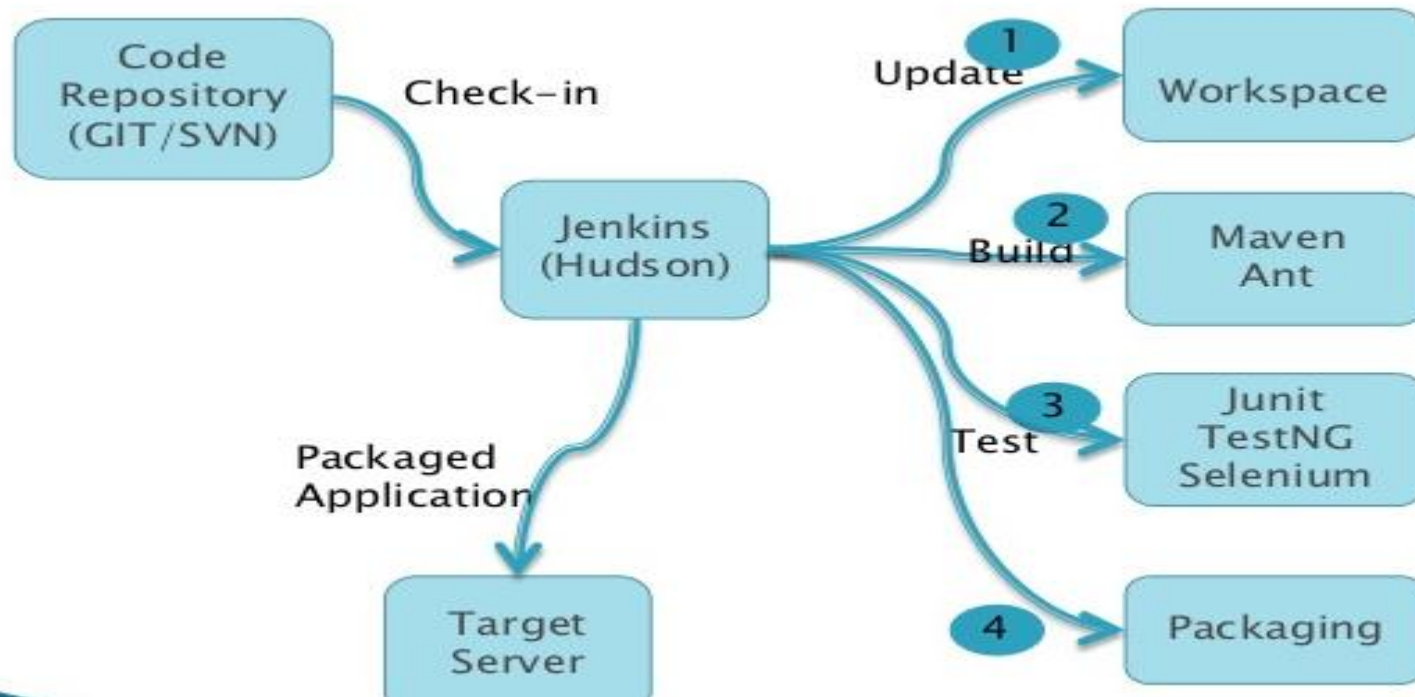
1 Idle
2 Idle



4.3: Creating Job in Jenkins

Git, Maven with Jenkins


Integration Git repository with Jenkins & Build using Maven





4.4: Adding plugin in Jenkins

Manage plugins

**Jenkins**


search


?


Rahul Vikash | log out


Jenkins


ENABLE AUTO REFRESH


 New Item

 People

 Build History

 **Manage Jenkins**

 My Views

 Credentials

Build Queue —

No builds in the queue.


Build Executor Status —


1 Idle


2 Idle


Click on Manage Jenkins


Manage Jenkins


 New version of Jenkins (2.32.2) is available for [download](#) ([changelog](#)). **Or Upgrade Automatically**


 [Configure System](#)
Configure global settings and paths.


 [Configure Global Security](#)
Secure Jenkins; define who is allowed to access/use the system.

 [Configure Credentials](#)
Configure the credential providers and types

 [Global Tool Configuration](#)
Configure tools, their locations and automatic installers.

 [Reload Configuration from Disk](#)
Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.

 [Manage Plugins](#)
Add, remove, disable or enable plugins that can extend the functionality of Jenkins.

 [System Information](#)
Displays various environmental information to assist trouble-shooting.

Click on Manage Plugin, Go to Available & such for plugins



4.4: Adding plugin in Jenkins

Manage plugins

Download Maven ,Git plugin

Download GIT Maven
Sonar plugin

| | | | |
|-------------------------------------|--|------------|-------------------------|
| <input checked="" type="checkbox"/> | Git client plugin | 2.2.1 | |
| | Shared library plugin for other Git related Jenkins plugins. | | |
| <input checked="" type="checkbox"/> | Git plugin | 3.0.5 | Downgrade to 3.0.4 |
| | This plugin integrates Git with Jenkins. | | |
| <input checked="" type="checkbox"/> | GitHub API Plugin | 1.84 | |
| | This plugin provides GitHub API for other plugins. | | |
| <input checked="" type="checkbox"/> | GitHub Integration Plugin | 0.1.0-rc20 | Downgrade to 0.1.0-rc19 |
| | Advanced trigger for GitHub Pull Requests and Branches. | | |
| <input checked="" type="checkbox"/> | GitHub plugin | 1.26.0 | Downgrade to 1.25.1 |
| | This plugin integrates GitHub to Jenkins. | | |
| <input checked="" type="checkbox"/> | Javadoc Plugin | 1.4 | |
| | This plugin adds Javadoc support to Jenkins. | | |
| <input checked="" type="checkbox"/> | JUnit Plugin | 1.19 | |
| | Allows JUnit-format test results to be published. | | |
| <input checked="" type="checkbox"/> | Mailer Plugin | 1.19 | |
| | This plugin allows you to configure email notifications for build results. This is a break-out of the original core based email component. | | |
| <input checked="" type="checkbox"/> | Matrix Project Plugin | 1.8 | |
| | Multi-configuration (matrix) project type. | | |
| <input checked="" type="checkbox"/> | Maven Integration plugin | 2.15.1 | |
| | This plugin provides an advanced integration for Maven 2/3 projects. | | |



4.4: Adding plugin in Jenkins

Manage plugins

Setting Configuration

- Go to Manage Jenkin->Global Tools Configuration

JDK installations

| | |
|--|-----------------------------------|
| JDK Name | JDK1.8 |
| JAVA_HOME | C:\Program Files\Java\jdk1.8.0_31 |
| <input type="checkbox"/> Install automatically | |
| <div>Put JDK Path</div> | |
| <div>Add JDK</div> | |
| List of JDK installations on this system | |

Git

Git installations

| | |
|--|----------------------------------|
| Git Name | Default |
| Path to Git executable | C:\Program Files\Git\bin\git.exe |
| <input type="checkbox"/> Install automatically | |
| <div>Use GIT.Exe Path</div> | |
| <div>Add Git</div> | |

Maven

Maven installations

| | |
|--|-----------------------------|
| Maven Name | Maven3.2.5 |
| MAVEN_HOME | D:\maven\apache-maven-3.2.5 |
| <input type="checkbox"/> Install automatically | |
| <div>Use Maven Path</div> | |




4.5: Creating Job with Maven & Git


Creating Maven Project


Create a Job, Give Job Name ,Select Maven Project & press Ok

Enter an item name

» Required field

**Freestyle project**
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

**Maven project**
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

**Multi-configuration project**
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.



4.5: Creating Job with Maven & Git

Creating Maven Project

Integrating Git with Jenkins by giving repository name & path of pom.xml in project

Source Code Management

☐ None
☒ Git

Repositories

Repository URL:

Credentials:

Advanced...

Add Repository

Branch to use:

Branch Specifier (blank for 'any'):

Add Branch

Repository browser:

Additional Behaviours:

Choose Source code management .Give the GIT Repository URL & Then press add give user name & Password of Github repository

Build

Root POM:

Goals and options:

Advanced...

In Build give path where pom.xml is there



4.5: Creating Job with Maven & Git

Creating Maven Project

Save & check in workspace all data fetched from Git Repository & then build

The screenshot displays the Maven IDE interface. On the left, a sidebar contains several icons and labels: 'Back to Dashboard' (green up arrow), 'Status' (magnifying glass), 'Changes' (pencil), 'Workspace' (blue folder), 'Wipe Out Current Workspace' (red X), 'Build Now' (green play button), 'Delete Maven project' (red circle with slash), 'Configure' (gear), and 'Modules' (document). The 'Workspace' tab is active, showing a folder icon, a text input field, and a green right arrow. Below this, there are icons for '.git', 'DemoOne', and '(all files in zip)'. A large black arrow points from the 'Build Now' icon in the sidebar to the 'Build now' text label on the right. At the bottom, a 'Build History' panel is visible, featuring a search bar with the text 'find', a list item '#1' dated '22-Feb-2017 10:07', and two RSS feed links: 'RSS for all' and 'RSS for failures'.

Back to Dashboard

Status

Changes

Workspace

Wipe Out Current Workspace

Build Now

Delete Maven project

Configure

Modules

Workspace of Maven_Git_Demo on master

Build now

Build History

find

#1 22-Feb-2017 10:07

RSS for all RSS for failures



4.5: Creating Job with Maven & Git

Creating Maven Project

```
T E S T S
-----
Running com.cg.demoone.AppTest
Tests run: 1, Failures: 0, Errors: 0, Time elapsed: 0.002 sec

Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0


[JENKINS] Recording test results
[INFO]
[INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ DemoOne ---
[INFO] Building jar: C:\Users\rv830051\.jenkins\jobs\Maven_Git_Demo\workspace\DemoOne\target\DemoOne-1.0-SNAPSHOT.jar
[INFO]
[INFO] --- maven-install-plugin:2.4:install (default-install) @ DemoOne ---
[INFO] Installing C:\Users\rv830051\.jenkins\jobs\Maven_Git_Demo\workspace\DemoOne\target\DemoOne-1.0-SNAPSHOT.jar to
C:\Users\rv830051\.m2\repository\com\cg\demoone\DemoOne\1.0-SNAPSHOT\DemoOne-1.0-SNAPSHOT.jar
[INFO] Installing C:\Users\rv830051\.jenkins\jobs\Maven_Git_Demo\workspace\DemoOne\pom.xml to C:\Users\rv830051\.m2\repository\com\cg\demoone\DemoOne\1.0-
SNAPSHOT\DemoOne-1.0-SNAPSHOT.pom
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 19.984 s
[INFO] Finished at: 2017-02-22T10:07:57+05:30
[INFO] Final Memory: 14M/34M
[INFO] -----
[JENKINS] Archiving C:\Users\rv830051\.jenkins\jobs\Maven_Git_Demo\workspace\DemoOne\pom.xml to com.cg.demoone/DemoOne/1.0-SNAPSHOT/DemoOne-1.0-
SNAPSHOT.pom
[JENKINS] Archiving C:\Users\rv830051\.jenkins\jobs\Maven_Git_Demo\workspace\DemoOne\target\DemoOne-1.0-SNAPSHOT.jar to com.cg.demoone/DemoOne/1.0-
SNAPSHOT/DemoOne-1.0-SNAPSHOT.jar
channel stopped
Finished: SUCCESS
```






4.5: Creating Job with Maven & Git




Creating Maven Project

Check & get all feedback when it changes
Build result obtained from the dashboard

 [add description](#)

| S | W | Name ↓ | Last Success | Last Failure | Last Duration | |
|---|---|--------------------------------|----------------------------------|--------------|---------------|---|
|  |  | Maven Git Demo | 8 min 3 sec - #1 | N/A | 55 sec |  |

Icon: [S](#) [M](#) [L](#)

[Legend](#)  [RSS for all](#)  [RSS for failures](#)  [RSS for just latest builds](#)



Demo

Demo on Maven-Git-Jenkins integration