

Jenkins

Continuous Integration Continuous Deployment

Steps Wise Process

1. Create an GITHUB account → Login GITHUB → create a repository → import repository from another repository → copy the URL of your repository.
2. Login to instance via putty ---> download JDK & tomcat and set class path and update path.
3. Login to instance via putty configure GIT
4. Login to Jenkins as admin --> manage Jenkins --> Global Tool Configuration --> add JDK installations, GIT Installations & Maven Installations
5. Go to login dashboard → New Item → Enter an item name --> free style project ---> ok

After Creating GITHUB account → Login to GITHUB Account

The screenshot shows the GitHub homepage for a user named jagarlamudirajesh. The browser address bar shows the URL https://github.com. The page header includes the GitHub logo, a search bar, and navigation links for Pull requests, Issues, Marketplace, and Explore. The main content area is divided into three sections:

- Repositories:** A list of repositories owned by jagarlamudirajesh, including trainingproject, project1, automaticdeployment, sampleproject, and autotest. A red box highlights this list, with the text "List of repo's in your account" below it.
- New:** A green button with a plus icon and the word "New", highlighted with a red box.
- This is GIT login page:** A large light blue box with the text "Click on 'New' for creating new account" and "Learn Git and GitHub without any code!". It includes a green "Read the guide" button and a grey "Start a project" button.

On the right side, there is a "Discover repositories" section listing popular repositories like web-platform-tests/wpt, springload/react-accessible-accordion, and mamedev/mame.

Enter your Repository name

Create a new repository

A repository contains all project files, including the revision history.

Owner

 jagarlamudirajesh34 ▾

Repository name *

cicd



Great repository names are short and memorable. Need inspiration? How about [improved-octo-enigma?](#)

Description (optional)

☒  **Public**

Anyone can see this repository. You choose who can commit.

☐  **Private**

You choose who can see and commit to this repository.

Public : any one can read your code and take your code. for write and submit code, you need to give permissions and public is completely free

Private : no read and write permissions, unless you give. its a cost one

☐ **Initialize this repository with a README**

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None ▾

Add a license: None ▾



Create repository

Importing code from Other Repo's

[Code](#) [Issues 0](#) [Pull requests 0](#) [Projects 0](#) [Wiki](#) [Insights](#) [Settings](#)

After creating your Repo, below options will appear

Quick setup — if you've done this kind of thing before

[Set up in Desktop](#) or [HTTPS](#) [SSH](#) `https://github.com/jagarlamudirajesh34/CICD.git`

This is your GIT REPO URL

this URL is given in JENKINS for pulling code from GIT

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# CICD" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/jagarlamudirajesh34/CICD.git
git push -u origin master
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/jagarlamudirajesh34/CICD.git
git push -u origin master
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Import code

In order to import code from other repo's user import code option

Provide URL of repo from which you are importing

Import your project to GitHub

Import all the files, including the revision history, from another version control system.

from below URL, i'm importing repo

Your old repository's clone URL

`https://github.com/pythonAwsBoto3/webapp.git`

Learn more about the types of [supported VCS](#).

Your existing repository

 `jagarlamudirajesh34/CICD`

[Change repository](#)

This is our REPO.

[Cancel](#)

Begin import

You will receive mail and importing status is shown below

Preparing your new repository

There is no need to keep this window open, we'll email you when the import is done.

 jagarlamudirajesh34/CICD

✓ Importing complete! Your new repository [jagarlamudirajesh34/CICD](#) is ready.

Go back to your GITHUB login dashboard and check the REPO and code

jagarlamudirajesh34 / CICD **Path of Repo in GITHUB** Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

No description, website, or topics provided. **This is your REPO** Edit

Total number of commits Manage topics **How many branches is having for this repo**

24 commits 1 branch 0 releases 2 contributors MIT

Branch: master New pull request Create new file Upload files Find file **Clone or download**

in which branch you are

src	Updated statement on index page
.gitignore	Updates .gitignore to exclude target
.travis.yml	Rename ,travis.yml to .travis.yml
LICENSE	Initial commit
README.md	Update README.md
pom.xml	Add log4j to pom.xml

Copy URL of repo and paste some where

Clone with HTTPS Use Git or checkout with SVN using the web URL
https://github.com/jagarlamudirajesh34/CICD

Open in Desktop Download ZIP

3 years ago 3 years ago

README.md

Login to Instance via putty and download JDK & tomcat, set path for both JDK and tomcat.

Go to webapps dir of tomcat and download Jenkins.war and start tomcat

This process is covered in previous class and for any info, please revise previous class materials

Add user in tomcat-users.xml which is in conf dir of tomcat

root@ip-1/2-31-93-24: /opt/tomcat/conf

to operate the "/manager/html" web application. If you wish to use this app, you must define such a user - the username and password are arbitrary. It is strongly recommended that you do NOT use one of the users in the commented section below since they are intended for use with the examples web application.

->

!—

NOTE: The sample user and role entries below are intended for use with the examples web application. They are wrapped in a comment and thus are ignored when reading this file. If you wish to configure these users for use with the examples web application, do not forget to remove the <!-- ... --> that surround them. You will also need to set the passwords to something appropriate.

->

!—

```
<role rolename="tomcat"/>
<role rolename="role1"/>
<user username="tomcat" password="<must-be-changed>" roles="tomcat"/>
<user username="both" password="<must-be-changed>" roles="tomcat,role1"/>
<user username="role1" password="<must-be-changed>" roles="role1"/>
```

->

```
user username="deployer" password="deployer" roles="manager-script"/>
user username="manager" password="manager" roles="manager-gui"/>
/tomcat-users>
```

Installing GIT Instance - use below commands to install

Login to instance and switch to root user and update instance

```
#sudo su root
```

```
#cd /
```

```
#apt-get update
```

```
#apt-get install git-core -y
```

```
#git --version
```

```
#git config --global user.name <your git account name>
```

```
#git config --global user.email <your git login mail>
```

```
#git config --list
```

```
#mkdir repos
```

```
#cd repos
```

```
#git init
```

```
#ls -a
```

```
#git clone <your git repo url>
```

Eg

```
#git clone https://github.com/jagarlamudirajesh34/CICD.git
```


```
#ls
```

```
#cd <repo name> eg cd CICD
```











```
#cd CICD/src/main/webapp
```


Login to Jenkins and configure JDK, GIT MAVEN


← → ↻ ⓘ Not secure | 3.83.109.115:8080/jenkins/manage ☆ ⓘ ⓘ ⓘ ⓘ

 **Jenkins** [admin](#) | [log](#)

Jenkins ▸ [ENABLE AUTO REF](#)

-  New Item
-  People
-  Build History
-  Project Relationship
-  Check File Fingerprint
-  **Manage Jenkins**
-  My Views
-  Lockable Resources
-  Credentials
-  New View

Build Queue 
No builds in the queue.







Build Executor Status 
1 Idle
2 Idle

Manage Jenkins

Login page of Jenkins

It appears that your reverse proxy set up is broken. [More Info](#) [Dismiss](#)

⚠ Your container doesn't use UTF-8 to decode URLs. If you use non-ASCII characters as a job name etc, this will cause problems. See Containers and Tomcat i18n for more details.

-  **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**

Default global settings provider Use default maven global settings ▾

JDK

JDK installations

Add JDK



Name

JDK1.8

JAVA_HOME

/opt/jdk1.8

☐ Install automatically

if JDK is already installed in your insatnce, then you the location where you installed and uncheck install automatically
if Not installed, check install automatically



Delete JDK

Add JDK

List of JDK installations on this system

Git

Git installations



Git

Name

Default

Path to Git executable

git

☐ Install automatically



Delete Git

Add Git ▾

Gradle

Add Git ▾

Gradle

Gradle installations

Add Gradle

List of Gradle installations on this system

Ant

Ant installations

Add Ant

List of Ant installations on this system

Maven

Maven installations

Add Maven

Maven

Name

☒ Install automatically

☐ Install from Apache

Version



after changes -- click on apply
and save

Add Installer ▾

Add Maven


List of Maven installations on this system

Delete Installer


Delete Maven


Now, create a JOB for CICD


← → ↻ ⓘ Not secure | 3.83.109.115:8080/jenkins/ ☆ ⓘ ⓘ ⓘ ⓘ


 **Jenkins** ⓘ admin | log o


Jenkins ▸ [ENABLE AUTO REFRES](#)


 New Item


 People


 Build History


 Project Relationship


 Check File Fingerprint

 Manage Jenkins

 My Views

 Lockable Resources


 Credentials

 New View

Welcome to Jenkins!

Please [create new jobs](#) to get started.

Now, create a job for CICD

 [add description](#)

Build Queue

Enter an item name

AutomaticDeployment

* 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.



Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.



Multi-configuration project

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



Folder

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.



GitHub Organization

Scans a GitHub organization (or user account) for all repositories matching some defined markers.



Multibranch Pipeline

Creates a set of Pipeline projects according to detected branches in one SCM repository.

OK

Now, configure GIT, job triggering details, Tomcat details

The screenshot shows the Jenkins 'Source Code Management' configuration page. The 'Source Code Management' tab is selected and highlighted with a red box. Under the 'Repositories' section, the 'Git' radio button is selected and highlighted with a red box. The 'Repository URL' field is highlighted with a red box and contains the text 'https://github.com/rsreddys/automationDeployment.git'. A red annotation 'This is your REPO url from GITHIB account' points to this field. The 'Credentials' dropdown is set to '- none -'. The 'Branches to build' section has a 'Branch Specifier (blank for \'any\')' field set to '*/master', which is also highlighted with a red box. A red annotation 'from which branch jenkins has to take code' points to this field. The 'Repository browser' is set to '(Auto)'. Other tabs visible include 'General', 'Build Triggers', 'Build Environment', 'Build', and 'Post-build Actions'.

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

Source Code Management

☐ None
☒ **Git**

Repositories

Repository URL **https://github.com/rsreddys/automationDeployment.git**

Credentials - none - Add

Advanced...

Add Repository

Branches to build

Branch Specifier (blank for 'any') ***/master**

Add Branch

Repository browser (Auto)

This is your REPO url from GITHIB account

from which branch jenkins has to take code

General

Source Code Management

Build Triggers

Build Environment

Build

Post-build Actions

Repository browser

(Auto)

Additional Behaviours

Add ▾

☐ Subversion

Build Triggers

☐ Trigger builds remotely (e.g., from scripts)

☐ Build after other projects are built

☐ Build periodically

☐ GitHub hook trigger for GITScm polling

☒ Poll SCM

Schedule

* * * * *

 **Do you really mean "every minute" when you say "* * * * *"? Perhaps you meant "H * * * * *" to poll once per hour**

Would last have run at Saturday, February 9, 2019 10:32:43 AM UTC; would next run at Saturday, February 9, 2019 10:32:43 AM UTC.

General

Source Code Management



Build Triggers

Build Environment

Build

Post-build Actions

Build Environment

- ☐ Delete workspace before build starts
- ☐ Use secret text(s) or file(s) 
- ☐ Abort the build if it's stuck
- ☐ Add timestamps to the Console Output
- ☐ With Ant 

Build

 Invoke top-level Maven targets  

Maven Version **Maven3.6.0** ▼

Goals **package** ▼

Advanced...

Add build step ▼

Configuring tomcat

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

Build Environment

- ☐ Delete workspace before build starts
- ☐ Use secret text(s) or file(s)
- ☐ Abort the build if it's stuck
- ☐ Add timestamps to the Console Output
- ☐ With Ant

Build

- Aggregate downstream test results
- Archive the artifacts
- Build other projects
- Publish JUnit test result report
- Record fingerprints of files to track usage
- Git Publisher
- Deploy war/ear to a container**
- E-mail Notification
- Editable Email Notification
- Set GitHub commit status (universal)
- Set build status on GitHub commit [deprecated]
- Delete workspace when build is done

Add post-build action

If, "Deploy war/ear to a container" is not available in list/dropdown, go to manage plugin and install by searching "deploy to container"

Pointing where the .war / build is present

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

Invoke top-level Maven targets

Maven Version

Maven3.6.0

Goals

package

Advanced...

Add build step ▾

Post-build Actions

Deploy war/ear to a container

WAR/EAR files

target/mvn-hello-world.war

Context path

mvn-hello-world

Containers

Add Container ▾

Deploy on failure

☐

Add post-build action ▾

giving the path where the war id present

click on add container -- to give server details

Giving Server Details

Post-build Actions

Deploy war/ear to a container X

WAR/EAR files ?

Context path ?

Containers **Add Container** ▼

Deploy on failure ☐

Add post-build action ▼

- GlassFish 3.x
- JBoss AS 3.x
- JBoss AS 4.x
- JBoss AS 5.x
- JBoss AS 6.x
- JBoss AS 7.x
- Tomcat 4.x
- Tomcat 5.x
- Tomcat 6.x
- Tomcat 7.x**
- Tomcat 8.x

Save **Apply**

Select Which server you installed in your instance

Post-build Actions

Deploy war/ear to a container

WAR/EAR files

Context path

Containers

Tomcat 7.x

Credentials

 Add

Tomcat URL

 Jenkins



Jenkins Credentials Provider: Jenkins

Add Credentials

Domain

Kind

Scope

Username

Password

ID

Description

Provide user name and password of server,
whose role is "manager-script"

Post-build Actions


Deploy war/ear to a container

WAR/EAR files

Context path

Containers

Tomcat 7.x

Credentials 

Tomcat URL

Select credentials and give
server url

 Add Container

Deploy on failure ☐

Add post-build action

Save

Apply

- New Item
- People
- Build History
- Project Relationship
- Check File Fingerprint
- Manage Jenkins
- My Views
- Lockable Resources
- Credentials
- New View

All +					
S	W	Name ↓	Last Success	Last Failure	Last Duration
		AutomaticDeployment	N/A	N/A	N/A

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

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

this is JOB created for CICD

Build Queue

No builds in the queue.

ANY builds are in queue,
you can observe here

Build Executor Status

- 1 Idle
- 2 Idle

Goto putty → goto git repo where you configured and open index.jsp page
and do some changes and add, commit and push to repo

```
root@ip-172-31-93-24:/repos/automaticdeployment/src/main/webapp# ls
index.jsp WEB-INF
root@ip-172-31-93-24:/repos/automaticdeployment/src/main/webapp# vi index.js
root@ip-172-31-93-24:/repos/automaticdeployment/src/main/webapp# vi index.js
root@ip-172-31-93-24:/repos/automaticdeployment/src/main/webapp# git add index.jsp
root@ip-172-31-93-24:/repos/automaticdeployment/src/main/webapp# git commit -m "comments"
[master cb72631] comments
1 file changed, 1 insertion(+), 1 deletion(-)
root@ip-172-31-93-24:/repos/automaticdeployment/src/main/webapp# git push origin master
Username for 'https://github.com': jagarlamudirajesh34
Password for 'https://jagarlamudirajesh34@github.com':
Counting objects: 6, done.
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 478 bytes | 0 bytes/s, done.
Total 6 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/jagarlamudirajesh34/automaticdeployment.git
741022f..cb72631 master -> master
root@ip-172-31-93-24:/repos/automaticdeployment/src/main/webapp#
```

**After doing this, your
job will start triggering
and deploys to server**

Commands is given in next slide if you not get from above snap

Go to dir in which index.jsp is available

```
#vi index.jsp
```

Change some code

```
#git add index.jsp
```

```
#git commit -m "Comments"
```

```
#git push origin master
```

Provide your git credentials

For cross check, go to webapps dir of tomcat and see the war is deployed or not.

Open tomcat in browser : <ip>:8080/mvn-hello-world

```
writing objects: 100% (6/6), 478 bytes | 0 bytes/s, done.
total 6 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
to https://github.com/jagarlamudirajesh34/automaticdeployment.git
   741022f..cb72631  master -> master
oot@ip-172-31-93-24:/repos/automaticdeployment/src/main/webapp# ls
index.jsp  WEB-INF
oot@ip-172-31-93-24:/repos/automaticdeployment/src/main/webapp# cd /
oot@ip-172-31-93-24:/# cd opt/tomcat7/webapps/
oot@ip-172-31-93-24:/opt/tomcat7/webapps# ls
docs  examples  host-manager  jenkins  jenkins.war  manager  mvn-hello-world  mvn-hello-world.war  ROOT
oot@ip-172-31-93-24:/opt/tomcat7/webapps#
```