

## ASSIGNMENT 6 DevOps

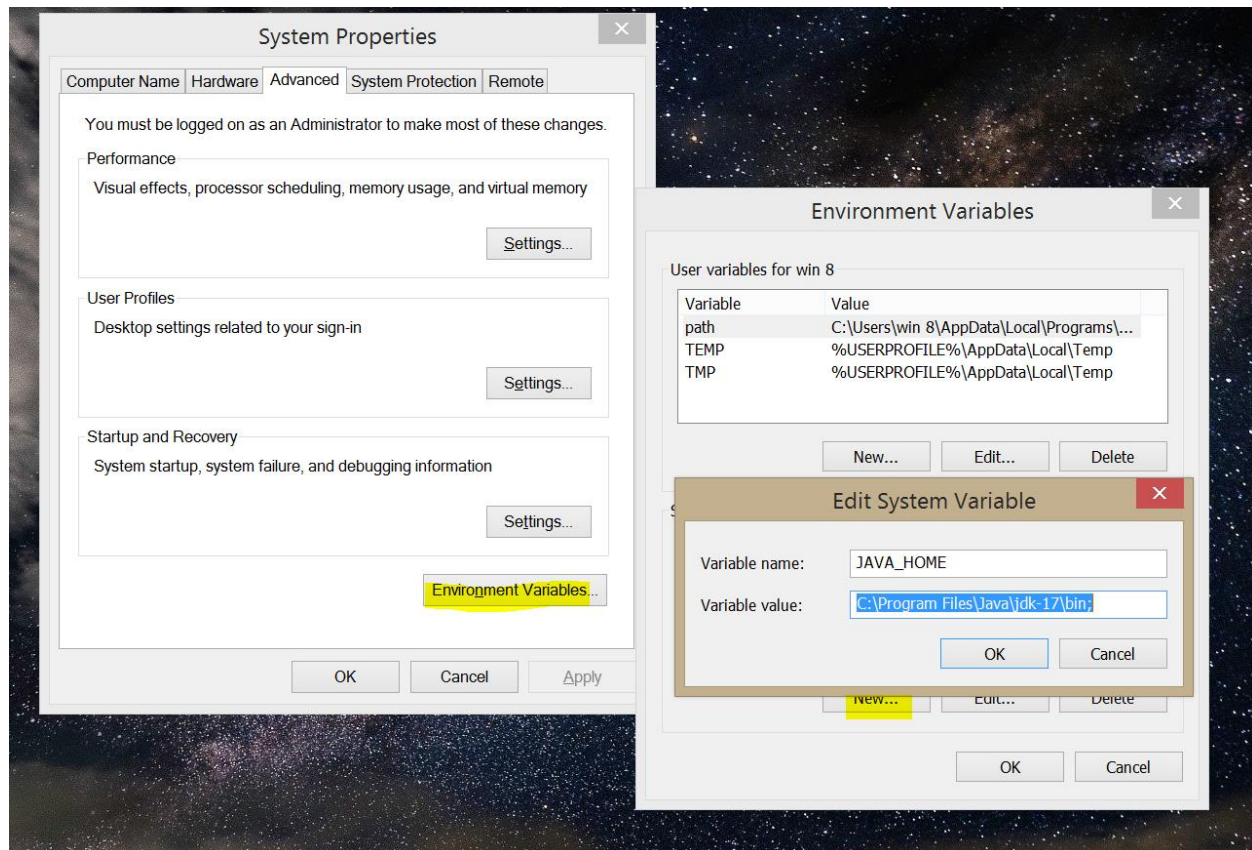
IT\_BE\_41\_Himanshu Shukla

Create a web application, Commit changes to GitHub/GitLab repository. Use GitHub/GitLab, Maven, Tomcat with Jenkins. Automate Deployment of the above-mentioned application to a container using Jenkins plugin, "Deploy to container".

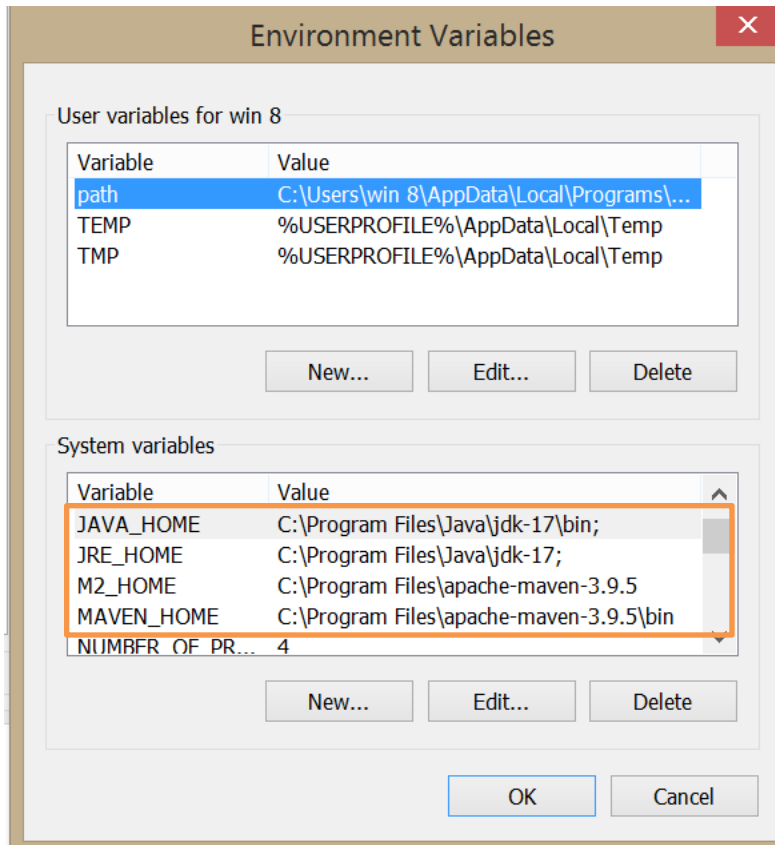
**Prerequisite:** Eclipse, Gitbash, apache maven([apache-maven-3.9.5-bin.zip](#)), tomcat server([32-bit/64-bit Windows Service Installer](#) ([pgp](#), [sha512](#))), **mandatory note:while downloading change the tomcat port to 8081**),jenkins, github account ,os:windows8+

1)After downloading maven and tomcat make sure the folder are in C drive under program files and set up the environment variable

Search edit environment variable and place the jdk/bin directory



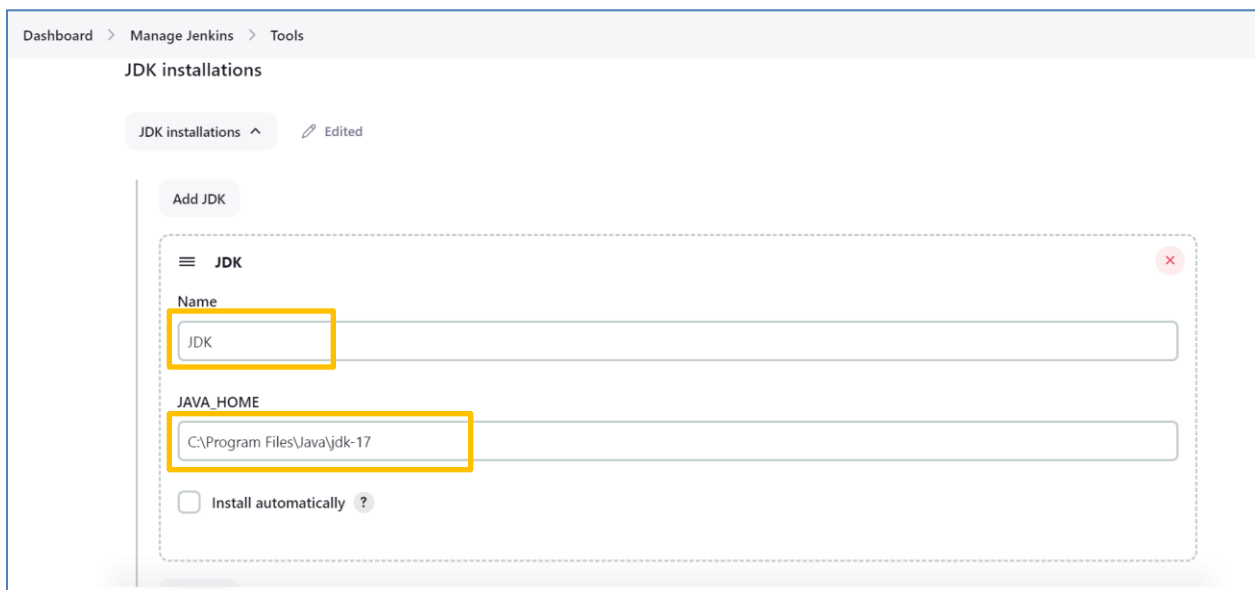
Like this place all four environment variable



Click ok and save

2)open jenkins localhost:8080 , go to DashBoard>Manage Jenkins>Tools(global tool)

Click on jdk installation add the name and paste jdk directory



Click on git installation and write the name and place the path of git

Dashboard > Manage Jenkins > Tools

### Git installations

≡ Git

Name

default

Path to Git executable ?

C:\Program Files\Git\bin\git.exe

☐ Install automatically ?

Add Git ▾

Click on maven installation and add the name and place the directory

Maven installations

Maven installations ^ Edited

Add Maven

≡ Maven

Name

Maven

MAVEN\_HOME

C:\Program Files\apache-maven-3.9.5

☐ Install automatically ?

Click on Apply and save

3)Again go to Manage jenkins>Plugin and install the “**deploy to container plugin**”

Dashboard > Manage Jenkins > Plugins

## Plugins

Updates 26

Available plugins

Installed plugins

Advanced settings

Q deploy

Name ↓ Enabled

Deploy to container Plugin 1.16

This plugin allows you to deploy a war to a container after a successful build.  
Glassfish 3.x remote deployment

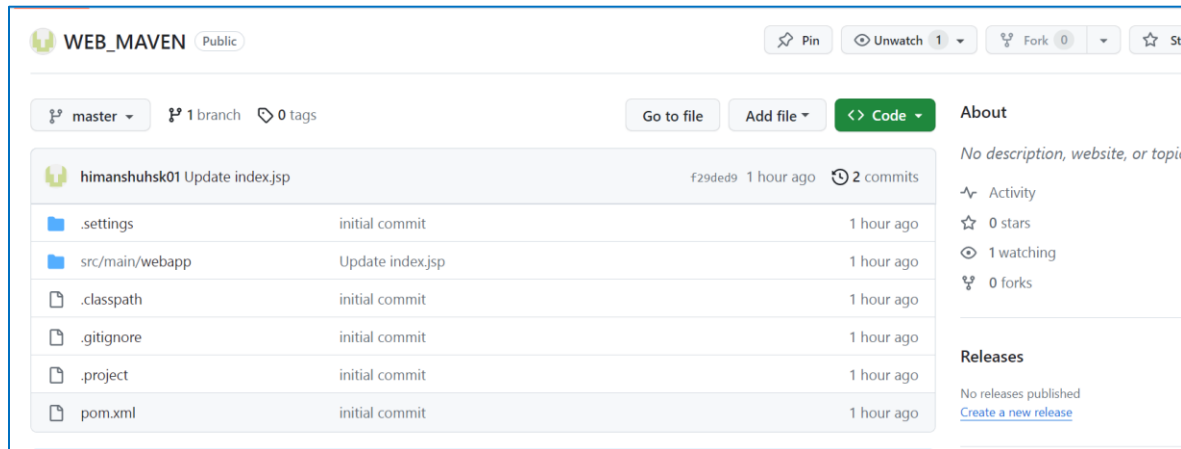
Report an issue with this plugin

☒

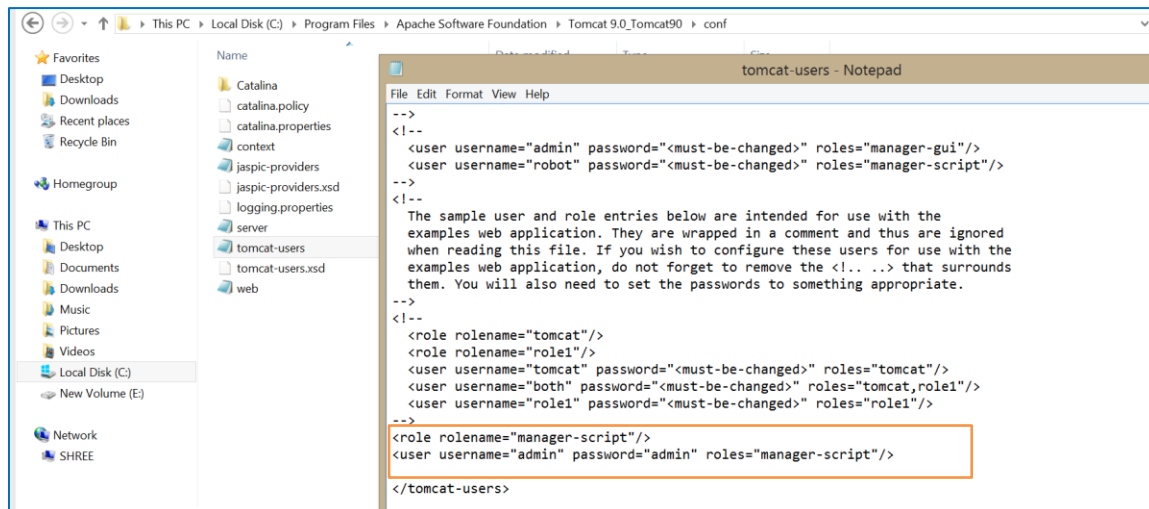
✕

4) go to Eclipse create a maven project([creation link](#)) and push to git hub([link](#))

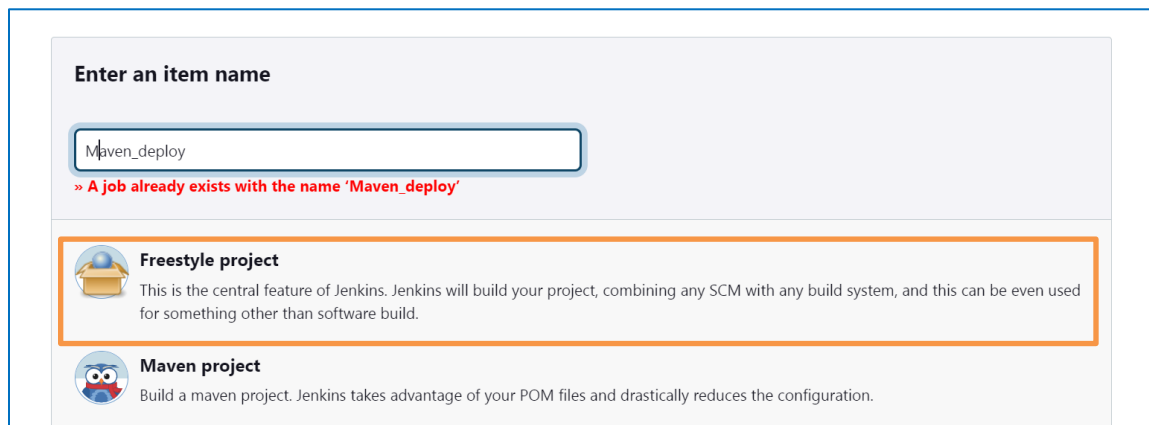
Here is my repo([https://github.com/himanshuhs01/WEB\\_MAVEN](https://github.com/himanshuhs01/WEB_MAVEN))



5) go to C:>Program Files>Apache Software Foundation>Tomcat 9.0\_Tomcat90>conf and under conf click on tomcat-users.xml add this lines



6) go to Dashboard>new item write the project name and choose free style project



7)add the github account which you created

The screenshot shows the Jenkins Configuration page for Source Code Management. The left sidebar has a 'Configure' section with options: General, Source Code Management (selected), Build Triggers, Build Environment, Build Steps, and Post-build Actions. The main content area is titled 'Source Code Management' and has two radio buttons: 'None' and 'Git' (selected). Below the 'Git' button is a 'Repositories' section with a 'Repository URL' field containing 'https://github.com/himanshuhs01/WEB\_MAVEN.git' and a 'Credentials' dropdown menu set to '- none -'. There is an '+ Add' button and an 'Advanced' dropdown at the bottom.

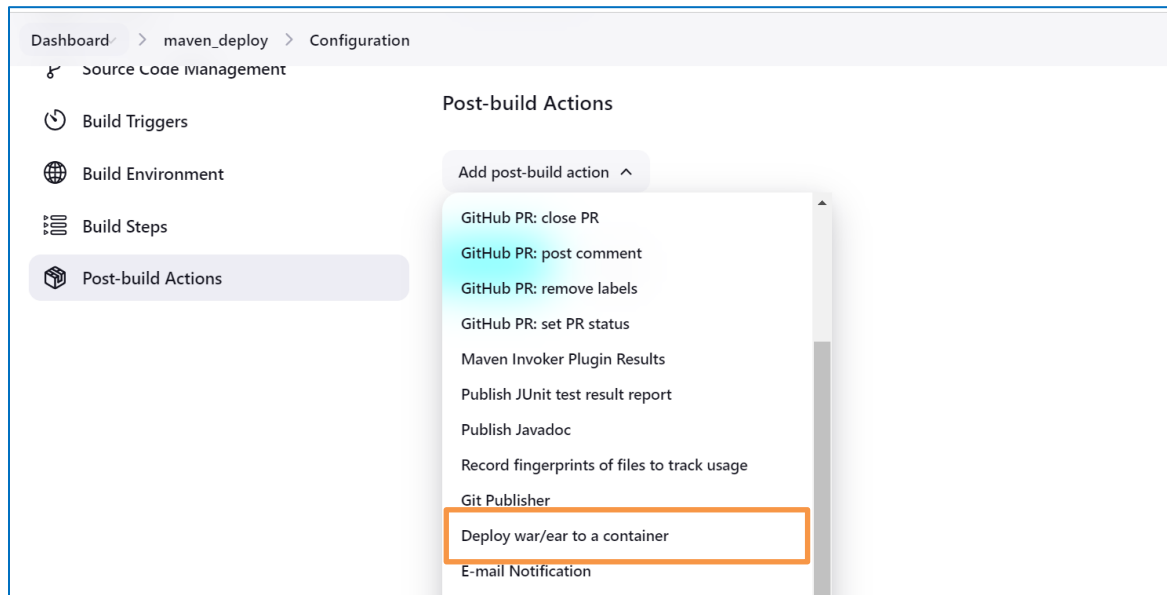
under build Triggers go to poll scm and write five \* with single space

The screenshot shows the Jenkins Configuration page for Build Triggers. The left sidebar has a 'Configure' section with options: General, Source Code Management, Build Triggers (selected), Build Environment, Build Steps, and Post-build Actions. The main content area is titled 'Build Triggers' and has several checkboxes: 'Trigger builds remotely (e.g., from scripts)', 'Build after other projects are built', 'Build periodically' (selected), 'GitHub Branches', 'GitHub Pull Requests', 'GitHub hook trigger for GITScm polling', and 'Poll SCM' (checked). Below the 'Poll SCM' checkbox is a 'Schedule' field containing '\*\*\*\*\*'. A warning message at the bottom says: 'Do you really mean "every minute" when you say "\*\*\*\*\*"? Perhaps you meant "H \* \* \* \*"'.

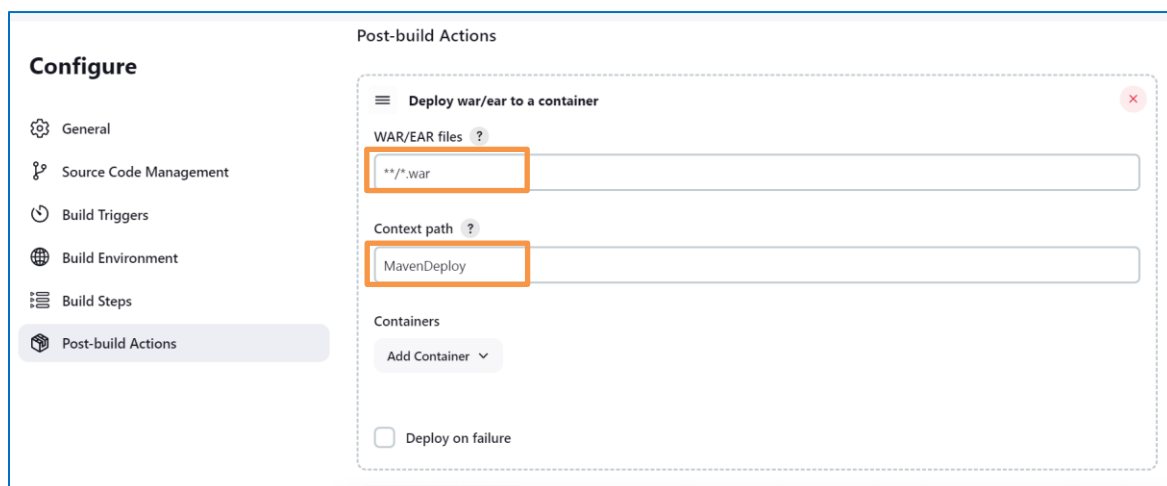
Add a build step select Maven and in goal write “clean install”

The screenshot shows the Jenkins Configuration page for Build Steps. The left sidebar has a 'Configure' section with options: General, Source Code Management, Build Triggers, Build Environment, Build Steps (selected), and Post-build Actions. The main content area is titled 'Build Steps' and has a 'Build Steps' section with a 'Maven Version' dropdown menu set to 'Maven' and a 'Goals' dropdown menu set to 'clean install'. There is an 'Advanced' dropdown and an 'Add build step' button at the bottom.

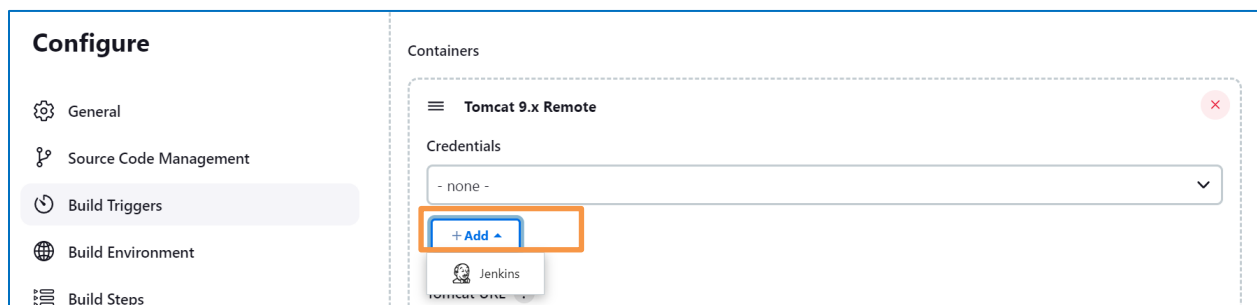
In post build select deploy war/ear to a container



Write this `**/*.war` and in context path you can write any name but keep in mind for deployment



Select add container and select the tomcat version which you have installed (in this case it is 9.0 version)



Now write the username(admin) and password(admin) which you have written in the tomcat-user.xml click on add

**Configure**

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

Scope ?

Global (Jenkins, nodes, items, all child items, etc)

Username ?

admin

☐ Treat username as secret ?

Password ?

\*\*\*\*

ID ?

Now you will be able to see that under credentials click on it and write the tomcat url as it is (as suggested while downloading you have to change the port to 8081)

**Configure**

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

Containers

Tomcat 9.x Remote

Credentials

none -

admin/\*\*\*\*\*

deployer/\*\*\*\*\*

Tomcat URL ?

http://localhost:8081/manager/text

Advanced

Now save and run it is successfully build

Dashboard > maven\_deploy > #2 > Console Output

Status

Changes

Console Output

View as plain text

Edit Build Information

Delete build '#2'

Git Build Data

Previous Build

**Console Output**

Started by user Himanshu

Running as SYSTEM

Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\maven\_deploy

The recommended git tool is: NONE

No credentials specified

> C:\Program Files\Git\bin\git.exe rev-parse --resolve-git-dir

C:\ProgramData\Jenkins\jenkins\workspace\maven\_deploy\.git # timeout=10

Fetching changes from the remote Git repository

> C:\Program Files\Git\bin\git.exe config remote.origin.url https://github.com/himanshu01/WEB\_MAVEN.git # timeout=10

Fetching upstream changes from https://github.com/himanshu01/WEB\_MAVEN.git

> C:\Program Files\Git\bin\git.exe --version # timeout=10

> git --version # 'git version 2.39.0.windows.2'

> C:\Program Files\Git\bin\git.exe fetch --tags --force --progress -- https://github.com/himanshu01/WEB\_MAVEN.git +refs/heads/\*:refs/remotes/origin/\* # timeout=10

> C:\Program Files\Git\bin\git.exe rev-parse "refs/remotes/origin/master"{commit}" # timeout=10

Now to deploy just write “<http://localhost:8081/MavenDeploy/>” in browser

