# Automating your CI/CD Stack with Java and Groovy Hands on Lab

Automating your CI/CD Stack with Java and Groovy Hands on Lab 1

Step 0: Install tools 2

Docker 2

Docker images 2

Groovy 3

Disk space 3

The ability to run curl 3

Step 1: Install Tools 4

Install Nexus 4

Install Jenkins 4

Final step: Cleanup 4

Stop containers 4

If want to reclaim disk space 4

References 5

## Step 0: Install tools

This lab requires the following:

1. Docker
2. Docker images
3. Groovy
4. The ability to run curl

### Docker

To install Docker, follow the instructions at: <https://docs.docker.com/install/>

Validation:

At the command line, run **docker –version**. (Small version differences are ok in the output)

$ docker --version

Docker version 18.06.0-ce, build 0ffa825

### Docker images

These images take up 1.2GB combined. So as not to tax the conference network, please pull them in advance. This will also let you deal with any corporate internet proxies while you are still at work and can ask for help.

At the command line run:

**docker pull sonatype/nexus3:3.13.0**

**docker pull jenkins/jenkins:2.140**

**Note: “latest” will probably work. These version numbers are the ones I tested with.**

$ docker pull sonatype/nexus3:3.13.0

3.13.0: Pulling from sonatype/nexus3

256b176beaff: Pull complete

18d124afa1e9: Pull complete

9bb412307f82: Pull complete

Digest: sha256:19d186d5bc8be1ea4f7bae72756baa830e79bf20aae0e9e7b1a0c7d3ce7ac136

Status: Downloaded newer image for sonatype/nexus3:3.13.0

$ docker pull jenkins/jenkins:2.140

2.140: Pulling from jenkins/jenkins

55cbf04beb70: Pull complete

1607093a898c: Pull complete

9a8ea045c926: Pull complete

d4eee24d4dac: Pull complete

c58988e753d7: Pull complete

794a04897db9: Pull complete

70fcfa476f73: Pull complete

806029475e0c: Pull complete

67959b355155: Pull complete

4d217ccd3d4c: Pull complete

0261bb88a4a5: Pull complete

96f2a3ae5539: Pull complete

f6bf99db32d5: Pull complete

bb47d4bbb0e1: Pull complete

4b48ec5d60cf: Pull complete

7280a8dfb767: Pull complete

91091f8d44ca: Pull complete

8ca02cad320f: Pull complete

46009bfec329: Pull complete

f9860b79812e: Pull complete

89ac8103ea67: Pull complete

Digest: sha256:161cb25fbb23a1c5ac5fdd0feebd713edd62c235e199e68b34d1a78205a42da7

Status: Downloaded newer image for jenkins/jenkins:2.140

### Groovy

Follow the instructions at: <http://groovy-lang.org/install.html>

Note: Please download Groovy 2.X rather than 3.X. Version 3.X is in alpha at the time of writing this lab.

Validation:

At the command line, run **groovy –version**.(Small version differences are ok in the output)

$ groovy -version

Groovy Version: 2.5.2 JVM: 1.8.0\_45 Vendor: Oracle Corporation OS: Mac OS X

### Disk space

This lab uses about 1.5 GB of disk space (not counting Docker itself.) The last step of the lab explains how to recover disk space.

### The ability to run curl

If you are running Mac, Linux or Windows 10, you don’t need to do anything. If you are running an older version of Windows, you can install <https://www.cygwin.com>.

Validation:

At the command line, run **curl https://www.oracle.com/code-one/index.html** and ensure the output isn’t an error message.

## Step 1: Start tools

In this lab, we use basic Docker to make setup simple. We do not set up a volume to persist the data since the emphasis is Groovy setup.

### Install Nexus

At the command line, run:

**docker run -d -p 8081:8081 --name nexus sonatype/nexus3:3.13.0**

Validation

In a browser, go to <http://localhost:8081>. If you do not see Nexus, wait 15 seconds and try again. (It could take up to a minute)

### Install Jenkins

At the command line, run

**docker run -d -p 8080:8080 --name jenkins jenkins/jenkins:2.140**

### Unlock Jenkins

1. At the command line, run:

**docker exec -it jenkins bash**

1. Print the password:

**cat /var/jenkins\_home/secrets/initialAdminPassword**

1. In a browser, go to [http://localhost:8080](http://localhost:8081)



1. Paste in the password and click continue
2. Allow Jenkins to install the suggested plugins
3. Pick a username/password for your admin test user. Enter all the fields and click “Save and Continue”
4. Click to Finish
5. Continue to Jenkins
6. In the command line window, type “exit” to get out of bash.



## Step 2: Basic Nexus Scripting

### Run your first Groovy script in Nexus

Let’s confirm what version of Groovy is being used.

1. Login using admin/admin123.
2. Click the gear icon to get the administration options



1. In the left navigation, scroll down and choose the last time “Tasks”
2. Click “Create Task”
3. Choose “Admin: Execute Script”
4. Enter a task name.
5. For the script source, enter:

**log.info 'Hello Oracle Code One!** **'**

**log.info GroovySystem.version**

1. For task frequency, choose “Manual”
2. Save
3. Click Run



1. Click “Yes” to confirm
2. Click Summary tab. Depending on how fast you are, it will either say Running or Waiting. Refresh if it is still running to confirm it completes.



1. In the left navigation go to Support > Logging > Log Viewer
2. Notice your logging is there:



### Looking in the task log on the command line

1. In a new command line tab/window, run:

**docker exec –it nexus bash**

1. Print the log file:

**cat /nexus-data/log/tasks/<script name from the log>**

1. Confirm you see 8 lines.
2. Now find the main nexus.log and run

**tail –f nexus.log**

1. In your browser, re-run your Groovy task. How many lines do you see added to the tail output? It’s not 8 lines. Can you figure out the difference between the nexus log and task log?

### Writing your own script in Nexus

Now try to write your own Groovy script that prints out a countdown. The output includes the arrows and should be:

--> 10

--> 9

--> 8

--> 7

--> 6

--> 5

--> 4

--> 3

--> 2

--> 1

Blast off!

Tip for Groovy developers: the log.info method takes a Java String and not a GString.

Check the output in both a browser (nexus log) and command line (task log).

**Challenge**: can you write the Groovy code in two lines of no more than 40 characters each? You can use <https://www.lettercount.com> to check the number of characters.

### log vs println

1. In your prior example, change **log.info** to **println**.
2. Run the task
3. Check the output in both the browser and command line task log. Where do you see it?

## Step 3: Basic Jenkins Scripting

### Run your first Groovy script in Jenkins

Let’s confirm what version of Groovy is being used in Jenkins.

1. On the left navigation, click “Manage Jenkins”
2. Click “Script Console”
3. Write:

**println 'Hello Oracle Code One!** **'**

**println GroovySystem.version**

1. The result is right below the text area:



1. Note that this is not the same version as Nexus used; but it is close enough to not cause us difficulties.

### Setup Jenkins so can run jobs with Groovy steps

We need the Groovy plugin to create jobs with Groovy steps.

1. In the left navigation, click “Manage Jenkins”
2. Click “Manage Plugins”
3. Click the “Available” tab
4. Select “Groovy” (note the plugins are not in alphabetical order)



1. Click the button “Download now and install after restart”
2. Check to restart



1. Refresh the page after a few seconds
2. Click “Manage Jenkins” again
3. Click “Global Tool Configuration”
4. Click “Add Groovy”



1. Set the name to “Groovy 2.X”. Leave the version as the default. (This gives us yet another slightly different of Groovy, but this is fine as long as we stay on the 2.X series.)



1. Click “Save”

### Run your first job containing Groovy scripting

1. In the left navigation, click “New item”
2. Enter “my-first-script” as the job name and choose “Freestyle Project”



1. Scroll down and click “Ok”
2. Add a Groovy build step



1. Choose “Groovy 2.X” to run
2. In the textbox, enter the code

**println 'Hello Oracle Code One!** **'**

**println GroovySystem.version**

****

1. Click “Save”
2. Click “Build Now”
3. Click your job number and view the console output.

Tip: You can click the blue circle to go directly to the console.



1. Note that Jenkins installed Groovy the first time it was needed.



### Writing your own script in Jenkins

This time we are going to write a script where you fill in the blank to print out all the strings that end with “day”.

def list = ['monday', 'wednesday', 'chocolate', 'friday']

println \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Use the Jenkins admin script console to test your script until you are happy with it. Then create a job named “days” that runs the same Groovy code.

Tip for Java developers: Groovy uses the method grep where Java uses filter.

## Final step: Cleanup

### Stop containers

Remember you can enter the first couple characters of the hash rather than the whole thing.

**docker ps**

**docker stop <hash1>**

**docker stop <hash2>**

### If want to reclaim disk space

Remove containers and images

**docker ps –a**

**docker rm <hash1>**

**docker rm <hash2>**

**docker images**

**docker rmi <hash1>**

**docker rmi <hash2>**

## References

<https://hub.docker.com/r/jenkins/jenkins>

<https://github.com/jenkinsci/docker/blob/master/README.md>

<https://hub.docker.com/r/sonatype/nexus3>

<https://support.sonatype.com/hc/en-us/articles/115015812727-Nexus-3-Groovy-Script-development-environment-setup>