CLOUDFOUNDRY

Pivotal Cloud Foundry Developer

Lab Instructions

Application Deployment using Pivotal Cloud Foundry

Version 1.11.a

Pivotal

Copyright Notice

Pivotal

Table of Contents

Chapter 1. Continuous Delivery

Estimated Time: 45 minutes

1.1. Preface

There's something special about setting up the automation to produce a new build artifact for an application as soon as a commit is made to version control, and then automatically deploying that application to cloudfoundry. In this lab, you'll do precisely that. You're going to use the Jenkins CI tool for the build job, and CloudFoundry as your target deployment platform. Enjoy.

1.2. Exercises

1.2.1. What is Continuous Delivery?

For a brief explanation, click here.

1.2.2. Setup Github Repository

1. Locate the https://github.com/pivotal-education/pcf-articulate-code on GitHub
2. Use the `Fork` button to make your own private copy of the project in your own GitHub account.

1.2.3. Install Jenkins

Install Jenkins on your local machine or use AWS Jenkins AMI.

1.2.3.1. Local Installation

1. Prior to installing Jenkins the following should have been installed. If not, you will need to do so now.
   - JDK 1.8
   - git
2. A copy of `jenkins.war` should have been provided for you - skip to next step. If not, use this link to

Download `Jenkins` version 1.642.4 war file.

3. Copy the file to folder: `pivotal-cloud-foundry-developer-workshop/jenkins/`

4. Change the working directory to where you copied the `jenkins.war` file.

```
Unix, Mac: cd ~/pivotal-cloud-foundry-developer-workshop/jenkins/
Windows:   cd C:\pivotal-cloud-foundry-developer-workshop\jenkins\
```

5. Run Jenkins.

```
java -jar jenkins.war
```

6. Open a browser to http://localhost:8080.

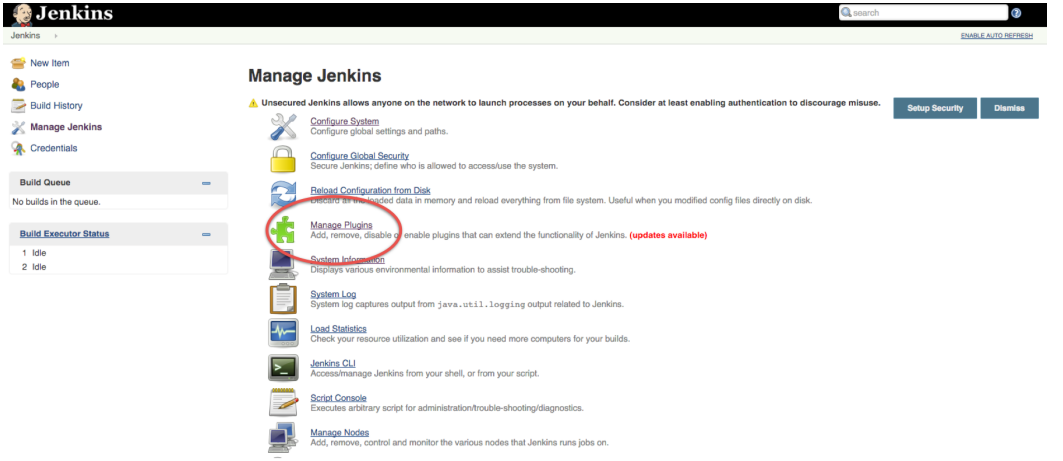
1.2.3.2. AWS Jenkins AMI

Provision a Jenkins instance on AWS.

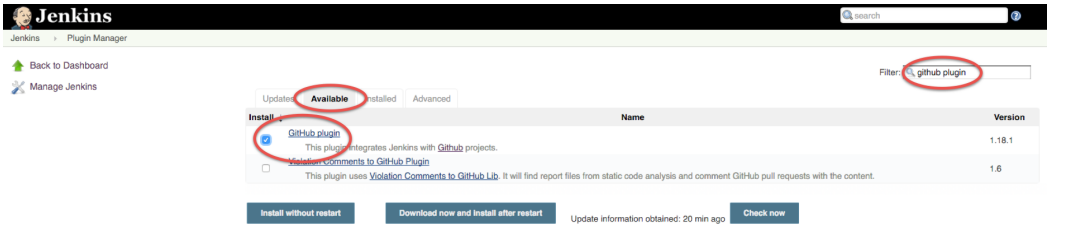
1.2.4. Configure Jenkins

You may skip this section if using the AWS Jenkins AMI.

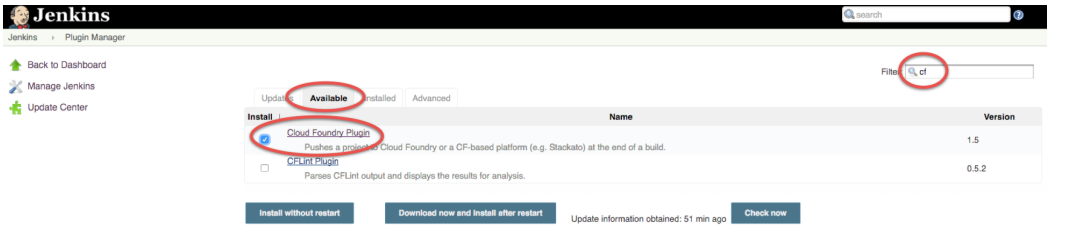
1. Click on the `Manage Jenkins` link, then click the `Manage Plugins` link.



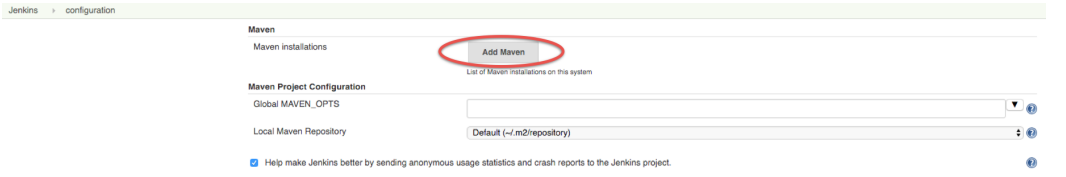
2. Click on the `Available` tab, and find the GitHub plugin. (You can search using the Filter box in the top right corner.) Select it and click `Install without restart`.



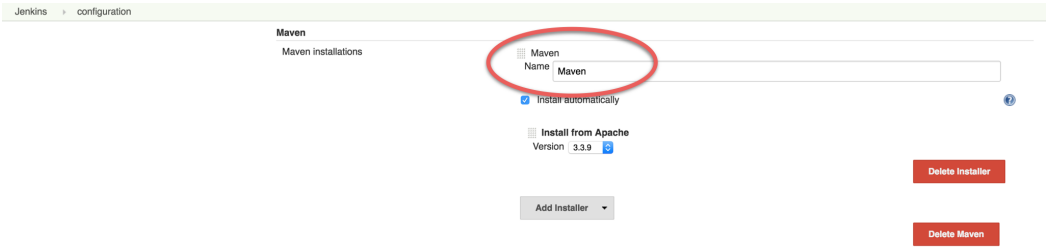
3. Also install the Cloud Foundry Plugin. Select it and click `Install without restart`.



4. From the `Manage Jenkins` page, click on `Configure System`. Scroll down to the `Maven` section, and click the `Add Maven` button.



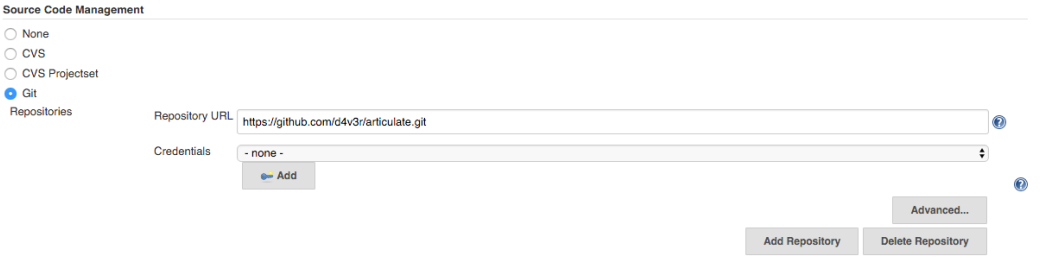
5. Enter a name into the name field then, at the bottom of the page, click `Save`.



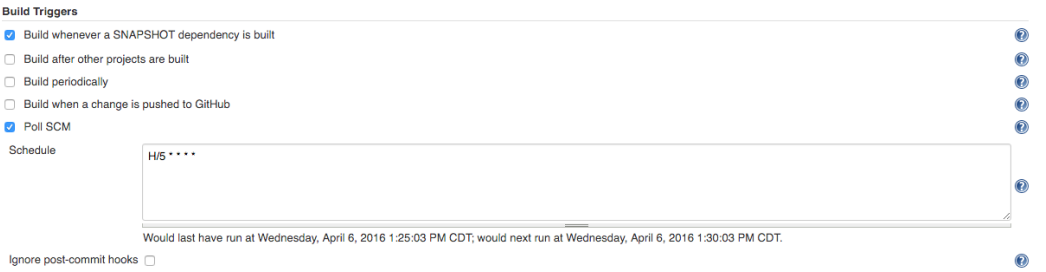
1.2.5. Create Build Job

1. From the Jenkins dashboard, click `New Item` (in top left corner), name it `articulate-{{initials}}` and select `Maven project` Then click `OK`.

2. Under `Source Code Management`, select `Git`, and supply your forked repository URL.



3. Under `Build Triggers`, select `Poll SCM`. In the `Schedule`, enter the CRON formatted string such as `* * * * *`. This will poll your Github repository every minute for changes, and if any are detected, will execute the build.



4. Under `Build`, add the project path to the `Root POM`, so it becomes `pom.xml`.

5. Under `Post-build Actions`, click the `Add post-build action`, and select `Push to Cloud Foundry`.

6. Fill in the parameters to target and log into the Cloud Foundry instance you'll be using. You will have to add your credentials. Test the connection to make sure you can connect. Also check the `Reset app if already exists` checkbox. This allows for app bits and configuration to be updated/reset with each deployment; creating a more dependable way to redeploy the application (see the context sensitive help in Jenkins for more details).

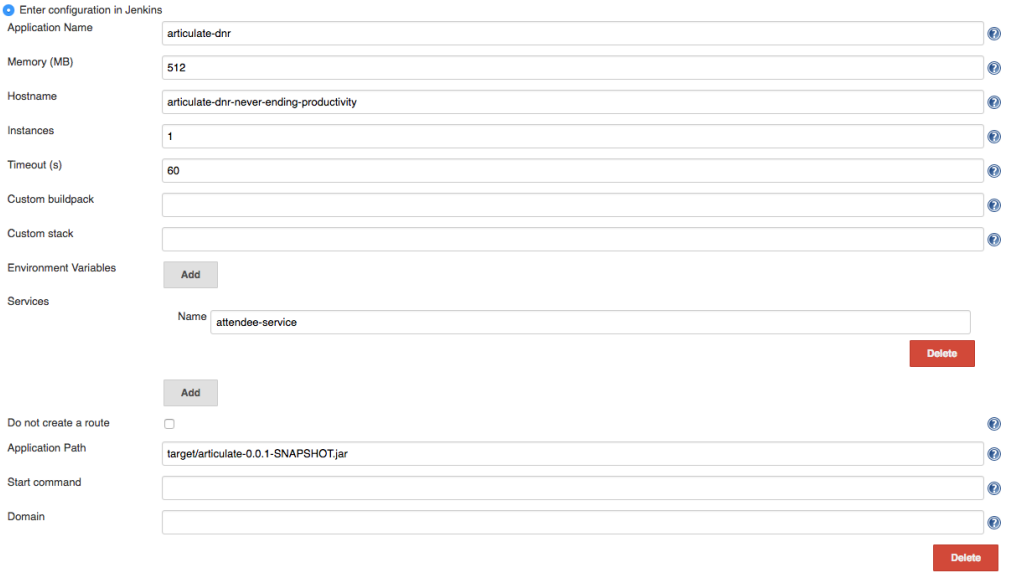
7. Make sure that `Enter configuration in Jenkins` is selected.

   Fill in the following fields:

   - `Application Name` = `articulate-{{initials}}`
   - `Memory` = `768M`
   - `Hostname` = come up with something original and unique
   - `Instance` = `1`
   - `Timeout` = `60`
   - `Services` = `attendee-service`

Advanced Settings:

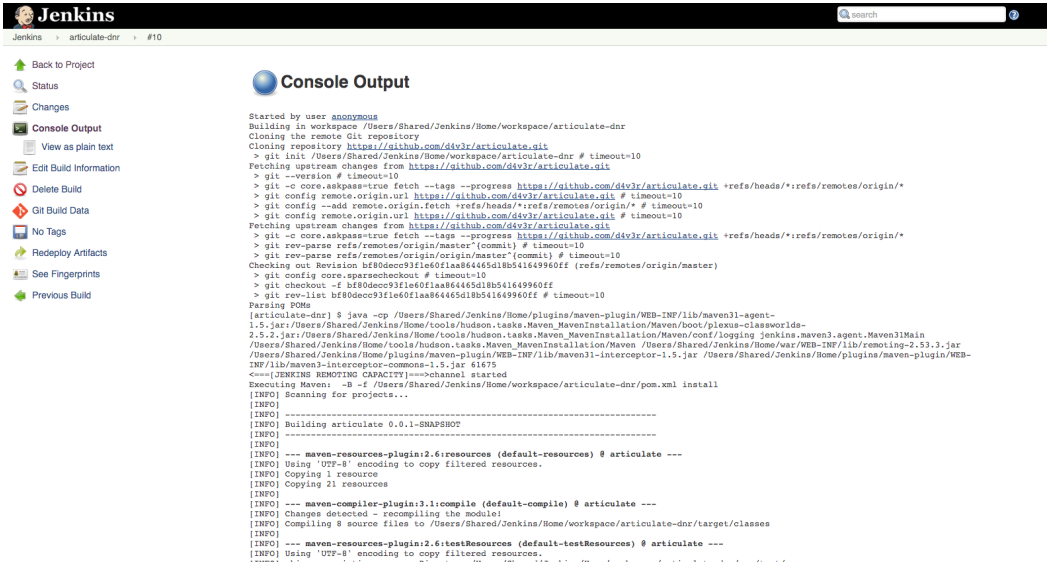
- `Application Path` = `target/articulate-0.0.1-SNAPSHOT.jar`



8. Save the config and try running the build by clicking `Build Now`. Do not proceed past this step until you have a successful build and deployment to Pivotal Cloud Foundry. Confirm the application is deployed by viewing it in your browser.

   Make sure to view the Build details (Left side of screen -> `Build History` -> `Build #`).

   `Console Output` can be viewed there (for active or completed jobs). This is very useful for debugging failing builds.



9. In your forked repo, edit the Welcome message for Articulate.

   a. Edit the following file (can be done with a browser): https://github.com/{{github_username}}/pcf-articulate-code/blob/master/src/main/resources/templates/index.html

   b. Change the welcome message from `Welcome to Articulate!` to `Welcome to My Articulate Application!` Commit and push the change to GitHub, wait until the polling detects it, and watch the magic. Verify the build in Jenkins now succeeds. Also verify your change in the deployed application with a browser.

Congratulations, you have finished this exercise!

1.2.6. Questions

- What are some of the benefits of continuous delivery?
- Does continuous delivery mean continuous deployment?

1.2.7. Cleanup

1. Delete the application that the pipeline deployed. For example:

```
cf delete articulate-{{initials}}
```

2. If provisioned, terminate your AWS Jenkins instance by going to your AWS EC2 dashboard, selecting the Jenkins instance, and clicking Actions -> Instance State -> Terminate.

1.3. Beyond the Class

The CD exercise above is very simplistic and should be expanded for real projects.

1.3.1. Artifact Repository

- Ideally, you want to build your artifacts (jars/wars) and publish them to a repository like Artifactory.
- Artifacts should be versioned to match the app deployments on PCF.
- All pushes to PCF should be using the same artifacts. Artifacts should be built once and used throughout the lifecycle.

1.3.2. Code promotion

- Jobs should be established in Jenkins to deploy/promote code to different phases like dev to test to prod.
- Jobs should use the same artifact published to Artifactory.
- Jobs can be triggered automatically or manually but should be fully automated. There should be no manual steps beyond clicking `build now`.

1.3.3. Notification

- Jenkins supports many notification plugins. It is important for code owners to be aware of build status.

1.3.4. Pivotal Cloud Foundry

Try out the CloudBees Jenkins Operations Center tile.

