

Contents

| | |
|-----------------------------------------------------|----|
| Software..... | 4 |
| eclipse-SDK-4.7-win32-x86_64 | 4 |
| First OSGi bundle..... | 5 |
| Create a new Bundle | 6 |
| Coding..... | 8 |
| Run..... | 11 |
| Karaf Installation | 14 |
| Installed Bundle in equinox/Felix/Karaf..... | 24 |
| Karaf – Server Mode | 31 |
| Using bnd and Bndtools..... | 35 |
| OSGI Bundle Using Maven - Felix | 37 |
| OSGI Bundle Using Maven - Karaf..... | 49 |
| Bundle Lifecycle - felix or equinox..... | 63 |
| Bundle Lifecycle - Karaf..... | 64 |
| Services Using Equinox..... | 66 |
| Define the service interface..... | 66 |
| Create service | 68 |
| Install service bundles | 74 |
| Use your service..... | 75 |
| Services Using Karaf..... | 82 |

| | |
|--------------------------------------------------------------|------------|
| Import / Required / Version Bundle - Virgo | 107 |
| Import / Required / Version Bundle - Karaf | 111 |
| Declarative OSGi Service - Eclipse..... | 122 |
| Define a declarative OSGi Service | 122 |
| Using services via declarative services | 130 |
| Declarative OSGi Service – Karaf Archetype..... | 136 |
| Service Tracker Using Eclipse PDE..... | 153 |
| Service Tracker Using Maven Plugin Karaf | 161 |
| OSGI HTTP Service with Equinox | 174 |
| OSGI HTTP Service with Karaf | 183 |
| Embedding Karaf in a WebContainer - Tomcat | 194 |
| REST Web Services – Producer - Karaf | 204 |
| REST Client – Jersey..... | 226 |
| Enabling REST API Security with Shiro | 248 |
| Testing – Mock & Pax Exam | 274 |
| Enabling Proxy in karaf..... | 295 |
| Configuration | 297 |
| Security..... | 298 |
| Troubleshooting Issues | 301 |
| Performance Tuning..... | 304 |
| Monitoring..... | 305 |
| Creating bundles for non OSGi third party dependencies | 311 |

| | |
|---------------|-----|
| Command | 312 |
|---------------|-----|

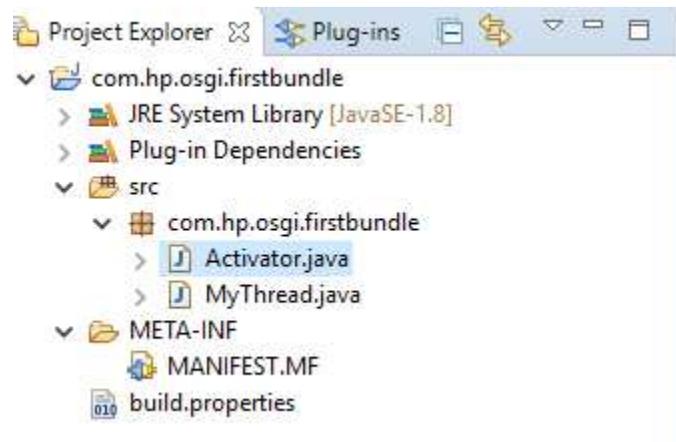
Software:

eclipse-SDK-4.7-win32-x86_64

First OSGi bundle

We will create a simple OSGi bundle and deploy it within Eclipse runtime. At the end of this tutorial you will also learn how to export a bundle and use it later to deploy in a standalone OSGi server – Karaf.

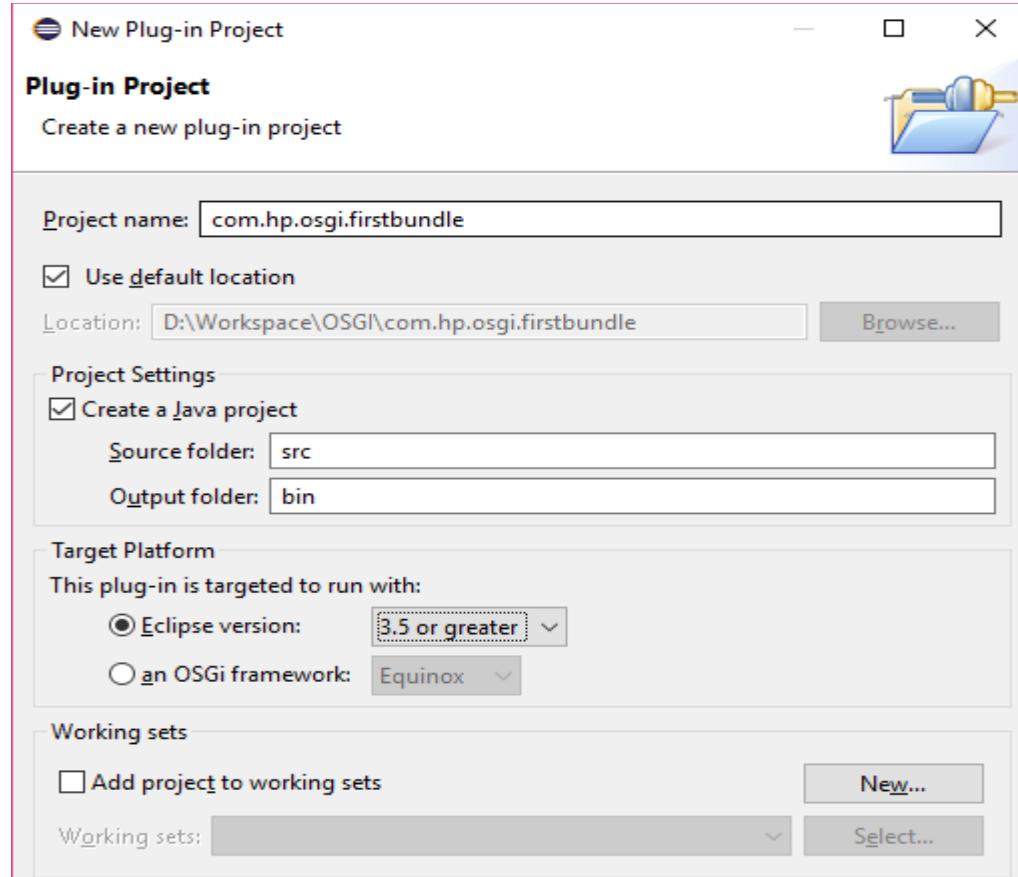
At the end of the lab, you will have the following project structure.



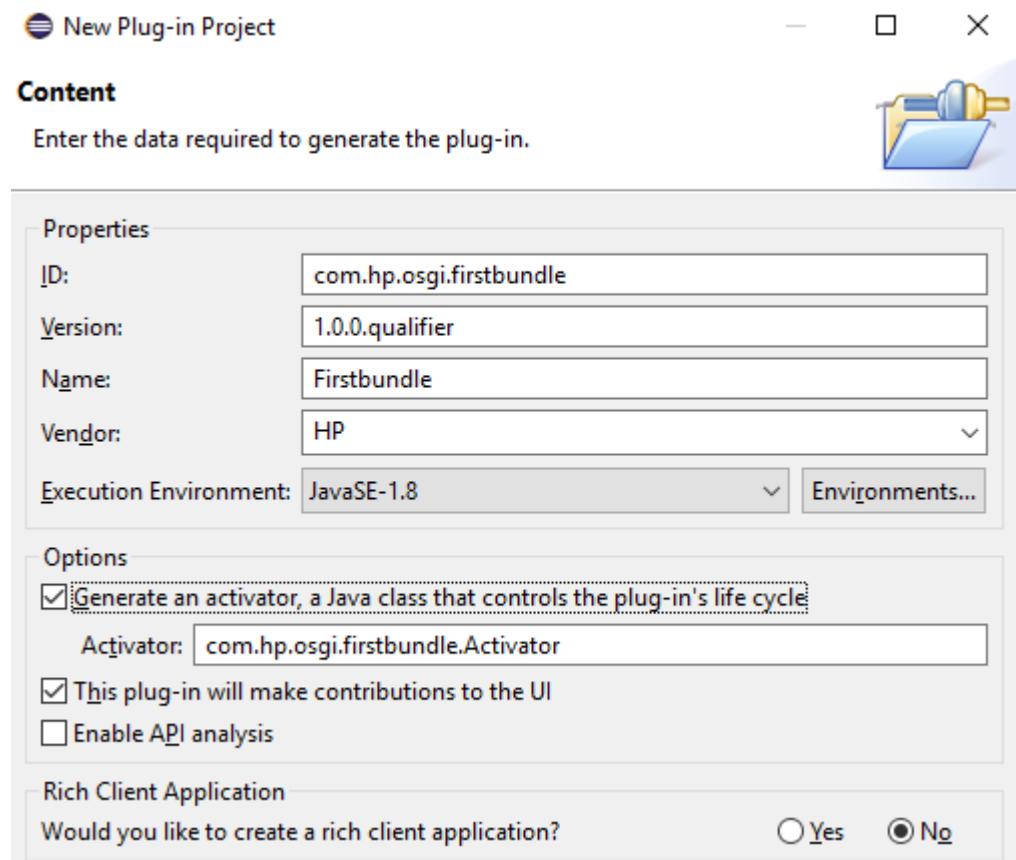
Start the eclipse IDE.

Create a new Bundle

Create a new plugin project "com.hp.osgi.firstbundle" via File → New → Other → Plug-in Development → Plug-in Project



Next



Next → Unselect the template and Finish. Accept all defaults setting.

Coding

Create the following thread class and update with the following code.

```
package com.hp.osgi.firstbundle;

public class MyThread extends Thread {
    private volatile boolean active = true;

    public void run() {
        while (active) {
            System.out.println("Hello OSGI console");
            try {
                Thread.sleep(5000);
            } catch (Exception e) {
                System.out.println("Thread interrupted " + e.getMessage());
            }
        }
    }

    public void stopThread() {
        active = false;
    }
}
```

Change the class Activator.java with the following code.

```
package com.hp.osgi.firstbundle;

import org.osgi.framework.BundleActivator;
import org.osgi.framework.BundleContext;

/**
 * The activator class controls the plug-in life cycle
 */
public class Activator implements BundleActivator {

    // The plug-in ID
    public static final String PLUGIN_ID = "com.hp.osgi.firstbundle"; //NON-NLS-1$
    private MyThread myThread;
    // The shared instance
    private static Activator plugin;

    /**
     * The constructor
     */
    public Activator() {
    }

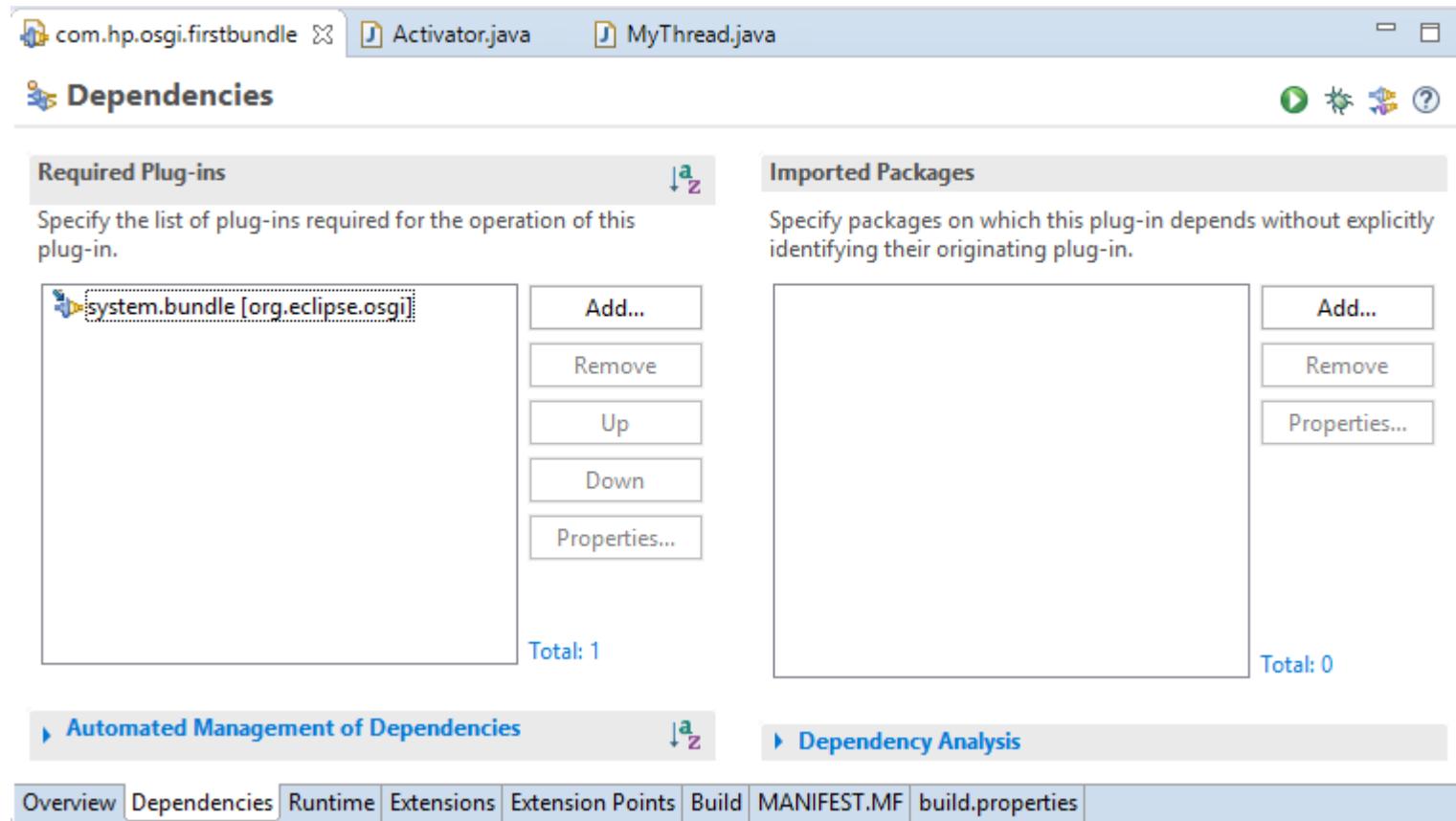
    /*
     * (non-Javadoc)
     * @see org.eclipse.ui.plugin.AbstractUIPlugin#start(org.osgi.framework.BundleContext)
     */
    public void start(BundleContext context) throws Exception {
        //super.start(context);
        plugin = this;
        System.out.println("Starting osgi.firstbundle");
        myThread = new MyThread();
        myThread.start();
    }
}
```

```
/*
 * (non-Javadoc)
 * @see org.eclipse.ui.plugin.AbstractUIPlugin#stop(org.osgi.framework.BundleContext)
 */
public void stop(BundleContext context) throws Exception {
    System.out.println("Stopping osgi.firstbundle");
    myThread.stopThread();
    myThread.join();

    plugin = null;
    super.stop(context);
}

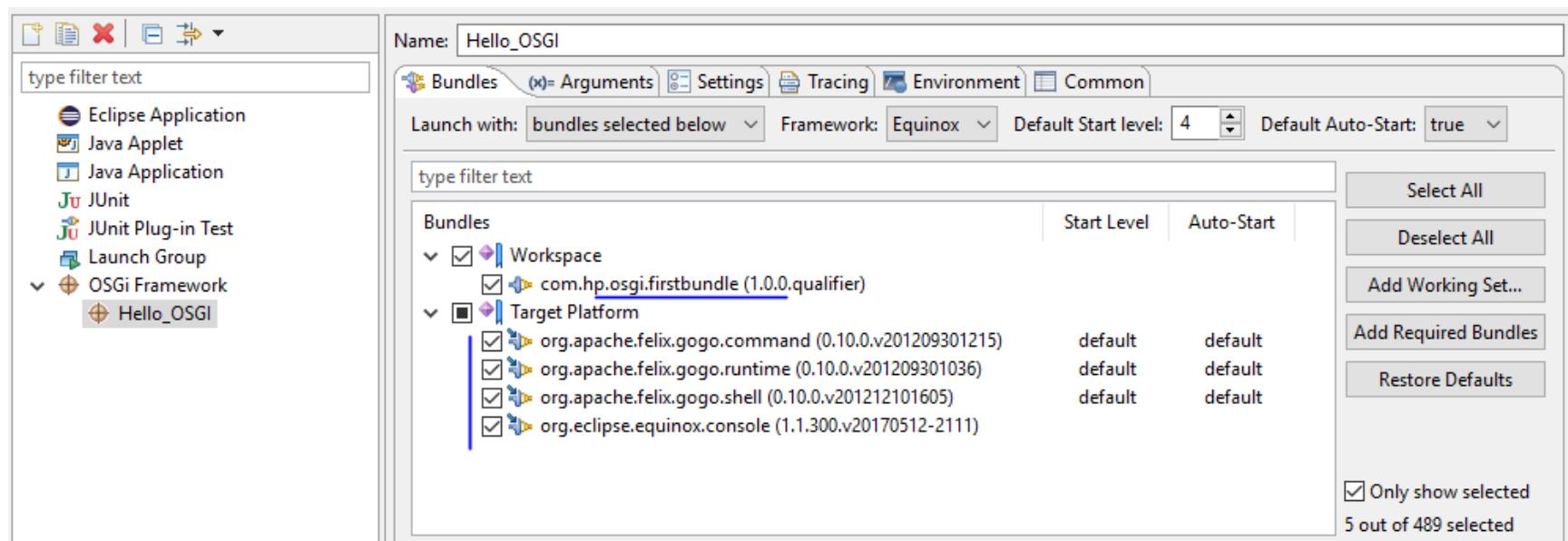
/**
 * Returns the shared instance
 *
 * @return the shared instance
 */
public static Activator getDefault() {
    return plugin;
}

}
```

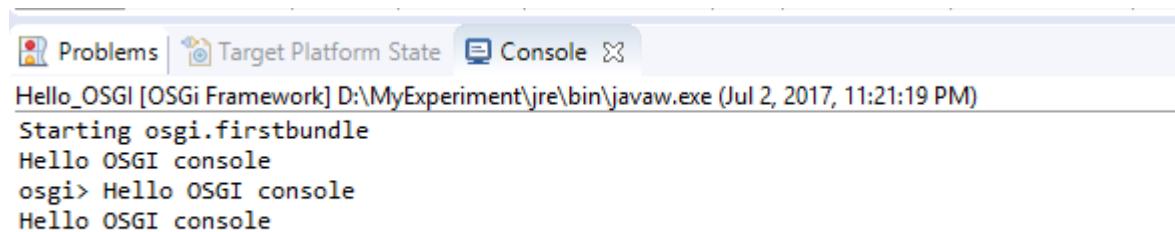


Run

Select your manifest.mf, right-click, select Run As → Run Configuration . Create a OSGi Framework launch configuration. Deselect all bundles except your com.hp.osgi.firstbundle. Press then "Add Required bundles". This should select the org.eclipse.osgi bundle.



Run this configuration. This should give you the following output. Every 5 second a new message should be written to the console.



The screenshot shows the Eclipse IDE interface with the 'Console' tab selected. The output window displays the following text:

```
Hello_OSGI [OSGi Framework] D:\MyExperiment\jre\bin\javaw.exe (Jul 2, 2017, 11:21:19 PM)
Starting osgi.firstbundle
Hello OSGI console
osgi> Hello OSGI console
Hello OSGI console
```

You have successfully created a bundle and executed it in the equinox.

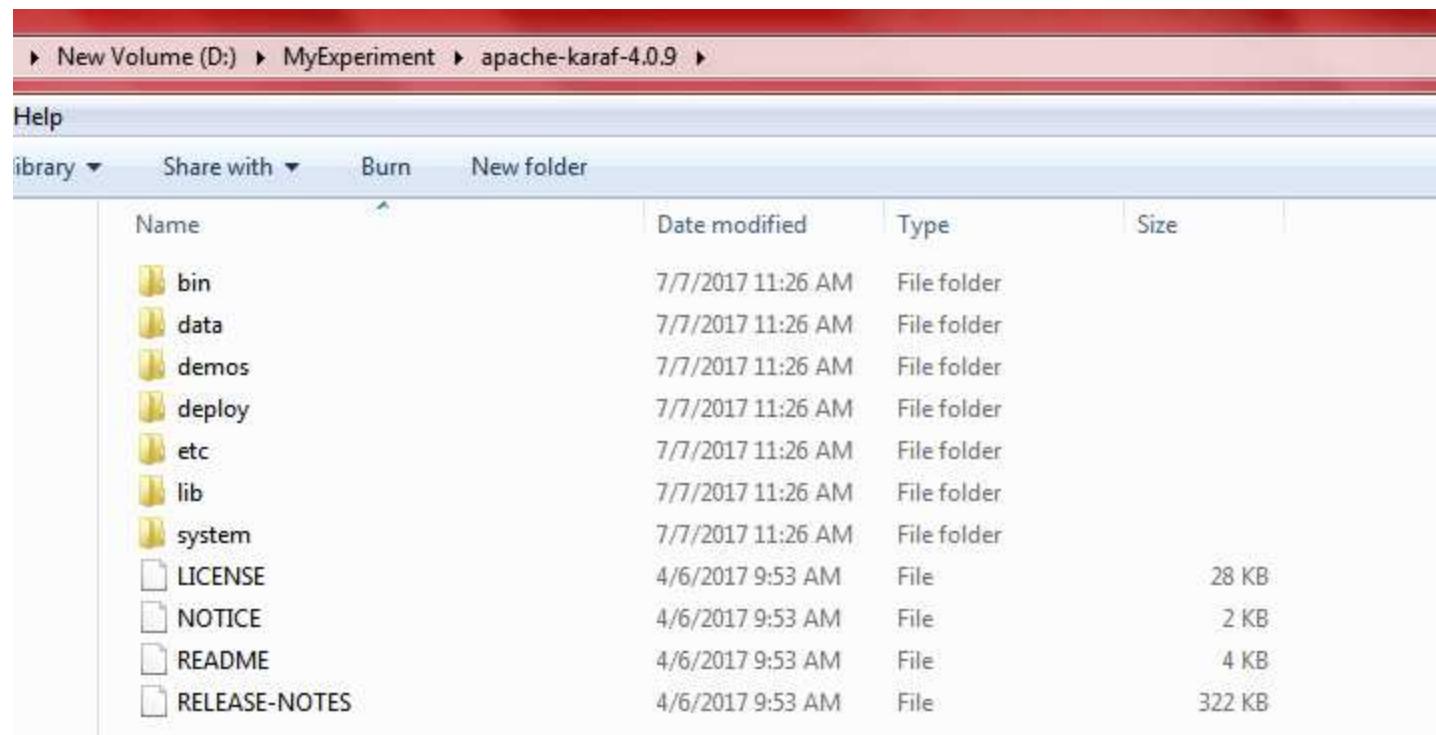
Karaf Installation

Karaf requires Java SE 7 or Java SE 8 environment to run it.

Download the binary distribution that matches your system (zip for windows,tar.gz for unixes)

Extract the archive to a new folder on your hard drive; for example in

D:\MyExperiment\karaf - from now on this directory will be referenced as <KARAF_HOME>.



The screenshot shows a Windows File Explorer window with the following details:

Path: New Volume (D:) > MyExperiment > apache-karaf-4.0.9

File Explorer ribbon: Help, library ▾, Share with ▾, Burn, New folder

Table of contents:

| Name | Date modified | Type | Size |
|---------------|-------------------|-------------|--------|
| bin | 7/7/2017 11:26 AM | File folder | |
| data | 7/7/2017 11:26 AM | File folder | |
| demos | 7/7/2017 11:26 AM | File folder | |
| deploy | 7/7/2017 11:26 AM | File folder | |
| etc | 7/7/2017 11:26 AM | File folder | |
| lib | 7/7/2017 11:26 AM | File folder | |
| system | 7/7/2017 11:26 AM | File folder | |
| LICENSE | 4/6/2017 9:53 AM | File | 28 KB |
| NOTICE | 4/6/2017 9:53 AM | File | 2 KB |
| README | 4/6/2017 9:53 AM | File | 4 KB |
| RELEASE-NOTES | 4/6/2017 9:53 AM | File | 322 KB |

```
Directory of D:\MyExperiment\apache-karaf-4.0.9\bin

07/07/2017  11:26 AM    <DIR>      .
07/07/2017  11:26 AM    <DIR>      ..
04/06/2017  09:53 AM            9,548 client
04/06/2017  09:53 AM            4,240 client.bat
07/07/2017  11:26 AM    <DIR>      contrib
04/06/2017  09:53 AM            11,181 instance
04/06/2017  09:53 AM            5,089 instance.bat
04/06/2017  09:53 AM            17,609 karaf
04/06/2017  09:53 AM            14,163 karaf.bat
04/06/2017  09:53 AM            2,154 setenv
04/06/2017  09:53 AM            2,159 setenv.bat
04/06/2017  09:53 AM            10,610 shell
04/06/2017  09:53 AM            4,698 shell.bat
04/06/2017  09:53 AM            4,286 start
04/06/2017  09:53 AM            2,495 start.bat
04/06/2017  09:53 AM            4,016 status
04/06/2017  09:53 AM            2,448 status.bat
04/06/2017  09:53 AM            4,010 stop
04/06/2017  09:53 AM            2,444 stop.bat
               16 File(s)   101,150 bytes
               3 Dir(s)  91,440,201,728 bytes free

D:\MyExperiment\apache-karaf-4.0.9\bin>karaf
```

Starting Karaf:change to KARAF_HOME/bin and execute the following command:

```
#karaf
```

```
D:\MyExperiment\apache-karaf-4.0.9\bin>karaf
Apache Karaf (4.0.9)

Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'system:shutdown' or 'logout' to shutdown Karaf.

karaf@root()>
```

SHELL BASICS

You can now run your first command. Simply type the <tab> key in the console. Y

```
karaf@root>>
Display all 311 possibilities? (y or n)
addCommand                                alias
bundle                                    bundle:capabilities
bundle:classes                            bundle:diag
bundle:dynamic-import                      bundle:find-class
bundle:headers                             bundle:id
bundle:info                               bundle:install
bundle:list                               bundle:load-test
bundle:refresh                            bundle:requirements
bundle:resolve                            bundle:restart
bundle:services                           bundle:start
bundle:start-level                        bundle:status
bundle:stop                               bundle:tree-show
bundle:uninstall                           bundle:update
bundle:watch                             cancel
capabilities                            cat
cl                                      classes
clear                                   clone
completion                            config
config:cancel                           config:delete
config:edit                             config:install
config:list                            config:meta
config:property-append                  config:property-delete
config:property-get                     config:property-list
config:property-set                     config:update
connect                                 create
date                                    delete
destroy                                 dev
dev:dump-create                         diag
display                                 dump-create
dynamic-import                          each
echo                                    edit
env                                     eval
exception-display                      exec
exit                                    export-bundles
exports                                 feature
```

You can then grab more specific help for a given command using the --help option for this command:

```
# bundle:list --help
```

```
karaf@root<*> bundle:list --help
DESCRIPTION
    bundle:list
        Lists all installed bundles.

SYNTAX
    bundle:list [options] [ids]

ARGUMENTS
    ids
        The list of bundle (identified by IDs or name or name/version)
        separated by whitespaces

OPTIONS
    -t
        Specifies the bundle threshold; bundles with a start-level less
        than this value will not get printed out.
    -n, --name
        Show bundle name
    --no-format
        Disable table rendered output
    -s
        Shows the symbolic name
    -l
        Show the locations
    --no-ellipsis
    --help
        Display this help message
    -u
        Shows the update locations
    --context, -c
        Use the given bundle context
        (defaults to 0)
    -r
        Shows the bundle revisions

karaf@root<*>
```

DEPLOY A SAMPLE APPLICATION

Let's install a sample Apache Camel application for now:

In the console, run the following commands

feature:repo-add camel 2.15.2



feature:install camel-spring

```
karaf@root>> feature:repo-add camel 2.15.2
Adding feature url mvn:org.apache.camel/karaf/apache-camel/2.15.2/xml/features
karaf@root>> feature:install camel-spring
```

bundle:install -s mvn:org.apache.camel/camel-example-osgi/2.15.2

```
karaf@root>> bundle:install -s mvn:org.apache.camel/camel-example-osgi/2.15.2
Bundle ID: 77
karaf@root>>
```

The example installed is using Camel to start a timer every 2 seconds and output a message in the log.

The previous commands download the Camel features descriptor and install the example feature.

You can display the log in the shell:

#log:display

```
2017-07-07 12:02:17,473 | INFO  | timer://myTimer | ExampleRouter
| 54 - org.apache.camel.camel-core - 2.15.2 | Exchange[ExchangePattern: In
Only, BodyType: String, Body: SpringDSL set body: Fri Jul 07 12:02:17 IST 2017]
2017-07-07 12:02:19,472 | INFO  | timer://myTimer | MyTransform
| 77 - camel-example-osgi - 2.15.2 | >>> SpringDSL set body: Fri Jul 07
12:02:19 IST 2017
2017-07-07 12:02:19,473 | INFO  | timer://myTimer | ExampleRouter
| 54 - org.apache.camel.camel-core - 2.15.2 | Exchange[ExchangePattern: In
Only, BodyType: String, Body: SpringDSL set body: Fri Jul 07 12:02:19 IST 2017]
```

Stopping and uninstalling the sample application

To stop and uninstall the demo, run the following command: You just verify the status of the bundle after each command.

```
#bundle:list
```

```
karaf@root>> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State | Lvl | Version | Name
---+-----+----+-----+
52 | Active | 50 | 2.15.2 | camel-catalog
53 | Active | 50 | 2.15.2 | camel-commands-core
54 | Active | 50 | 2.15.2 | camel-core
55 | Active | 50 | 2.15.2 | camel-spring
56 | Active | 50 | 2.15.2 | camel-karaf-commands
57 | Active | 50 | 1.1.1 | geronimo-jta_1.1_spec
77 | Active | 80 | 2.15.2 | camel-example-osgi
karaf@root>>
```

```
# bundle:stop camel-example-osgi
```

```
#bundle:list
```

```
# bundle:uninstall camel-example-osgi
```

```
#bundle:list
```

```

karaf@root>> bundle:stop camel-example-osgi
karaf@root>> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State | Lvl | Version | Name
-----+
52 | Active | 50 | 2.15.2 | camel-catalog
53 | Active | 50 | 2.15.2 | camel-commands-core
54 | Active | 50 | 2.15.2 | camel-core
55 | Active | 50 | 2.15.2 | camel-spring
56 | Active | 50 | 2.15.2 | camel-karaf-commands
57 | Active | 50 | 1.1.1 | geronimo-jta_1.1_spec
77 | Resolved | 80 | 2.15.2 | camel-example-osgi
karaf@root>> bundle:uninstall camel-example-osgi
karaf@root>> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State | Lvl | Version | Name
-----+
52 | Active | 50 | 2.15.2 | camel-catalog
53 | Active | 50 | 2.15.2 | camel-commands-core
54 | Active | 50 | 2.15.2 | camel-core
55 | Active | 50 | 2.15.2 | camel-spring
56 | Active | 50 | 2.15.2 | camel-karaf-commands
57 | Active | 50 | 1.1.1 | geronimo-jta_1.1_spec
karaf@root>>

```

#feature:list | grep -i war

```

karaf@root>> feature:list | grep -i war
war | 4.0.9 | Uninstal
led | standard-4.0.9 | Turn Karaf as a full WebContainer
blueprint-web | 4.0.9 | Uninstal
led | standard-4.0.9 | Provides an OSGI-aware Servlet ContextListener
fo |
pax-war | 4.3.0 | Uninstal
led | org.ops4j.pax.web-4.3.0 | Provide support of a full WebContainer
pax-war-tomcat | 4.3.0 | Uninstal
led | org.ops4j.pax.web-4.3.0 |
karaf@root>>

```

To display the current OSGi framework in use

```
#system:framework
```

Enabling Webconsole:

```
#feature:install webconsole
```

By default, you can **use karaf/karaf.** To connect to the web console.

<http://localhost:8181/system/console/vmstat>

The screenshot shows the Apache Karaf Web Console System Information page. At the top, there's a navigation bar with links for Main, OSGi, and Web Console, and a Log out button. Below the navigation is a yellow banner stating "System is up and running!". The main content area is divided into sections: "Start Level Information" and "Server Information". Under "Start Level Information", there are two rows: "System Start Level" set to 100 with a "Change" button, and "Default Bundle Start Level" set to 80 with a "Change" button. Under "Server Information", there are three rows: "Last Started" at July 7, 2017 3:31:38 PM IST, "Uptime" at 0 days 00:01:49.909, and "Framework" with "Restart" and "Stop" buttons. The Apache Karaf logo is visible in the top right corner of the page.

STOPPING KARAF

To stop Karaf from the console, **enter ^D** in the console:

^D

Alternatively, you can also run the following command:

```
# system:shutdown
```

halt is also an alias for system:shutdown:

```
# halt
```

```
karaf@root>> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State | Lvl | Version | Name
---+-----+----+-----+
52 | Active | 50 | 2.15.2 | camel-catalog
53 | Active | 50 | 2.15.2 | camel-commands-core
54 | Active | 50 | 2.15.2 | camel-core
55 | Active | 50 | 2.15.2 | camel-spring
56 | Active | 50 | 2.15.2 | camel-karaf-commands
57 | Active | 50 | 1.1.1 | geronimo-jta_1.1_spec
karaf@root>> halt
karaf@root>>

D:\MyExperiment\apache-karaf-4.0.9\bin>
```

Cleaning the Karaf state

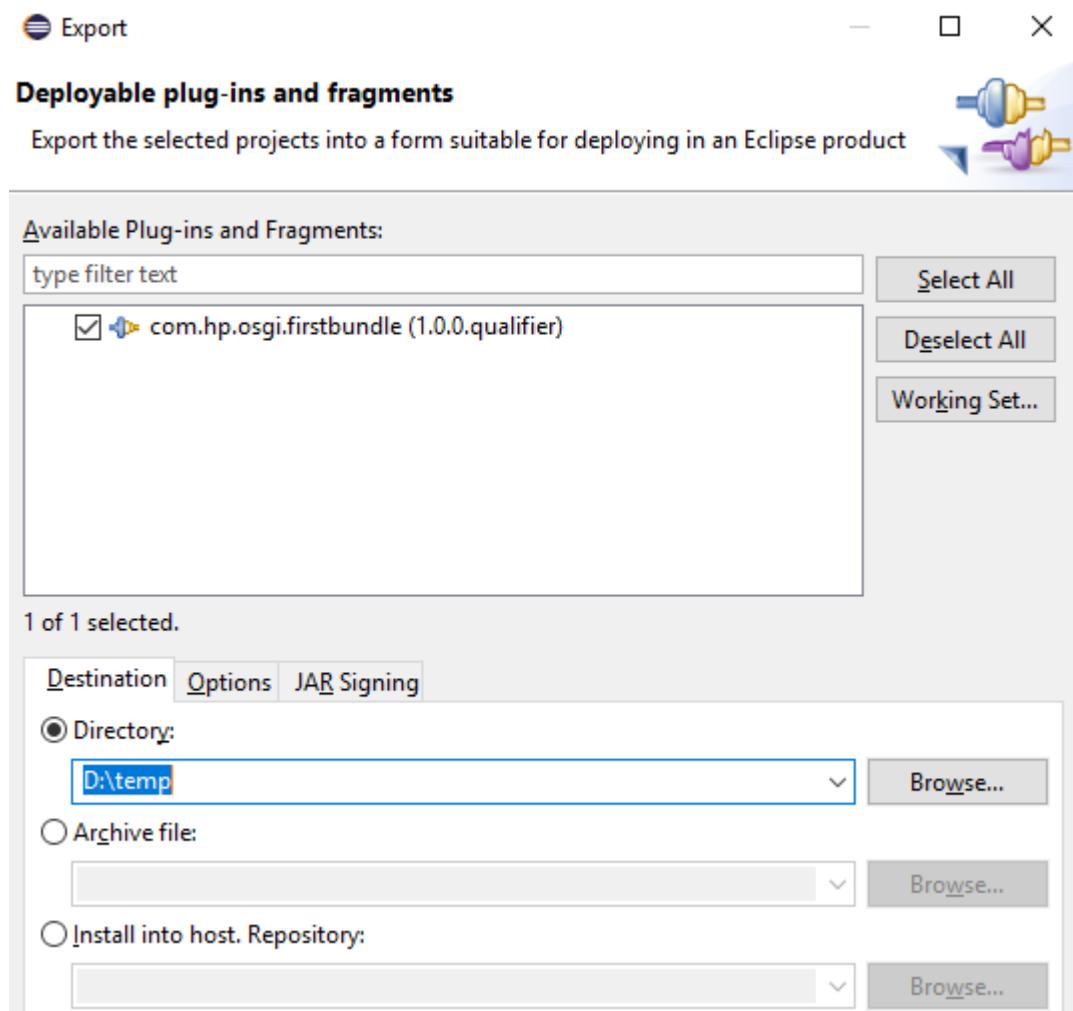
Normally Karaf remembers the features and bundles you installed and started. To reset Karaf into a clean state, just delete the data directory when Karaf is not running.

Installed Bundle in equinox/Felix/Karaf

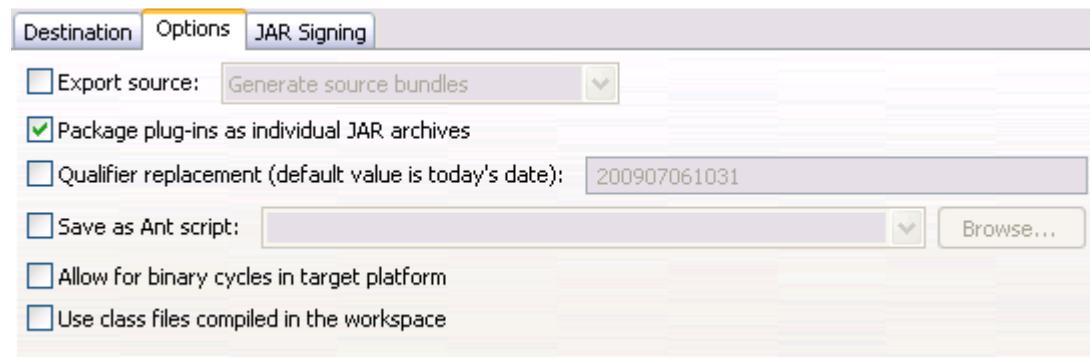
Export your bundle (The one you have created earlier)

Export your bundle. This will allow you to install it into a OSGi runtime. Select your bundle and choose File -> Export -> Plug-in Development -> "Deployable plug-ins and fragment".



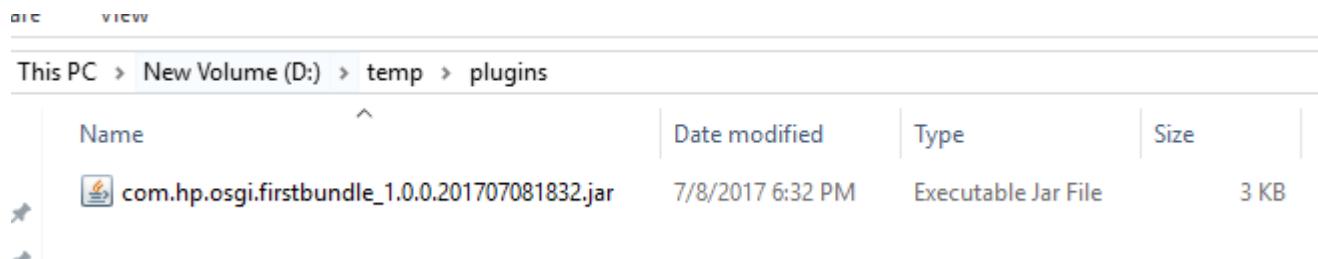


Uncheck the option to export the source.



Finish

You will find the bundle in the folder specified in the export configuration wizard.



Let us deploy this bundle in an OSGI server.

Running a standalone OSGi server

In your Eclipse installation directory identify the file org.eclipse.osgi*.jar. This file should be in the "plugin" folder. Copy this jar file to a new place, e.g. c:\temp\osgi-server. Rename the file to "org.eclipse.osgi.jar".

Start your OSGi server via the following command.

```
cd c:\temp\osgi-server
```

```
java -jar org.eclipse.osgi.jar -console
```

You can use "install URL" to install a bundle from a certain URL. For example to install your bundle from "c:\temp\bundles" use:

```
install file:c:\temp\bundles\plugins\de.vogella.osgi.firstbundle_1.0.0.jar
```

start:

```
osgi> ss
```

(it will list all the bundle installed in the equinox)

Verify the bundle id of the newly installed bundle and start using the following command.

```
start {id}
```

```
ss
```

```
stop {id}
```

Unzip: org.apache.felix.main.distribution-4.0.3.zip in D:\OSGI

Open a shell, and change directory to Felix's installation home. You can run Felix with the following command:

```
felix-framework-3.0.6$ java -jar bin/felix.jar
```

```
_____
```

Welcome to Apache Felix Gogo

g!

Create "dist" folder inside the installation folder and dump the earlier bundle in that folder

```
g! felix:install file:dist/helloworld.jar
```

Bundle ID: 5

```
g! felix:start 5
```

Installation in Karaf:

You can install the bundle with bundle:list command.

```
#bundle:install -s file:///d://temp/plugins/com.hp.osgi.firstbundle_1.0.0.201707081832.jar
```

Once installed, you can verify the bundle using list command

```
#bundle:list
```

```
karaf@root()> bundle:install -s file:///d://temp/plugins/com.hp.osgi.firstbundle_1.0.0.201707081832.jar
Starting osgi.firstbundle
Bundle ID: 66
Hello OSGI console
karaf@root()> Hello OSGI console
Hello OSGI console
Hello OSGI console
bundle:list
START LEVEL 100 , List Threshold: 50
ID | State | Lvl | Version          | Name
--+
66 | Active | 80 | 1.0.0.201707081832 | Firstbundle
karaf@root()> Hello OSGI console
```

The thread is printing the statement “Hello OSGI Console” repetitively creating clumsiness on the console let us stop the bundle .

```
#bundle:list
```

Let us determine the bundle ID using the above command.

```
#bundle:stop 66
```

```
karaf@root()> bundle:stop 60Hello OSGI console
6
Stopping osgi.firstbundle
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State    | Lvl | Version      | Name
-----
```

Karaf – Server Mode

Start the Apache Karaf in server mode

```
#karaf server
```



D:\MyExperiment\apache-karaf-4.0.9\bin>
D:\MyExperiment\apache-karaf-4.0.9\bin>
D:\MyExperiment\apache-karaf-4.0.9\bin>karaf server

Connect, Even if you start Apache Karaf without the console (using server or background modes),
you can connect to the console.

This connection can be local or remote. It means that you can access to Karaf console remotely.

To connect to the console, you can use the bin/client Unix script (bin\client.bat on Windows).

```
#bin/client
```

```
#bundle:list
```

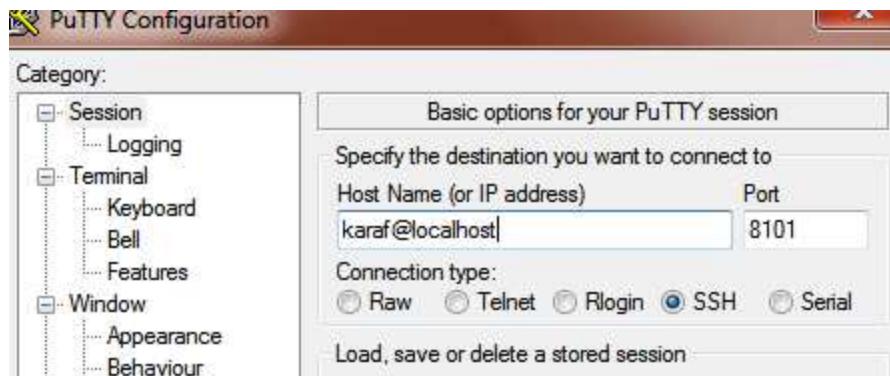
```
D:\MyExperiment>cd apache-karaf-4.0.9
D:\MyExperiment\apache-karaf-4.0.9>cd bin
D:\MyExperiment\apache-karaf-4.0.9\bin>client
Logging in as karaf

Apache Karaf <4.0.9>
Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit 'system:shutdown' to shutdown Karaf.
Hit '<ctrl-d>' or type 'logout' to disconnect shell from current session.

karaf@root<>> bundle:list
START LEVEL 100, List Threshold: 50
ID | State | Lvl | Version | Name
-----
52 | Active | 50 | 2.15.2 | camel-catalog
53 | Active | 50 | 2.15.2 | camel-commands-core
54 | Active | 50 | 2.15.2 | camel-core
55 | Active | 50 | 2.15.2 | camel-spring
56 | Active | 50 | 2.15.2 | camel-karaf-commands
57 | Active | 50 | 1.1.1 | geronimo-jta_1.1_spec
karaf@root<>>
```

By default, client tries to connect on localhost, on port 8101 (the default Apache Karaf SSH port). client accepts different options to let you connect on a remote Apache Karaf instance. You can use --help to get details about the options:

Another ways of connection: Use putty and specify the connection parameter:



Enter the password as karaf.

```
Using username "karaf".
SSH server: Password authentication
Using keyboard-interactive authentication.
Password:
Apache Karaf (4.0.9)

Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit 'system:shutdown' to shutdown Karaf.
Hit '<ctrl-d>' or type 'logout' to disconnect shell from current session.

karaf@root()> [REDACTED]
```

#shutdown -h

```
Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit 'system:shutdown' to shutdown
Hit '<ctrl-d>' or type 'logout' to exit

karaf@root()> bundle:list
START LEVEL 100 , List Threshold:
ID | State | Lvl | Version | Nam
-----
52 | Active | 50 | 2.15.2 | cam
53 | Active | 50 | 2.15.2 | cam
54 | Active | 50 | 2.15.2 | cam
55 | Active | 50 | 2.15.2 | camel-karaf-commands
56 | Active | 50 | 2.15.2 | camel-karaf-commands
57 | Active | 50 | 1.1.1 | geronimo-jta_1.1_spec
karaf@root()> shutdown -h
Confirm: halt instance root (yes/no): yes
karaf@root()>
```

A red error dialog box titled "PuTTY Fatal Error" is overlaid on the terminal window. It contains a red "X" icon and the message "Network error: Software caused connection abort". A blue "OK" button is at the bottom right of the dialog.

Using bnd and Bndtools.

Download [eclipse-java-neon-3-win32-x86_64.zip](#)

Install Bndtools

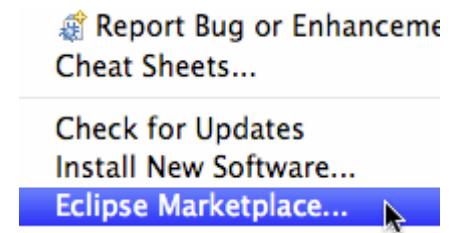
Via Marketplace

The recommended way to install Bndtools.

[View details »](#)

Eclipse Marketplace

1.



From the Help menu select Eclipse Marketplace.

If this is the first time you have used the marketplace, you *may** be asked to choose a solutions catalog. Select **Eclipse Marketplace** and click next.

2.



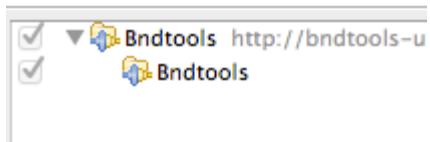
Type **bnd** into the search field and click Go.

Click "Install" next to the Bndtools entry.

3.

Confirm Selected Features

Confirm the features to include in thi



Drive the wizard to completion, following the on-screen instructions.

<http://bndtools.org/concepts.html>

OSGI Bundle Using Maven - Felix



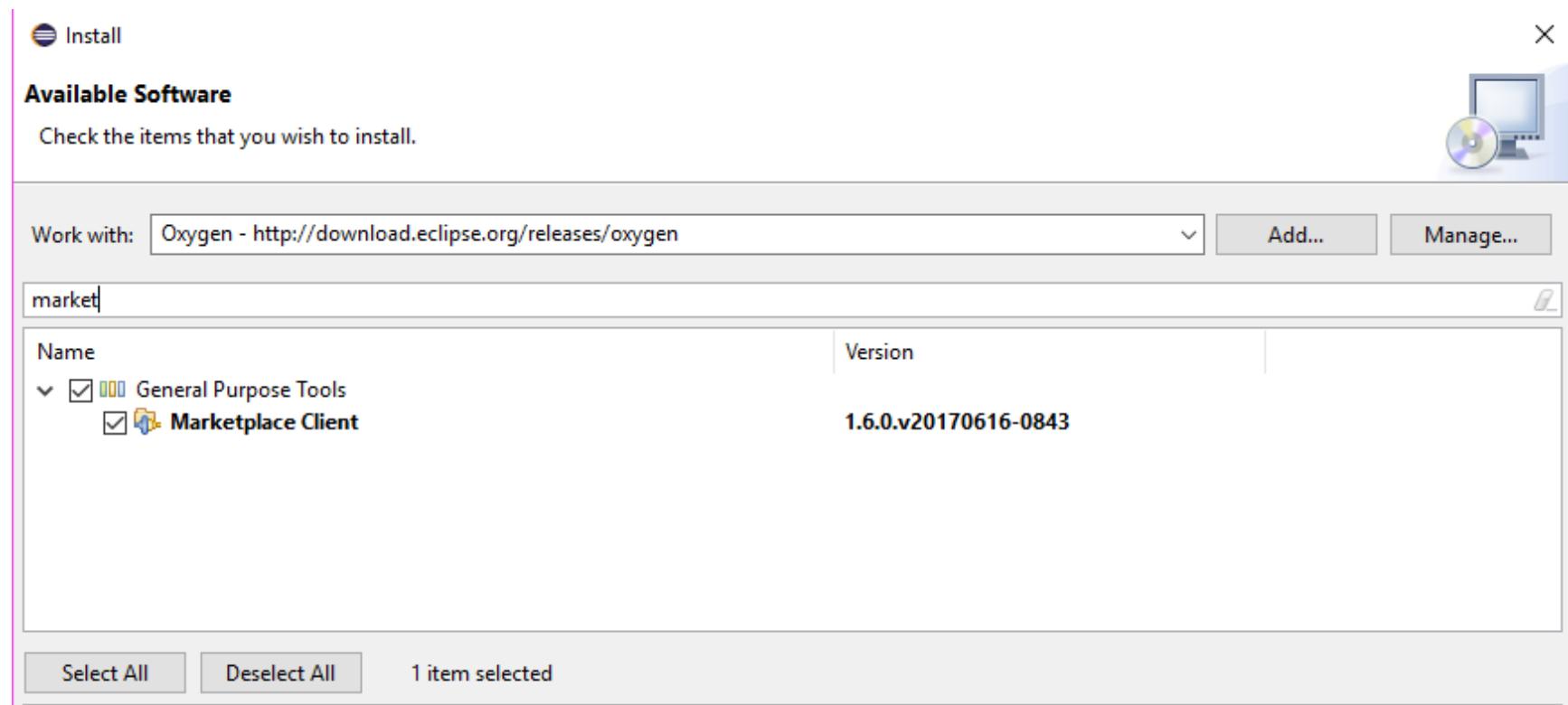
You need to install Maven Plugin for this project.

The screenshot shows the 'Available Software' window in the Eclipse Marketplace. The window title is 'OSGI Bundle Using Maven - Felix'. It includes an 'Install' button, a close button (X), and a feedback icon. A message at the top says 'Check the items that you wish to install.' On the right, there's an icon of a computer monitor with a CD. Below the message, there's a 'Work with:' dropdown set to 'Oxygen - http://download.eclipse.org/releases/oxygen' and buttons for 'Add...' and 'Manage...'. A search bar contains the text 'maven'. The main area displays a list of Maven-related plugins:

| Name | Version |
|----------------------------------------------------------------------|----------------------------------------|
| General Purpose Tools | |
| m2e - Maven Integration for Eclipse (includes Incubating components) | 1.8.0.20170516-2043 |
| m2e - slf4j over logback logging (Optional) | 1.8.0.20170516-2043 |
| Web, XML, Java EE and OSGi Enterprise Development | |
| m2e connector for mavenarchiver pom properties | 0.17.2.201606141937-signed-20160830... |
| m2e-wtp - Maven Integration for WTP | 1.3.2.20170517-2015 |

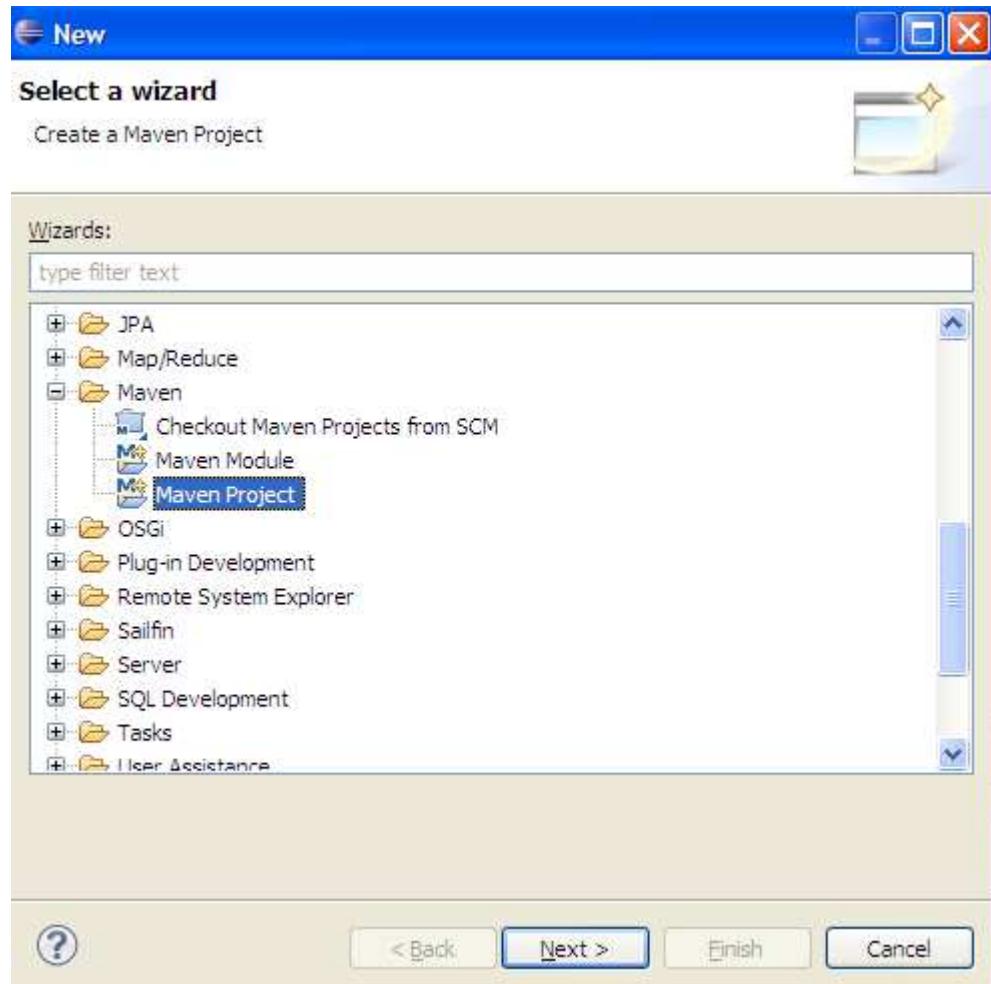
At the bottom, there are 'Select All' and 'Deselect All' buttons, and a status message '4 items selected'.

Install Market Place if not installed

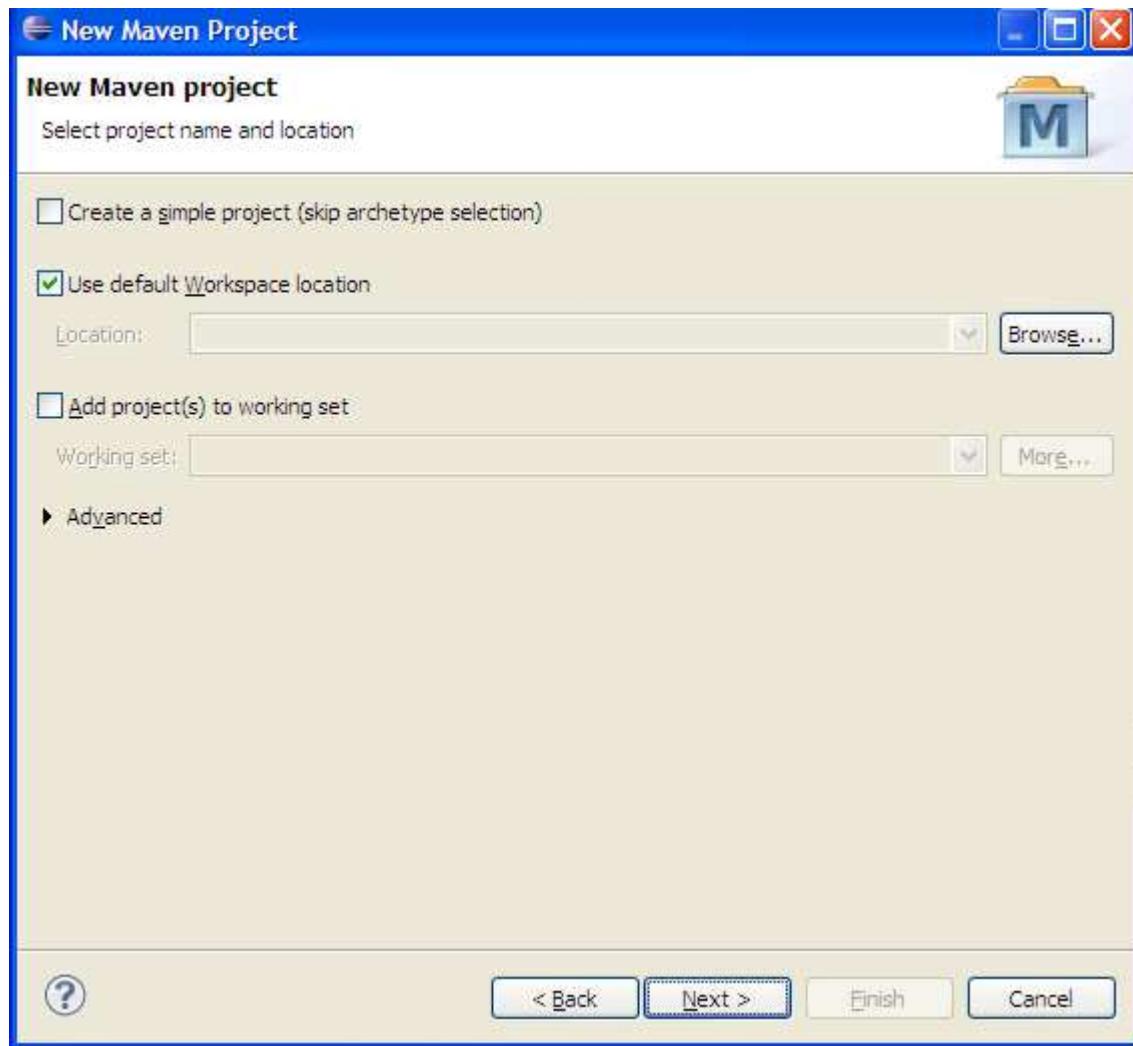


After the installation, you can create maven project as shown below:

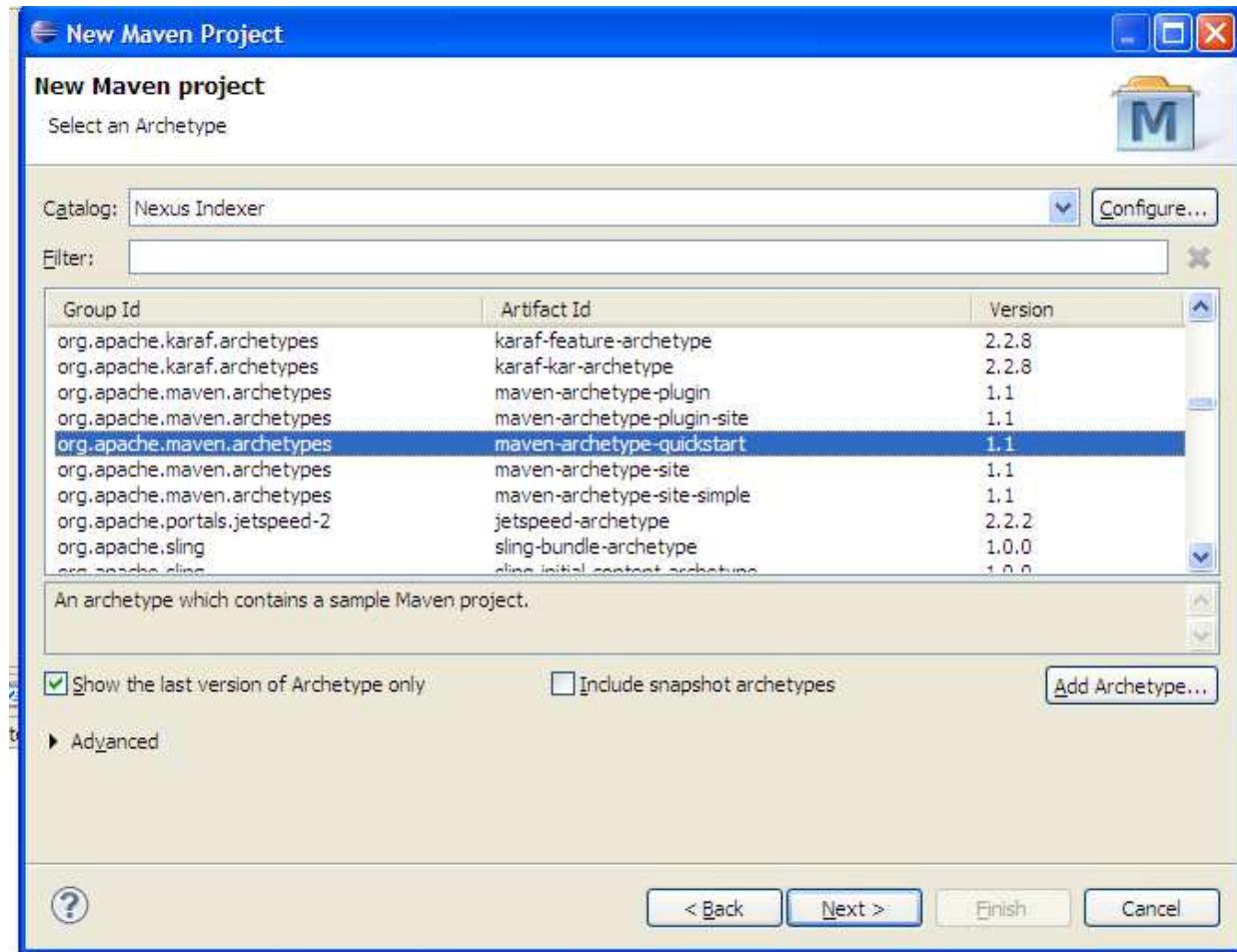
File -> New -> Others – Maven –Maven Project → Next → Select Archetype



Next



Next



Next

New Maven project

Specify Archetype parameters

Maven icon

| | |
|--------------|--------------------|
| Group Id: | com.hp.osgi |
| Artifact Id: | com.hp.osgi.bundle |
| Version: | 0.0.1-SNAPSHOT |
| Package: | com.hp.osgi.bundle |

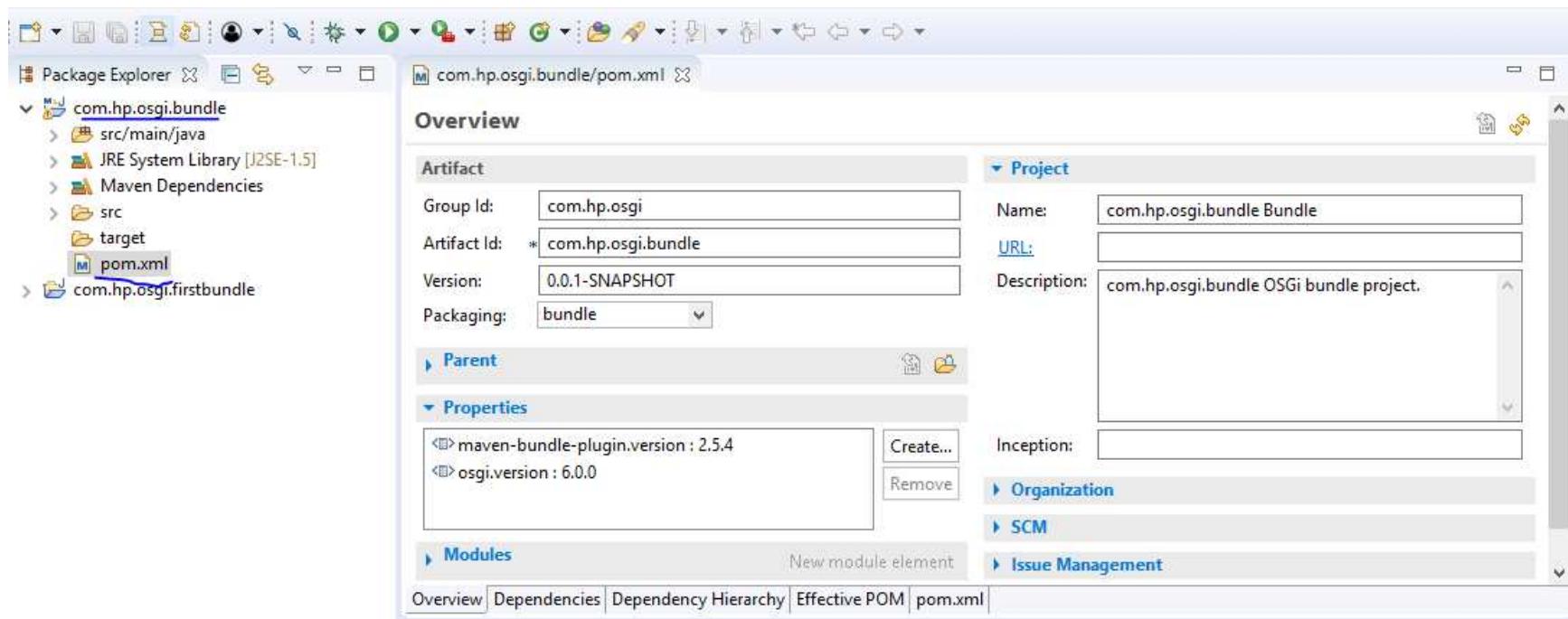
Properties available from archetype:

| Name | Value |
|---------|--------------------|
| package | com.hp.osgi.bundle |

Add... Remove

Finish

At the end, you will have the following project structure:



Replace the POM.xml, we have define all the dependencies library in this file.

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.hp.osgi</groupId>
  <artifactId>com.hp.osgi.bundle</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>bundle</packaging>

  <name>com.hp.osgi.bundle</name>
  <url>http://maven.apache.org</url>

  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <equinox.version>3.5.1.R35x_v20090827</equinox.version>
  </properties>

  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>3.8.1</version>
      <scope>test</scope>
    </dependency>

    <dependency>
      <groupId>org.eclipse.osgi</groupId>
      <artifactId>org.eclipse.osgi</artifactId>
      <version>${equinox.version}</version>
      <scope>test</scope>
    </dependency>
    <dependency>
      <groupId>org.apache.felix</groupId>
      <artifactId>org.osgi.core</artifactId>
      <version>1.0.0</version>
    </dependency>
  </dependencies>
  <build>
    <plugins>
```

```

<plugin>    <!-- (2) START It Help To generate the Manifest file and the Bundle Jar-->
  <groupId>org.apache.felix</groupId>
  <artifactId>maven-bundle-plugin</artifactId>
  <version>2.1.0</version>
  <extensions>true</extensions>
  <configuration>
    <instructions>
      <Export-Package>com.hp.services</Export-Package>
      <Bundle-Activator>com.hp.services.MyBundleActivator</Bundle-Activator>
      <Import-Package>
        org.osgi.framework,
        *;resolution:=optional
      </Import-Package>
    </instructions>
  </configuration>
</plugin>
</plugins>
</build>

<repositories>
  <repository>
    <id>com.springsource.repository.bundles.release</id>
    <name>SpringSource EBR - SpringSource Bundle Releases</name>
    <url>http://repository.springsource.com/maven/bundles/release</url>
  </repository>

  <repository>
    <id>com.springsource.repository.bundles.external</id>
    <name>SpringSource EBR - External Bundle Releases</name>
    <url>http://repository.springsource.com/maven/bundles/external</url>
  </repository>

  <repository>
    <id>spring-maven-milestone</id>
    <name>Springframework Maven Repository</name>
    <url>http://s3.amazonaws.com/maven.springframework.org/milestone</url>
  </repository>

  <repository>
    <id>spring-osgified-artifacts</id>
    <snapshots>

```

```
        <enabled>true</enabled>
    </snapshots>
    <name>Springframework Maven OSGified Artifacts Repository</name>
    <url>http://maven.springframework.org/osgi</url>
</repository>
</repositories>

</project>
```

Create an activator Class

```
com.hp.services.MyBundleActivator
```

Update the Activator class with the following:

```
package com.hp.osgi.bundle;

import org.osgi.framework.BundleActivator;
import org.osgi.framework.BundleContext;

public class Activator implements BundleActivator {

    public void start(BundleContext context) {
        System.out.println(" Your Maven Bundle Activated");

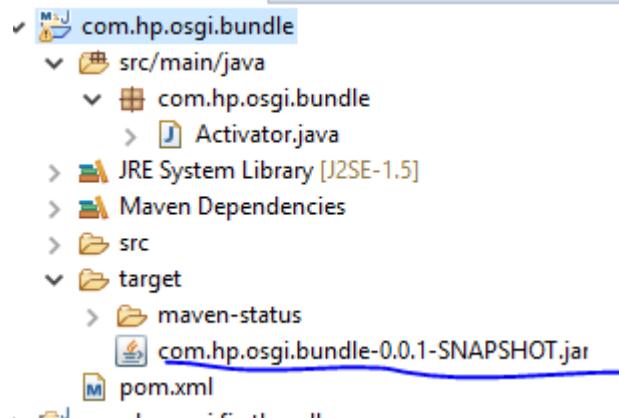
    }

    public void stop(BundleContext context) {
        System.out.println(" Your Maven Bundle DeActivated");
    }

}
```

Clean Maven

Install Maven



Deploy the bundle in the Felix and start

You should get the following result. (lb to list all the bundle installed in the felix)

```
C:\WINDOWS\system32\cmd.exe - java -jar bin/felix.jar
1!Active      1!Apache Felix Bundle Repository <1.6.6>
2!Active      1!Apache Felix Gogo Command <0.12.0>
3!Active      1!Apache Felix Gogo Runtime <0.10.0>
4!Active      1!Apache Felix Gogo Shell <0.10.0>
5!Resolved    1!Firstbundle <1.0.0.201208211326>
6!Resolved    1!henry <0.0.1.SNAPSHOT>
7!Resolved    1!com.hp.osgi.bundle <0.0.1.SNAPSHOT>
g! uninstall 7
gogo: CommandNotFoundException: Command not found: uninstall
g! uninstall 7
g! uninstall 6
g! lb
START LEVEL 1
ID!State      Level!Name
0!Active      0!System Bundle <4.0.3>
1!Active      1!Apache Felix Bundle Repository <1.6.6>
2!Active      1!Apache Felix Gogo Command <0.12.0>
3!Active      1!Apache Felix Gogo Runtime <0.10.0>
4!Active      1!Apache Felix Gogo Shell <0.10.0>
5!Resolved    1!Firstbundle <1.0.0.201208211326>
g! install file:dist/com.hp.osgi.bundle-0.0.1-SNAPSHOT.jar
Bundle ID: 8
g! start 8
Your Maven Bundle Activated
g!
```



OSGI Bundle Using Maven - Karaf

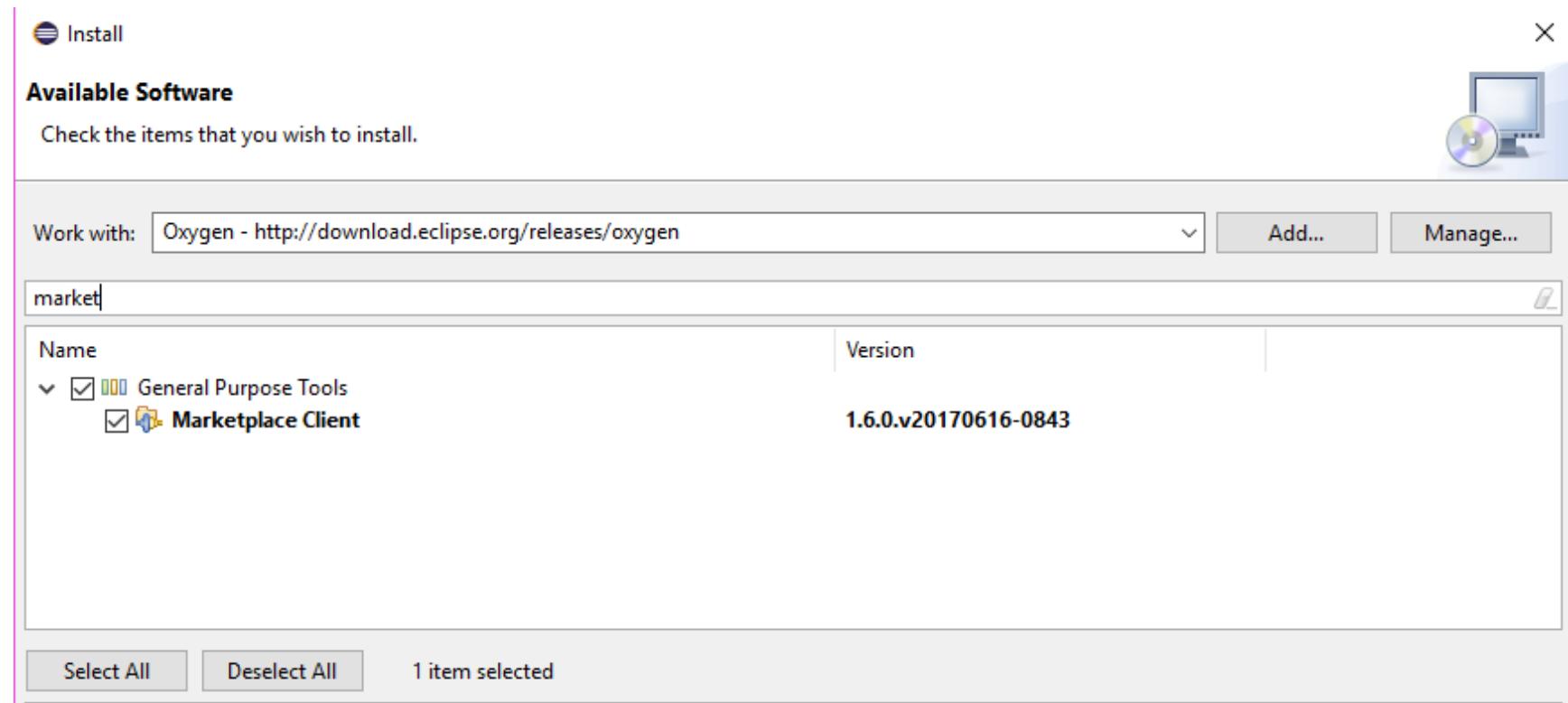
You need to install Maven Plugin for this project.

The screenshot shows the 'Available Software' window in the Eclipse Marketplace. The window title is 'Install'. It displays a list of Maven-related plugins under the search term 'maven'. The selected items are:

| Name | Version |
|----------------------------------------------------------------------|----------------------------------------|
| General Purpose Tools | |
| m2e - Maven Integration for Eclipse (includes Incubating components) | 1.8.0.20170516-2043 |
| m2e - slf4j over logback logging (Optional) | 1.8.0.20170516-2043 |
| Web, XML, Java EE and OSGi Enterprise Development | |
| m2e connector for mavenarchiver pom properties | 0.17.2.201606141937-signed-20160830... |
| m2e-wtp - Maven Integration for WTP | 1.3.2.20170517-2015 |

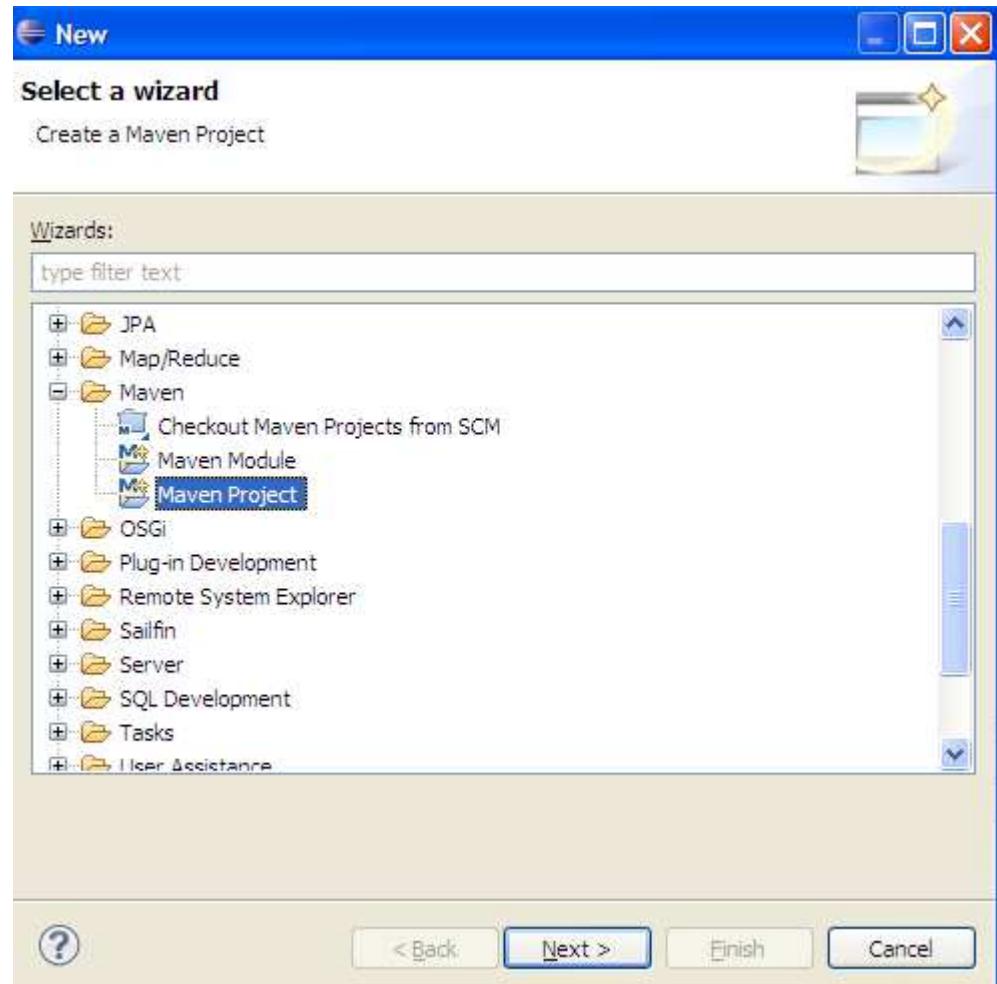
At the bottom, there are buttons for 'Select All', 'Deselect All', and '4 items selected'.

Install Market Place if not installed

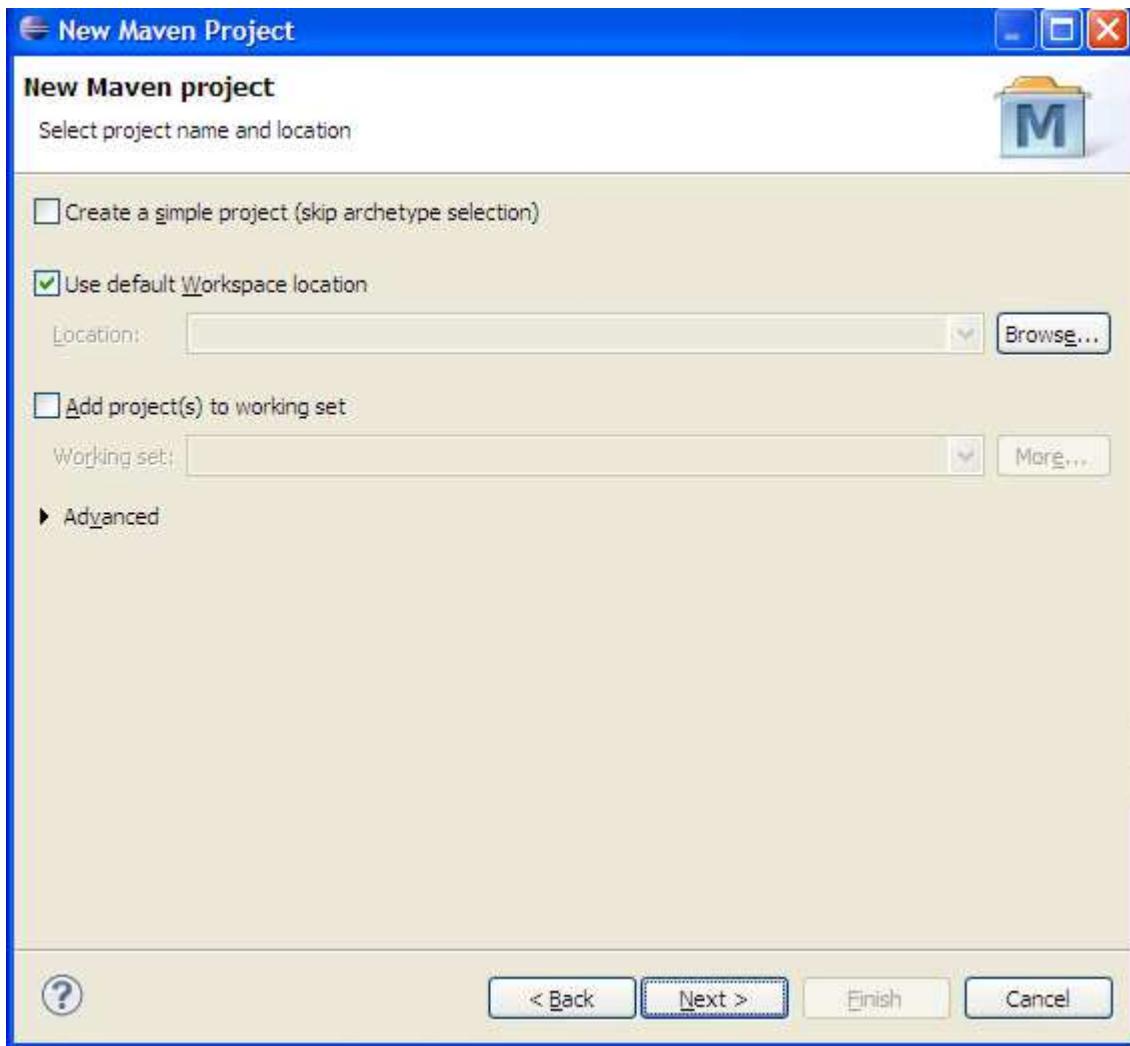


After the installation, you can create maven project as shown below:

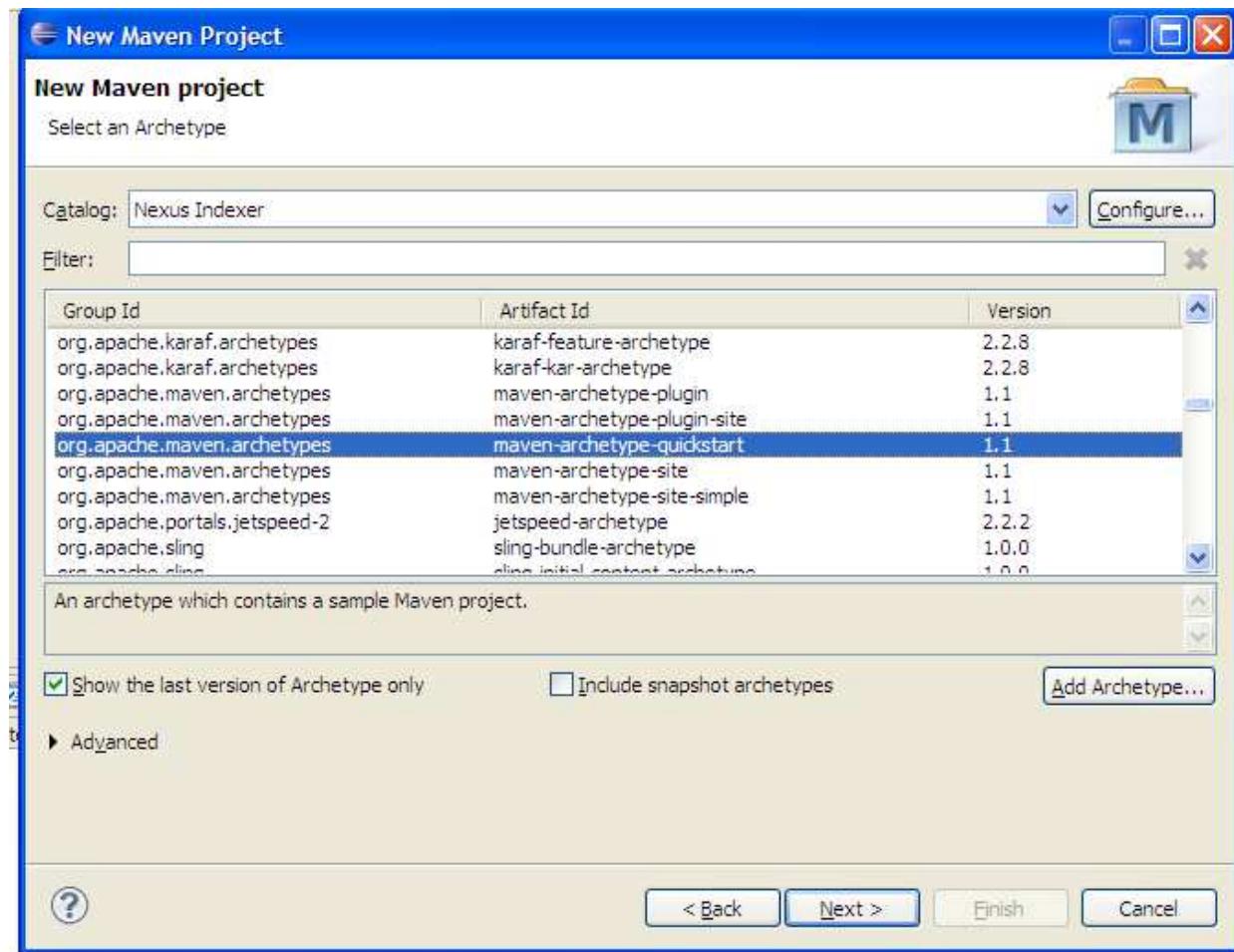
File -> New -> Others – Maven –Maven Project → Next → Select Archetype



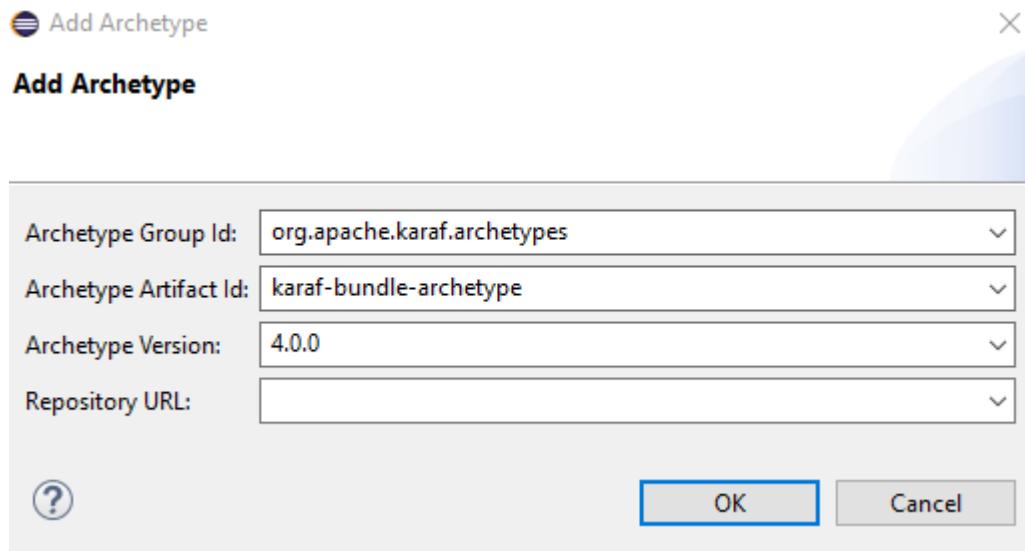
Next



Next

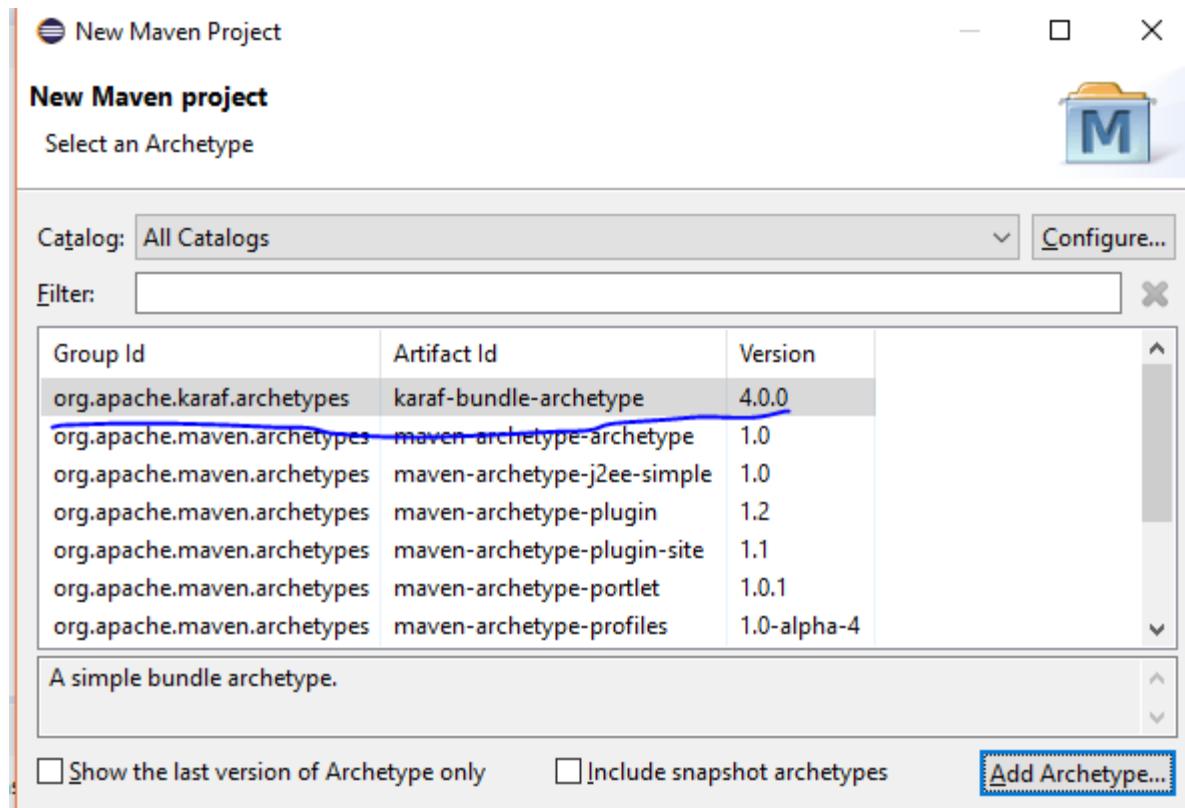


Click on Add Archetype and fill up as shown below



Ok

It will download the karaf Archetype and you can select as shown below



Next

New Maven project

Specify Archetype parameters

Group Id: com.hp.osgi

Artifact Id: com.hp.osgi.bundle

Version: 0.0.1-SNAPSHOT

Package: com.hp.osgi.bundle

Properties available from archetype:

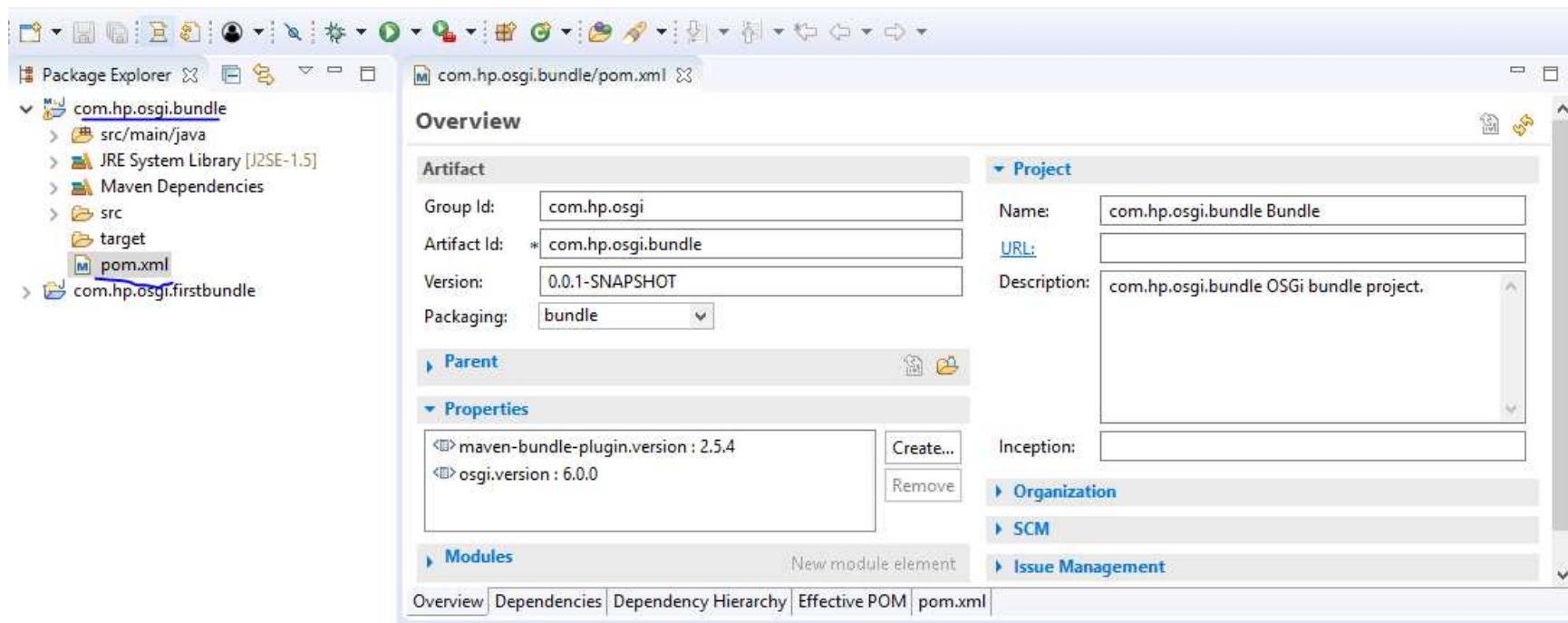
| Name | Value |
|---------|--------------------|
| package | com.hp.osgi.bundle |

Add... Remove



Finish

At the end, you will have the following project structure:



Replace the POM.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<!--
    Licensed to the Apache Software Foundation (ASF) under one
    or more contributor license agreements. See the NOTICE file
    distributed with this work for additional information
    regarding copyright ownership. The ASF licenses this file
    to you under the Apache License, Version 2.0 (the
    "License"); you may not use this file except in compliance
    with the License. You may obtain a copy of the License at

        http://www.apache.org/licenses/LICENSE-2.0

    Unless required by applicable law or agreed to in writing,
    software distributed under the License is distributed on an
    "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
    KIND, either express or implied. See the License for the
    specific language governing permissions and limitations
    under the License.
-->

<modelVersion>4.0.0</modelVersion>

<groupId>com.hp.osgi</groupId>
<artifactId>com.hp.osgi.bundle</artifactId>
<version>0.0.1-SNAPSHOT</version>
<packaging>bundle</packaging>

<name>com.hp.osgi.bundle Bundle</name>
<description>
    com.hp.osgi.bundle OSGi bundle project.
</description>

<properties>
    <maven-bundle-plugin.version>2.5.4</maven-bundle-plugin.version>
```

```
<osgi.version>6.0.0</osgi.version>
</properties>

<dependencies>
    <dependency>
        <groupId>org.osgi</groupId>
        <artifactId>org.osgi.core</artifactId>
        <version>${osgi.version}</version>
        <scope>provided</scope>
    </dependency>
</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.felix</groupId>
            <artifactId>maven-bundle-plugin</artifactId>
            <version>${maven-bundle-plugin.version}</version>
            <extensions>true</extensions>
            <configuration>
                <instructions>
                    <Bundle-SymbolicName>${project.artifactId}</Bundle-SymbolicName>
                    <Bundle-Version>${project.version}</Bundle-Version>
                    <Bundle-Activator>com.hp.osgi.bundle.Activator</Bundle-Activator>
                    <Export-Package>
                        com.hp.osgi.bundle*;version=${project.version}
                    </Export-Package>
                    <Import-Package>
                        *
                    </Import-Package>
                </instructions>
            </configuration>
        </plugin>
    </plugins>
</build>

</project>
```

Explore what is exported here?

Update the Activator class with the following:

```
package com.hp.osgi.bundle;

import org.osgi.framework.BundleActivator;
import org.osgi.framework.BundleContext;

public class Activator implements BundleActivator {

    public void start(BundleContext context) {
        System.out.println(" Your Maven Bundle Activated");

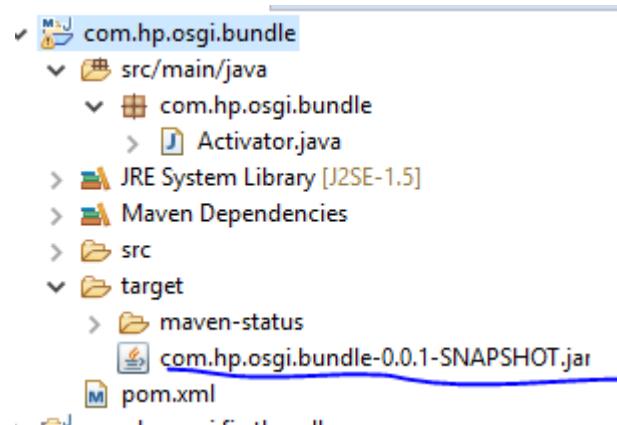
    }

    public void stop(BundleContext context) {
        System.out.println(" Your Maven Bundle DeActivated");
    }

}
```

Clean Maven

Install Maven



In Apache Karaf:

You can deploy the bundle using command: bundle:install or dumping in the deploy folder. Since we have already installed using the command, let us copy the bundle and dump in the deploy folder as shown below:

| Name | Date modified | Type | Size |
|---------------------------------------|------------------|---------------------|------|
| com.hp.osgi.bundle-0.0.1-SNAPSHOT.jar | 7/8/2017 7:53 PM | Executable Jar File | 4 KB |
| README | 4/6/2017 9:53 AM | File | 1 KB |

As soon as you dump the file, you will get the following display in the karaf console. Let us list the bundle and verify the installation of the bundle.

```
karaf@root()> Your Maven Bundle Activated  
  
karaf@root()>  
karaf@root()> bundle:list  
START LEVEL 100 , List Threshold: 50  
ID | State      | Lvl | Version          | Name  
---  
66 | Resolved   | 80  | 1.0.0.201707081832 | Firstbundle  
67 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle  
karaf@root()
```

Try stopping the bundle:

```
karaf@root()> bundle:list  
START LEVEL 100 , List Threshold: 50  
ID | State      | Lvl | Version          | Name  
---  
66 | Resolved   | 80  | 1.0.0.201707081832 | Firstbundle  
67 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle  
karaf@root()> bundle:stop 67  
Your Maven Bundle DeActivated  
karaf@root()
```

What does it display? Clarify with the trainer for the event?



Bundle Lifecycle - felix or equinox

Use the earlier created Bundle.

Install the Bundle :in felix or equinox

```
OSGI> Install file:dist/ com.hp.osgi.bundle-0.0.1-SNAPSHOT.jar
```

```
OSGI> ss / lb (Verify the bundle status – It should be resolved)
```

```
OSGI> start {bundle id}
```

```
SS (Verify that is should be in active mode)
```

```
Refresh {bundle id}
```

```
Ss/ls
```

```
Uninstall {bundle id}
```

```
Ss
```

Bundle Lifecycle - Karaf

You can use any of the earlier created Bundles. If it's not deploy, you need to install it before going ahead.

Install the Bundle in karaf using `bundle:install`

After installation, your bundle should be in installed state in my case i.e com.hp.osgi.bundle Bundle

`bundle:list`

you can choose any of the bundles that is installed in your container.

Let us stop this bundle. You can verify the status, it will remain same since the bundle is already in the Installed state only.

```
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
+-----+
| ID | State      | Lvl | Version          | Name           |
+-----+
66 | Installed   | 80  | 1.0.0.201707081832 | Firstbundle
67 | Installed   | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle
karaf@root()> bundle:stop 67
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
+-----+
| ID | State      | Lvl | Version          | Name           |
+-----+
66 | Installed   | 80  | 1.0.0.201707081832 | Firstbundle
67 | Installed   | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle
karaf@root()>
```

Now let us start the bundle:

`#bundle:start 67`

```
#bundle:list
```

```
karaf@root()> bundle:start 67
Your Maven Bundle Activated
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version          | Name
-----
66 | Installed  | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle
karaf@root()>
```

SO, what is the state of that bundle? It should be Active

Let us refresh the bundle, observe the statement printed in the console.

```
# bundle:refresh 67
```

```
karaf@root()> bundle:refresh 67
karaf@root()> Your Maven Bundle DeActivated
Your Maven Bundle Activated
```

What does it mean? The bundle was stopped and started again.

You can un install the bundle using the following command.

```
#bundle:uninstall {bundle id}
```

And you can verify the status using `bundle:list` command

Services Using Equinox

Define the service interface

Create a plugin project "de.vogella.osgi.quote" and the package "de.vogella.osgi.quote". Do not use a template. You do not need an activator. Afterwards select the MANIFEST.MF and the Runtime tab. Add "de.vogella.osgi.quote" to the exported packages.

The screenshot shows the Eclipse OSGI Runtime configuration interface. At the top, there are four icons: a green circle with a play button, a green starburst, a blue gear, and a question mark. Below these are two main sections: "Exported Packages" and "Package Visibility (Eclipse 3.1 or later)".

Exported Packages: This section lists packages that the plug-in exposes to clients. It currently contains one entry: "de.vogella.osgi.quote". To the right of this entry are four buttons: "Add...", "Remove", "Properties...", and "Calculate Uses".

Package Visibility (Eclipse 3.1 or later): This section describes package visibility rules. It states: "When the runtime is in strict mode, the selected package is:" followed by two radio button options: "visible to downstream plug-ins" and "hidden from all plug-ins except:". Below these options is a large empty rectangular box with "Add..." and "Remove" buttons on its right side.

Classpath: This section specifies the libraries and folders that constitute the plug-in classpath. It includes the following text: "Specify the libraries and folders that constitute the plug-in classpath. If unspecified, the classes and resources are assumed to be at the root of the plug-in." A small dropdown arrow is located to the right of this text.

At the bottom of the interface, there is a navigation bar with tabs: Overview, Dependencies, Runtime (which is selected), Build, MANIFEST.MF, and build.properties.

Create the following interface "IQuoteService".

```
package de.vogella.osgi.quote;

public interface IQuoteService {
    String getQuote();
}
```

Create service

We will now define a bundle which will provide the service.

Create a plugin project "de.vogella.osgi.quoteservice". Do not use a template.

Select the MANIFEST.MF and dependency tab. Add "de.vogella.osgi.quote" to the required plugins.

The screenshot shows the Eclipse IDE's Dependencies view for a plugin named "de.vogella.osgi.quo". The view is divided into two main sections: "Required Plug-ins" and "Imported Packages".

Required Plug-ins:

- de.vogella.osgi.quote (1.0.0)

Buttons for managing these dependencies include: Add..., Remove, Up, Down, and Properties... . A total count of 1 is displayed.

Imported Packages:

- org.osgi.framework (1.3.0)

Buttons for managing these packages include: Add..., Remove, and Properties... . A total count of 1 is displayed.

Below the main view, there are two tabs: "Automated Management of Dependencies" and "Dependency Analysis". The "Dependencies" tab is currently selected. At the bottom, there is a navigation bar with tabs: Overview, Dependencies, Runtime, Build, MANIFEST.MF, and build.properties.

Create the following class "QuoteService".

```
package de.vogella.osgi.quoteservice.internal;

import java.util.Random;

import de.vogella.osgi.quote.IQuoteService;

public class QuoteService implements IQuoteService {

    @Override
    public String getQuote() {
        Random random = new Random();
        // Create a number between 0 and 2
        int nextInt = random.nextInt(3);
        switch (nextInt) {
        case 0:
            return "Tell them I said something";
        case 1:
            return "I feel better already";
        }
    }
}
```

```
    default:  
        return "Hubba Bubba, Baby!";  
    }  
  
}  
}
```

Register the service in the class Activator.

```
package de.vogella.osgi.quoteservice;

import java.util.Hashtable;

import org.osgi.framework.BundleActivator;
import org.osgi.framework.BundleContext;

import de.vogella.osgi.quote.IQuoteService;
import
de.vogella.osgi.quoteservice.internal.QuoteService;

public class Activator implements BundleActivator {

    public void start(BundleContext context) throws
Exception {
        IQuoteService service = new QuoteService();
        // Third parameter is a hashmap which allows
        to configure the service
```

```
// Not required in this example

context.registerService(IQuoteService.class.getName(), service,
                      null);
System.out.println("IQuoteService is
registered");
}

public void stop(BundleContext context) throws
Exception {
}
```

Install service bundles

Export your bundles and install them on your server. Start the service bundle.

```
osgi> install file:c:\temp\bundles\plugins\de.vogella.osgi.quote_1.0.0.jar
Bundle id is 3

osgi> install file:c:\temp\bundles\plugins\de.vogella.osgi.quoteservice_1.0.0.jar
Bundle id is 4

osgi> start 4
IQuoteService is registered

osgi>
```

Yet to consume our service.

Use your service

Create a new plugin "de.vogella.osgi.quoteconsumer". Add also a dependency to the package "de.vogella.osgi.quote".

Dependencies

Required Plug-ins

Specify the list of plug-ins required for the operation of this plug-in.

| | |
|------------------|-------------------------------|
| org.eclipse.osgi | Add... |
| | Remove |
| | Up |
| | Down |
| | Properties... |

Total: 1

Imported Packages

Specify packages on which this plug-in depends without explicitly identifying their originating plug-in.

| | |
|----------------------------|-------------------------------|
| de.vogella.osgi.quote | Add... |
| org.osgi.framework (1.3.0) | Remove |
| | Properties... |

Total: 2

▶ Automated Management of Dependencies

▶ Dependency Analysis

Overview Dependencies Runtime Build MANIFEST.MF build.properties

Let's register directly to the service and use it.

```
package de.vogella.osgi.quoteconsumer;

import org.osgi.framework.BundleActivator;
import org.osgi.framework.BundleContext;
import org.osgi.framework.ServiceReference;

import de.vogella.osgi.quote.IQuoteService;

public class Activator implements BundleActivator {

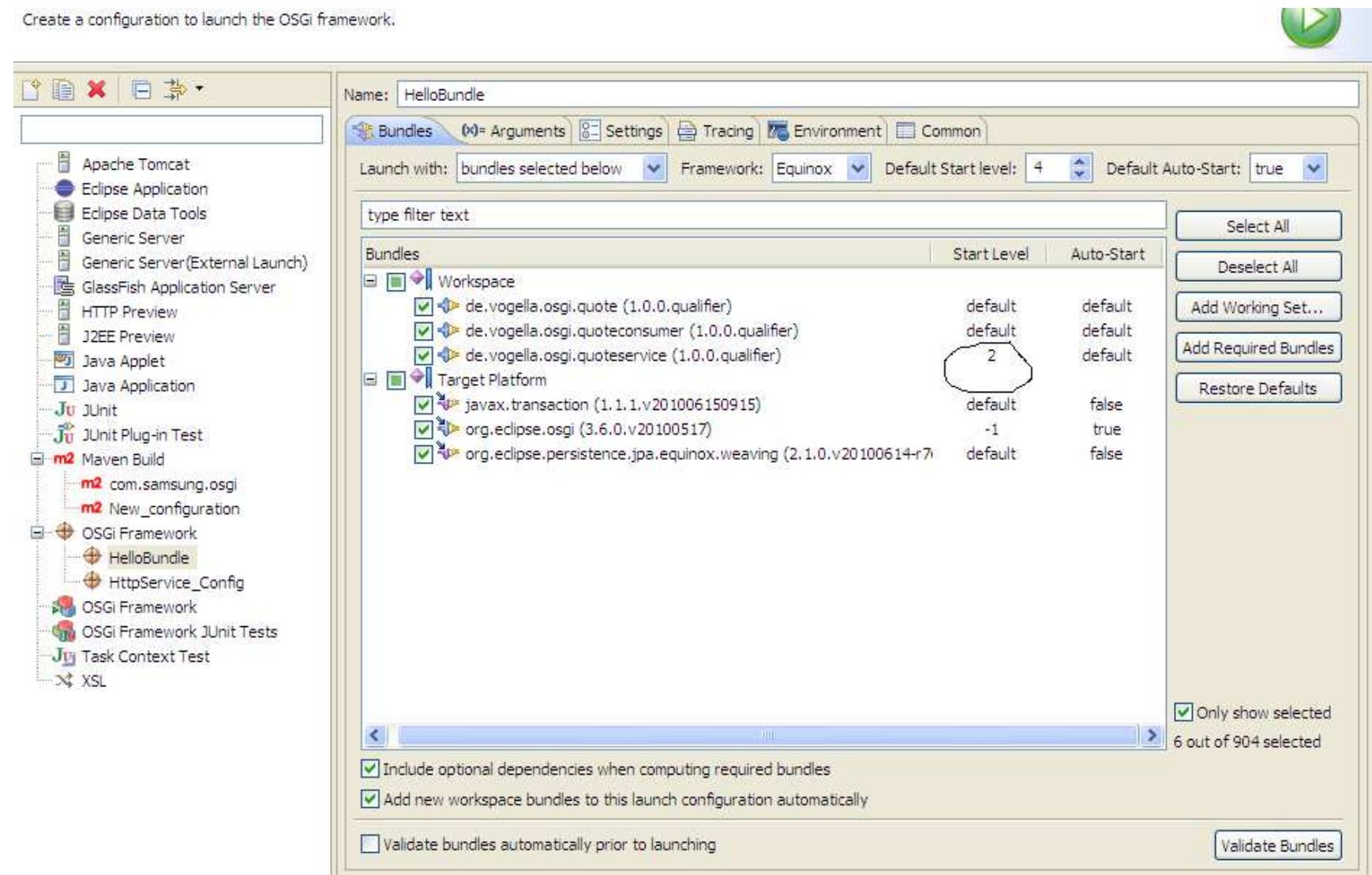
    private BundleContext context;
    private IQuoteService service;

    public void start(BundleContext context) throws Exception {
        this.context = context;
        // Register directly with the service
        ServiceReference reference = context
                .getServiceReference(IQuoteService.class.getName());
        service = (IQuoteService) context.getService(reference);
        System.out.println(service.getQuote());
    }

    public void stop(BundleContext context) throws Exception {
        System.out.println(service.getQuote());
    }
}
```

Run the Bundle in the Eclipse

Create a configuration to launch the OSGI framework.



Export this bundle, install it and start and stop it. Everything work. But if you stop the service bundle then your receive an error.

```
osgi> install file:c:\temp\bundles\plugins\de.vogella.osgi.quoteconsumer_1.0.0.jar
Bundle id is 5

osgi> start 5
Tell them I said something

osgi> stop 5
Tell them I said something

osgi> stop 4

osgi> stop 5

osgi> start 5
org.osgi.framework.BundleException: Exception in de.vogella.osgi.quoteconsumer.Activator.start() of bundle de.vogella.osgi.quoteconsumer.
    at org.eclipse.osgi.framework.internal.core.BundleContextImpl.startActivator(BundleContextImpl.java:1028)
    at org.eclipse.osgi.framework.internal.core.BundleContextImpl.start(BundleContextImpl.java:984)
    at org.eclipse.osgi.framework.internal.core.BundleHost.startWorker(BundleHost.java:346)
```

The reason for this is that OSGi is a very dynamic environment and service may be registered and de-registered any time.

Try the following:

Stop the service

Refresh the consumer

Start service

Refresh the consumer

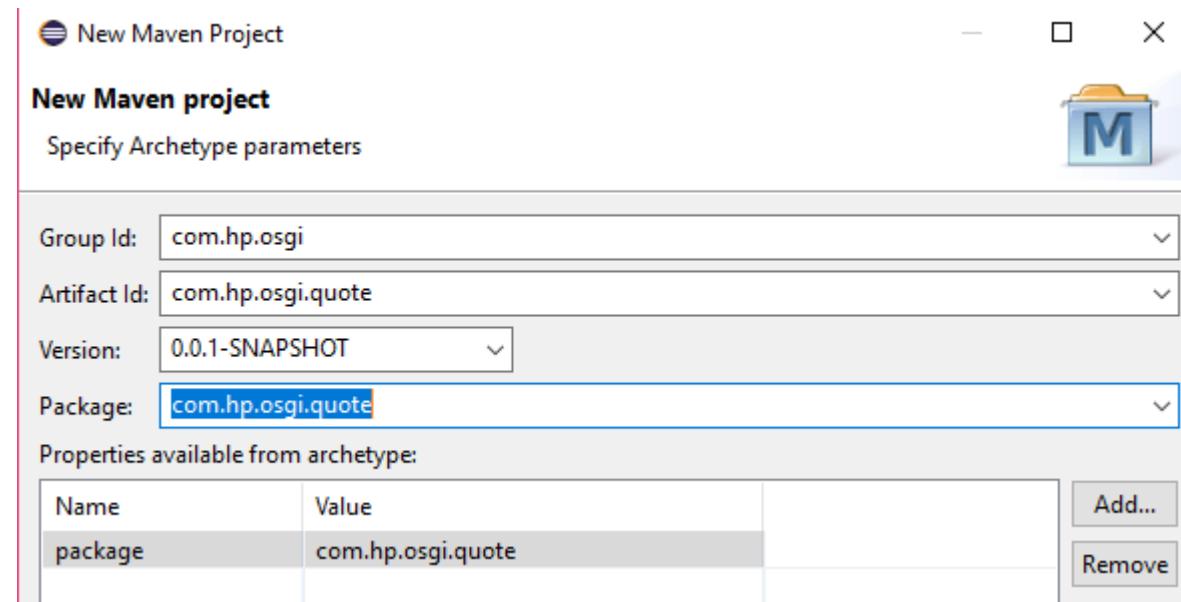
Services Using Karaf

In this lab, we are going to create three bundles; to demonstrate the services creation in OSGI

1. Quote bundle: It will contain the interface of the business functions
2. Quote Service bundle: It will implement the business functions.
3. Quote Consumer: It will consume the service by looking up from the service registry.

Define the service interface

Create a maven project using karaf archetype "com.hp.osgi.quote" and the package " com.hp.osgi.quote". Do not use a template. You do not need an activator. Afterwards add " com.hp.osgi.quote" to the exported packages. You need to specify this in the pom.xml, and the maven plugin will generate the Manifest.mf file for our plugin.



Update pom.xml with the following

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<!--
    Licensed to the Apache Software Foundation (ASF) under one
    or more contributor license agreements. See the NOTICE file
    distributed with this work for additional information
    regarding copyright ownership. The ASF licenses this file
    to you under the Apache License, Version 2.0 (the
    "License"); you may not use this file except in compliance
    with the License. You may obtain a copy of the License at

        http://www.apache.org/licenses/LICENSE-2.0

    Unless required by applicable law or agreed to in writing,
    software distributed under the License is distributed on an
    "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
    KIND, either express or implied. See the License for the
    specific language governing permissions and limitations
    under the License.
-->

<modelVersion>4.0.0</modelVersion>

<groupId>com.hp.osgi</groupId>
<artifactId>com.hp.osgi.quote</artifactId>
<version>0.0.1-SNAPSHOT</version>
<packaging>bundle</packaging>

<name>com.hp.osgi.quote Bundle</name>
<description>
    com.hp.osgi.quote OSGi bundle project.
</description>

<properties>
    <maven-bundle-plugin.version>2.5.4</maven-bundle-plugin.version>
```

```
<osgi.version>6.0.0</osgi.version>
</properties>

<dependencies>
    <dependency>
        <groupId>org.osgi</groupId>
        <artifactId>org.osgi.core</artifactId>
        <version>${osgi.version}</version>
        <scope>provided</scope>
    </dependency>
</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.felix</groupId>
            <artifactId>maven-bundle-plugin</artifactId>
            <version>${maven-bundle-plugin.version}</version>
            <extensions>true</extensions>
            <configuration>
                <instructions>
                    <Bundle-SymbolicName>${project.artifactId}</Bundle-SymbolicName>
                    <Bundle-Version>${project.version}</Bundle-Version>
                    <Bundle-Activator>com.hp.osgi.quote.Activator</Bundle-Activator>
                    <Export-Package>
                        com.hp.osgi.quote;version=${project.version}
                    </Export-Package>
                    <Import-Package>
                        *
                    </Import-Package>
                </instructions>
            </configuration>
        </plugin>
    </plugins>
</build>

</project>
```

Verify that you have exported the correct package as shown below:

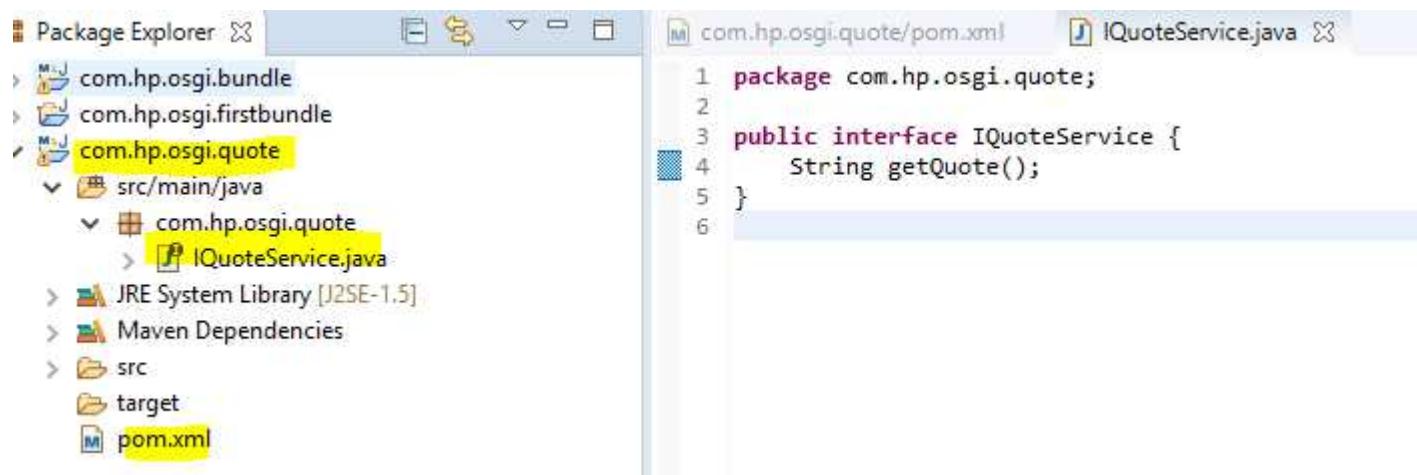
```
40
49  <build>
50    <plugins>
51      <plugin>
52        <groupId>org.apache.felix</groupId>
53        <artifactId>maven-bundle-plugin</artifactId>
54        <version>${maven-bundle-plugin.version}</version>
55        <extensions>true</extensions>
56      </plugin>
57      <instructions>
58        <Bundle-SymbolicName>${project.artifactId}</Bundle-SymbolicName>
59        <Bundle-Version>${project.version}</Bundle-Version>
60        <Bundle-Activator>com.hp.osgi.quote.Activator</Bundle-Activator>
61        <Export-Package>
62          com.hp.osgi.quote;version=${project.version}
63        </Export-Package>
64        <Import-Package>
65          *
66          </Import-Package>
67        </instructions>
68      </configuration>
69    </plugin>
70  </plugins>
71 </build>
```

Create the following interface "IQuoteService".

```
package com.hp.osgi.quote;

public interface IQuoteService {
    String getQuote();
}
```

We don't require the Activator class in this bundle, so ensure that you delete it from the project. At the end of the above step, your project should be as shown below:



Create service

We will now define a bundle which will provide the service.

Create a maven karaf project " com.hp.osgi.quoteservice". Do not use a template.

New Maven project

Specify Archetype parameters

Group Id: com.hp.osgi

Artifact Id: com.hp.osgi.quoteservice

Version: 0.0.1-SNAPSHOT

Package: com.hp.osgi.quoteservice

Properties available from archetype:

| Name | Value |
|---------|--------------------------|
| package | com.hp.osgi.quoteservice |

Add... Remove



Update the pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<!--
    Licensed to the Apache Software Foundation (ASF) under one
    or more contributor license agreements. See the NOTICE file
    distributed with this work for additional information
    regarding copyright ownership. The ASF licenses this file
    to you under the Apache License, Version 2.0 (the
    "License"); you may not use this file except in compliance
    with the License. You may obtain a copy of the License at

        http://www.apache.org/licenses/LICENSE-2.0

    Unless required by applicable law or agreed to in writing,
    software distributed under the License is distributed on an
    "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
    KIND, either express or implied. See the License for the
    specific language governing permissions and limitations
    under the License.
-->

<modelVersion>4.0.0</modelVersion>

<groupId>com.hp.osgi</groupId>
<artifactId>com.hp.osgi.quoteservice</artifactId>
<version>0.0.1-SNAPSHOT</version>
<packaging>bundle</packaging>

<name>com.hp.osgi.quoteservice Bundle</name>
<description>
    com.hp.osgi.quoteservice OSGi bundle project.
</description>

<properties>
    <maven-bundle-plugin.version>2.5.4</maven-bundle-plugin.version>
```

```
<osgi.version>6.0.0</osgi.version>
</properties>

<dependencies>
    <dependency>
        <groupId>org.osgi</groupId>
        <artifactId>org.osgi.core</artifactId>
        <version>${osgi.version}</version>
        <scope>provided</scope>
    </dependency>
</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.felix</groupId>
            <artifactId>maven-bundle-plugin</artifactId>
            <version>${maven-bundle-plugin.version}</version>
            <extensions>true</extensions>
            <configuration>
                <instructions>
                    <Bundle-SymbolicName>${project.artifactId}</Bundle-SymbolicName>
                    <Bundle-Version>${project.version}</Bundle-Version>
                    <Bundle-Activator>com.hp.osgi.quoteservice.Activator</Bundle-Activator>
                    <Export-Package>
                        com.hp.osgi.quoteservice*;version=${project.version}
                    </Export-Package>
                    <Require-Bundle>com.hp.osgi.quote</Require-Bundle>
                    <Import-Package>
                        *
                    </Import-Package>
                </instructions>
            </configuration>
        </plugin>
    </plugins>
</build>

</project>
```

Add "com.hp.osgi.quote" to the required plugins.

```
50      <plugins>
51        <plugin>
52          <groupId>org.apache.felix</groupId>
53          <artifactId>maven-bundle-plugin</artifactId>
54          <version>${maven-bundle-plugin.version}</version>
55          <extensions>true</extensions>
56        <configuration>
57          <instructions>
58            <Bundle-SymbolicName>${project.artifactId}</Bundle-SymbolicName>
59            <Bundle-Version>${project.version}</Bundle-Version>
60            <Bundle-Activator>com.hp.osgi.quoteservice.Activator</Bundle-Activator>
61          <Export-Package>
62            com.hp.osgi.quoteservice*;version=${project.version}
63          </Export-Package>
64          <Require-Bundle>com.hp.osgi.quote</Require-Bundle>
65          <Import-Package>
66            *
67          </Import-Package>
68        </instructions>
69      </configuration>
70    </plugin>
```

Include the interface project as dependency in the Service project:

```
<dependency>
    <groupId>com.hp.osgi</groupId>
    <artifactId>com.hp.osgi.quote</artifactId>
    <version>0.0.1-SNAPSHOT</version>
</dependency>
```

Create the following class "QuoteService".

```
package com.hp.osgi.quoteservice.internal;

import java.util.Random;

import com.hp.osgi.quote.IQuoteService;
public class QuoteService implements IQuoteService {

    public String getQuote() {
        Random random = new Random();
        // Create a number between 0 and 2
        int nextInt = random.nextInt(3);
        switch (nextInt) {
        case 0:
            return "Tell them I said something";
        case 1:
            return "I feel better already";
        default:
            return "Oh, You are in default!";
        }
    }
}
```

Register the service in the class Activator.

```
package com.hp.osgi.quoteservice;

import org.osgi.framework.BundleActivator;
import org.osgi.framework.BundleContext;

import com.hp.osgi.quote.IQuoteService;
import com.hp.osgi.quoteservice.internal.QuoteService;

public class Activator implements BundleActivator {

    public void start(BundleContext context) throws Exception {
        IQuoteService service = new QuoteService();
        // Third parameter is a hashmap which allows to configure the service
        // Not required in this example
        context.registerService(IQuoteService.class.getName(), service,
                               null);
        System.out.println("IQuoteService is registered");
    }

    public void stop(BundleContext context) throws Exception {
}
```

At the end, you should have the service bundle as shown below:

The screenshot shows the Eclipse IDE interface with the Package Explorer and the Java editor side-by-side.

Package Explorer:

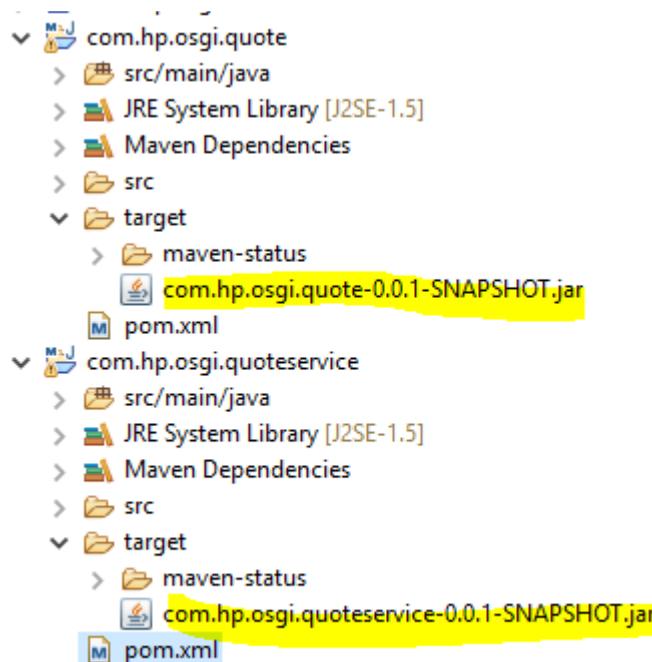
- com.hp.osgi.bundle
- com.hp.osgi.firstbundle
- com.hp.osgi.quote
- com.hp.osgi.quoteservice (selected)
 - src/main/java
 - com.hp.osgi.quoteservice
 - Activator.java
 - com.hp.osgi.quoteservice.internal
 - QuoteService.java
 - JRE System Library [J2SE-1.5]
 - Maven Dependencies
 - src
 - target
 - pom.xml

Java Editor (Activator.java):

```
* Licensed to the Apache Software Foundation (ASF) under one or more
*  package com.hp.osgi.quoteservice;
import org.osgi.framework.BundleActivator;
import org.osgi.framework.BundleContext;
import com.hp.osgi.quote.IQuoteService;
import com.hp.osgi.quoteservice.internal.QuoteService;
public class Activator implements BundleActivator {
    public void start(BundleContext context) throws Exception {
        IQuoteService service = new QuoteService();
        // Third parameter is a hashmap which allows to configure the service
        // Not required in this example
        context.registerService(IQuoteService.class.getName(), service,
                               null);
        System.out.println("IQuoteService is registered");
    }
    public void stop(BundleContext context) throws Exception {
    }
}
```

Now let us install Quote and Service bundles after exporting the bundle.

How do you export the bundle? You need to run maven install command in both the project to create bundle jar. Generate the bundle in sequential manner i.e Quote then Quote Service bundle.



Export your bundles and install them on your server. Start the service bundle.

```
#bundle:install file:///D:/temp/com.hp.osgi.quote-0.0.1-SNAPSHOT.jar  
#bundle:install file:///D:/temp/com.hp.osgi.quoteservice-0.0.1-SNAPSHOT.jar
```

```
karaf@root()> bundle:install file:///D:/temp/com.hp.osgi.quote-0.0.1-SNAPSHOT.jar  
Bundle ID: 68  
karaf@root()> bundle:install file:///D:/temp/com.hp.osgi.quoteservice-0.0.1-SNAPSHOT.jar  
Bundle ID: 69  
karaf@root()> bundle:list  
START LEVEL 100 , List Threshold: 50  
ID | State      | Lvl | Version           | Name  
---  
66 | Installed   | 80  | 1.0.0.201707081832 | Firstbundle  
67 | Active      | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle  
68 | Installed   | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quote Bundle  
69 | Installed   | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quoteservice Bundle  
karaf@root()>
```

```
bundle:start 69  
bundle:start 68
```

```

karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version            | Name
--+
66 | Installed   | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.bundle Bundle
68 | Installed   | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quote Bundle
69 | Installed   | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quoteservice Bundle
karaf@root()> bundle:start 69
Error executing command: Error executing command on bundles:
    Error starting bundle 69: Unable to resolve com.hp.osgi.quoteservice [69](R 69.0): missing requirement [com.hp.osgi.quoteservice [69](R 69.0)] osgi.wiring.bundle; (osgi.wiring.bundle=org.apache.cxf.bundle) Unresolved requirements: [[com.hp.osgi.quoteservice [69](R 69.0)] osgi.wiring.bundle; (osgi.wiring.bundle=org.apache.cxf.bundle)]
karaf@root()> bundle:start 68
karaf@root()> bundle:start 69
Error executing command: Error executing command on bundles:
    Error starting bundle 69: Unable to resolve com.hp.osgi.quoteservice [69](R 69.0): missing requirement [com.hp.osgi.quoteservice [69](R 69.0)] osgi.wiring.bundle; (osgi.wiring.bundle=org.apache.cxf.bundle) Unresolved requirements: [[com.hp.osgi.quoteservice [69](R 69.0)] osgi.wiring.bundle; (osgi.wiring.bundle=org.apache.cxf.bundle)]
karaf@root()>

```

You get the above error why? Verify the Required Bundle is installed properly before the service Bundle is started. Once you resolved the issue it should be as shown below when you start the service bundle:

```

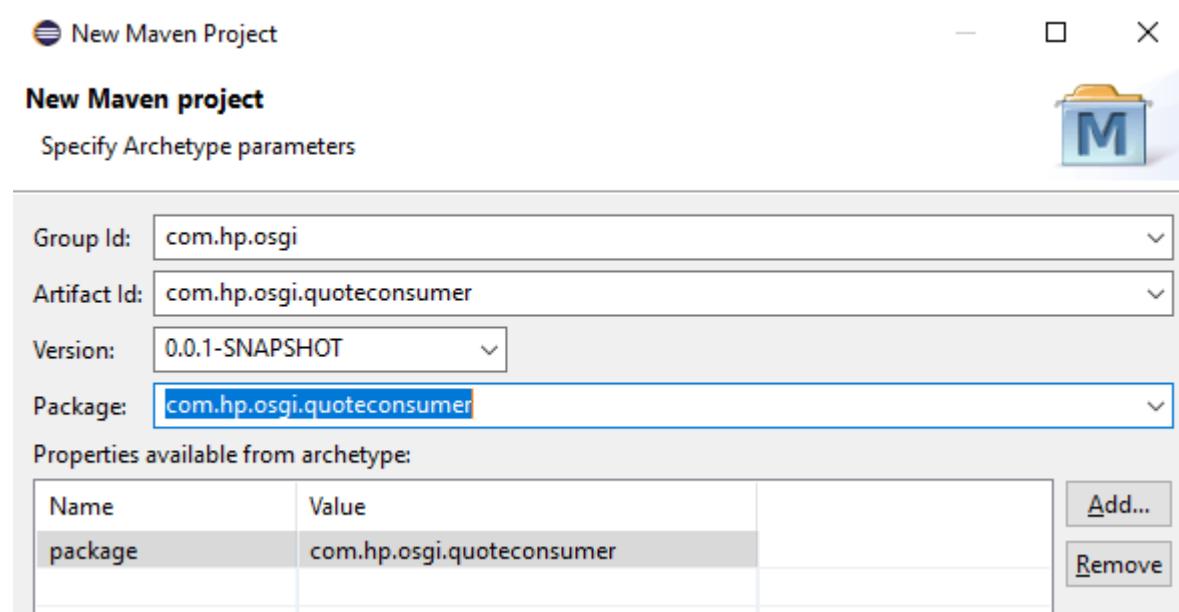
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version            | Name
--+
66 | Installed   | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.bundle Bundle
68 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quote Bundle
70 | Installed   | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quoteservice Bundle
karaf@root()> bundle:start 70
IQuoteService is registered
karaf@root()>

```

We have successfully installed two bundles i.e Quote and Service . Let us develop consumer bundle to consume our service.

Use your service

Create a new maven project " com.hp.osgi.quoteconsumer".



Update the pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<!--
    Licensed to the Apache Software Foundation (ASF) under one
    or more contributor license agreements. See the NOTICE file
    distributed with this work for additional information
    regarding copyright ownership. The ASF licenses this file
    to you under the Apache License, Version 2.0 (the
    "License"); you may not use this file except in compliance
    with the License. You may obtain a copy of the License at

        http://www.apache.org/licenses/LICENSE-2.0

    Unless required by applicable law or agreed to in writing,
    software distributed under the License is distributed on an
    "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
    KIND, either express or implied. See the License for the
    specific language governing permissions and limitations
    under the License.
-->

<modelVersion>4.0.0</modelVersion>

<groupId>com.hp.osgi</groupId>
<artifactId>com.hp.osgi.quoteconsumer</artifactId>
<version>0.0.1-SNAPSHOT</version>
<packaging>bundle</packaging>

<name>com.hp.osgi.quoteconsumer Bundle</name>
<description>
    com.hp.osgi.quoteconsumer OSGi bundle project.
</description>

<properties>
    <maven-bundle-plugin.version>2.5.4</maven-bundle-plugin.version>
```

```
<osgi.version>6.0.0</osgi.version>
</properties>

<dependencies>
    <dependency>
        <groupId>org.osgi</groupId>
        <artifactId>org.osgi.core</artifactId>
        <version>${osgi.version}</version>
        <scope>provided</scope>
    </dependency>
</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.felix</groupId>
            <artifactId>maven-bundle-plugin</artifactId>
            <version>${maven-bundle-plugin.version}</version>
            <extensions>true</extensions>
            <configuration>
                <instructions>
                    <Bundle-SymbolicName>${project.artifactId}</Bundle-SymbolicName>
                    <Bundle-Version>${project.version}</Bundle-Version>
                    <Bundle-Activator>com.hp.osgi.quoteconsumer.Activator</Bundle-Activator>
                    <Export-Package>
                        com.hp.osgi.quoteconsumer*;version=${project.version}
                    </Export-Package>
                    <Import-Package>
                        com.hp.osgi.quote,*</Import-Package>
                </instructions>
            </configuration>
        </plugin>
    </plugins>
</build>

</project>
```

Add also a dependency to the package "com.hp.osgi.quote" i.e of the Interface Quote.

```
i9@      <plugin>
i0        <groupId>org.apache.felix</groupId>
i1        <artifactId>maven-bundle-plugin</artifactId>
i2        <version>${maven-bundle-plugin.version}</version>
i3        <extensions>true</extensions>
i4        <configuration>
i5          <instructions>
i6            <Bundle-SymbolicName>${project.artifactId}</Bundle-SymbolicName>
i7            <Bundle-Version>${project.version}</Bundle-Version>
i8            <Bundle-Activator>com.hp.osgi.quoteconsumer.Activator</Bundle-Activator>
i9          <Export-Package>
i0            com.hp.osgi.quoteconsumer*;version=${project.version}
i1          </Export-Package>
i2          <Import-Package>
i3            com.hp.osgi.quote,*<!-- com.hp.osgi.quote -->
i4          </Import-Package>
i5        </instructions>
i6      </configuration>
i7    </plugin>
i8  </plugins>
i9 </build>
```

Update the pom.xml to specify the dependency of the project – Quote interface.

```
<dependency>
    <groupId>com.hp.osgi</groupId>
    <artifactId>com.hp.osgi.quote</artifactId>
    <version>0.0.1-SNAPSHOT</version>
</dependency>

32
33<dependencies>
34<dependency>
35    <groupId>org.osgi</groupId>
36    <artifactId>org.osgi.core</artifactId>
37    <version>${osgi.version}</version>
38    <scope>provided</scope>
39</dependency>
40<dependency>
41    <groupId>com.hp.osgi</groupId>
42    <artifactId>com.hp.osgi.quote</artifactId>
43    <version>0.0.1-SNAPSHOT</version>
44</dependency>
45</dependencies>
46
```

Let's look up the register for the service , refer the service and use it from the consumer bundle. Update the Activator class with the following code.

```
package com.hp.osgi.quoteconsumer;

import org.osgi.framework.BundleActivator;
import org.osgi.framework.BundleContext;
import org.osgi.framework.ServiceReference;

import com.hp.osgi.quote.IQuoteService;

public class Activator implements BundleActivator {

    private BundleContext context;
    private IQuoteService service;

    public void start(BundleContext context) throws Exception {
        this.context = context;
        // Register directly with the service
        ServiceReference reference = context
            .getServiceReference(IQuoteService.class.getName());
        service = (IQuoteService) context.getService(reference);
        System.out.println(service.getQuote());
    }

    public void stop(BundleContext context) throws Exception {
        System.out.println(service.getQuote());
    }
}
```

Export the Consumer Bundle using install command and deploy in the container.

```
#bundle:install file:///D:/temp/com.hp.osgi.quoteconsumer-0.0.1-SNAPSHOT.jar
```

Start the bundle

```
# bundle:start 72
```

```
70 | ACTIVE   | 80 | 0.0.1.SNAPSHOT      | com.hp.osgi.quoteservice Bundle  
karaf@root()> bundle:install file:///D:/temp/com.hp.osgi.quoteconsumer-0.0.1-SNAPSHOT.jar  
Bundle ID: 72  
karaf@root()> bundle:start 72  
Oh, You are in default!  
karaf@root()> bundle:list  
START LEVEL 100 , List Threshold: 50  
ID | State    | Lvl | Version          | Name  
---  
66 | Installed | 80 | 1.0.0.201707081832 | Firstbundle  
67 | Active    | 80 | 0.0.1.SNAPSHOT     | com.hp.osgi.bundle Bundle  
68 | Active    | 80 | 0.0.1.SNAPSHOT     | com.hp.osgi.quote Bundle  
70 | Active    | 80 | 0.0.1.SNAPSHOT     | com.hp.osgi.quoteservice Bundle  
72 | Active    | 80 | 0.0.1.SNAPSHOT     | com.hp.osgi.quoteconsumer Bundle  
karaf@root()>
```

Let us stop the consumer bundle:

```

karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version           | Name
---+-----+-----+-----+-----+
66 | Installed  | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle
68 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quote Bundle
70 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quoteservice Bundle
72 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quoteconsumer Bundle
karaf@root()> bundle:stop 72
Oh, You are in default!
karaf@root()> bundle:stop 72
karaf@root()

```

Nothing happened, everything is working well. Now Let us stop the Quote Service bundle i.e 70 and stop/refresh the Consumer bundle then start the Consumer i.e 72

```

karaf@root()> bundle:stop 72
karaf@root()> bundle:stop 70
karaf@root()> bundle:refresh 72
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version           | Name
---+-----+-----+-----+-----+
66 | Installed  | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle
68 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quote Bundle
70 | Resolved   | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quoteservice Bundle
72 | Installed  | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quoteconsumer Bundle
karaf@root()> bundle:start 72
Error executing command: Error executing command on bundles:
    Error starting bundle 72: Activator start error in bundle com.hp.osgi.quoteconsumer [72].
karaf@root()

```

You should get an error. Why? It's because the Service is down. You can start the service bundle now.

```
#bundle:start 70  
#bundle:start 72
```

```
karaf@root()> bundle:start 70  
IQuoteService is registered  
karaf@root()> bundle:refresh 72  
karaf@root()> bundle:list  
START LEVEL 100 , List Threshold: 50  
ID | State | Lvl | Version | Name  
---  
66 | Installed | 80 | 1.0.0.201707081832 | Firstbundle  
67 | Active | 80 | 0.0.1.SNAPSHOT | com.hp.osgi.bundle Bundle  
68 | Active | 80 | 0.0.1.SNAPSHOT | com.hp.osgi.quote Bundle  
70 | Active | 80 | 0.0.1.SNAPSHOT | com.hp.osgi.quoteservice Bundle  
72 | Installed | 80 | 0.0.1.SNAPSHOT | com.hp.osgi.quoteconsumer Bundle  
karaf@root()> bundle:start 72  
Oh, You are in default!  
karaf@root()
```

The reason for this is that OSGi is a very dynamic environment and service may be registered and de-registered any time.

Try the following:

Stop the service bundle
Refresh the consumer
Start service
Refresh the consumer

```
68 | Active    | 80 | 0.0.1.SNAPSHOT | com.hp.osgi.quote Bundle
70 | Active    | 80 | 0.0.1.SNAPSHOT | com.hp.osgi.quoteservice Bundle
72 | Active    | 80 | 0.0.1.SNAPSHOT | com.hp.osgi.quoteconsumer Bundle
karaf@root()> bundle:stop 70
karaf@root()> bundle:refresh 72
I feel better already
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State    | Lvl | Version      | Name
-----
66 | Installed | 80 | 1.0.0.201707081832 | Firstbundle
67 | Active    | 80 | 0.0.1.SNAPSHOT | com.hp.osgi.bundle Bundle
68 | Active    | 80 | 0.0.1.SNAPSHOT | com.hp.osgi.quote Bundle
70 | Resolved   | 80 | 0.0.1.SNAPSHOT | com.hp.osgi.quoteservice Bundle
72 | Resolved   | 80 | 0.0.1.SNAPSHOT | com.hp.osgi.quoteconsumer Bundle
karaf@root()> bundle:start 70
IQuoteService is registered
karaf@root()> bundle:refresh 72
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State    | Lvl | Version      | Name
-----
66 | Installed | 80 | 1.0.0.201707081832 | Firstbundle
67 | Active    | 80 | 0.0.1.SNAPSHOT | com.hp.osgi.bundle Bundle
68 | Active    | 80 | 0.0.1.SNAPSHOT | com.hp.osgi.quote Bundle
70 | Active    | 80 | 0.0.1.SNAPSHOT | com.hp.osgi.quoteservice Bundle
72 | Installed  | 80 | 0.0.1.SNAPSHOT | com.hp.osgi.quoteconsumer Bundle
karaf@root()> bundle:start 72
Tell them I said something
karaf@root()
```

What yours observation? You can start the consumer now.

Import / Required / Version Bundle - Virgo

- 1) Export the bundles as follows: (Created earlier)
 - a. Quote
 - b. Quote service
 - c. Quote consumer

Install the above bundles using telnet to Virgo server.

Install file:d:/OSGi/{Quote}.jar

Make sure that everything is working as expected.

Refresh the Quote consumer bundle.

```
osgi> refresh 157
osgi> [2012-08-25 18:38:50.828] Refresh Packages <DE0010I> Stopping
bundle 'de.vogella.osgi.quoteconsumer' version '1.0.0.201208251831'.
Hubba Bubba, Baby!
[2012-08-25 18:38:50.828] Refresh Packages <DE0011I> Stopped bundle
'de.vogella.osgi.quoteconsumer' version '1.0.0.201208251831'.
[2012-08-25 18:38:50.843] Refresh Packages <DE0004I> Starting bundle
'de.vogella.osgi.quoteconsumer' version '1.0.0.201208251831'.
I feel better already
```

osgi> packages {bundle id} [replace the bundle id of with all the three bundles you have installed above]

```
gw Telnet localhost
143 ACTIVE org.eclipse.equinox.http.registry_1.1.0.v20100503
144 ACTIVE MyActivator_3.0.0.201208200009
145 ACTIVE com.hp.springdm-web-sample.simple-web_1.0.1.SNAPSHOT
146 ACTIVE com.manning.sdmia.springdm-sample_0.0.1.SNAPSHOT
147 ACTIVE javax.ws.rs.jsr311-api_1.1.1
148 ACTIVE com.sun.jersey.jersey-core_1.12.0
149 ACTIVE com.sun.jersey.jersey-server_1.12.0
150 ACTIVE com.sun.jersey.jersey-servlet_1.12.0
151 ACTIVE com.hp.springdm-web-rest.simple-web-rest_2.0.1.SNAPSHOT
152 ACTIVE com.hp.osgi.com.hp.mock_0.0.1.SNAPSHOT
154 ACTIVE com.hp.osgi.log.client.com.hp.mock.log.client_0.0.1.SNAPSHOT
155 ACTIVE de.vogella.osgi.quote_1.0.0.201208251831
156 ACTIVE de.vogella.osgi.quoteservice_1.0.0.201208251831
157 ACTIVE de.vogella.osgi.quoteconsumer_1.0.0.201208251831
osgi> packages 157
Export-Package: org.osgi.framework; version="1.7.0": 1
Export-Package: de.vogella.osgi.quote; version="0.0.0": 1
osgi> packages 156
Export-Package: org.osgi.framework; version="1.7.0": 1
Export-Package: de.vogella.osgi.quote; version="0.0.0": 1
osgi> package 155
gogo: CommandNotFoundException: Command not found: package
osgi> packages 155
Export-Package: org.osgi.framework; version="1.7.0": 1
osgi>
```

- Change the version of Quote Bundle as follows

META-INF\MANIFEST.mf

```
Export-Package: de.vogella.osgi.quote;version="1.0.0"
```

- Uninstalled the earlier Quote Bundle (> `uninstall {bundle id}`)
- Export the above bundle and install the bundle.
- Refresh the Consumer Bundle and start again.

The screenshot shows a Telnet session connected to 'localhost'. The window title is 'Telnet localhost'. Inside, the OSGi service registry is displayed, listing various services with their IDs, states, and service names. Below the registry, several commands are entered and executed:

```

142 ACTIVE org.eclipse.equinox.registry_3.5.0.v20100503
143 ACTIVE org.eclipse.equinox.http.registry_1.1.0.v20100503
144 ACTIVE MyActivator_3.0.0.201208200009
145 ACTIVE com.hp.springdm-web-sample.simple-web_1.0.1.SNAPSHOT
146 ACTIVE com.manning.sdmia.springdm-sample_0.0.1.SNAPSHOT
147 ACTIVE javax.ws.rs.jsr311-api_1.1.1
148 ACTIVE com.sun.jersey.jersey-core_1.12.0
149 ACTIVE com.sun.jersey.jersey-server_1.12.0
150 ACTIVE com.sun.jersey.jersey-servlet_1.12.0
151 ACTIVE com.hp.springdm-web-rest.simple-web-rest_2.0.1.SNAPSHOT
152 ACTIVE com.hp.osgi.com.hp.mock_0.0.1.SNAPSHOT
154 ACTIVE com.hp.osgi.log.client.com.hp.mock.log.client_0.0.1.SNAPSHOT
156 ACTIVE de.vogella.osgi.quoteservice_1.0.0.201208251831
157 RESOLVED de.vogella.osgi.quoteconsumer_1.0.0.201208251831
158 ACTIVE de.vogella.osgi.quote_1.1.0

osgi> start 157
[2012-08-25 18:59:05.046] Thread-35 <DE0004I> Starting bundle
'de.vogella.osgi.quoteconsumer' version '1.0.0.201208251831'.
[2012-08-25 18:59:05.062] Thread-35 <DE0010I> Stopping bundle
'de.vogella.osgi.quoteconsumer' version '1.0.0.201208251831'.
[2012-08-25 18:59:05.062] Thread-35 <DE0011I> Stopped bundle
'de.vogella.osgi.quoteconsumer' version '1.0.0.201208251831'.
gogo: BundleException: Exception in de.vogella.osgi.quoteconsumer.Activator.star
t() of bundle de.vogella.osgi.quoteconsumer.
osgi>

```

- Changes the Consumer Bundle Manifest file with the following:
 - Bundle-Version: 1.1.0
 - `de.vogella.osgi.quote;version="1.0.0,1.2.0"`
- Export the above bundle and deploy it and start / refresh the bundle

```
Telnet localhost
143 ACTIVE org.eclipse.equinox.http.registry_1.1.0.v20100503
144 ACTIVE MyActivator_3.0.0.201208200009
145 ACTIVE com.hp.springdm-web-sample.simple-web_1.0.1.SNAPSHOT
146 ACTIVE com.manning.sdmia.springdm-sample_0.0.1.SNAPSHOT
147 ACTIVE javax.ws.rs.Jsr311-api_1.1.1
148 ACTIVE com.sun.jersey.jersey-core_1.12.0
149 ACTIVE com.sun.jersey.jersey-server_1.12.0
150 ACTIVE com.sun.jersey.jersey-servlet_1.12.0
151 ACTIVE com.hp.springdm-web-rest.simple-web-rest_2.0.1.SNAPSHOT
152 ACTIVE com.hp.osgi.com.hp.mock_0.0.1.SNAPSHOT
154 ACTIVE com.hp.osgi.log.client.com.hp.mock.log.client_0.0.1.SNAPSHOT
156 ACTIVE de.vogella.osgi.quoteservice_1.0.0.201208251831
157 ACTIVE de.vogella.osgi.quoteconsumer_1.0.0.201208251831
159 ACTIVE de.vogella.osgi.quote_1.0.0
161 ACTIVE de.vogella.osgi.quoteconsumer_1.1.0

osgi> refresh 161
osgi> [2012-08-25 19:08:46.750] Refresh Packages <DE0010I> Stopping
bundle 'de.vogella.osgi.quoteconsumer' version '1.1.0'.
Tell them I said something
[2012-08-25 19:08:46.750] Refresh Packages <DE0011I> Stopped bundle
'de.vogella.osgi.quoteconsumer' version '1.1.0'.
[2012-08-25 19:08:46.765] Refresh Packages <DE0004I> Starting bundle
'de.vogella.osgi.quoteconsumer' version '1.1.0'.
Hubba Bubba, Baby!
```

Import / Required / Version Bundle - Karaf

In this lab, we will learn how to specify version ranges in the bundle and how to determine the export/import packages of a bundle

We are going to use the bundles created during the service Lab, export the bundles as follows and install in the karaf container if you have not done earlier.

- Quote
- Quote service
- Quote consumer

Make sure that everything is working as expected.

Refresh the Quote consumer bundle.

```
#bundle:refresh 72
```

```
karaf@root()>
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State   | Lvl | Version           | Name
--+
66 | Installed | 80 | 1.0.0.201707081832 | Firstbundle
67 | Active    | 80 | 0.0.1.SNAPSHOT     | com.hp.osgi.bundle Bundle
68 | Active    | 80 | 0.0.1.SNAPSHOT     | com.hp.osgi.quote Bundle
70 | Active    | 80 | 0.0.1.SNAPSHOT     | com.hp.osgi.quoteservice Bundle
72 | Active    | 80 | 0.0.1.SNAPSHOT     | com.hp.osgi.quoteconsumer Bundle
karaf@root()> bundle:refresh 72
I feel better already
karaf@root()> Oh, You are in default!
```

What is printed in your console? Why two statements? Verified the reason for it?

Let us find out all the packages export by the three bundles using the package:exports command. First get the bundle id using bundle: list command.

```
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version           | Name
---+-----+-----+-----+-----+
66 | Installed | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active    | 80  | 0.0.1.SNAPSHOT   | com.hp.osgi.bundle Bundle
68 | Active    | 80  | 0.0.1.SNAPSHOT   | com.hp.osgi.quote Bundle
70 | Active    | 80  | 0.0.1.SNAPSHOT   | com.hp.osgi.quoteservice Bundle
72 | Active    | 80  | 0.0.1.SNAPSHOT   | com.hp.osgi.quoteconsumer Bundle
karaf@root()> package:exports | grep 68
com.hp.osgi.quote           | 0.0.1.SNAPSHOT | 68 | com.hp.osgi.quote
karaf@root()> package:exports | grep 70
com.hp.osgi.quoteservice.internal | 0.0.1.SNAPSHOT | 70 | com.hp.osgi.quoteservice
com.hp.osgi.quoteservice     | 0.0.1.SNAPSHOT | 70 | com.hp.osgi.quoteservice
karaf@root()> package:exports | grep 72
com.hp.osgi.quoteconsumer    | 0.0.1.SNAPSHOT | 72 | com.hp.osgi.quoteconsumer
karaf@root()
```

Let us change the version of Quote Bundle as follows in pom.xml

```
<Export-Package>
    com.hp.osgi.quote;version=1.0.0
</Export-Package>
```

- Uninstalled the earlier Quote Bundle (bundle:uninstall {bundle id})
#bundle:stop 68
#bundle:uninstall 68

```
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version           | Name
-----
66 | Installed   | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.bundle Bundle
68 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quote Bundle
70 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quoteservice Bundle
72 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quoteconsumer Bundle
karaf@root()> bundle:stop 68
karaf@root()> bundle:uninstall 68
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version           | Name
-----
66 | Installed   | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.bundle Bundle
70 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quoteservice Bundle
72 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quoteconsumer Bundle
karaf@root()>
```

- Export the above bundle , install and start it.
 - o [hints: mvn clean , mvn install]
 - o #bundle:install file:///D:/temp/com.hp.osgi.quote-0.0.1-SNAPSHOT.jar
 - o bundle:start 73

```
karaf@root()> bundle:install file:///D:/temp/com.hp.osgi.quote-0.0.1-SNAPSHOT.jar
Bundle ID: 73
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version           | Name
-----
66 | Installed   | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active      | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle
70 | Active      | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quoteservice Bundle
72 | Active      | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quoteconsumer Bundle
73 | Installed   | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quote Bundle
karaf@root()> bundle:start 73
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version           | Name
-----
66 | Installed   | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active      | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle
70 | Active      | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quoteservice Bundle
72 | Active      | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quoteconsumer Bundle
73 | Active      | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quote Bundle
```

- Refresh the Consumer Bundle and start again. It will try to resolve with the existing bundle information.

```
#bundle:refresh 72
```

```
#bundle:start 72
```

As you can see from the console, the consumer wont start it because of the version dependency violation.

```
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version           | Name
-----
66 | Installed   | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active      | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle
70 | Active      | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quoteservice Bundle
72 | Active      | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quoteconsumer Bundle
73 | Active      | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quote Bundle
karaf@root()> bundle:refresh 72
Oh, You are in default!
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version           | Name
-----
66 | Installed   | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active      | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle
70 | Active      | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quoteservice Bundle
72 | Installed   | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quoteconsumer Bundle
73 | Active      | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quote Bundle
karaf@root()> bundle:start 72
Error executing command: Error executing command on bundles:
      Error starting bundle 72: Unable to resolve com.hp.osgi.quoteconsumer [72](R 72.0): missing requirement [com.hp.osgi.quoteconsumer [72](R 72.0)] osgi.wiring.package; (&(osgi.wiring.package=com.hp.osgi.quote)(version>=0.0.0)(&!(version>=1.0.0))) Unresolved requirements: [[com.hp.osgi.quoteconsumer [72](R 72.0)] osgi.wiring.package; (&(osgi.wiring.package=com.hp.osgi.quote)(version>=0.0.0)(&!(version>=1.0.0)))]
karaf@root()
```

How do you resolve this issue? Hints :- Specify the version in the consumer bundle and restart it.

- Changes the Consumer Bundle Manifest file with the following so that it satisfied the version of Quote package:

```
Bundle-Version: 1.1.0  
com.hp.osgi.quote;version="[1.0.0,1.2.0)",*
```

```
18  <plugins>  
19    <plugin>  
20      <groupId>org.apache.felix</groupId>  
21      <artifactId>maven-bundle-plugin</artifactId>  
22      <version>${maven-bundle-plugin.version}</version>  
23      <extensions>true</extensions>  
24    </configuration>  
25    <instructions>  
26      <Bundle-SymbolicName>${project.artifactId}</Bundle-SymbolicName>  
27      <Bundle-Version>1.1.0</Bundle-Version>  
28      <Bundle-Activator>com.hp.osgi.quoteconsumer.Activator</Bundle-Activator>  
29      <Export-Package>  
30        com.hp.osgi.quoteconsumer*;version=${project.version}  
31      </Export-Package>  
32      <Import-Package>  
33        com.hp.osgi.quote;version="[1.0.0,1.2.0)",*  
34      </Import-Package>  
35    </instructions>  
36  </configuration>  
37 </plugin>  
38 </plugins>
```

- Export the above bundle and deploy it and start / refresh the bundle, it should start without any exception.
Hints : mvn clean and mvn install.

Let us remove the earlier consumer and install the fresh bundle. [Notes: Always execute bundle:list to determine the id of the bundle and verify its status before executing any command]

```
#bundle:uninstall 72
```

```
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version            | Name
-----
66 | Installed   | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.bundle Bundle
70 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quoteservice Bundle
72 | Installed   | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quoteconsumer Bundle
73 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quote Bundle
karaf@root()> bundle:uninstall 72
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version            | Name
-----
66 | Installed   | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.bundle Bundle
70 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quoteservice Bundle
73 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quote Bundle
karaf@root()>
```

```
#bundle:install file:///D:/temp/com.hp.osgi.quoteconsumer-0.0.1-SNAPSHOT.jar
```

Before starting the consumer, let us change the version of that Service Bundle, and follow the following steps:

Uninstall the service, and install the fresh bundle:

Update the version in the Service Bundle as shown below

The screenshot shows the Eclipse IDE interface. On the left, the Package Explorer view displays several projects: com.hp.osgi.bundle, com.hp.osgi.firstbundle, com.hp.osgi.quote, com.hp.osgi.quoteconsumer, and com.hp.osgi.quoteservice. The com.hp.osgi.quoteservice project is expanded, showing its src/main/java, JRE System Library [J2SE-1.5], Maven Dependencies, and target folders. Inside target, there is a maven-status folder and a com.hp.osgi.quoteservice-0.0.1-SNAPSHOT folder containing a pom.xml file. The central area has two tabs open: com.hp.osgi.quoteconsumer/pom.xml and com.hp.osgi.quoteservice/pom.xml. The com.hp.osgi.quoteservice/pom.xml tab is active, showing Maven configuration code. The line containing the bundle version is highlighted with a yellow background: <Bundle-Version>1.0.0</Bundle-Version>. The code also includes configurations for export and import packages, as well as dependencies on other bundles.

```
<plugin>
    <groupId>org.apache.felix</groupId>
    <artifactId>maven-bundle-plugin</artifactId>
    <version>${maven-bundle-plugin.version}</version>
    <extensions>true</extensions>
    <configuration>
        <instructions>
            <Bundle-SymbolicName>${project.artifactId}</Bundle-SymbolicName>
            <Bundle-Version>1.0.0</Bundle-Version>
            <Bundle-Activator>com.hp.osgi.quoteservice.Activator</Bundle-Activator>
            <Export-Package>
                com.hp.osgi.quoteservice*;version=1.0.0
            </Export-Package>
            <Require-Bundle>com.hp.osgi.quote</Require-Bundle>
            <Import-Package>
                *
            </Import-Package>
        </instructions>
    </configuration>
</plugin>
</plugins>
```

Perform maven clean and install, then copy the bundle in the temp folder to install in the container.

Uninstall the earlier service bundle.

```

karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version            | Name
--+
66 | Installed   | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.bundle Bundle
70 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quoteservice Bundle
73 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quote Bundle
75 | Resolved    | 80  | 1.1.0               | com.hp.osgi.quoteconsumer Bundle
karaf@root()> bundle:uninstall 70
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version            | Name
--+
66 | Installed   | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.bundle Bundle
73 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quote Bundle
75 | Resolved    | 80  | 1.1.0               | com.hp.osgi.quoteconsumer Bundle
karaf@root()> bundle:install file:///D:/temp/com.hp.osgi.quoteservice-0.0.1-SNAPSHOT.jar
Bundle ID: 76
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version            | Name
--+
66 | Installed   | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.bundle Bundle
73 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quote Bundle
75 | Resolved    | 80  | 1.1.0               | com.hp.osgi.quoteconsumer Bundle
76 | Installed   | 80  | 1.0.0               | com.hp.osgi.quoteservice Bundle
karaf@root()> bundle:start

```

#bundle:uninstall 70

Then install the new bundle and start it:

#bundle:install <file:///D:/temp/com.hp.osgi.quoteservice-0.0.1-SNAPSHOT.jar>

#bundle:start 76

Now, you can start the consumer bundle.

```
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version            | Name
---+-----+-----+-----+-----+
66 | Installed | 80 | 1.0.0.201707081832 | Firstbundle
67 | Active    | 80 | 0.0.1.SNAPSHOT     | com.hp.osgi.bundle Bundle
73 | Active    | 80 | 0.0.1.SNAPSHOT     | com.hp.osgi.quote Bundle
75 | Resolved   | 80 | 1.1.0               | com.hp.osgi.quoteconsumer Bundle
76 | Installed  | 80 | 1.0.0               | com.hp.osgi.quoteservice Bundle
karaf@root()> bundle:start
bundle:start          bundle:start-level
karaf@root()> bundle:start 76
IQuoteService is registered
karaf@root()> bundle:start 75
I feel better already
karaf@root()>
```

Great if you can see the statement printed as above.

Before ending this lab, if you want to find out what are the packages export by a bundle and imported by a bundle in case there is a dependency problem. You can use the following command.

```
#package:exports | grep osgi.quote
#package:imports | grep osgi.quote
```

```
karaf@root()>
karaf@root()> package:exports | grep osgi.quote
com.hp.osgi.quote | 1.0.0 | 73 | com.hp.osgi.quote
com.hp.osgi.quoteinternal | 1.0.0 | 76 | com.hp.osgi.quoteservice
com.hp.osgi.quoteservice | 1.0.0 | 76 | com.hp.osgi.quoteservice
karaf@root()> package:imports | grep osgi.quote
com.hp.osgi.quote | [1.0.0,1.2.0) | 75 | com.hp.osgi.quoteconsumer
com.hp.osgi.quote | [1.0.0,2.0.0) | 76 | com.hp.osgi.quoteservice
org.osgi.framework | [1.8.0,2.0.0) | 75 | com.hp.osgi.quoteconsumer
org.osgi.framework | [1.8.0,2.0.0) | 76 | com.hp.osgi.quoteservice
karaf@root()>
```

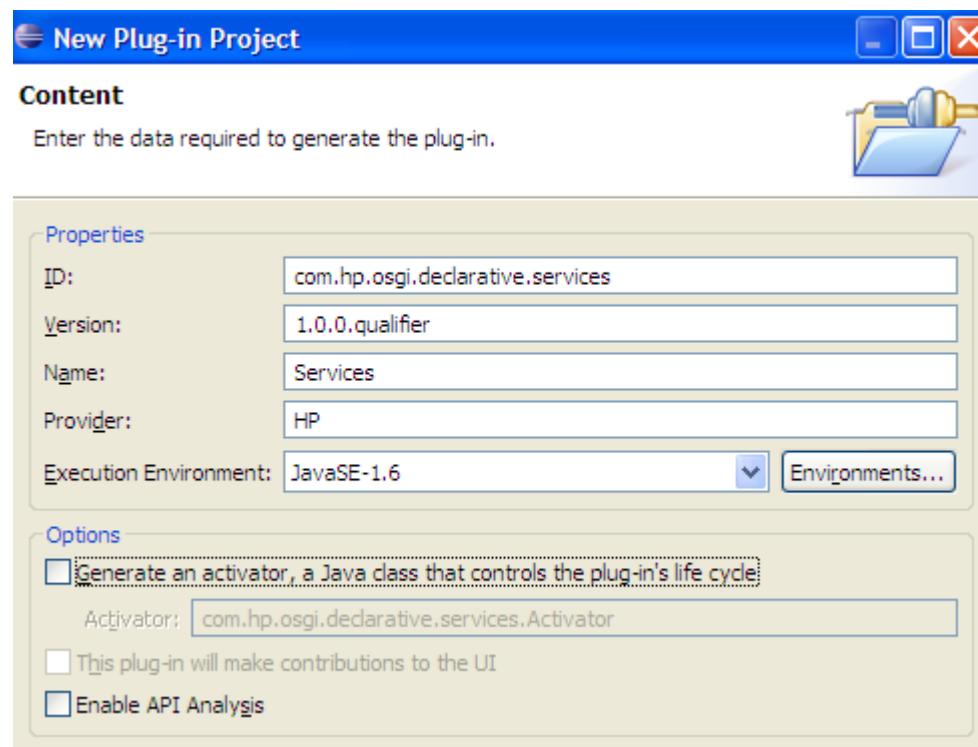
You can observe it clearly that, `com.hp.osgi.quote` exported by bundle 73 is of version 1.0.0 and the `com.hp.osgi.quote` imported by bundle 75 i.e consumer bundle is in the version range of 1.0.0. This way you can verify the package version information to determine any conflict in the bundle dependencies resolutions.

Declarative OSGi Service - Eclipse

Define a declarative OSGi Service

The following will define a DS service based on the quote example. It is therefore required that you have created the "de.vogella.osgi.quote" project which contains the interface definition.

Create a new plugin project "com.hp.osgi.declarative.services". Do not use a template, do not create an activator. Import package "de.vogella.osgi.quote" in MANIFEST.MF on the tab Dependencies.



Create the `OSGI-INF` folder in your project.

The implementing class is `de.vogella.osgi.ds.quoteservice.QuoteService` which provides the service for `IQuoteService`.

Create the class "QuoteService" which implements the interface `IQuoteService`.

```
package de.vogella.osgi.ds.quoteservice;

import java.util.Random;

import de.vogella.osgi.quote.IQuoteService;

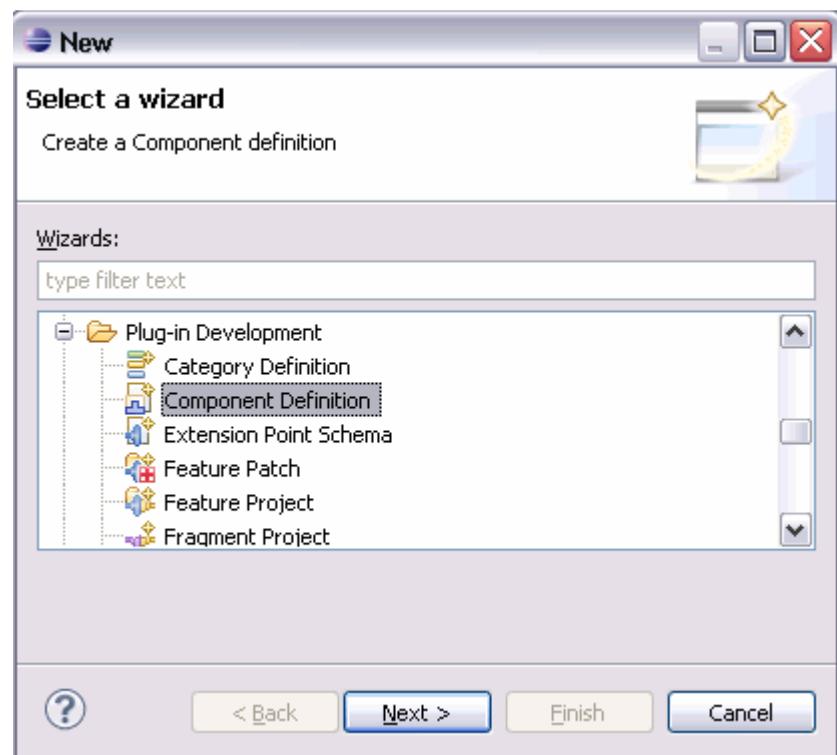
public class QuoteService implements IQuoteService {

    @Override
    public String getQuote() {
        Random random = new Random();
        // Create a number between 0 and 2
        int nextInt = random.nextInt(3);
        switch (nextInt) {
        case 0:
            return "Ds: Tell them I said something";
        case 1:
            return "Ds: I feel better already";
        default:
            return "Ds: Hubba Bubba, Baby!";
        }
    }
}
```

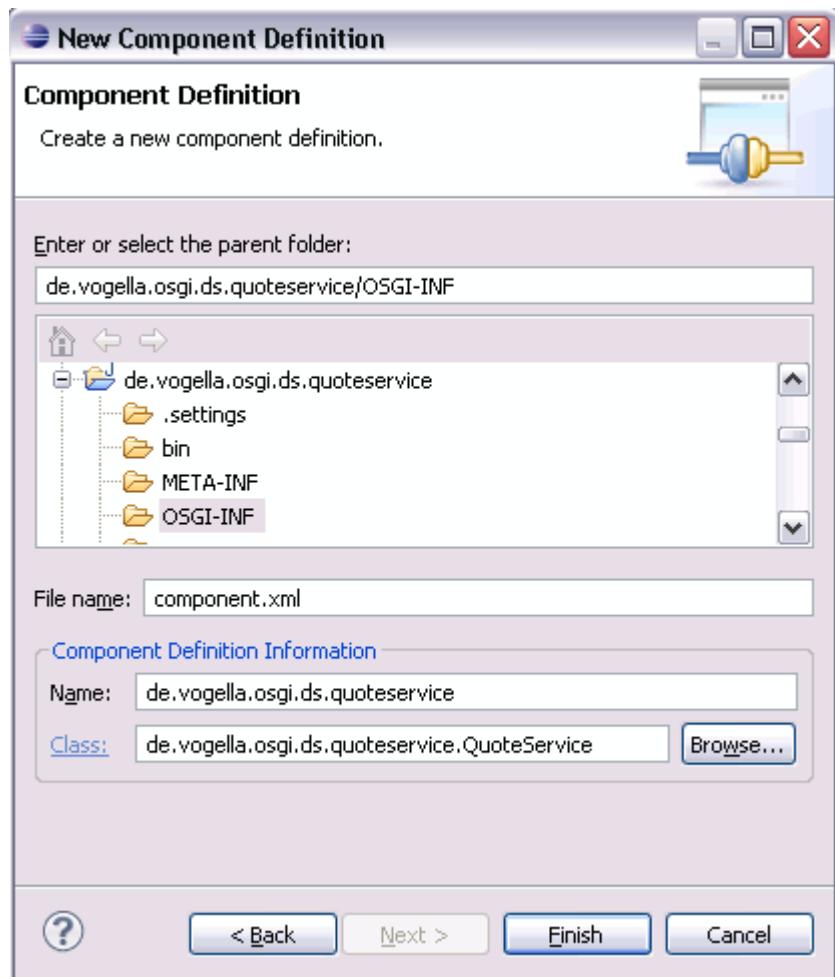
Create a new component definition as described below.

Create a new component definition

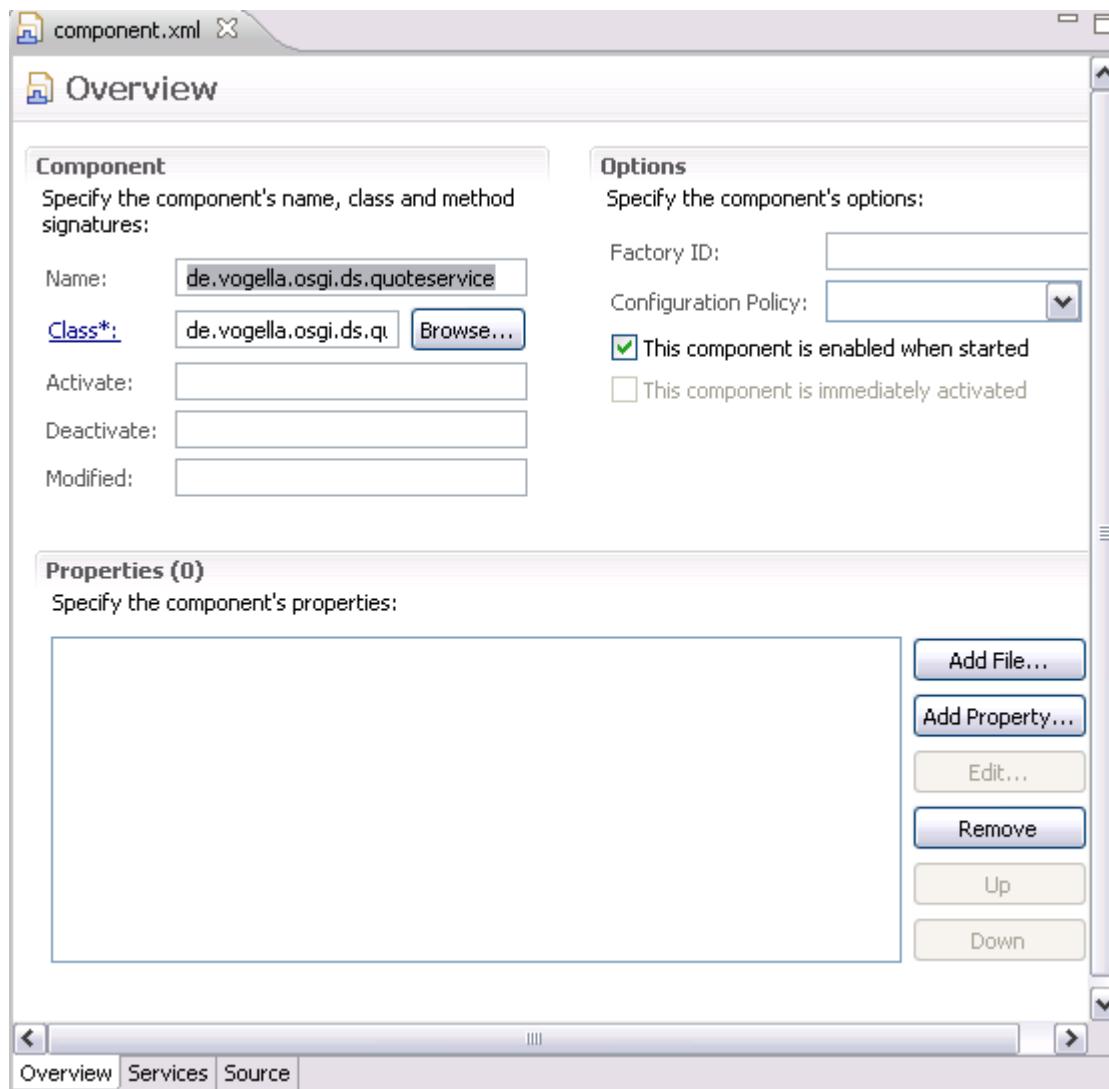
Typically component definitions are created in the project `OSGi-INF` folder via `New → Other → Plug-in Development → Component Definition`. The wizard will also add the `Service-Component` entry to the `MANIFEST.MF` file.



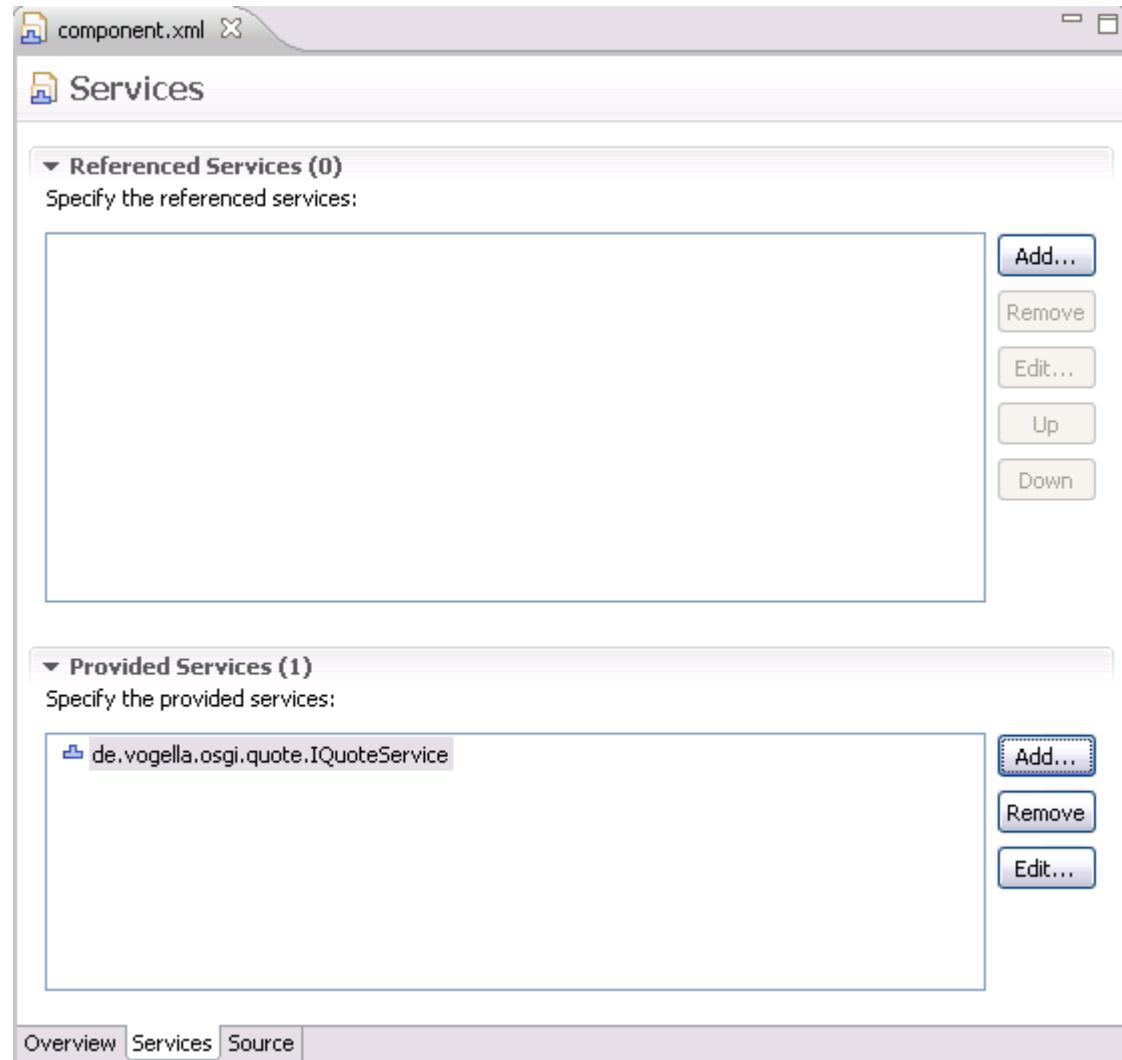
On the next page of the wizard, you can maintain the name of the component definition file, a name and the class which implements the services (or consumes the services).



If you press finish, the service editor opens.



On the Services tab you can maintain provided services or referred services. press the Add button under Provided Services and select the service interface you want to implement.



As a final step you would implement the class which would provide the service. (Code shown above)

Open component.xml and select the tab "Source". The final result should look like the following.

```
<?xml version="1.0" encoding="UTF-8"?>
<scr:component xmlns:scr="http://www.osgi.org/xmlns/scr/v1.1.0"
    name="de.vogella.osgi.ds.quoteservice">
    <implementation class="de.vogella.osgi.ds.quoteservice.QuoteService"/>
    <service>
        <provide interface="de.vogella.osgi.quote.IQuoteService"/>
    </service>
</scr:component>
```

Copy the "org.eclipse.equinox.ds*.jar", "org.eclipse.osgi.services.jar" and "org.eclipse.equinox.util*.jar" from your Eclipse/plugin installation directory into a folder, e.g. "D:\OSGI\DS" and install the bundle into your OSGi runtime via.

```
install file:D:\OSGI\DS\org.eclipse.equinox.ds_1.2.0.v20100507.jar  
install file:D:\OSGI\DS\org.eclipse.equinox.util_1.0.200.v20100503.jar  
install file:D:\OSGI\DS\org.eclipse.osgi.services_3.2.100.v20100503.jar  
  
install file:D:\OSGI\userBundles\de.vogella.osgi.quote_1.0.0.jar
```

Start the bundles manually so that declarative services are available.

Export your own bundle to " com.hp.osgi.declarative.services_1.0.0.201208311043.jar". and install it via:

```
install file:D:\OSGI\userBundles\com.hp.osgi.declarative.services_1.0.0.201208311043.jar
```

To check if your service was registered use the command "services". This will list all installed and available services.

If you stop / uninstall the old service provider and start the new one your service should be picked up by the consumer.

Using services via declarative services

Create a new plug-in "de.vogella.osgi.ds.quoteconsumer". Do not use a template, do not create an activator. Import the package "de.vogella.osgi.quote" in MANIFEST.MF on the *Dependencies* tab.

Create the following class.

```
package de.vogella.osgi.ds.quoteconsumer;

import de.vogella.osgi.quote.IQuoteService;

public class QuoteConsumer {
    private IQuoteService service;

    public void quote() {
        System.out.println(service.getQuote());
    }

    // Method will be used by DS to set the quote service
    public synchronized void setQuote(IQuoteService service) {
        System.out.println("Service was set. Thank you DS!");
        this.service = service;
        System.out.println(service.getQuote());
    }

    // Method will be used by DS to unset the quote service
    public synchronized void unsetQuote(IQuoteService service) {
        System.out.println("Service was unset. Why did you do this to me?");
        if (this.service == service) {
            this.service = null;
        }
    }
}
```

Create the *OSGI-INF* folder and create a new *Component Definition* in this folder.

The screenshot shows the 'Overview' tab of the Eclipse OSGI Component Definition editor. The 'Component' section contains fields for Name (de.vogella.osgi.ds.quoteconsumer), Class* (de.vogella.osgi.ds.quoteconsumer.QuoteConsumer), and various activation/deactivation status fields. The 'Options' section includes fields for Factory ID and Configuration Policy, and two checkboxes: 'This component is enabled when started' (checked) and 'This component is immediately activated'. The 'Properties (0)' section is empty. On the right, there is a toolbar with buttons for Add File..., Add Property..., Edit..., Remove, Up, and Down. At the bottom, tabs for Overview, Services, and Source are visible, with Overview selected.

Component
Specify the component's name, class and method signatures:

Name: de.vogella.osgi.ds.quoteconsumer

Class*: de.vogella.osgi.ds.quoteconsumer.QuoteConsumer | [Browse...](#)

Activate:

Deactivate:

Modified:

Options
Specify the component's options:

Factory ID:

Configuration Policy:

This component is enabled when started

This component is immediately activated

Properties (0)
Specify the component's properties:

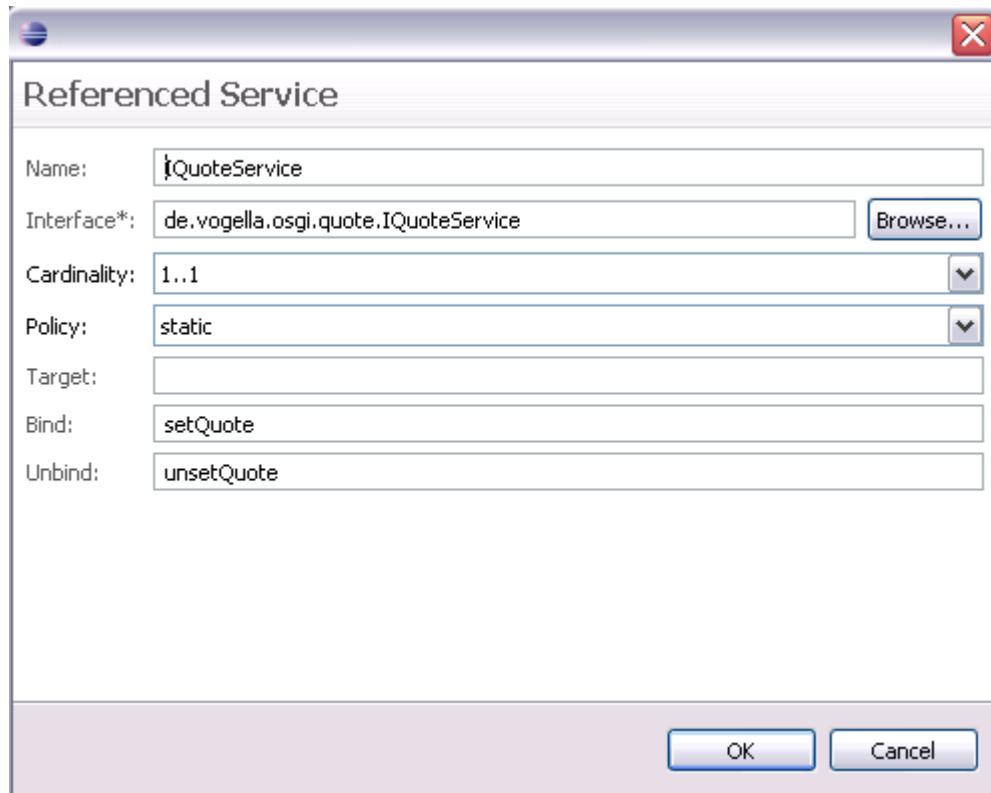
Add File...
Add Property...
Edit...
Remove
Up
Down

Overview Services Source

This time we will use a service. Maintain the "Referenced Services".

The screenshot shows a software interface titled "Services". Under the "Referenced Services" section, there is one entry: "IQuoteService [setQuote,unsetQuote]". To the right of this entry are five buttons: "Add...", "Remove", "Edit...", "Up", and "Down".

Make the relationship to the `bind()` and `unbind()` method by selecting your entry can by pressing the *Edit* button.



The result component.xml should look like:

```
<?xml version="1.0" encoding="UTF-8"?>
<scr:component xmlns:scr="http://www.osgi.org/xmlns/scr/v1.1.0" name="de.vogella.osgi.ds.quoteconsumer">
    <implementation class="de.vogella.osgi.ds.quoteconsumer.QuoteConsumer"/>
    <reference bind="setQuote" cardinality="1..1" interface="de.vogella.osgi.quote.IQuoteService"
name="IQuoteService" policy="static" unbind="unsetQuote"/>
</scr:component>
```

The result MANIFEST.MF should look like:

```
Manifest-Version: 1.0
Bundle-ManifestVersion: 2
Bundle-Name: Quoteconsumer
Bundle-SymbolicName: de.vogella.osgi.ds.quoteconsumer
Bundle-Version: 1.0.0.qualifier
Bundle-RequiredExecutionEnvironment: JavaSE-1.6
Import-Package: de.vogella.osgi.quote;version="1.0.0"
Service-Component: Osgi-INF/component.xml
```

Export your plug-in and install it via:

install file:D:\OSGI\userBundles\de.vogella.osgi.ds.quoteconsumer_1.0.0.201208311056.jar

"If you start the bundle now with "start id_of_your_bundle" you should get the feedback that the service was set and one quote should be returned

```
osgi> ss
Framework is launched.

id      State    Bundle
0      ACTIVE   org.eclipse.osgi_3.5.1.R35x_v20090827
5      RESOLVED de.vogella.osgi.firstbundle_1.0.0.200912211101
7      ACTIVE   de.vogella.osgi.quote_1.0.0
9      RESOLVED de.vogella.osgi.quoteservice_1.0.0
10     ACTIVE   de.vogella.osgi.quoteconsumer_1.0.0
22     ACTIVE   org.eclipse.equinox.ds_1.1.1.R35x_v20090806
23     RESOLVED de.vogella.osgi.ds.quoteservice_1.0.0.200912221341
26     ACTIVE   org.eclipse.equinox.util_1.0.100.v20090520-1800
27     RESOLVED org.eclipse.osgi.services_3.2.0.v20090520-1800

osgi> start 23
osgi> Ds: Tell them I said something
Ds: I feel better already
```

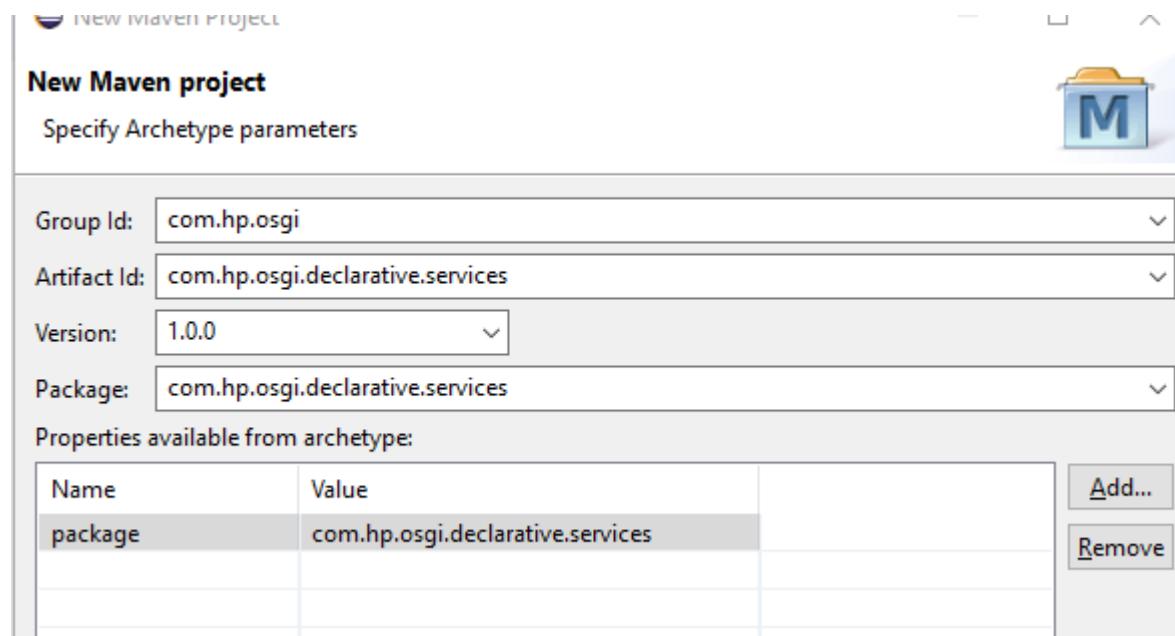
To run declarative services you require the following bundles.

- org.eclipse.equinox.util
- org.eclipse.equinox.ds - Declarative service

Declarative OSGi Service – Karaf Archetype

Define a declarative OSGi Service

The following will define a DS service based on the quote example. It is therefore required that you have created the "com.hp.osgi.quote" project which contains the interface definition before going ahead with this lab. Create a new maven karaf project "com.hp.osgi.declarative.services". Do not use a template, do not create an activator.



Remove the Activator class.

We are going to use interface declare in the Quote bundle, Import package "com.hp.osgi.quote" that need to be specify in pom.xml.

```

<configuration>
    <instructions>
        <Bundle-SymbolicName>${project.artifactId}</Bundle-SymbolicName>
        <Bundle-Version>${project.version}</Bundle-Version>
        <Export-Package>
            com.hp.osgi.declarative.services*;version=${project.version}
        </Export-Package>
        <Import-Package>
            com.hp.osgi.quote,* // This line is highlighted in yellow
        </Import-Package>
    </instructions>

```

Add the dependency of the Quote bundle in this project.

```

<dependency>
    <groupId>com.hp.osgi</groupId>
    <artifactId>com.hp.osgi.quote</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <scope>provided</scope>

</dependency>

```

We will add service component annotation dependency also.

```

<!--
    Contains the SCR Annotations
-->
<dependency>
    <groupId>biz.aQute</groupId>
    <artifactId>bndlib</artifactId>
    <version>2.0.0.20130123-133441</version>

</dependency>

```

The implementing class is com.hp.osgi.ds.quoteservice.QuoteService which provides the service for IQuoteService.

Create the class "QuoteService" which implements the interface IQuoteService,

```
package com.hp.osgi.declarative.services;

import java.util.Random;

import com.hp.osgi.quote.IQuoteService;

import aQute.bnd.annotation.component.Component;

@Component
public class QuoteService implements IQuoteService {

    public String getQuote() {
        Random random = new Random();
        // Create a number between 0 and 2
        int nextInt = random.nextInt(3);
        switch (nextInt) {
        case 0:
            return "Ds: Tell them I said something";
        case 1:
            return "Ds: I feel better already";
        default:
            return "Ds: Hubba Bubba, Quote!";
        }
    }
}
```

Update the pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

  <!-- Licensed to the Apache Software Foundation (ASF) under one or more
      contributor license agreements. See the NOTICE file distributed with this
      work for additional information regarding copyright ownership. The ASF licenses
      this file to you under the Apache License, Version 2.0 (the "License"); you
      may not use this file except in compliance with the License. You may obtain
      a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless
      required by applicable law or agreed to in writing, software distributed
      under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES
      OR CONDITIONS OF ANY KIND, either express or implied. See the License for
      the specific language governing permissions and limitations under the License. -->

  <modelVersion>4.0.0</modelVersion>

  <groupId>com.hp.osgi</groupId>
  <artifactId>com.hp.osgi.declarative.services</artifactId>
  <version>1.0.0</version>
  <packaging>bundle</packaging>

  <name>com.hp.osgi.declarative.services Bundle</name>
  <description>
    com.hp.osgi.declarative.services OSGi bundle project.
  </description>

  <properties>
    <maven-bundle-plugin.version>2.5.4</maven-bundle-plugin.version>
    <osgi.version>6.0.0</osgi.version>
  </properties>

  <dependencies>
    <dependency>
      <groupId>org.osgi</groupId>
      <artifactId>org.osgi.core</artifactId>
      <version>${osgi.version}</version>
```

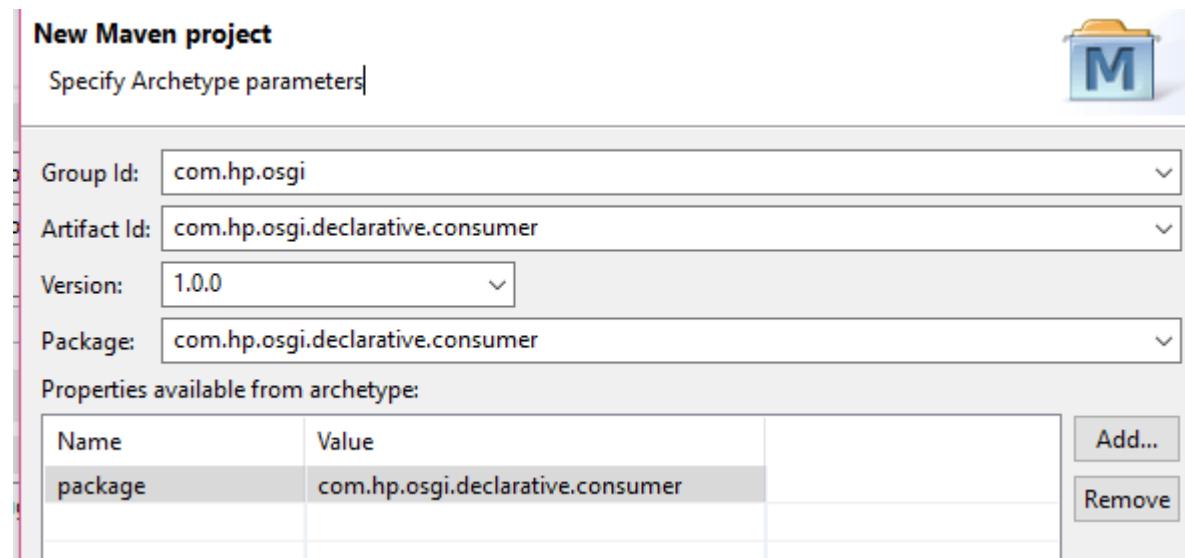
```
<scope>provided</scope>
</dependency>
<dependency>
    <groupId>com.hp.osgi</groupId>
    <artifactId>com.hp.osgi.quote</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <scope>provided</scope>
</dependency>
<!--
    Contains the SCR Annotations
-->
<dependency>
    <groupId>biz.aQute</groupId>
    <artifactId>bndlib</artifactId>
    <version>2.0.0.20130123-133441</version>
</dependency>
</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.felix</groupId>
            <artifactId>maven-bundle-plugin</artifactId>
            <version>${maven-bundle-plugin.version}</version>
            <extensions>true</extensions>
            <configuration>
                <instructions>
                    <Bundle-SymbolicName>${project.artifactId}</Bundle-SymbolicName>
                    <Bundle-Version>${project.version}</Bundle-Version>
                    <Service-Component>*</Service-Component>
                    <Export-Package>
                        com.hp.osgi.declarative.services*;version=${project.version}
                    </Export-Package>
                    <Import-Package>
                        com.hp.osgi.quote,*
                    </Import-Package>
                </instructions>
            </configuration>
        </plugin>
    </plugins>
</build>
```

```
        </plugins>
    </build>
</project>
```

Create a new component definition as described below which will be our consumer bundle

Create a new component definition using maven karaf Plugin.



Finish.

As a final step you would implement the class which would provide the service. (Code shown below)

Update the pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

  <!-- Licensed to the Apache Software Foundation (ASF) under one or more
      contributor license agreements. See the NOTICE file distributed with this
      work for additional information regarding copyright ownership. The ASF licenses
      this file to you under the Apache License, Version 2.0 (the "License"); you
      may not use this file except in compliance with the License. You may obtain
      a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless
      required by applicable law or agreed to in writing, software distributed
      under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES
      OR CONDITIONS OF ANY KIND, either express or implied. See the License for
      the specific language governing permissions and limitations under the License. -->

  <modelVersion>4.0.0</modelVersion>

  <groupId>com.hp.osgi</groupId>
  <artifactId>com.hp.osgi.declarative.consumer</artifactId>
  <version>1.0.0</version>
  <packaging>bundle</packaging>

  <name>com.hp.osgi.declarative.consumer Bundle</name>
  <description>
    com.hp.osgi.declarative.consumer OSGi bundle project.
  </description>

  <properties>
    <maven-bundle-plugin.version>2.5.4</maven-bundle-plugin.version>
    <osgi.version>6.0.0</osgi.version>
  </properties>

  <dependencies>
    <dependency>
      <groupId>org.osgi</groupId>
      <artifactId>org.osgi.core</artifactId>
      <version>${osgi.version}</version>
```

```
<scope>provided</scope>
</dependency>
<!-- Contains the SCR Annotations -->
<dependency>
    <groupId>biz.aQute</groupId>
    <artifactId>bndlib</artifactId>
    <version>2.0.0.20130123-133441</version>
</dependency>
<!-- https://mvnrepository.com/artifact/org.slf4j/slf4j-log4j12 -->
<dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>slf4j-api</artifactId>
    <version>1.7.25</version>
</dependency>
<dependency>
    <groupId>com.hp.osgi</groupId>
    <artifactId>com.hp.osgi.quote</artifactId>
    <version>0.0.1-SNAPSHOT</version>
</dependency>

</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.felix</groupId>
            <artifactId>maven-bundle-plugin</artifactId>
            <version>${maven-bundle-plugin.version}</version>
            <extensions>true</extensions>
            <configuration>
                <instructions>
                    <Bundle-SymbolicName>${project.artifactId}</Bundle-SymbolicName>
                    <Bundle-Version>${project.version}</Bundle-Version>
                    <Service-Component>*</Service-Component>
                    <Export-Package>
                        com.hp.osgi.declarative.consumer*;version=${project.version}
                    </Export-Package>
                    <Import-Package>
                        *
                    </Import-Package>
                </instructions>
            </configuration>
        </plugin>
    </plugins>
</build>
```

```

        </Import-Package>
    </instructions>
</configuration>
</plugin>
</plugins>
</build>

</project>
```

Create a component consumer class that will consume the Quote Service:

```

package com.hp.osgi.declarative.consumer;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import com.hp.osgi.quote.IQuoteService;

import aQute.bnd.annotation.component.Activate;
import aQute.bnd.annotation.component.Component;
import aQute.bnd.annotation.component.Deactivate;
import aQute.bnd.annotation.component.Reference;

@Component(name = ConsumerComponent.COMPONENT_NAME)
public class ConsumerComponent {

    public static final String COMPONENT_NAME = "ConsumerComponent";

    public static final String COMPONENT_LABEL = "Consumer Component";

    private static final Logger LOG = LoggerFactory.getLogger(ConsumerComponent.class);

    private IQuoteService quoteService;

    /**
     * Called when all of the SCR Components required dependencies have been
     * satisfied.
```

```

        */
    @Activate
    public void activate() {
        LOG.info("Activating the " + COMPONENT_LABEL);
        System.out.println(quoteService.getQuote());
        LOG.info("Activating the Quote Value " + quoteService.getQuote());
    }

    /**
     * Called when any of the SCR Components required dependencies become
     * unsatisfied.
     */
    @Deactivate
    public void deactivate() {
        LOG.info("Deactivating the " + COMPONENT_LABEL);
    }

    @Reference
    public void setQuoteService(final IQuoteService quoteService) {
        this.quoteService = quoteService;
    }

    public void unsetQuoteService(final IQuoteService quoteService) {
        this.quoteService = null;
    }
}

```

The `ConsumerComponent.java` class is our service consumer class. It is annotated with the `@Component` annotation just like `QuoteService` but considered an **Immediate Component**. BND will configure `ConsumerComponent.java` as an Immediate Component because it doesn't implement an interface. Therefore once all of its required dependencies are available this component will be activated.

Again, this can be overridden if you need to by adding the `immediate` attribute set to false on the `@Component` annotation.

The `ConsumerComponent` also introduces 3 more annotations: `@Activate`, `@Deactivate` & `@Reference`.

The `@Reference` annotation is used to configure a dependency requirement for a component, in this case the `QuoteService`. As such the defaults will cause the `ConsumerComponent` to remain deactivated if the `QuoteService` is not available.

When the QuoteService dependency becomes satisfied, the SCR service will call the configured activation method denoted by the **@Activate** annotation. Conversely if the dependency becomes unsatisfied post activation the SCR service will call the components configured deactivate method denoted by the **@Deactivate** annotation.

You need to enable the SCR for declarative service in Apache Karaf before installing the bundles.

Once it is up and running we can add the SCR Feature:

```
#feature:install scr
```

After installing the SCR feature you can verify everything installed correctly by typing scr: in the shell and then hitting the TAB key. This will display the new commands that were installed:

```
karaf@root()> scr:
scr:activate      scr:components    scr:config        scr:deactivate    scr:details      scr:disable     scr:enable      scr:info
scr:list
karaf@root()> scr:
```

Let us install the bundles into our OSGi runtime via our usual methods; Hints – You can use bundle:install or dump in the **deploy** folder.

```
com.hp.osgi.declarative.consumer
com.hp.osgi.declarative.services
```

| This PC > New Volume (D:) > MyExperiment > apache-karaf-4.0.9 > deploy | | | | |
|------------------------------------------------------------------------|--------------------------------------------|--------------------|---------------------|------|
| | Name | Date modified | Type | Size |
| | com.hp.osgi.bundle-0.0.1-SNAPSHOT.jar | 7/8/2017 7:53 PM | Executable Jar File | 4 KB |
| | com.hp.osgi.declarative.consumer-1.0.0.jar | 7/10/2017 11:27 PM | Executable Jar File | 5 KB |
| | com.hp.osgi.declarative.services-1.0.0.jar | 7/10/2017 11:27 PM | Executable Jar File | 5 KB |
| | README | 4/6/2017 9:53 AM | File | 1 KB |
| | | | | |

Start the bundles manually so that declarative services are available. At the end of this step, you should have the following three bundles get installed:

```
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version           | Name
--+
66 | Installed  | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle
73 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quote Bundle
75 | Active     | 80  | 1.1.0              | com.hp.osgi.quoteconsumer Bundle
76 | Active     | 80  | 1.0.0              | com.hp.osgi.quoteservice Bundle
81 | Active     | 80  | 1.0.0              | com.hp.osgi.declarative.services Bundle
83 | Active     | 80  | 1.0.0              | com.hp.osgi.declarative.consumer Bundle
karaf@root()>
```

Now if we execute the scr:list command we should see two components listed:

```
#scr:list
```

```
karaf@root()> scr:list
BundleId Component Name Default State
  Component Id State      PIDs (Factory PID)
[ 80] ScrServiceMBean enabled
  [ 0] [active      ]
[ 81] com.hp.osgi.declarative.services.QuoteService enabled
  [ 1] [satisfied   ]
[ 82] ConsumerComponent enabled
  [ 2] [active      ]
[ 83] com.hp.osgi.declarative.consumer.Ds: I feel better already
```

When you deploy for the first time the declarative consumer will display the following:

```
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version           | Name
---+-----+-----+-----+-----+
66 | Installed  | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle
73 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quote Bundle
75 | Active     | 80  | 1.1.0              | com.hp.osgi.quoteconsumer Bundle
76 | Resolved   | 80  | 1.0.0              | com.hp.osgi.quoteservice Bundle
83 | Active     | 80  | 1.0.0              | com.hp.osgi.declarative.consumer Bundle
karaf@root()> Ds: I feel better already
```

You can also verify the karaf log file apache-karaf-4.0.9\data\log\

```
| 4 - org.apache.felix.fileinstall - 3.5.8 | Installing bundle com.hp.osgi.declarative.services / 1.0.0
| 83 - com.hp.osgi.declarative.consumer - 1.0.0 | Activating the Consumer Component
| 83 - com.hp.osgi.declarative.consumer - 1.0.0 | Activating the Quote Value Ds: I feel better already
| 4 - org.apache.felix.fileinstall - 3.5.8 | Started bundle: file:/D:/MyExperiment/apache-karaf-4.0.9/deploy/com.hp.osgi.declarative.services-1.0.0.jar
```

So, why this is printed and at what conditions?

Hints[

```
@Activate : Method

/**
 * Called when all of the SCR Components required dependencies have been
 * satisfied.
 */

]
```

This Quote message is served by Declarative Service, we can make it out as the message is prepended with DS:

```
83 | Active    | 80 | 1.0.0          | com.hp.o
karaf@root()> Ds: I feel better already
```

The **scr:details** command will list the current configuration and state of a given component. Taking a look at the **ConsumerComponent** we see it is currently active and that it has a satisfied/Active reference to the **ConsumerService**: In this state it should be in Active

scr:details ConsumerComponent

```
karaf@root()> scr:list
BundleId Component Name Default State
  Component Id State      PIDs (Factory PID)
[ 80] ScrServiceMBean enabled
  [ 0] [active      ]
[ 83] ConsumerComponent enabled
  [ 2] [active      ]
[ 84] com.hp.osgi.declarative.services.QuoteService enabled
  [ 4] [active      ]
karaf@root()>
```

Let us stop the Declarative Service now, determine the bundle ID using `bundle:list` to stop it. When we stop the declarative service and restart the normal Quote Service, it should print out message from the Earlier Quote service.

```
#bundle:stop 84 [Stopping the Declarative Service]
#bundle:list [Verify the bundle status]
#bundle:restart 76 [restart the Normal Quote service ]
```

```
karaf@root()> scr:list
BundleId Component Name Default State
  Component Id State      PIDs (Factory PID)
[ 80] ScrServiceMBean enabled
    [ 0] [active      ]
[ 83] ConsumerComponent enabled
    [ 2] [active      ]
[ 84] com.hp.osgi.declarative.services.QuoteService enabled
    [ 4] [active      ]
karaf@root()> bundle:stop 84
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version           | Name
-----
66 | Installed   | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.bundle Bundle
73 | Active      | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quote Bundle
75 | Active      | 80  | 1.1.0              | com.hp.osgi.quoteconsumer Bundle
76 | Resolved    | 80  | 1.0.0              | com.hp.osgi.quoteservice Bundle
83 | Active      | 80  | 1.0.0              | com.hp.osgi.declarative.consumer Bundle
84 | Resolved    | 80  | 1.0.0              | com.hp.osgi.declarative.services Bundle
karaf@root()> bundle:restart 76
Tell them I said something IQuoteService is registered
karaf@root()>
```

Verify the message it should not begin with **DS**:

```
| nsole user karaf | ConsumerComponent          | 83 - com.hp.osgi.declarative.consumer - 1.0.0 | Activating the Consumer Component  
| nsole user karaf | ConsumerComponent          | 83 - com.hp.osgi.declarative.consumer - 1.0.0 | Activating the Quote Value Tell them I said something
```

Now verify the consumer component state:

```
scr:deactivate    scr:details  
karaf@root()> scr:details  ConsumerComponent  
Component Details  
  Name           : ConsumerComponent  
  State          : ACTIVE  
References  
  Reference     : greeterService  
    State        : satisfied  
    Multiple      : single  
    Optional       : mandatory  
    Policy         : static  
    Service Reference : Bound Service ID 108  
karaf@root()>
```

If you stop / uninstall the old service provider and start the new one your service should be picked up by the consumer.

Service Tracker Using Eclipse PDE

Use the earlier project com.hp.osgi.osgi.quote

Create a new plug-in "com.hp.osgi.st.quoteconsumer".

Declare a package dependency to the package "org.osgi.util.tracker" and "org.eclipse.osgi.util", com.hp.osgi.quote in your bundle.

Define the following class "MyQuoteServiceTrackerCustomizer"

```
package com.hp.osgi.st.quoteconsumer;

import org.osgi.framework.BundleContext;
import org.osgi.framework.ServiceReference;
import org.osgi.util.tracker.*;

import de.vogella.osgi.quote.IQuoteService;

public class MyQuoteServiceTrackerCustomizer implements
    ServiceTrackerCustomizer {

    private final BundleContext context;

    public MyQuoteServiceTrackerCustomizer(BundleContext context) {
        this.context = context;
    }

    private MyThread thread;

    @Override
    public Object addingService(ServiceReference reference) {
        IQuoteService service = (IQuoteService) context.getService(reference);
        thread = new MyThread(service);
        thread.start();
    }
}
```

```
        return service;
    }

@Override
public void modifiedService(ServiceReference reference, Object service) {
    // removedService(reference, service);
    // addingService(reference);
}

@Override
public void removedService(ServiceReference reference, Object service) {
    context.ungetService(reference);
    System.out.println("How sad. Service for quote is gone");
    thread.stopThread();
}

public static class MyThread extends Thread {

    private volatile boolean active = true;
    private final IQuoteService service;

    public MyThread(IQuoteService service) {
        this.service = service;
    }

    public void run() {
        while (active) {
            System.out.println(service.getQuote());
            try {
                Thread.sleep(5000);
            } catch (Exception e) {
                System.out.println("Thread interrupted " + e.getMessage());
            }
        }
    }

    public void stopThread() {
        active = false;
    }
}
```

}

Register a service tracker in your activator of your service consumer.

```
package com.hp.osgi.st.quoteconsumer;

import org.osgi.framework.BundleActivator;
import org.osgi.framework.BundleContext;
import org.osgi.util.tracker.ServiceTracker;

import de.vogella.osgi.quote.IQuoteService;

public class Activator implements BundleActivator {

    private static BundleContext context;
    private ServiceTracker serviceTracker;
    static BundleContext getContext() {
        return context;
    }

    /*
     * (non-Javadoc)
     * @see org.osgi.framework.BundleActivator#start(org.osgi.framework.BundleContext)
     */
    public void start(BundleContext bundleContext) throws Exception {
        Activator.context = bundleContext;
        System.out.println("Starting quoteconsumer bundles");
        // Register directly with the service
        MyQuoteServiceTrackerCustomizer customer = new MyQuoteServiceTrackerCustomizer(context);
        serviceTracker = new ServiceTracker(context, IQuoteService.class
                .getName(), customer);
        serviceTracker.open();
    }

    /*
     * (non-Javadoc)
```

```
* @see org.osgi.framework.BundleActivator#stop(org.osgi.framework.BundleContext)
*/
public void stop(BundleContext bundleContext) throws Exception {
    Activator.context = null;
    System.out.println("Stopping quoteconsumer bundles");
    serviceTracker.close();
}

}
```

Manifest.MF

```
Manifest-Version: 1.0
Bundle-ManifestVersion: 2
Bundle-Name: Quotecconsumer
Bundle-SymbolicName: com.hp.osgi.st.quoteconsumer
Bundle-Version: 1.0.0.qualifier
Bundle-Activator: com.hp.osgi.st.quoteconsumer.Activator
Import-Package: com.hp.osgi.quote;version="1.0.0",
    org.eclipse.osgi.util;version="1.1.0",
    org.osgi.framework;version="1.3.0",
    org.osgi.util.tracker;version="1.4.0"
Bundle-RequiredExecutionEnvironment: JavaSE-1.6
```

Export your bundle again.

Start the OSGi console. All the bundle need to be active as shown below

```
C:\WINDOWS\system32\cmd.exe - java -jar org.eclipse.osgi-3.6.0.v20100517.jar -console
osgi>
osgi> install file:D:\OSGI\userBundles\de.vogella.osgi.ds.quoteconsumer_1.0.0.201208311056.jar
Bundle id is 23
osgi> ss
Framework is launched.
id      State      Bundle
0      ACTIVE      org.eclipse.osgi_3.6.0.v20100517
1      ACTIVE      org.eclipse.equinox.common_3.6.0.v20100503
2      ACTIVE      org.osgi.compendium_4.1.0
3      ACTIVE      com.hpe.osgi.declarative.services_1.0.0.201208311043
4      ACTIVE      org.eclipse.equinox.ds_1.2.0.v20100507
5      ACTIVE      org.eclipse.equinox.util_1.0.200.v20100503
6      ACTIVE      org.osgi.services_3.2.100.v20100503
7      ACTIVE      com.hpe.osgi.declarative.services_1.0.0.201208311043
8      ACTIVE      de.vogella.osgi.quote_1.0.0
23     INSTALLED   de.vogella.osgi.ds.quoteconsumer_1.0.0.201208311056

osgi> start 23
Service was set. Thank you DS!
Ds: Tell them I said something
osgi>
```

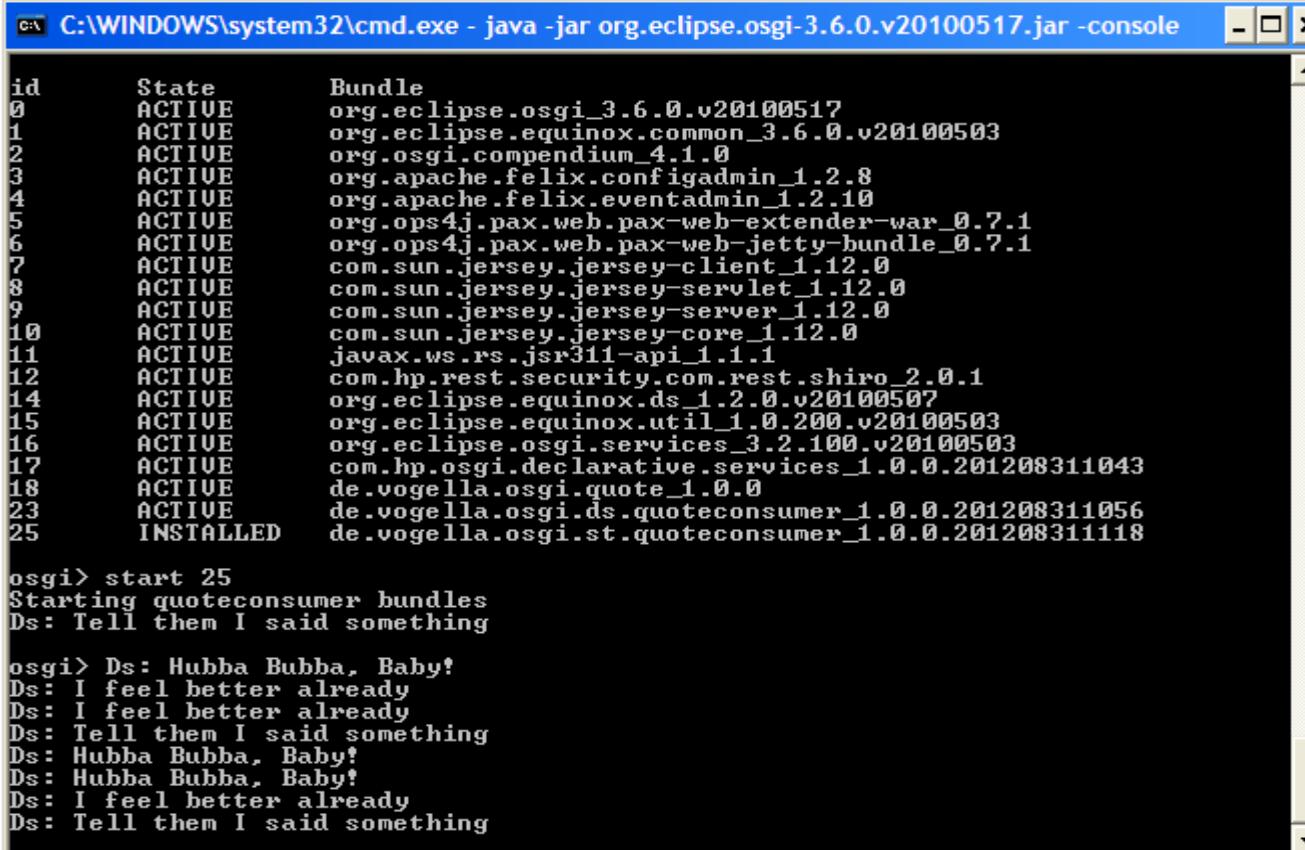
Install the bundle:

Install file:userBunldes/ com.hpe.osgi.st.quoteconsumer_1.0.0.201208311118.jar

Use the update command or the install command to get the new version of your bundle and start it.

You are using the earlier Quote services, so all the dependent bundles of Quote services as earlier tutorial need to be active.

Once you start your service the tracker will be called and the consumer bundle will start writing messages to the console. Stop the service (Quote) and verify that the consumer does not use the service anymore.(It won't display the message and start the service it will again display the message)



The screenshot shows a Windows command prompt window titled "cmd.exe - java -jar org.eclipse.osgi-3.6.0.v20100517.jar -console". The window displays the following content:

```
id      State    Bundle
0      ACTIVE   org.eclipse.osgi_3.6.0.v20100517
1      ACTIVE   org.eclipse.equinox.common_3.6.0.v20100503
2      ACTIVE   org.osgi.compendium_4.1.0
3      ACTIVE   org.apache.felix.configadmin_1.2.8
4      ACTIVE   org.apache.felix.eventadmin_1.2.10
5      ACTIVE   org.ops4j.pax.web.pax-web-extender-war_0.7.1
6      ACTIVE   org.ops4j.pax.web.pax-web-jetty-bundle_0.7.1
7      ACTIVE   com.sun.jersey.jersey-client_1.12.0
8      ACTIVE   com.sun.jersey.jersey-servlet_1.12.0
9      ACTIVE   com.sun.jersey.jersey-server_1.12.0
10     ACTIVE  com.sun.jersey.jersey-core_1.12.0
11     ACTIVE  javax.ws.rs.jsr311-api_1.1.1
12     ACTIVE  com.hpe.rest.security.com.rest.shiro_2.0.1
14     ACTIVE  org.eclipse.equinox.ds_1.2.0.v20100507
15     ACTIVE  org.eclipse.equinox.util_1.0.200.v20100503
16     ACTIVE  org.eclipse.osgi.services_3.2.100.v20100503
17     ACTIVE  com.hpe.osgi.declarative.services_1.0.0.201208311043
18     ACTIVE  de.vogella.osgi.quote_1.0.0
23     ACTIVE  de.vogella.osgi.ds.quoteconsumer_1.0.0.201208311056
25     INSTALLED de.vogella.osgi.st.quoteconsumer_1.0.0.201208311118

osgi> start 25
Starting quoteconsumer bundles
Ds: Tell them I said something

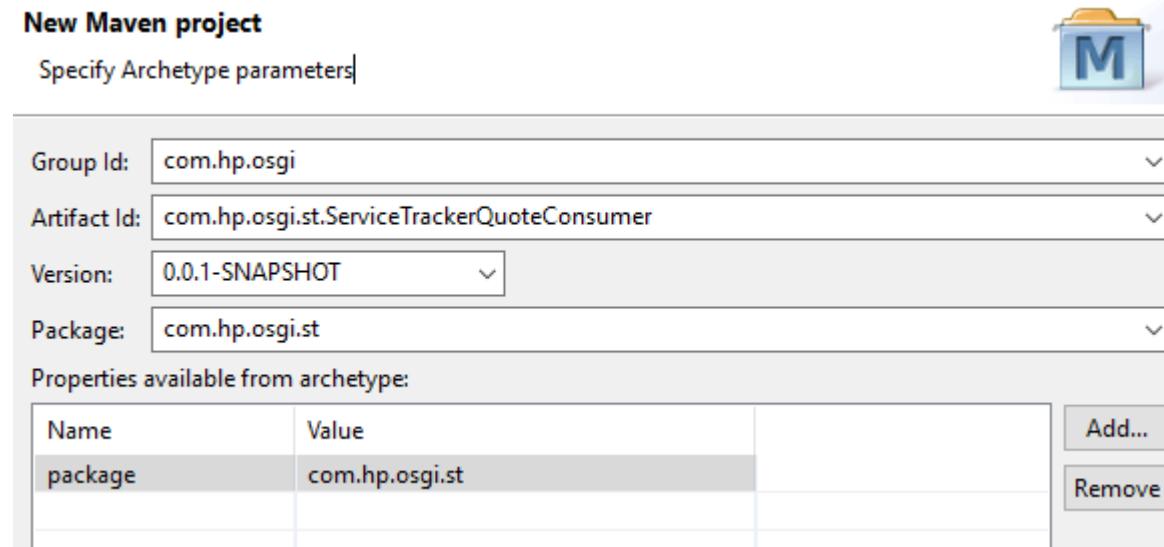
osgi> Ds: Hubba Bubba, Baby!
Ds: I feel better already
Ds: I feel better already
Ds: Tell them I said something
Ds: Hubba Bubba, Baby!
Ds: Hubba Bubba, Baby!
Ds: I feel better already
Ds: Tell them I said something
```

Service Tracker Using Maven Plugin Karaf

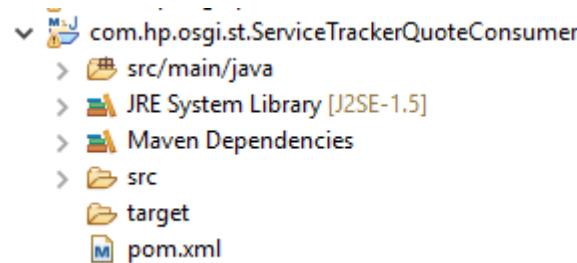
In this lab we are going to access the Service using Servicetracker. We will be using the earlier Quote service, ensure that you have completed the earlier lab before proceeding ahead.

Use the earlier project com.hp.osgi.quote

Create a new plug-in using apache karaf plugin archetype " com.hp.osgi.st.ServiceTrackerQuoteConsumer".



Click Finish



You should have the project structure as shown above.

Declare a package dependency to the package "org.osgi.util.tracker" and “org.eclipse.osgi.util”, com.hp.osgi.quote in your bundle.

```
<Bundle-Activator>com.hp.osgi.st.Activator</Bundle-Activator>
<Export-Package>
    com.hp.osgi.st*;version=${project.version}
</Export-Package>
<Import-Package>
    org.osgi.util.tracker,org.eclipse.osgi.util,com.hp.osgi.quote,*
```

Update the pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

  <!-- Licensed to the Apache Software Foundation (ASF) under one or more
      contributor license agreements. See the NOTICE file distributed with this
      work for additional information regarding copyright ownership. The ASF licenses
      this file to you under the Apache License, Version 2.0 (the "License"); you
      may not use this file except in compliance with the License. You may obtain
      a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless
      required by applicable law or agreed to in writing, software distributed
      under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES
      OR CONDITIONS OF ANY KIND, either express or implied. See the License for
      the specific language governing permissions and limitations under the License. -->

  <modelVersion>4.0.0</modelVersion>

  <groupId>com.hp.osgi</groupId>
  <artifactId>com.hp.osgi.st.ServiceTrackerQuoteConsumer</artifactId>
  <version>1.0.0</version>
  <packaging>bundle</packaging>

  <name>com.hp.osgi.st.ServiceTrackerQuoteConsumer Bundle</name>
  <description>
    com.hp.osgi.st.ServiceTrackerQuoteConsumer OSGi bundle project.
  </description>

  <properties>
    <maven-bundle-plugin.version>2.5.4</maven-bundle-plugin.version>
    <osgi.version>6.0.0</osgi.version>
  </properties>

  <dependencies>
    <dependency>
      <groupId>org.osgi</groupId>
      <artifactId>org.osgi.core</artifactId>
      <version>${osgi.version}</version>
```

```
<scope>provided</scope>
</dependency>
<dependency>
    <groupId>com.hp.osgi</groupId>
    <artifactId>com.hp.osgi.quote</artifactId>
    <version>0.0.1-SNAPSHOT</version>
</dependency>
</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.felix</groupId>
            <artifactId>maven-bundle-plugin</artifactId>
            <version>${maven-bundle-plugin.version}</version>
            <extensions>true</extensions>
            <configuration>
                <instructions>
                    <Bundle-SymbolicName>${project.artifactId}</Bundle-SymbolicName>
                    <Bundle-Version>${project.version}</Bundle-Version>
                    <Bundle-Activator>com.hp.osgi.st.Activator</Bundle-Activator>
                    <Export-Package>
                        com.hp.osgi.st*;version=${project.version}
                    </Export-Package>
                    <Import-Package>
                        org.osgi.util.tracker,org.eclipse.osgi.util,com.hp.osgi.quote,*
                    </Import-Package>
                </instructions>
            </configuration>
        </plugin>
    </plugins>
</build>

</project>
```

Define the following class "MyQuoteServiceTrackerCustomizer"

```
package com.hp.osgi.st;

import org.osgi.framework.BundleContext;
import org.osgi.framework.ServiceReference;
import org.osgi.util.tracker.ServiceTrackerCustomizer;

import com.hp.osgi.quote.IQuoteService;

public class MyQuoteServiceTrackerCustomizer implements
    ServiceTrackerCustomizer {

    private final BundleContext context;

    public MyQuoteServiceTrackerCustomizer(BundleContext context) {
        this.context = context;
    }

    private MyThread thread;

    public Object addingService(ServiceReference reference) {
        IQuoteService service = (IQuoteService) context.getService(reference);
        thread = new MyThread(service);
        thread.start();
        return service;
    }

    public void modifiedService(ServiceReference reference, Object service) {
        // removedService(reference, service);
        // addingService(reference);
    }

    public void removedService(ServiceReference reference, Object service) {
        context.ungetService(reference);
        System.out.println("How sad. Service for quote is gone");
        thread.stopThread();
    }
}
```

```
}
```

```
public static class MyThread extends Thread {
```

```
    private volatile boolean active = true;
```

```
    private final IQuoteService service;
```

```
    public MyThread(IQuoteService service) {
```

```
        this.service = service;
```

```
    }
```

```
    public void run() {
```

```
        while (active) {
```

```
            System.out.println(service.getQuote());
```

```
            try {
```

```
                Thread.sleep(5000);
```

```
            } catch (Exception e) {
```

```
                System.out.println("Thread interrupted " + e.getMessage());
```

```
            }
```

```
        }
```

```
    }
```

```
    public void stopThread() {
```

```
        active = false;
```

```
    }
```

```
}
```

Register a service tracker in your activator so that it can consume the QuoteService using Service Tracker

```
package com.hp.osgi.st;

import org.osgi.framework.BundleActivator;
import org.osgi.framework.BundleContext;
import org.osgi.util.tracker.ServiceTracker;

import com.hp.osgi.quote.IQuoteService;

public class Activator implements BundleActivator {

    private static BundleContext context;
    private ServiceTracker serviceTracker;
    static BundleContext getContext() {
        return context;
    }

    /*
     * (non-Javadoc)
     * @see org.osgi.framework.BundleActivator#start(org.osgi.framework.BundleContext)
     */
    public void start(BundleContext bundleContext) throws Exception {
        Activator.context = bundleContext;
        System.out.println("Starting quoteconsumer bundles");
        // Register directly with the service
        MyQuoteServiceTrackerCustomizer customer = new MyQuoteServiceTrackerCustomizer(context);
        serviceTracker = new ServiceTracker(context, IQuoteService.class
            .getName(), customer);
        serviceTracker.open();
    }
}
```

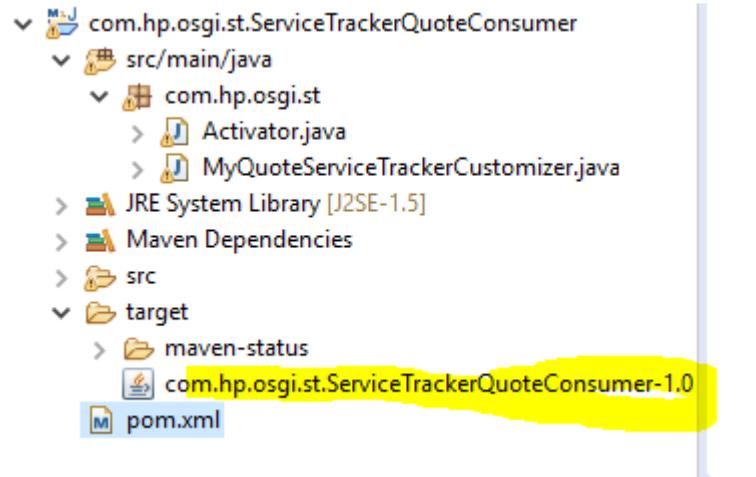
```
/*
 * (non-Javadoc)
 * @see org.osgi.framework.BundleActivator#stop(org.osgi.framework.BundleContext)
 */
public void stop(BundleContext bundleContext) throws Exception {
    Activator.context = null;
    System.out.println("Stopping quoteconsumer bundles");
    serviceTracker.close();
}

}
```

Manifest.MF

```
Manifest-Version: 1.0
Bundle-ManifestVersion: 2
Bundle-Name: Quoteconsumer
Bundle-SymbolicName: de.vogella.osgi.st.quoteconsumer
Bundle-Version: 1.0.0.qualifier
Bundle-Activator: com.hp.osgi.st.quoteconsumer.Activator
Import-Package: com.hp.osgi.quote;version="1.0.0",
    org.eclipse.osgi.util;version="1.1.0",
    org.osgi.framework;version="1.3.0",
    org.osgi.util.tracker;version="1.4.0"
Bundle-RequiredExecutionEnvironment: JavaSE-1.6
```

Export your bundle again.



Start the OSGi apache karaf console. All the bundle need to be active as shown below:

```
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version           | Name
--+
66 | Installed  | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle
73 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quote Bundle
75 | Active     | 80  | 1.1.0              | com.hp.osgi.quoteconsumer Bundle
76 | Resolved   | 80  | 1.0.0              | com.hp.osgi.quoteservice Bundle
83 | Active     | 80  | 1.0.0              | com.hp.osgi.declarative.consumer Bundle
84 | Active     | 80  | 1.0.0              | com.hp.osgi.declarative.services Bundle
karaf@root()>
```

Install our bundle:

```
#bundle:install file:///d://temp/com.hp.osgi.st.ServiceTrackerQuoteConsumer-1.0.0.jar
```

Verify that the bundle is installed

```
#bundle:list
```

```
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version           | Name
---+-----+-----+-----+-----+
66 | Installed  | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle
73 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quote Bundle
75 | Active     | 80  | 1.1.0              | com.hp.osgi.quoteconsumer Bundle
76 | Resolved   | 80  | 1.0.0              | com.hp.osgi.quoteservice Bundle
83 | Active     | 80  | 1.0.0              | com.hp.osgi.declarative.consumer Bundle
84 | Active     | 80  | 1.0.0              | com.hp.osgi.declarative.services Bundle
karaf@root()> bundle:install file:///d://temp/com.hp.osgi.st.ServiceTrackerQuoteConsumer-1.0.0.jar
Bundle ID: 86
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version           | Name
---+-----+-----+-----+-----+
66 | Installed  | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle
73 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quote Bundle
75 | Active     | 80  | 1.1.0              | com.hp.osgi.quoteconsumer Bundle
76 | Resolved   | 80  | 1.0.0              | com.hp.osgi.quoteservice Bundle
83 | Active     | 80  | 1.0.0              | com.hp.osgi.declarative.consumer Bundle
84 | Active     | 80  | 1.0.0              | com.hp.osgi.declarative.services Bundle
86 | Installed  | 80  | 1.0.0              | com.hp.osgi.st.ServiceTrackerQuoteConsumer Bundle
karaf@root()>
```

If everything is well, then the bundle is in Installed status, now let us start it

```
#bundle:start 86
```

You are using the earlier Quote services, so all dependent bundles of Quote services need to be in active state.

When you start the bundle it will give error:

```

66 | Installed | 80 | 1.0.0.201707081832 | Firstbundle
67 | Active   | 80 | 0.0.1.SNAPSHOT      | com.hp.osgi.bundle Bundle
73 | Active   | 80 | 0.0.1.SNAPSHOT      | com.hp.osgi.quote Bundle
75 | Active   | 80 | 1.1.0                | com.hp.osgi.quoteconsumer Bundle
76 | Resolved  | 80 | 1.0.0                | com.hp.osgi.quoteservice Bundle
83 | Active   | 80 | 1.0.0                | com.hp.osgi.declarative.consumer Bundle
84 | Active   | 80 | 1.0.0                | com.hp.osgi.declarative.services Bundle
karaf@root()> bundle:install file:///d:///temp/com.hp.osgi.st.ServiceTrackerQuoteConsumer-1.0.0.jar
Bundle ID: 87
karaf@root()> bundle:st
bundle:start          bundle:start-level    bundle:status        bundle:stop
karaf@root()> bundle:start 87
Error executing command: Error executing command on bundles:
    Error starting bundle 87: Unable to resolve com.hp.osgi.st.ServiceTrackerQuoteConsumer [87](R 87.0): missing requirement [com.hp.osgi.st.ServiceTrackerQuoteConsumer [87](R 87.0)] osgi.wiring.package; (osgi.wiring.package=org.eclipse.osgi.util) Unresolved requirements: [[com.hp.osgi.st.ServiceTrackerQuoteConsumer [87](R 87.0)] osgi.wiring.package; (osgi.wiring.package=org.eclipse.osgi.util)]

```

So, how do you debug it? You can use the bundle:tree-show to determine the dependencies resolution as shown below:
#bundle:tree-show 87

```

karaf@root()> bundle:tree-show 87
Bundle com.hp.osgi.st.ServiceTrackerQuoteConsumer [87] is currently INSTALLED
- import org.osgi.util.tracker;version="[1.5,2)": resolved using org.apache.felix.framework [0]
- import com.hp.osgi.quote;version="[1.0,2)": resolved using com.hp.osgi.quote [73]
- import org.osgi.framework;version="[1.8,2)": resolved using org.apache.felix.framework [0]
- import org.eclipse.osgi.util: WARNING - unable to find matching export

Warning: the below tree is a rough approximation of a possible resolution
com.hp.osgi.st.ServiceTrackerQuoteConsumer [87]
+- org.apache.felix.framework [0]
+- com.hp.osgi.quote [73]
karaf@root()>

```

You can see that there is a package that is unable to resolve.

We have included a package; org.eclipse.osgi.util which is not satisfy by any bundle, let us remove it from the import configuration and re export the bundle.

```
58             <Bundle-Activator>com.hp.osgi.st.Activator</Bundle-Activator>
59             <Export-Package>
60                 com.hp.osgi.st*;version=${project.version}
61             </Export-Package>
62             <Import-Package>
63                 org.osgi.util.tracker,com.hp.osgi.quote,org.osgi.framework,*  
64             </Import-Package>
65             </instructions>
66         </configuration>
```

Then, uninstall the earlier bundle and install again.

```
karaf@root()> bundle:uninstall 87
karaf@root()> bundle:install file:///d://temp/com.hp.osgi.st.ServiceTrackerQuoteConsumer-1.0.0.jar
Bundle ID: 88
karaf@root()> bundle:start 88
Starting quoteconsumer bundles
Ds: I feel better already
karaf@root()> Ds: Tell them I said something
Ds: Tell them I said something
Ds: I feel better already
Ds: Hubba Bubba, Quote!
Ds: Hubba Bubba, Quote!
Ds: Hubba Bubba, Quote!
```

Once you start your service the tracker will be called and the consumer bundle will start writing messages to the console. Stop the service (Quote Declarative in this case) and verify that the consumer does not use the service anymore.(It won't display the message and start the service it will again display the message)

```
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version           | Name
--+
66 | Installed  | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle
73 | Resolved   | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quote Bundle
75 | Resolved   | 80  | 1.1.0              | com.hp.osgi.quoteconsumer Bundle
76 | Installed  | 80  | 1.0.0              | com.hp.osgi.quoteservice Bundle
83 | Active     | 80  | 1.0.0              | com.hp.osgi.declarative.consumer Bundle
84 | Active     | 80  | 1.0.0              | com.hp.osgi.declarative.services Bundle
88 | Active     | 80  | 1.0.0              | com.hp.osgi.st.ServiceTrackerQuoteConsumer Bundle
karaf@root()> bundle:stop 73Ds: I feel better already
Ds: Hubba Bubba, Quote!
Ds: Tell them I said something

Error executing command: No matching bundles
karaf@root()> bundle:stop 84Ds: I feel better already

How sad. Service for quote is gone
karaf@root()>
```

After you start the service again

```
karaf@root()> bundle:start 84
Ds: I feel better alreadyDs: Hubba Bubba, Quote!
karaf@root()> Ds: I feel better already
Ds: Tell them I said something
Ds: I feel better already
Ds: I feel better already
Ds: Tell them I said something
Ds: I feel better already
Ds: Hubba Bubba, Quote!
```

OSGI HTTP Service with Equinox



Create one “Plugin-Project” :- com.hp.httpservice

Create one servlet: HelloWorldServlet

```
package com.hp.servlet;

import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class HelloWorldServlet extends HttpServlet {

    protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
        resp.setContentType("text/html");
        resp.getWriter().write("<html><body>Hello World -- sample servlet</body></html>"); //NON-NLS-1$
    }
}
```

Create one Activator:

```
package myactivator;

import org.osgi.framework.BundleActivator;
import org.osgi.framework.BundleContext;
import org.osgi.framework.ServiceReference;
import org.osgi.service.http.HttpService;
import org.osgi.util.tracker.ServiceTracker;

import com.hp.servlet.HelloWorldServlet;

public class Activator implements BundleActivator {

    private ServiceTracker httpServiceTracker;

    public void start(BundleContext context) throws Exception {
        httpServiceTracker = new HttpServiceTracker(context);
        httpServiceTracker.open();
    }

    public void stop(BundleContext context) throws Exception {
        httpServiceTracker.close();
        httpServiceTracker = null;
    }

    private class HttpServiceTracker extends ServiceTracker {

        public HttpServiceTracker(BundleContext context) {
            super(context, HttpService.class.getName(), null);
            System.out.println(" hELLO 1 ");
        }
    }
}
```

```
public Object addingService(ServiceReference reference) {
    HttpService httpService = (HttpService) context.getService(reference);
    System.out.println(" hELLO 2 " + httpService);
    try {
        httpService.registerServlet("/helloworld", new HelloWorldServlet(),
null, null); //NON-NLS-1$
        System.out.println(" hELLO 3 " + httpService);
    } catch (Exception e) {
        e.printStackTrace();
    }
    return httpService;
}

public void removedService(ServiceReference reference, Object service) {
    HttpService httpService = (HttpService) service;
    httpService.unregister("/helloworld"); //NON-NLS-1$
    super.removedService(reference, service);
}

}
```

Create a manifest file: META-INF/MANIFEST.MF

```
Manifest-Version: 1.0
Bundle-ManifestVersion: 2
Bundle-Name: MyActivator
Bundle-SymbolicName: MyActivator;singleton:=true
Bundle-Version: 3.0.0.qualifier
Bundle-Activator: myactivator.Activator
Import-Package: javax.servlet,
    javax.servlet.http,
    org.osgi.framework;version="1.3.0",
    org.osgi.service.http;version="1.2.1",
    org.osgi.util.tracker;version="1.4.0"
Bundle-RequiredExecutionEnvironment: JavaSE-1.6
Require-Bundle: org.eclipse.equinox.http.registry;bundle-version="[1.0.0,2.0.0)"
```

Create an extension. (Optional if you want to register your servlet using extension)

Open MANIFEST.MF -> Overview -> Extensions – Add (All extensions) -

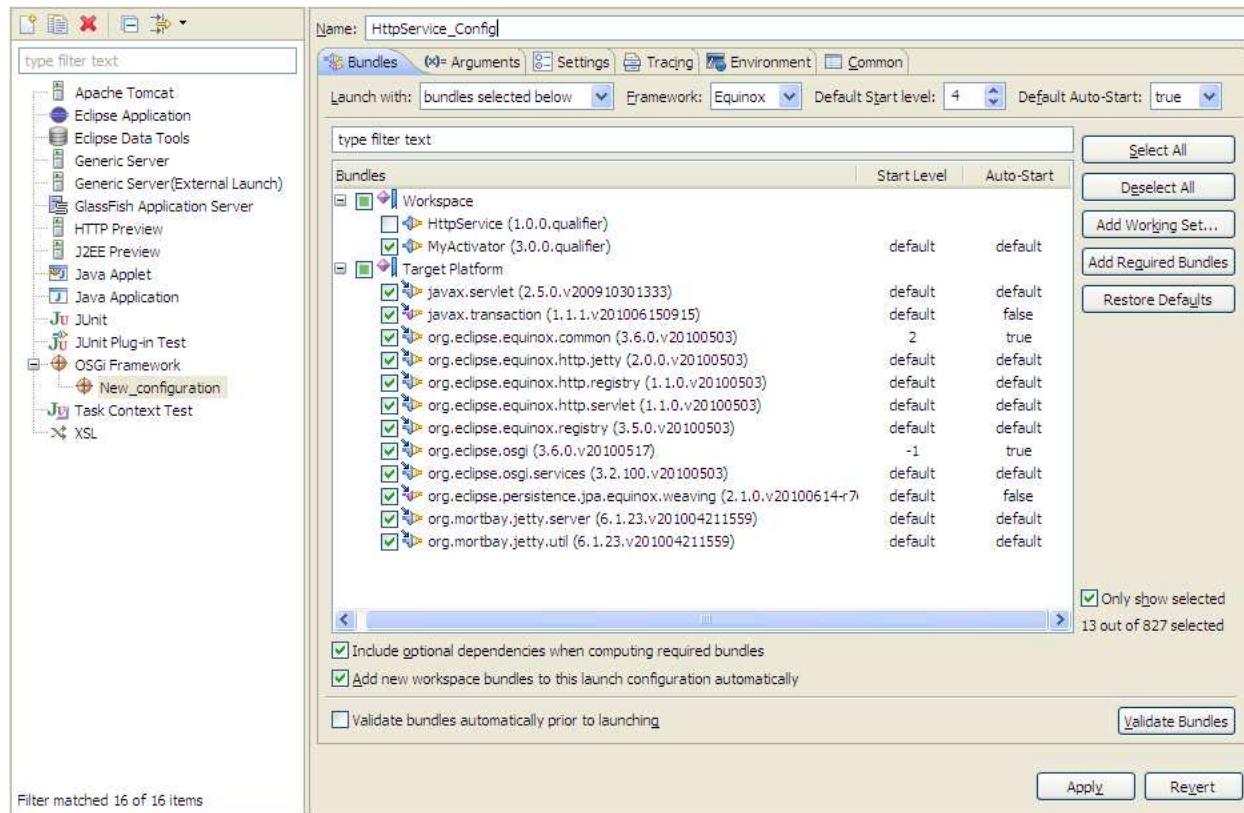
`org.eclipse.equinox.http.registry.resources.`

(replace with following in the plugin.xml file)

```
<?xml version="1.0" encoding="UTF-8"?>
<?eclipse version="3.4"?>
<plugin>
  <extension
    point="org.eclipse.equinox.http.registry.resources">
    <resource
      alias="/files"      base-name="/web_files"/>
  </extension>
  <extension point="org.eclipse.equinox.http.registry.servlets">
    <servlet
      alias="/test"
      class="com.hp.servlet.HelloWorldServlet"/>
  </extension>
</plugin>
```

Running the application:

Click on **HttpService** -> **Run Configurations** -> **OSGI Frameworks** -> Name (**HttpService_Config**) -> as follows



Change port: (Arguments tab in above figure) – (Optional if you want to change the default port no)

-Declipse.ignoreApp=true -Dosgi.noShutdown=true -Dorg.osgi.service.http.port=8080

<http://localhost:8080//helloworld>

Deploy the plugin to equinox. (Execute the bundles outside Eclipse)

Install and start all the following bundles:

javax.servlet_2.5.0.v200910301333.jar
javax.transaction_1.1.1.v201006150915.jar
org.eclipse.equinox.common_3.6.0.v20100503.jar
org.eclipse.equinox.http.jetty_2.0.0.v20100503.jar
org.eclipse.equinox.http.registry_1.1.0.v20100503.jar
org.eclipse.equinox.http.servlet_1.1.0.v20100503.jar
org.eclipse.equinox.registry_3.5.0.v20100503.jar
org.eclipse.equinox.util_1.0.300.v20111010-1614.jar
org.eclipse.osgi.services-3.1.200.v20071203.jar
org.eclipse.osgi.services_3.2.100.v20100503.jar
org.eclipse.osgi.util_3.2.100.v20100503.jar
org.eclipse.persistence.jpa.equinox.weaving_2.1.0.v20100614-r7608.jar
org.mortbay.jetty.server_6.1.23.v201004211559.jar
org.mortbay.jetty.util_6.1.23.v201004211559.jar
com.hp.httpservice_3.0.0.201208301650.jar

[hints:- install file:{jar} and start {id}]

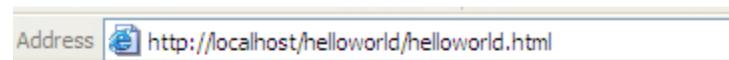
```
0:\OSGI>java -jar org.eclipse.osgi-3.6.0.v20100517.jar -console
osgi> hELLO 1
hELLO 2 org.eclipse.equinox.http.servlet.internal.HttpServiceImpl@cbf30e
hELLO 3 org.eclipse.equinox.http.servlet.internal.HttpServiceImpl@cbf30e
```

```
osgi> ss
Framework is launched.

id      State    Bundle
0      ACTIVE   org.eclipse.osgi_3.6.0.v20100517
       Fragments=17, 23
16     ACTIVE   javax.servlet_2.5.0.v200910301333
17     RESOLVED  javax.transaction_1.1.1.v201006150915
       Master=@0
18     ACTIVE   org.eclipse.equinox.common_3.6.0.v20100503
19     ACTIVE   org.eclipse.equinox.http.jetty_2.0.0.v20100503
20     ACTIVE   org.eclipse.equinox.http.registry_1.1.0.v20100503
21     ACTIVE   org.eclipse.equinox.http.servlet_1.1.0.v20100503
22     ACTIVE   org.eclipse.equinox.registry_3.5.0.v20100503
23     RESOLVED  org.eclipse.persistence.jpa.equinox.weaving_2.1.0.v20100614-
r7608
       Master=@0
24     ACTIVE   org.mortbay.jetty.server_6.1.23.v201004211559
25     ACTIVE   org.mortbay.jetty.util_6.1.23.v201004211559
26     ACTIVE   org.eclipse.osgi.util_3.2.100.v20100503
27     ACTIVE   org.eclipse.equinox.util_1.0.300.v20111010-1614
28     ACTIVE   org.eclipse.osgi.services_3.1.200.v20071203
29     ACTIVE   com.hp.httpservice_3.0.0.201208301650

osgi> _
```

<http://localhost:8080//helloworld>

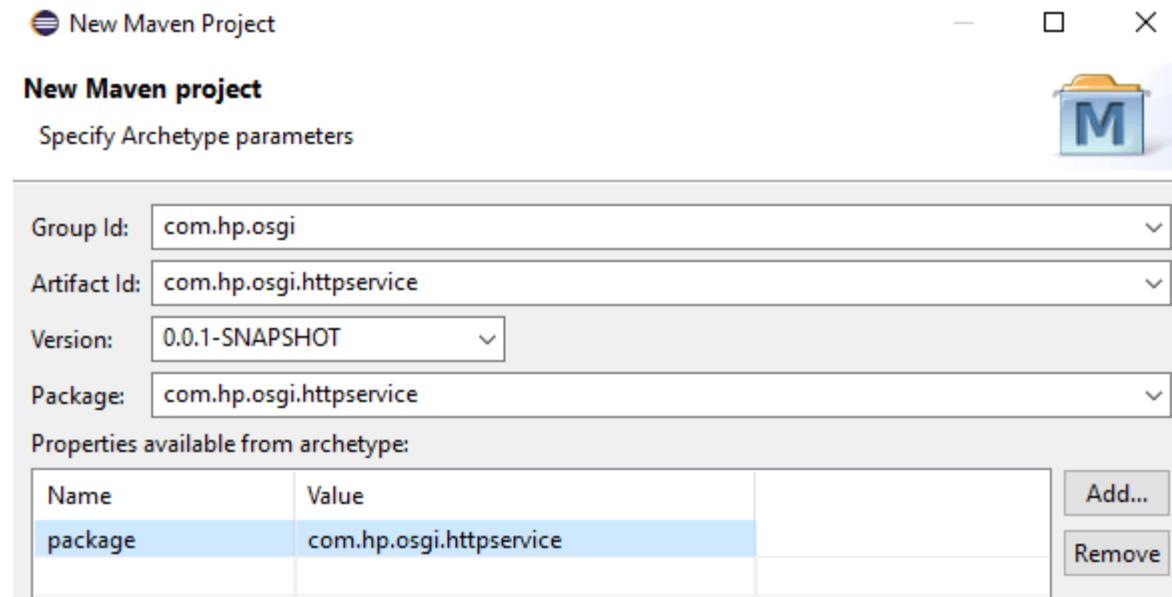


Hello World -- sample servlet

OSGI HTTP Service with Karaf

In this lab, we will expose a servlet using HTTP Service.

Let us create a “Plugin-Project” using maven karaf plugin generator with the following specs:



Click Finish.

Create a servlet name HelloWorldServlet.java, it will extend HttpServlet and we will be printing out Hello World – OSGI Servlet when a get request is received.

```
package com.hp.osgi.httpservice;

import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class HelloWorldServlet extends HttpServlet {

    protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
        resp.setContentType("text/html");
        resp.getWriter().write("<html><body>Hello World – OSGI servlet</body></html>");
    }
}
```

You need to define the dependency of HttpServlet in the pom.xml. You can update the pom.xml as follows.

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

  <!-- Licensed to the Apache Software Foundation (ASF) under one or more
       contributor license agreements. See the NOTICE file distributed with this
       work for additional information regarding copyright ownership. The ASF licenses
       this file to you under the Apache License, Version 2.0 (the "License"); you
       may not use this file except in compliance with the License. You may obtain
       a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless
       required by applicable law or agreed to in writing, software distributed
       under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES
       OR CONDITIONS OF ANY KIND, either express or implied. See the License for
       the specific language governing permissions and limitations under the License. -->

  <modelVersion>4.0.0</modelVersion>

  <groupId>com.hp.osgi</groupId>
  <artifactId>com.hp.osgi.httpservice</artifactId>
  <version>1.0.0</version>
  <packaging>bundle</packaging>

  <name>com.hp.osgi.httpservice Bundle</name>
  <description>
    com.hp.osgi.httpservice OSGi bundle project.
  </description>

  <properties>
    <maven-bundle-plugin.version>2.5.4</maven-bundle-plugin.version>
    <osgi.version>6.0.0</osgi.version>
  </properties>

  <dependencies>
    <dependency>
      <groupId>org.osgi</groupId>
      <artifactId>org.osgi.core</artifactId>
      <version>${osgi.version}</version>
```

```

        <scope>provided</scope>
    </dependency>
    <!-- https://mvnrepository.com/artifact/javax.servlet/servlet-api -->
    <dependency>
        <groupId>javax.servlet</groupId>
        <artifactId>servlet-api</artifactId>
        <version>2.5</version>
        <scope>provided</scope>
    </dependency>
    <!-- https://mvnrepository.com/artifact/org.osgi/org.osgi.service.http -->
    <dependency>
        <groupId>org.osgi</groupId>
        <artifactId>org.osgi.service.http</artifactId>
        <version>1.2.1</version>
        <scope>provided</scope>
    </dependency>

</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.felix</groupId>
            <artifactId>maven-bundle-plugin</artifactId>
            <version>${maven-bundle-plugin.version}</version>
            <extensions>true</extensions>
            <configuration>
                <instructions>
                    <Bundle-SymbolicName>${project.artifactId}</Bundle-SymbolicName>
                    <Bundle-Version>${project.version}</Bundle-Version>
                    <Bundle-Activator>com.hp.osgi.httpservice.Activator</Bundle-Activator>
                    <Export-Package>
                        com.hp.osgi.httpservice*;version=${project.version}
                    </Export-Package>
                    <Import-Package>
                        javax.servlet.http,
                        org.osgi.util.tracker,

```

```
        *
        </Import-Package>
        </instructions>
    </configuration>
</plugin>
</plugins>
</build>

</project>
```

Update the Activator with the following code: In the activator class, we are getting a service tracker in the start method and this object is used to register our servlet in the addingService method.

```
package myactivator;

import org.osgi.framework.BundleActivator;
import org.osgi.framework.BundleContext;
import org.osgi.framework.ServiceReference;
import org.osgi.service.http.HttpService;
import org.osgi.util.tracker.ServiceTracker;

import com.hp.servlet.HelloWorldServlet;

public class Activator implements BundleActivator {

    private ServiceTracker httpServiceTracker;

    public void start(BundleContext context) throws Exception {
        httpServiceTracker = new HttpServiceTracker(context);
        httpServiceTracker.open();

    }

    public void stop(BundleContext context) throws Exception {
        httpServiceTracker.close();
        httpServiceTracker = null;
    }

    private class HttpServiceTracker extends ServiceTracker {

        public HttpServiceTracker(BundleContext context) {
            super(context, HttpService.class.getName(), null);
        }
    }
}
```

```
        System.out.println(" Hello from HttpService Service Tracker! ");
    }

    public Object addingService(ServiceReference reference) {
        HttpService httpService = (HttpService) context.getService(reference);
        System.out.println(" Hello 2 " + httpService);
        try {
            httpService.registerServlet("/helloworld", new HelloWorldServlet(), null, null);
            System.out.println(" Hello 3 " + httpService);
        } catch (Exception e) {
            e.printStackTrace();
        }
        return httpService;
    }

    public void removedService(ServiceReference reference, Object service) {
        HttpService httpService = (HttpService) service;
        httpService.unregister("/helloworld");
        super.removedService(reference, service);
    }
}
```

We have already include the org.osgi.service.http.HttpService; in our pom.xml

Perform clean and maven install. Then you can install the bundle using mvn install command of bundle:install.

```
#bundle:install mvn:com.hp.osgi/com.hp.osgi.httpservice/1.0.0
```

```

karaf@root()> bundle:install mvn:com.hp.osgi/com.hp.osgi.httpservice/1.0.0
Bundle ID: 92
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State   | Lvl | Version      | Name
--+
66 | Installed | 80 | 1.0.0.201707081832 | Firstbundle
67 | Active    | 80 | 0.0.1.SNAPSHOT | com.hp.osgi.bundle Bundle
73 | Resolved  | 80 | 0.0.1.SNAPSHOT | com.hp.osgi.quote Bundle
75 | Resolved  | 80 | 1.1.0        | com.hp.osgi.quoteconsumer Bundle
76 | Installed | 80 | 1.0.0        | com.hp.osgi.quoteservice Bundle
83 | Active    | 80 | 1.0.0        | com.hp.osgi.declarative.consumer Bundle
84 | Resolved  | 80 | 1.0.0        | com.hp.osgi.declarative.services Bundle
88 | Active    | 80 | 1.0.0        | com.hp.osgi.st.ServiceTrackerQuoteConsumer Bundle
92 | Installed | 80 | 1.0.0        | com.hp.osgi.httpservice Bundle
karaf@root()>

```

Then you can start the bundle now:

```
# bundle:start 92
```

```

92 | Installed | 80 | 1.0.0          | com.hp.osgi.httpservice Bundle
karaf@root()> bundle:start
bundle:start           bundle:start-level
karaf@root()> bundle:start 92
Error executing command: Error executing command on bundles:
    Error starting bundle 92: Unable to resolve com.hp.osgi.httpservice [92](R 92.0): missing requirement [com.hp.osgi.httpservice [92](R 92.0)] osgi.wiring.package; (osgi.wiring.package=javax.servlet.http) Unresolved requirements: [[com.hp.osgi.httpservice [92](R 92.0)] osgi.wiring.package; (osgi.wiring.package=javax.servlet.http)]

```

If you get the above error, what do you do? Hints determine the dependencies issue using tree-show.

```
karaf@root()> bundle:tree-show 92
Bundle com.hp.osgi.httpservice [92] is currently INSTALLED
- import javax.servlet.http: WARNING - unable to find matching export
- import org.osgi.util.tracker;version="[1.5,2)": resolved using org.apache.felix.framework [0]
- import javax.servlet: WARNING - unable to find matching export
- import org.osgi.framework;version="[1.8,2)": resolved using org.apache.felix.framework [0]
- import org.osgi.service.http;version="[1.2,2)": WARNING - unable to find matching export
- import org.osgi.service.http.HttpService: WARNING - unable to find matching export

Warning: the below tree is a rough approximation of a possible resolution
com.hp.osgi.httpservice [92]
+- org.apache.felix.framework [0]
karaf@root()>
```

It's clearly shown that lot of packages are not export by any of the bundles installed in the container. You can enable the http service feature using the following command.

```
#feature:install http
```

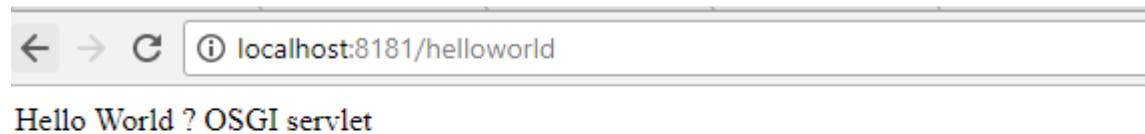
Now you can start the bundle:

```
karaf@root()> bundle:start 138
Hello from HttpService Service Tracker!
Hello 2 org.ops4j.pax.web.service.internal.HttpServiceProxy@7d3fdd91
Hello 3 org.ops4j.pax.web.service.internal.HttpServiceProxy@7d3fdd91
karaf@root()> bundle:tree-show 138
Bundle com.hp.osgi.httpservice [138] is currently ACTIVE

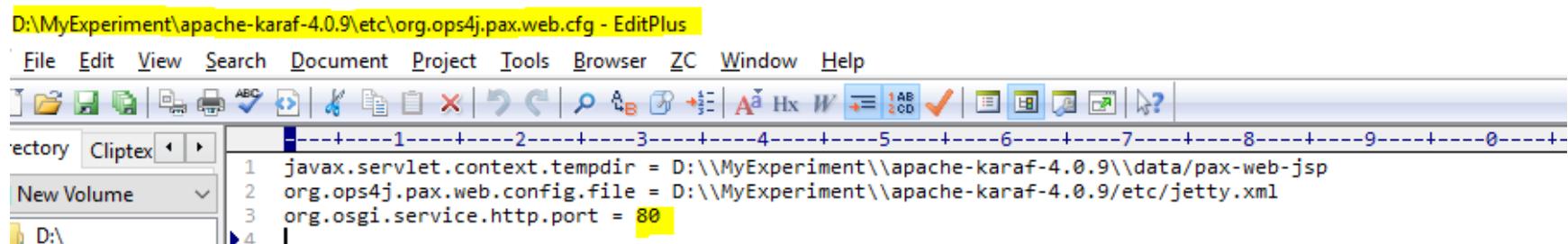
com.hp.osgi.httpservice [138]
+- org.ops4j.pax.web.pax-web-runtime [129]
| +- org.ops4j.pax.web.pax-web-api [127]
| | +- org.ops4j.pax.logging.pax-logging-api [1]
| | | +- org.apache.karaf.services.eventadmin [6]
| | | | +- org.apache.felix.configadmin [3]
| | | | +- org.apache.felix.metatype [5]
| | +- javax.servlet-api [94]
| +- org.ops4j.pax.web.pax-web-spi [130]
| | +- org.ops4j.pax.web.pax-web-api [127]
| | +- org.ops4j.pax.logging.pax-logging-api [1]
| | +- javax.servlet-api [94]
| +- org.apache.karaf.services.eventadmin [6]
| +- org.ops4j.pax.web.pax-web-jsp [137]
| | +- org.ops4j.pax.web.pax-web-api [127]
| | +- org.ops4j.pax.web.pax-web-spi [130]
| | +- org.ops4j.pax.logging.pax-logging-api [1]
| | +- javax.servlet-api [94]
| | +- org.eclipse.jdt.core.compiler.batch [132]
| +- org.ops4j.pax.logging.pax-logging-api [1]
| +- org.apache.felix.configadmin [3]
| +- javax.servlet-api [94]
```

Now, all the dependencies are resolved

<http://localhost:8181/helloworld>

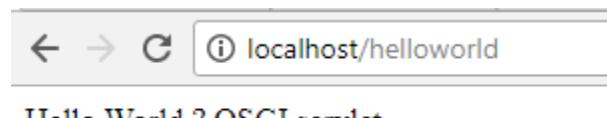


If you want to changes the port then, You can update [etc/org.ops4j.pax.web.cfg](#)



Now access the url with the new port:

<http://localhost/helloworld>

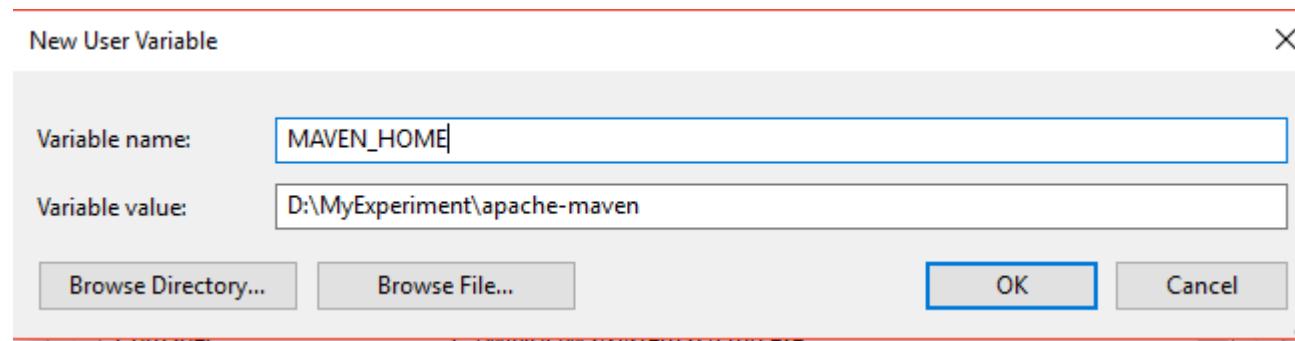


Embedding Karaf in a WebContainer - Tomcat

We will deploy apache karaf in apache tomcat server container, by completing the following steps:

1: Install Maven 3

Unzip apache-maven-3.5.0-bin.zip and set MAVEN_HOME



Include in the PATH Variable:

```
D:\MyExperiment\Vagrant\bin  
D:\MyExperiment\apache-maven\bin
```

2: Install Apache Tomcat

Unzip apache-tomcat-8.0.45-windows-x64.zip

This PC > New Volume (D:) > MyExperiment > apache-tomcat-8.0.45

| Name | Date modified | Type | Size |
|---------------|--------------------|---------------|-------|
| bin | 6/26/2017 11:08 PM | File folder | |
| conf | 6/26/2017 11:08 PM | File folder | |
| lib | 6/26/2017 11:08 PM | File folder | |
| logs | 6/26/2017 11:06 PM | File folder | |
| temp | 6/26/2017 11:08 PM | File folder | |
| webapps | 6/26/2017 11:08 PM | File folder | |
| work | 6/26/2017 11:06 PM | File folder | |
| LICENSE | 6/26/2017 11:08 PM | File | 57 KB |
| NOTICE | 6/26/2017 11:08 PM | File | 2 KB |
| RELEASE-NOTES | 6/26/2017 11:08 PM | File | 7 KB |
| RUNNING.txt | 6/26/2017 11:08 PM | Text Document | 17 KB |

3. Enter the following command: by going to the APACHE_KARAF/ demos/web folder using command line console

```
mvn package
```

```
D:\MyExperiment\apache-karaf-4.0.9\demos>cd web  
D:\MyExperiment\apache-karaf-4.0.9\demos\web>mvn package  
[INFO] Scanning for projects...  
Downloading: https://repo.maven.apache.org/maven2/org/apache/karaf/karaf/4.0.9/karaf-4.0.9.pom
```

```
| (53 kB at 10 kB/s)
Downloaded: https://repo.maven.apache.org/maven2/org/mortbay/jetty/jetty/6.1.25/jetty-6.1.25.jar (539 kB at 127 kB/s)
Downloaded: https://repo.maven.apache.org/maven2/org/apache/commons/commons-lang3/3.4/commons-lang3-3.4.jar (435 kB at 9
5 kB/s)
Downloaded: https://repo.maven.apache.org/maven2/org/mortbay/jetty/jetty-util/6.1.25/jetty-util-6.1.25.jar (177 kB at 34
kB/s)
[INFO] Skipping because packaging 'war' is not pom.
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 03:30 min
[INFO] Finished at: 2017-07-13T20:41:10+05:30
[INFO] Final Memory: 36M/314M
[INFO] -----
```

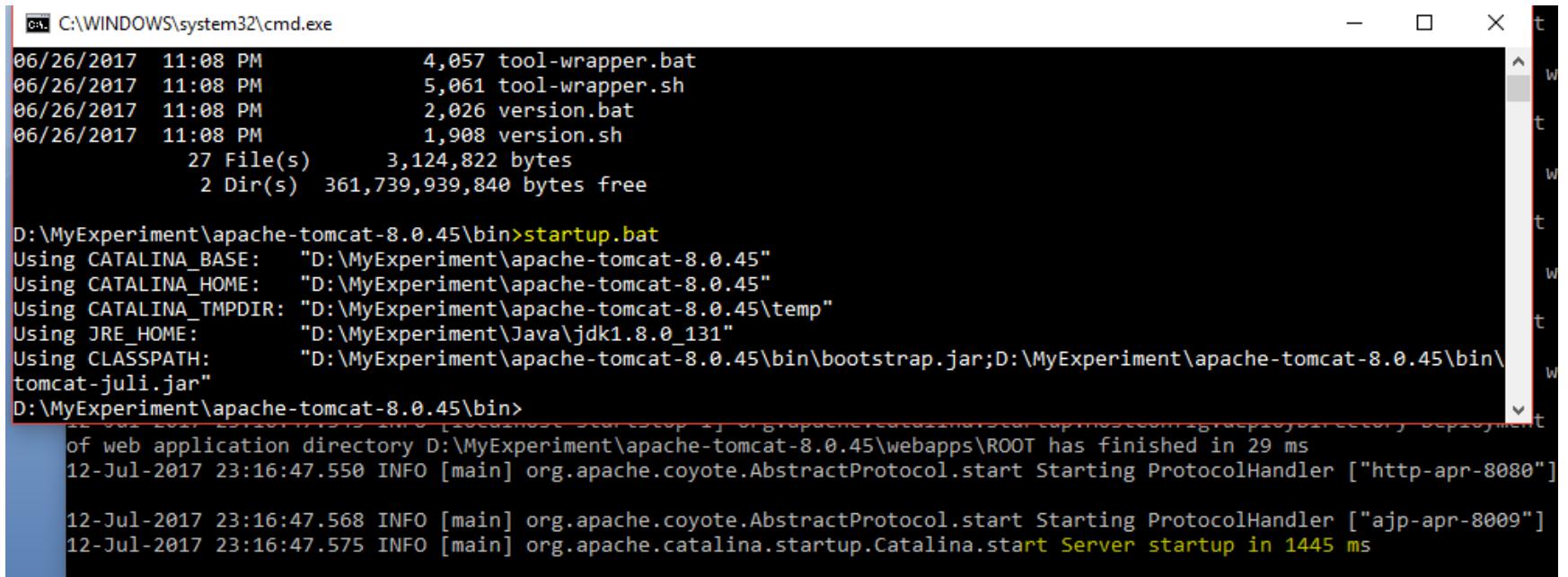
D:\MyExperiment\apache-karaf-4.0.9\demos\web>

Maven builds the web application, web-4.0.9.war, and saves it in the target directory of this example. Deploy this WAR file to your favorite web container. Once the application is running, you can test it using the Apache Karaf client as described in the "Running a Client" section above.

It will download all the dependencies required by this application, it may take some times depending upon your bandwidth.

Now, you can start the apache tomcat server. Open a new command prompt and go to the apache tomcat installation/bin folder and execute the following command:

```
#startup.bat
```



C:\WINDOWS\system32\cmd.exe

```
06/26/2017 11:08 PM      4,057 tool-wrapper.bat
06/26/2017 11:08 PM      5,061 tool-wrapper.sh
06/26/2017 11:08 PM      2,026 version.bat
06/26/2017 11:08 PM      1,908 version.sh
    27 File(s)     3,124,822 bytes
    2 Dir(s)  361,739,939,840 bytes free

D:\MyExperiment\apache-tomcat-8.0.45\bin>startup.bat
Using CATALINA_BASE: "D:\MyExperiment\apache-tomcat-8.0.45"
Using CATALINA_HOME: "D:\MyExperiment\apache-tomcat-8.0.45"
Using CATALINA_TMPDIR: "D:\MyExperiment\apache-tomcat-8.0.45\temp"
Using JRE_HOME: "D:\MyExperiment\Java\jdk1.8.0_131"
Using CLASSPATH: "D:\MyExperiment\apache-tomcat-8.0.45\bin\bootstrap.jar;D:\MyExperiment\apache-tomcat-8.0.45\bin\tomcat-juli.jar"
D:\MyExperiment\apache-tomcat-8.0.45\bin>
of web application directory D:\MyExperiment\apache-tomcat-8.0.45\webapps\ROOT has finished in 29 ms
12-Jul-2017 23:16:47.550 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler ["http-apr-8080"]

12-Jul-2017 23:16:47.568 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler ["ajp-apr-8009"]
12-Jul-2017 23:16:47.575 INFO [main] org.apache.catalina.startup.Catalina.start Server startup in 1445 ms
```

If everything is ok, you can refer the Apache tomcat website with the url:

<http://localhost:8080/>

localhost:8080

Home Documentation Configuration Examples Wiki Mailing Lists Find Help

Apache Tomcat/8.0.45

If you're seeing this, you've successfully installed Tomcat. Congratulations!



Recommended Reading:

- [Security Considerations HOW-TO](#)
- [Manager Application HOW-TO](#)
- [Clustering/Session Replication HOW-TO](#)

Server Status
Manager App
Host Manager

Developer Quick Start

| | | | |
|---------------------------------------|----------------------------------|--------------------------|----------------------------------------|
| Tomcat Setup | Realms & AAA | Examples | Servlet Specifications |
| First Web Application | JDBC DataSources | | Tomcat Versions |

Let us deploy the karaf web application in our tomcat server;

Copy web-4.0.9.war to apache tomcat webapps folder.

| This PC > New Volume (D:) > MyExperiment > apache-karaf-4.0.9 > demos > web > target | | | |
|--------------------------------------------------------------------------------------|-------------------|-------------|-----------|
| Name | Date modified | Type | Size |
| antrun | 7/13/2017 8:40 PM | File folder | |
| classes | 7/13/2017 8:40 PM | File folder | |
| dependency-maven-plugin-markers | 7/13/2017 8:39 PM | File folder | |
| generated-sources | 7/13/2017 8:40 PM | File folder | |
| karaf | 7/13/2017 8:39 PM | File folder | |
| maven-archiver | 7/13/2017 8:40 PM | File folder | |
| maven-shared-archive-resources | 7/13/2017 8:38 PM | File folder | |
| maven-status | 7/13/2017 8:40 PM | File folder | |
| test-classes | 7/13/2017 8:40 PM | File folder | |
| web-4.0.9 | 7/13/2017 8:40 PM | File folder | |
| .plxarc | 7/13/2017 8:38 PM | PLXARC File | 1 KB |
| <u>web-4.0.9.war</u> | 7/13/2017 8:40 PM | WAR File | 20,457 KB |

| This PC > New Volume (D:) > MyExperiment > apache-tomcat-8.0.45 > webapps | | | |
|---------------------------------------------------------------------------|--------------------|-------------|-----------|
| Name | Date modified | Type | Size |
| docs | 6/26/2017 11:08 PM | File folder | |
| examples | 6/26/2017 11:08 PM | File folder | |
| host-manager | 6/26/2017 11:08 PM | File folder | |
| manager | 6/26/2017 11:08 PM | File folder | |
| ROOT | 6/26/2017 11:08 PM | File folder | |
| web-4.0.9.war | 7/13/2017 8:40 PM | WAR File | 20,457 KB |

If you face issue while deploying the karaf war, ensure to update the start up config specified in next para.

D:\MyExperiment\apache-tomcat-8.0.45\webapps\web-4.0.9\WEB-INF\karaf\etc\ config.properties

```
# Location of the OSGi frameworks
#
karaf.framework.equinox=mvn\:\org.eclipse.birt.runtime\org.eclipse.osgi\3.10.2.v20150203-1939
karaf.framework.felix=mvn\:\org.apache.felix\org.apache.felix.framework\5.6.2

#
```

```

time.
contextInitialized
Root: D:\MyExperiment\apache-tomcat-8.0.45\webapps\web-4.0.9\WEB-INF\karaf
13-Jul-2017 21:02:03.952 INFO [localhost-startStop-1] org.apache.karaf.main.Main.launch Installing and starting initial
bundles
java.lang.RuntimeException: Error installing bundle listed in startup.properties with url: mvn:org.apache.karaf.services
/org.apache.karaf.services.eventadmin/4.0.4-SNAPSHOT and startlevel: 5
    at org.apache.karaf.main.Main.installAndStartBundles(Main.java:495)
    at org.apache.karaf.main.Main.launch(Main.java:311)
    at org.apache.karaf.web.WebAppListener.contextInitialized(WebAppListener.java:45)
    at org.apache.catalina.core.StandardContext.listenerStart(StandardContext.java:4853)
    at org.apache.catalina.core.StandardContext.startInternal(StandardContext.java:5314)
    at org.apache.catalina.util.LifecycleBase.start(LifecycleBase.java:145)
    at org.apache.catalina.core.ContainerBase.addChildInternal(ContainerBase.java:753)
    at org.apache.catalina.core.ContainerBase.addChild(ContainerBase.java:720)

15 #      See the License for the specific language governing permissions and
16 #      limitations under the License.
17 #
18 ######
19
20 # Bundles to be started on startup, with startlevel
21 mvn\:org.apache.sshd/sshd-core/0.14.0 = 20
22 mvn\:org.apache.felix/org.apache.felix.metatype/1.1.2 = 5
23 mvn\:org.apache.karaf.services/org.apache.karaf.services.eventadmin/4.0.9 = 5
24 mvn\:org.ops4j.pax.url/pax-url-aether/2.4.4 = 5
25 mvn\:org.ops4j.pax.logging/pax-logging-api/1.8.4 = 8
26 mvn\:org.ops4j.pax.logging/pax-logging-service/1.8.4 = 8
27 mvn\:org.apache.felix/org.apache.felix.configadmin/1.8.8 = 10
28 mvn\:org.apache.felix/org.apache.felix.fileinstall/3.5.0 = 11
29 mvn\:org.apache.karaf.features/org.apache.karaf.features.core/4.0.9 = 15
30 mvn\:org.apache.felix/org.apache.felix.http.bridge/3.0.0 = 15
31

```

Execute mvn package again then reinstall it

Running a Client

To test the example, you can use the Apache Karaf client to connect to the server and issue a Karaf command. For example, try executing the "features:list" command as follows:

1. In a command prompt/shell, change to your product installation directory. i.e karaf
2. Run the following commands:

Windows:

```
%KARAF_HOME%\bin\client.bat feature:list
```

In this case, you should see output similar to the following:

(abbreviated output below)

| State | Version | Name | Repository | Description |
|---------------|-------------------|-----------------|----------------|----------------------------------------------|
| [uninstalled] | [4.0.9] | wrapper | standard-4.0.9 | Provide OS integration |
| [uninstalled] | [4.0.9] | obr | standard-4.0.9 | Provide OSGi Bundle Repository (OBR) support |
| [uninstalled] | [4.0.9] | config | standard-4.0.9 | Provide OSGi ConfigAdmin support |
| [uninstalled] | [4.0.9] | region | standard-4.0.9 | Provide Region commands |
| [uninstalled] | [7.5.4.v20111024] | jetty | standard-4.0.9 | Provide Jetty engine support |
| [uninstalled] | [4.0.9] | http | standard-4.0.9 | Implementation of the OSGI HTTP Service |
| [uninstalled] | [4.0.9] | http-whiteboard | standard-4.0.9 | Provide HTTP Whiteboard pattern support |
| [uninstalled] | [4.0.9] | war | standard-4.0.9 | Turn Karaf as a full WebContainer |
| [uninstalled] | [4.0.9] | deployers | standard-4.0.9 | Provide Karaf deployer |

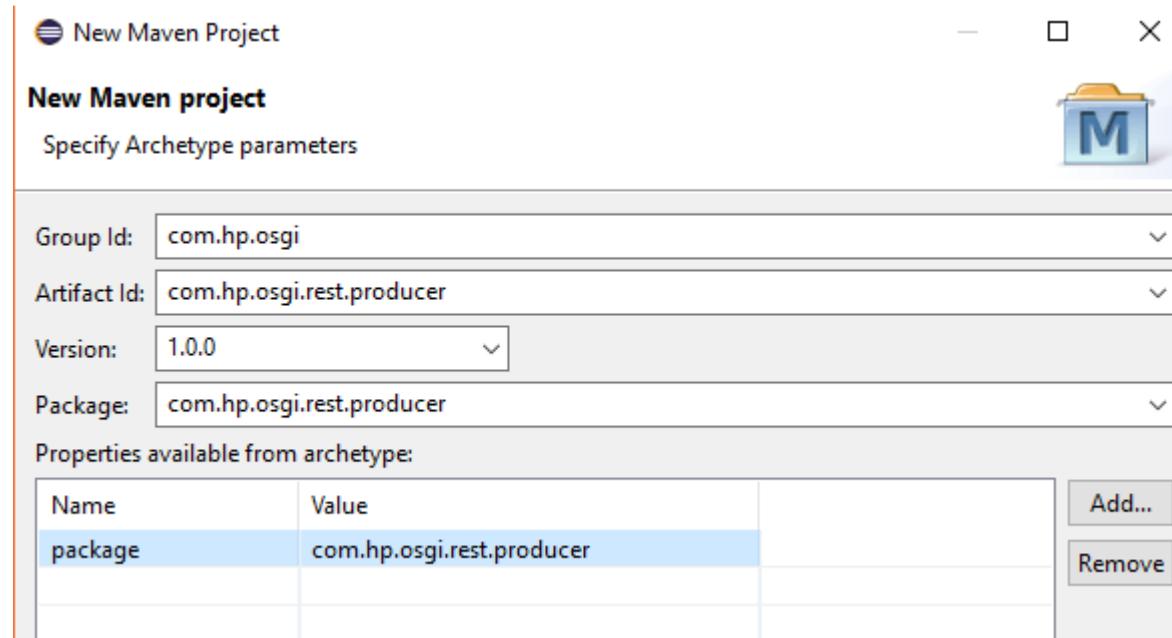
```
[uninstalled] [4.0.9 ] kar           standard-4.0.9   Provide KAR (KARaf archive) support
[uninstalled] [4.0.9 ] webconsole-base standard-4.0.9   Base support of the Karaf WebConsole
[uninstalled] [4.0.9 ] webconsole    standard-4.0.9   Karaf WebConsole for administration and monitoring
[uninstalled] [4.0.9 ] ssh           standard-4.0.9   Provide a SSHd server on Karaf
```

Congratulation!

REST Web Services – Producer - Karaf

Create jersey based – Producer. In this lab we will expose REST API and will be consumer by another Bundle in the next lab. We will be creating web services using REST API.

- 1) Create a maven project – archetype –quickstart. (simple-web-rest)



Click Finish

Create the following RestBO class which will be our business object, all the business logic for the web services will be creating in this class only. In this class it should be transparent of any framework i.e it should be a POJO class only.

```
package com.hp.bo;

import com.hp.osgi.rest.producer.CustomerVO;

public class RestBO {

    public static String createCustomer(CustomerVO vo){
        return "Successfully created, user " + vo.getName();
    }

    public static String updateCustomer(CustomerVO vo){
        return "Successfully updated, user " + vo.getName();
    }

    public static String deleteCustomer(String vo){
        return "Successfully deleted, user " + vo;
    }

    public static CustomerVO findCustomer(String name){
        CustomerVO vo = new CustomerVO(name, "", "");
        if(name.startsWith("a")){
            vo.setCity("Allahabad");
            vo.setProfession(" Consultant ");
        }else if (name.startsWith("b")){
            vo.setCity("Banglore");
            vo.setProfession(" IT Consultant ");
        }else if (name.startsWith("h")){
            vo.setCity("Mumbai");
            vo.setProfession(" Business Consultant ");
        } else {
            vo.setCity("Delhi");
            vo.setProfession(" Politician ");
        }

        return vo;
    }
}
```

Now, we will create CustomerVO class, which will be our Value object i.e we will be using this class to transfer value from one layer to another layer. It is again a POJO which will be used to contain our domain object.

```
package com.hp.vo;

public class CustomerVO {

    private String name;
    private String city;
    private String profession;

    public CustomerVO() {
        super();
    }

    public CustomerVO(String name, String city, String profession) {
        super();
        this.name = name;
        this.city = city;
        this.profession = profession;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getCity() {
        return city;
    }

    public void setCity(String city) {
        this.city = city;
    }

    public String getProfession() {
        return profession;
    }

    public void setProfession(String profession) {
        this.profession = profession;
    }
}
```

```

@Override
public String toString() {
    // TODO Auto-generated method stub
    return getName() + " - " + getCity() + " - " + getProfession();
}

}

```

Let us create a web services that will produce a text message when request is send for /another/time request path.

```

package com.hp.osgi.rest.producer;

import java.text.SimpleDateFormat;

import javax.ws.rs.GET;
import javax.ws.rs.Produces;
import javax.ws.rs.Path;

@Path("/another")
public class AnotherResource {

    @GET @Produces("text/plain")
    public String getAnotherMessage() {
        return "Another";
    }
    @Path("/time")
    @GET @Produces("text/plain")
    public String getWhatTime() {
        return new SimpleDateFormat("dd/mm/yyyy hh:mm:ss").
            format(System.currentTimeMillis());
    }
}

```

Then, we will again create another web services controller, where we expose multiple methods. This controller will be used to serve request path; /RestWorld/{}.

```
package com.hp.osgi.rest.producer;

import javax.ws.rs.Consumes;
import javax.ws.rs.FormParam;
import javax.ws.rs.GET;
import javax.ws.rs.POST;
import javax.ws.rs.PUT;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.core.Response;

import com.hp.bo.RestBO;
import com.hp.vo.CustomerVO;

@Path("/RestWorld")
public class HelloWorldResource {

    @GET @Produces("text/plain")
    @Path("Default")
    public String getClickedMessage() {
        return "Hello World From REST - JERSEY";
    }

    @POST
    @Produces("text/json")
    @Consumes("application/x-www-form-urlencoded")
    @Path("Create")
    public Response createCustomer(@FormParam("name") String name){
        //,@FormParam("city") String city, @FormParam("professional") String pro) {
        //System.out.print("-----" + name);
        CustomerVO vo = new CustomerVO(name,"","");
        System.out.print("--- RS SERVICES-----" + vo);
        String json = "{" + RestBO.createCustomer(vo) + "}";
        System.out.print("-----RS SERVICES 1-----" + json);
    }
}
```

```

        return Response.status(200).type("application/json").entity(json).build();
    }

    @PUT
    @Produces("text/json")
    @Path("Update/{userName}{path:.*}")
    @Consumes("application/x-www-form-urlencoded")
    public Response updateCustomer(@PathParam("name") String name,@FormParam("profession") String
profession,@FormParam("city") String city) {
        CustomerVO vo = new CustomerVO(name, city, profession);
        System.out.print("-----" + vo);
        String json = "{" + RestBO.updateCustomer(vo)+";" + vo.toString() + "}";
        return Response.status(200).type("application/json").entity(json).build();
    }

    @POST
    @Produces("text/json")
    @Path("Delete/{userName}")
    public Response deleteCustomer(@PathParam("userName") String name) {
        String json = "{" + RestBO.deleteCustomer(name) + "}";
        return Response.status(200).type("application/json").entity(json).build();
    }

    @GET
    @Produces("text/json")
    @Path("Find/{userName}{path:.*}")
    public Response findCustomer(@PathParam("userName") String name) {
        System.out.print("-----" + name);
        CustomerVO vo = RestBO.findCustomer(name);
        String json = "{ 'user' : { 'name' : '" + name + "' , 'city' :'" + vo.getCity() + "' , 'Profession' :'" +
vo.getProfession() + "' }}";
        System.out.print("-----" + json);
        return Response.status(200).type("application/json").entity(json).build();
    }
}

```

Finally let us create the web application context listener class that will help to capture bundle context in our application in case we need to refer from our code.

```
package com.hp.osgi.rest.producer;

import java.util.HashMap;
import javax.servlet.ServletContextEvent;
import javax.servlet.ServletContextListener;
import org.osgi.framework.BundleActivator;
import org.osgi.framework.BundleContext;
import org.osgi.framework.ServiceReference;
import org.osgi.service.event.Event;
import org.osgi.service.event.EventAdmin;

public class WebApplicationContextListener implements BundleActivator, ServletContextListener {

    static EventAdmin ea;

    BundleContext bc;
    ServiceReference eaRef;

    synchronized static EventAdmin getEa() {
        return ea;
    }

    synchronized static void setEa(EventAdmin ea) {
        WebApplicationContextListener.ea = ea;
    }

    public void contextInitialized(final ServletContextEvent sce) {
        if (getEa() != null) {
            getEa().sendEvent(new Event("jersey/test/DEPLOYED", new HashMap<String, String>(){put("context-path",
sce.getServletContext().getContextPath());}));
        }
    }

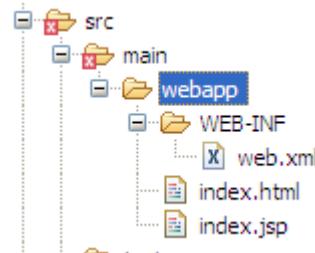
    // @Override
```

```
public void contextDestroyed(final ServletContextEvent sce) {
    if (getEa() != null) {
        getEa().sendEvent(new Event("jersey/test/UNDEPLOYED", new HashMap<String, String>(){put("context-path",
sce.getServletContext().getContextPath());})));
    }
}

// @Override
public void start(BundleContext context) throws Exception {
    bc = context;
    eaRef = bc.getServiceReference(EventAdmin.class.getName());
    if (eaRef != null) {
        setEa((EventAdmin)bc.getService(eaRef));
    }
}

// @Override
public void stop(BundleContext context) throws Exception {
    if (eaRef != null) {
        setEa(null);
        bc.ungetService(eaRef);
    }
}
}
```

In the src folder create the following folders structure:



Update the web.xml, which is mandatory for being a web application. In this file we define the servlet that will serve the http request for our application.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<web-app xmlns="http://java.sun.com/xml/ns/j2ee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
  http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
  version="2.4">
  <display-name>Spring DM web sample</display-name>
  <description>Spring DM web sample</description>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>
  <listener>
    <listener-class>com.hp.osgi.rest.producer.WebAppContextListener</listener-class>
  </listener>

  <servlet>
    <servlet-name>Jersey Web Application</servlet-name>
    <servlet-class>com.sun.jersey.spi.container.servlet.ServletContainer</servlet-class>
    <init-param>
      <param-name>com.sun.jersey.config.property.resourceConfigClass</param-name>
      <param-value>com.sun.jersey.api.core.ClassNamesResourceConfig</param-value>
    </init-param>
    <init-param>
      <param-name>com.sun.jersey.config.property.classnames</param-name>
      <param-
value>com.hp.osgi.rest.producer.HelloWorldResource;com.hp.osgi.rest.producer.AnotherResource</param-value>
    </init-param>
    <init-param>
      <param-name>com.sun.jersey.api.json.POJOMappingFeature</param-name>
      <param-value>true</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
  </servlet>
  <servlet-mapping>
    <servlet-name>Jersey Web Application</servlet-name>
    <url-pattern>/webresources/*</url-pattern>
  </servlet-mapping>
```

```
</web-app>
```

Update the default static page,index.html

```
<html>
<head><title>REST web sample</title></head>
<body>
<h1>Hello REST world REST! (HTML)</h1>
</body>
</html>
```

Update the default dynamic page,index.jsp

```
<html>
<head><title>Dynamic Web Page</title></head>
<body>
<h1><%=" Dynamic Web Page! (JSP) "%></h1>
</body>
</html>
```

Update the pom.xml, where we define all our dependencies related to our project.

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.hp.spring-web-rest</groupId>
  <artifactId>simple-web-rest</artifactId>
  <version>2.0.1</version>
  <packaging>war</packaging>

  <name>simple-web-rest</name>
  <url>http://maven.apache.org</url>

  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <equinox.version>3.5.1.R35x_v20090827</equinox.version>
    <jersey.version>1.12</jersey.version>
  </properties>

  <dependencies>
    <dependency>
      <groupId>com.sun.jersey</groupId>
      <artifactId>jersey-servlet</artifactId>
      <version>${jersey.version}</version>
      <scope>provided</scope>
    </dependency>

    <dependency>
      <groupId>com.sun.jersey</groupId>
      <artifactId>jersey-json</artifactId>
      <version>${jersey.version}</version>
    </dependency>

    <dependency>
      <groupId>junit</groupId>
```

```
<artifactId>junit</artifactId>
<version>3.8.1</version>
<scope>test</scope>
</dependency>

<dependency>
    <groupId>javax.servlet</groupId>
    <artifactId> servlet-api</artifactId>
    <version>2.5</version>
    <scope>provided</scope>
</dependency>

<dependency>
    <groupId>org.osgi</groupId>
    <artifactId>org.osgi.core</artifactId>
    <version>4.2.0</version>
    <scope>provided</scope>
</dependency>
<dependency>
    <groupId>org.osgi</groupId>
    <artifactId>org.osgi.compendium</artifactId>
    <version>4.2.0</version>
    <scope>provided</scope>
</dependency>
<dependency>
    <groupId>org.apache.felix</groupId>
    <artifactId>org.apache.felix.http.bundle</artifactId>
    <version>2.0.4</version>
</dependency>

</dependencies>

<build>
    <plugins>

        <!-- to generate the MANIFEST-FILE required by the bundle -->
        <plugin>
            <groupId>org.apache.felix</groupId>
            <artifactId>maven-bundle-plugin</artifactId>
```

```

<version>2.5.4</version>
<extensions>true</extensions>
<executions>
    <execution>
        <id>bundle-manifest</id>
        <phase>process-classes</phase>
        <goals>
            <goal>manifest</goal>
        </goals>
    </execution>
</executions>
<configuration>
    <manifestLocation>${project.build.directory}/META-INF</manifestLocation>
    <supportedProjectTypes>
        <supportedProjectType>bundle</supportedProjectType>
        <supportedProjectType>war</supportedProjectType>
    </supportedProjectTypes>
    <instructions>
        <Import-
Package>com.sun.jersey.api.core,com.sun.jersey.spi.container.servlet,*</Import-Package>
        <Include-Resource>src/main/webapp</Include-Resource>
        <Webapp-Context>Restworld</Webapp-Context>
        <Web-ContextPath>Restworld</Web-ContextPath>
        <Bundle-Activator>com.hp.osgi.rest.producer.WebAppContextListener</Bundle-
Activator>
    </instructions>
    </configuration>
</plugin>

<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-war-plugin</artifactId>
    <configuration>
        <attachClasses>true</attachClasses>
        <archive>
            <manifestFile>${project.build.directory}/META-INF/MANIFEST.MF</manifestFile>
            <manifestEntries>
                <Bundle-ClassPath>WEB-INF/classes</Bundle-ClassPath>
            </manifestEntries>
        </archive>
    </configuration>
</plugin>

```

```
        </archive>
    </configuration>
</plugin>

<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-compiler-plugin</artifactId>
    <version>2.1</version>
    <configuration>
        <source>1.5</source>
        <target>1.5</target>
    </configuration>
</plugin>
</plugins>
</build>

<repositories>
    <repository>
        <id>com.springsource.repository.bundles.release</id>
        <name>SpringSource EBR - SpringSource Bundle Releases</name>
        <url>http://repository.springsource.com/maven/bundles/release</url>
        <releases>
            <enabled>true</enabled>
            <updatePolicy>always</updatePolicy>
        </releases>
        <snapshots>
            <enabled>true</enabled>
            <updatePolicy>always</updatePolicy>
        </snapshots>
    </repository>
    <repository>
        <id>com.springsource.repository.bundles.external</id>
        <name>SpringSource EBR - External Bundle Releases</name>
        <url>http://repository.springsource.com/maven/bundles/external</url>
        <releases>
            <enabled>true</enabled>
            <updatePolicy>always</updatePolicy>
        </releases>
```

```
<snapshots>
    <enabled>true</enabled>
    <updatePolicy>always</updatePolicy>
</snapshots>
</repository>

<repository>
    <id>spring-maven-milestone</id>
    <name>Springframework Maven Repository</name>
    <releases>
        <enabled>true</enabled>
        <updatePolicy>always</updatePolicy>
    </releases>
    <snapshots>
        <enabled>true</enabled>
        <updatePolicy>always</updatePolicy>
    </snapshots>
    <url>http://s3.amazonaws.com/maven.springframework.org/milestone</url>
</repository>

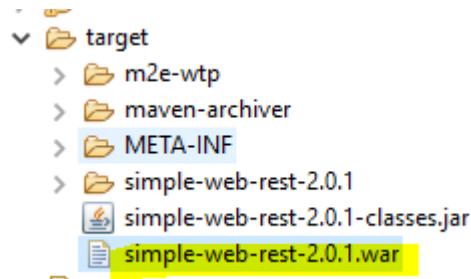
<repository>
    <id>maven2-repository.dev.java.net</id>
    <name>Java.net Repository for Maven</name>
    <url>http://download.java.net/maven/2/</url>
    <releases>
        <enabled>true</enabled>
        <updatePolicy>always</updatePolicy>
    </releases>
    <snapshots>
        <enabled>true</enabled>
        <updatePolicy>always</updatePolicy>
    </snapshots>
    <layout>default</layout>
</repository>

<repository>
    <id>jersey-bundle</id>
    <releases>
        <enabled>true</enabled>
```

```
        <updatePolicy>always</updatePolicy>
    </releases>
    <snapshots>
        <enabled>true</enabled>
        <updatePolicy>always</updatePolicy>
    </snapshots>
    <name>Springframework Maven OSGified Artifacts Repository</name>
    <url>http://maven.java.net</url>
</repository>

</repositories>
</project>
```

You can execute maven goals; Clean -> install – Maven and deploy the war using bundle install command. You will have the war file in the target folder, copy that in a folder and install it by specifying the file path.



Deploy the war to virgo.

- install jersey-core-1.12.jar
- jersey-server-1.12.jar
- jersey-servlet-1.12.jar
- jsr311-api-1.1.1.jar

Deploy the war to virgo.

Test The Following URL:

<http://localhost/Restworld/webresources/another/time>

<http://localhost/Restworld/webresources/RestWorld>

<http://localhost/Restworld/webresources/RestWorld/Default>

<http://localhost/Restworld/webresources/RestWorld/Find/henry>

Deploy in equinox:

Following bundles need to be install:

equinox/org.eclipse.equinox.common_3.6.0.v20100503.jar
equinox/org.osgi.compendium-1.4.0.jar

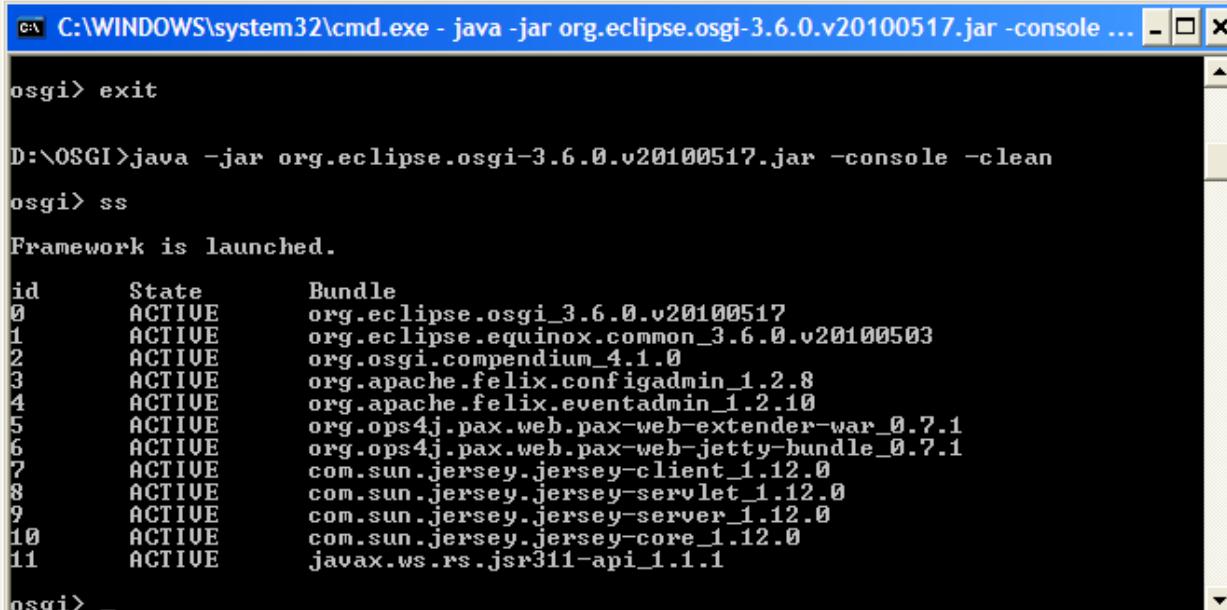
Bundles in apacheConfig folder

org.apache.felix.configadmin-1.2.8.jar
org.apache.felix.eventadmin-1.2.10.jar
pax-web-extender-war-0.7.1.jar
pax-web-jetty-bundle-0.7.1.jar

Bundles in Jersey1.2

- jersey-core-1.12.jar
- jersey-server-1.12.jar
- jersey-servlet-1.12.jar
- jsr311-api-1.1.1.jar

install file:userbundles/simple-web-rest-2.0.1.war



The screenshot shows a Windows command prompt window titled 'C:\WINDOWS\system32\cmd.exe - java -jar org.eclipse.osgi-3.6.0.v20100517.jar -console ...'. The window contains the following OSGI console session:

```
osgi> exit  
  
D:\OSGI>java -jar org.eclipse.osgi-3.6.0.v20100517.jar -console -clean  
osgi> ss  
Framework is launched.  


| id | State  | Bundle                                       |
|----|--------|----------------------------------------------|
| 0  | ACTIVE | org.eclipse.osgi_3.6.0.v20100517             |
| 1  | ACTIVE | org.eclipse.equinox.common_3.6.0.v20100503   |
| 2  | ACTIVE | org.osgi.compendium_4.1.0                    |
| 3  | ACTIVE | org.apache.felix.configadmin_1.2.8           |
| 4  | ACTIVE | org.apache.felix.eventadmin_1.2.10           |
| 5  | ACTIVE | org.ops4j.pax.web.pax-web-extender-war_0.7.1 |
| 6  | ACTIVE | org.ops4j.pax.web.pax-web-jetty-bundle_0.7.1 |
| 7  | ACTIVE | com.sun.jersey.jersey-client_1.12.0          |
| 8  | ACTIVE | com.sun.jersey.jersey-servlet_1.12.0         |
| 9  | ACTIVE | com.sun.jersey.jersey-server_1.12.0          |
| 10 | ACTIVE | com.sun.jersey.jersey-core_1.12.0            |
| 11 | ACTIVE | javax.ws.rs.jsr311-api_1.1.1                 |

  
osgi>
```

<http://localhost:8080/Restworld/webresources/another/time>

<http://localhost:8080/Restworld/webresources/RestWorld>

<http://localhost:8080/Restworld/webresources/RestWorld/Default>

<http://localhost:8080/Restworld/webresources/RestWorld/Find/henry>

Deploy in Karaf

Before you install your REST Bundle, you need to install the jersey runtime provider. For this your required library to execute the Jersey apis. You can install the three requires bundles as shown below:

```
#bundle:install -s mvn:com.sun.jersey/jersey-core/1.18.1
#bundle:install -s mvn:com.sun.jersey/jersey-server/1.18.1
#bundle:install -s mvn:com.sun.jersey/jersey-servlet/1.18.1
```

```
karaf@root()>
karaf@root()> bundle:install -s mvn:com.sun.jersey/jersey-core/1.18.1
Bundle ID: 154
karaf@root()> bundle:install -s mvn:com.sun.jersey/jersey-server/1.18.1
Bundle ID: 155
karaf@root()> bundle:install -s mvn:com.sun.jersey/jersey-servlet/1.18.1
Bundle ID: 156
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
   ID | State      | Lvl | Version           | Name
-----+-----+-----+-----+-----+
   66 | Installed  | 80  | 1.0.0.201707081832 | Firstbundle
   67 | Active     | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.bundle Bundle
   73 | Resolved   | 80  | 0.0.1.SNAPSHOT    | com.hp.osgi.quote Bundle
   75 | Resolved   | 80  | 1.1.0              | com.hp.osgi.quoteconsumer Bundle
   76 | Installed  | 80  | 1.0.0              | com.hp.osgi.quoteservice Bundle
   83 | Active     | 80  | 1.0.0              | com.hp.osgi.declarative.consumer Bundle
   84 | Installed  | 80  | 1.0.0              | com.hp.osgi.declarative.services Bundle
   88 | Active     | 80  | 1.0.0              | com.hp.osgi.st.ServiceTrackerQuoteConsumer Bundle
  138 | Active     | 80  | 1.0.0              | com.hp.osgi.httpservice Bundle
  153 | Active     | 80  | 2.0.1              | simple-web-rest
  154 | Active     | 80  | 1.18.1             | jersey-core
  155 | Active     | 80  | 1.18.1             | jersey-server
  156 | Active     | 80  | 1.18.1             | jersey-servlet
```

Finally you can list the bundles, using bundle:list and all required bundles should be in Active state as shown above.

```
# bundle:install -s file:///d://temp/simple-web-rest-2.0.1.war
```

```
karaf@root()> bundle:install -s file:///d://temp/simple-web-rest-2.0.1.war
Bundle ID: 162
karaf@root()> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State      | Lvl | Version           | Name
---+-----+-----+-----+-----+
66 | Installed | 80  | 1.0.0.201707081832 | Firstbundle
67 | Active    | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.bundle Bundle
73 | Resolved   | 80  | 0.0.1.SNAPSHOT     | com.hp.osgi.quote Bundle
75 | Resolved   | 80  | 1.1.0               | com.hp.osgi.quoteconsumer Bundle
76 | Installed  | 80  | 1.0.0               | com.hp.osgi.quoteservice Bundle
83 | Active    | 80  | 1.0.0               | com.hp.osgi.declarative.consumer Bundle
84 | Installed  | 80  | 1.0.0               | com.hp.osgi.declarative.services Bundle
88 | Active    | 80  | 1.0.0               | com.hp.osgi.st.ServiceTrackerQuoteConsumer Bundle
138 | Active    | 80  | 1.0.0               | com.hp.osgi.httpservice Bundle
154 | Active    | 80  | 1.18.1              | jersey-core
155 | Active    | 80  | 1.18.1              | jersey-server
156 | Active    | 80  | 1.18.1              | jersey-servlet
162 | Active    | 80  | 2.0.1               | simple-web-rest
karaf@root()
```

Then you can access the URL as shown below, if there is any issue you can verify the log using log:display and verify the dependencies using bundle:tree-show id.

<http://localhost/Restworld/webresources/another/time>



<http://localhost/Restworld/webresources/RestWorld/Default>



<http://localhost/Restworld/webresources/RestWorld/Find/henry>

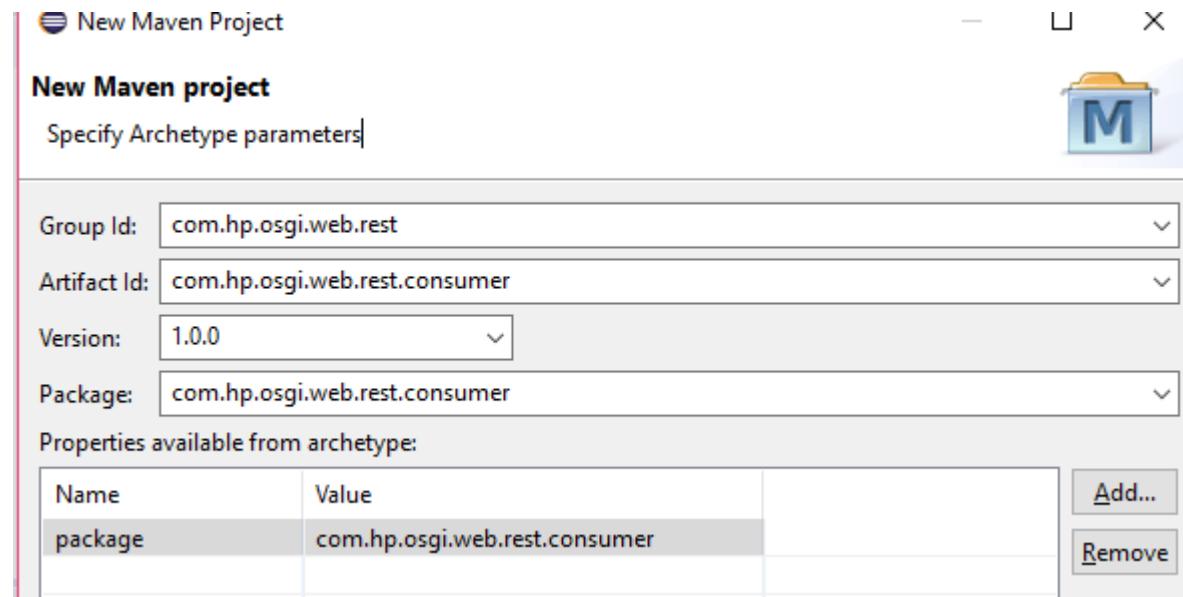


Congratulations, you have exposed web services Using REST API. We will consume this service from another bundle in the next lab.

REST Client – Jersey

In this lab, we are going to consume REST web services, which were exposed by the previous lab.

Create a maven-project archetype karaf – (com.hp.osgi.web.rest.consumer)



Click Finish.

Let us create a business object, that will connect to the REST API web services, which we have created earlier. This will have all the logic to send and receive information between the server and the client bundles.

```
package com.hp.bo;

import java.net.URI;

import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.UriBuilder;

import com.hp.vo.CustomerVO;
import com.sun.jersey.api.client.Client;
import com.sun.jersey.api.client.ClientResponse;
import com.sun.jersey.api.client.WebResource;
import com.sun.jersey.api.client.config.ClientConfig;
import com.sun.jersey.api.client.config.DefaultClientConfig;
import com.sun.jersey.api.representation.Form;

public class ConsumeRESTBO {

    private static URI getBaseURI() {
        return UriBuilder.fromUri("http://localhost/Restworld/webresources/RestWorld").build();
    }

    public String createConsumer(CustomerVO vo){

        ClientConfig config = new DefaultClientConfig();
        Client client = Client.create(config);
        WebResource service = client.resource(getBaseURI());
        Form f = new Form();
        f.add("name", vo.getName());
        f.add("profession", vo.getProfession());
        f.add("city", vo.getCity());
    }
}
```

```
ClientResponse response = service.path("Create").post(ClientResponse.class,f);

if (response.getStatus() != 200) {
    throw new RuntimeException("Failed : HTTP error code : "
        + response.getStatus());
}
String resp = response.getEntity(String.class);
System.out.println("-----ANANNANANn---" + resp);
return resp;
}

public String update(CustomerVO vo){

    ClientConfig config = new DefaultClientConfig();
    Client client = Client.create(config);
    WebResource service = client.resource(getBaseURI());
    Form f = new Form();
    f.add("name", vo.getName());
    f.add("profession", vo.getProfession());
    f.add("city", vo.getCity());

    ClientResponse response = service.path("Update").path(vo.getName()).put(ClientResponse.class,f);

    if (response.getStatus() != 200) {
        throw new RuntimeException("Failed : HTTP error code : "
            + response.getStatus());
    }
    String resp = response.getEntity(String.class);
    System.out.println("-----Updated---" + resp);
    return resp;
}
```

```
public String delete(String name){  
  
    ClientConfig config = new DefaultClientConfig();  
    Client client = Client.create(config);  
    WebResource service = client.resource(getBaseURI());  
    Form f = new Form();  
    f.add("name",""+ name +"");  
  
    ClientResponse response = service.path("Delete").path(name).post(ClientResponse.class,f);  
  
    if (response.getStatus() != 200) {  
        throw new RuntimeException("Failed : HTTP error code : "  
            + response.getStatus());  
    }  
    String resp = response.getEntity(String.class);  
    System.out.println("-----Deleted---" + resp);  
    return resp;  
}  
  
public String findByName(String name){  
  
    ClientConfig config = new DefaultClientConfig();  
    Client client = Client.create(config);  
    WebResource service = client.resource(getBaseURI());  
    ClientResponse response = service.path("Find").path(name).get(ClientResponse.class);  
  
    if (response.getStatus() != 200) {  
        throw new RuntimeException("Failed : HTTP error code : "  
            + response.getStatus());  
    }  
    String resp = response.getEntity(String.class);  
    System.out.println("-----Find By Name---" + resp);  
    return resp;  
}
```

```
}
```

Next, let us create a servlet that will call our business object, which will be our controller.

```
package com.hp.servlet;

import java.io.IOException;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.hp.bo.ConsumeRESTBO;
import com.hp.vo.CustomerVO;

public class ConsumerServlet extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
        // TODO Auto-generated method stub
        // super.doGet(req, resp);
        process(req, resp);
    }

    @Override
    protected void doPost(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
        // TODO Auto-generated method stub
        // super.doPost(req, resp);
        process(req, resp);
    }
}
```

```

}

void process(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
// TODO Auto-generated method stub
    ConsumeRESTBO bo = new ConsumeRESTBO();

    System.out.println("\n--- name - " + req.getParameter("name"));
    System.out.println("\n--- city - " + req.getParameter("city"));
    CustomerVO vo = new
CustomerVO(req.getParameter("name"),req.getParameter("city"),req.getParameter("Profession"));
    String result = "NA";
    String btn = req.getParameter("btn");
    System.out.println("\n--- Button - " + req.getParameter("btn"));

    if(btn.equalsIgnoreCase("add")){
        result = bo.createConsumer(vo);
    }else if(btn.equalsIgnoreCase("Find")){
        result = bo.findByName(vo.getName());
    } else if(btn.equalsIgnoreCase("Update")){
        result = bo.update(vo);
    } else if(btn.equalsIgnoreCase("Delete")){
        result = bo.delete(vo.getName());
    }

    process(resp, result);
}

protected void process(HttpServletResponse resp, String mesg) throws ServletException, IOException {
    resp.setContentType("text/html");
    resp.getWriter().write("<html><body>" + mesg + "</body></html>"); //NON-NLS-1$
}
}

```

Finally, let us create a Web application context listener that will help to get the bundle context in our web application i.e by expending the bundle Activator:

```
package com.hp.servlet;

import java.util.HashMap;
import javax.servlet.ServletContextEvent;
import javax.servlet.ServletContextListener;
import org.osgi.framework.BundleActivator;
import org.osgi.framework.BundleContext;
import org.osgi.framework.ServiceReference;
import org.osgi.service.event.Event;
import org.osgi.service.event.EventAdmin;

/**
 * @author Jakub.Podlesak@Sun.COM
 */
public class WebAppContextListener implements BundleActivator, ServletContextListener {

    static EventAdmin ea;

    BundleContext bc;
    ServiceReference eaRef;

    synchronized static EventAdmin getEa() {
        return ea;
    }

    synchronized static void setEa(EventAdmin ea) {
        WebAppContextListener.ea = ea;
    }

    public void contextInitialized(final ServletContextEvent sce) {
```

```
if (getEa() != null) {
    getEa().sendEvent(new Event("jersey/client/DEPLOYED",new HashMap<String, String>(){put("context-path",
sce.getServletContext().getContextPath());}));}
}

public void contextDestroyed(final ServletContextEvent sce) {
    if (getEa() != null) {
        getEa().sendEvent(new Event("jersey/client/UNDEPLOYED",new HashMap<String, String>(){put("context-
path", sce.getServletContext().getContextPath());}));}
}

public void start(BundleContext context) throws Exception {
    bc = context;
    eaRef = bc.getServiceReference(EventAdmin.class.getName());
    if (eaRef != null) {
        setEa((EventAdmin)bc.getService(eaRef));
    }
}

public void stop(BundleContext context) throws Exception {
    if (eaRef != null) {
        setEa(null);
        bc.ungetService(eaRef);
    }
}
```

Let us create Value Object that will be use to transfer data across the layer.

```
package com.hp.vo;

public class CustomerVO {

    private String name;
    private String city;
    private String profession;

    public CustomerVO() {
        super();
    }

    public CustomerVO(String name, String city, String profession) {
        super();
        this.name = name;
        this.city = city;
        this.profession = profession;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getCity() {
        return city;
    }

    public void setCity(String city) {
        this.city = city;
    }

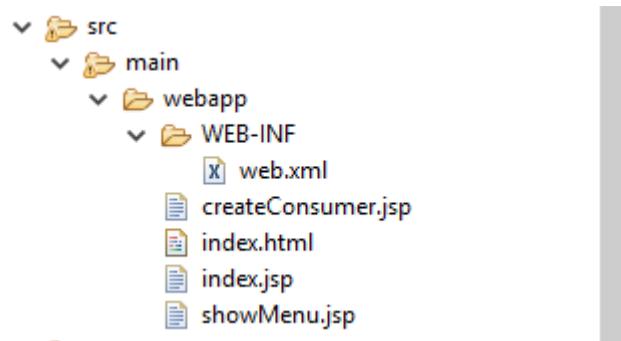
    public String getProfession() {
        return profession;
    }

    public void setProfession(String profession) {
        this.profession = profession;
    }
}
```

```
}
```

```
}
```

Create the following folders structure according to the norm of web application:



Update the web.xml with the following content.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<web-app xmlns="http://java.sun.com/xml/ns/j2ee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
  version="2.4">
  <display-name> ConsumerServlet </display-name>
  <description> ConsumerServlet </description>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>
  <servlet>
    <servlet-name>ConsumerServlet</servlet-name>
    <servlet-class>com.hp.servlet.ConsumerServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>ConsumerServlet</servlet-name>
    <url-pattern>/Consumer</url-pattern>
  </servlet-mapping>
</web-app>
```

Update createConsumer.jsp with the following code. This will be our UI.

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
<script type="text/javascript">
    function submit(val)
    {
        alert(val);
        frm.btn1.value=val ;
        frm.submit();
        return true;
    }
</script>
</head>
<body>
<form action="Consumer" method="post" name="frm">
<table>
    <tr>
        <td>Name</td> <td> <input type="text" id="name" name="name">
    </tr>
    <tr>
        <td>City</td> <td> <input type="text" id="city" name="city">
    </tr>
    <tr>
        <td>Profession</td> <td> <input type="text" id="profession" name="profession">
    </tr>
    <tr>
        <td colspan="2">
            <input type="hidden" name="btn1" value="0">
            <input type="submit" name="btn" value="Add" > &nbsp;&nbsp;
            <input type="submit" name="btn" value="Update" > &nbsp;&nbsp;
            <input type="submit" name="btn" value="Delete" >&nbsp;&nbsp;
            <input type="submit" name="btn" value="Find" >
        </td>
    </tr>
</table>
</form>

```

```
</table>
</form>

</body>
</html>
```

Update our Pom.xml which contains all the dependencies require for the bundle.

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.hp.osgi.web.rest</groupId>
  <artifactId>com.hp.osgi.web.rest.consumer</artifactId>
  <version>1.0.0</version>
  <packaging>bundle</packaging>

  <name>simple-web</name>
  <url>http://maven.apache.org</url>

  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

    <jersey.version>1.18.1</jersey.version>
  </properties>

  <dependencies>
    <dependency>
      <groupId>com.sun.jersey</groupId>
      <artifactId>jersey-client</artifactId>
      <version>${jersey.version}</version>
    </dependency>
    <dependency>
      <groupId>com.sun.jersey</groupId>
      <artifactId>jersey-servlet</artifactId>
```

```
<version>${jersey.version}</version>
<scope>provided</scope>
</dependency>

<dependency>
    <groupId>com.sun.jersey</groupId>
    <artifactId>jersey-json</artifactId>
    <version>${jersey.version}</version>
</dependency>

<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>3.8.1</version>
    <scope>test</scope>
</dependency>

<dependency>
    <groupId>javax.servlet</groupId>
    <artifactId>servlet-api</artifactId>
    <version>2.5</version>
    <scope>provided</scope>
</dependency>

<dependency>
    <groupId>org.osgi</groupId>
    <artifactId>org.osgi.core</artifactId>
    <version>4.2.0</version>
    <scope>provided</scope>
</dependency>
<dependency>
    <groupId>org.osgi</groupId>
    <artifactId>org.osgi.compendium</artifactId>
    <version>4.2.0</version>
    <scope>provided</scope>
</dependency>
<dependency>
    <groupId>org.apache.felix</groupId>
    <artifactId>org.apache.felix.http.bundle</artifactId>
```

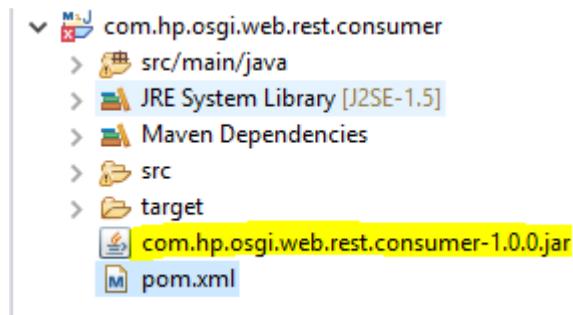
```
<version>2.0.4</version>
</dependency>

</dependencies>
<build>
    <plugins>
        <plugin>
            <groupId>org.apache.felix</groupId>
            <artifactId>maven-bundle-plugin</artifactId>
            <version>2.5.4</version>
            <extensions>true</extensions>
            <configuration>
                <instructions>
                    <Import-Package>*</Import-Package>
                    <Include-Resource>src/main/webapp</Include-Resource>
                    <Web-ContextPath>rest-consumer</Web-ContextPath>
                    <Webapp-Context>rest-consumer</Webapp-Context>
                    <Bundle-Activator>com.hp.servlet.WebAppContextListener</Bundle-Activator>
                </instructions>
            </configuration>
        </plugin>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-compiler-plugin</artifactId>
            <version>2.1</version>
            <configuration>
                <source>1.5</source>
                <target>1.5</target>
            </configuration>
        </plugin>
    </plugins>
</build>

</project>
```

Clean the pom.xml

Install the pom.xml and deploy the bundle in the container.



Install bundle:- jersey-client-1.18.1 in the karaf before deploying the consumer bundle:

```
#bundle:install -s mvn:com.sun.jersey/jersey-client/1.18.1
```

```
162 | Active   | 80 | 2.0.1           | simple-web-rest
karaf@root()> bundle:install -s mvn:com.sun.jersey/jersey-client/1.18.1
Bundle ID: 163
karaf@root()> bundle:list | grep jersey-client
163 | Active   | 80 | 1.18.1           | jersey-client
karaf@root()>
```

Deploy our consumer bundle in the container and test as follow:

```
#bundle:install -s file:///d://temp/plugins/com.hp.osgi.web.rest.consumer-1.0.0.jar
```

```
162 | Active   | 80 | 2.0.1           | jersey-client
163 | Active   | 80 | 1.18.1           | simple-web-rest
jersey-client
164 | Active   | 80 | 1.0.0           | simple-web
karaf@root()>
```

Log will display the following when the consumer is installed properly.

```
2017-07-15 00:37:52,553 | INFO  | ender-2-thread-1 | HttpServiceContext      | 128 - org.ops4j.pax.web.pax-web-jetty - 4.3.0 | registering JasperInitializer
2017-07-15 00:37:52,557 | INFO  | ender-2-thread-1 | ContextHandler           | 116 - org.eclipse.jetty.util - 9.2.19.v20160908 | Started HttpServiceContext{httpContext=WebAppHttpContext{com.hp.osgi.web.rest.consumer - 164}}
karaf@root()> log:display
```

Access the Client as follows:

<http://localhost/rest-consumer/createConsumer.jsp>

The screenshot shows a Microsoft Internet Explorer window with the title "Insert title here - Microsoft Internet Explorer". The address bar contains the URL "http://localhost:8081/rest-consumer/createConsumer.jsp". The page displays a form with three input fields: "Name" (Henry Pot), "City" (Mumbai), and "Profession" (IT Trainer). Below the form are four buttons: "Add" (highlighted in blue), "Update", "Delete", and "Find".

| | |
|------------|------------|
| Name | Henry Pot |
| City | Mumbai |
| Profession | IT Trainer |

Add Update Delete Find

If you click “Add”



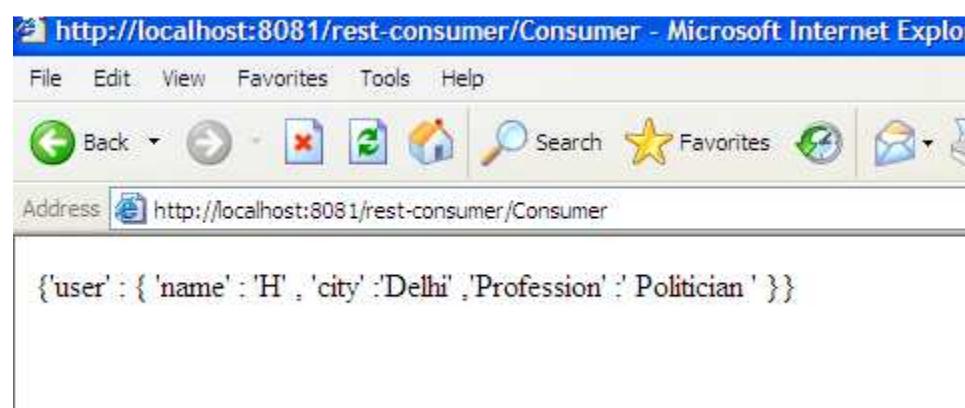
If you click “Update”



If you click “Delete”



If you click “Find”



For deploying in the equinox:

Following Bundles need to be installed:

```
osgi> ss
Framework is launched.

id      State    Bundle
0      ACTIVE   org.eclipse.osgi_3.6.0.v20100517
1      ACTIVE   org.eclipse.equinox.common_3.6.0.v20100503
2      ACTIVE   org.osgi.compendium_4.1.0
3      ACTIVE   org.apache.felix.configadmin_1.2.8
4      ACTIVE   org.apache.felix.eventadmin_1.2.10
5      ACTIVE   org.ops4j.pax.web.pax-web-extender-war_0.7.1
6      ACTIVE   org.ops4j.pax.web.pax-web-jetty-bundle_0.7.1
7      ACTIVE   com.sun.jersey.jersey-client_1.12.0
8      ACTIVE   com.sun.jersey.jersey-servlet_1.12.0
9      ACTIVE   com.sun.jersey.jersey-server_1.12.0
10     ACTIVE   com.sun.jersey.jersey-core_1.12.0
11     ACTIVE   javax.ws.rs.jsr311-api_1.1.1
12     ACTIVE   com.hp.spring-web-rest.simple-web-rest_2.0.1
16     ACTIVE   com.hp.simple.consumer.simple-web-rest-consumer_1.0.1.SNAPSHOT

osgi>
```

Console:

```
C:\WINDOWS\system32\cmd.exe - java -jar org.eclipse.osgi-3.6.0.v20100517.jar -console ...
9 ACTIVE com.sun.jersey.jersey-server_1.12.0
10 ACTIVE com.sun.jersey.jersey-core_1.12.0
11 ACTIVE javax.ws.rs.jsr311-api_1.1.1
12 ACTIVE com.hp.spring-web-rest.simple-web-rest_2.0.1

osgi> install file:D:/OSGI/userBundles/simple-web-rest-consumer-1.0.1-SNAPSHOT.j
ar
Bundle id is 16

osgi> start 16

osgi>
---- name = Henry
---- city = Mumbai
---- Button - Add
---- RS SERVICES-----Henry -----RS SERVICES 1-----{'Successfully crea
ted, user Henry'}-----ANANNANANn-----{'Successfully created, user Henry'}
---- name = Henry
---- city = Mumbai
---- Button - Update
---- null - Mumbai - -----Updated---{'Successfully updated, user nul
l;null - Mumbai - '}
---- name = Henry
---- city = Mumbai
---- Button - Delete
---- Deleted---{'Successfully deleted, user Henry'}
---- name = Henry
---- city = Mumbai
---- Button - Find
---- H-----{'user' : { 'name' : 'H', 'city' : 'Delhi', 'Professi
on' : ' Politician ' }}-----Find By Name---{'user' : { 'name' : 'H', 'city'
: 'Delhi', 'Profession' : ' Politician ' }}
---- name = dscasdc
---- city = Mumbai
---- Button - Find
```

Appendix:

```
osgi.bundles=equinox/org.eclipse.equinox.common_3.6.0.v20100503.jar@start,\n    equinox/org.osgi.compendium-1.4.0.jar@start,\n    apacheConfig/org.apache.felix.configadmin-1.2.8.jar@start,\n    apacheConfig/org.apache.felix.eventadmin-1.2.10.jar@start,\n    Jersey1.2/pax-web-extender-war-0.7.1.jar@start,\n    Jersey1.2/pax-web-jetty-bundle-0.7.1.jar@start,\n    Jersey1.2/jersey-client-1.12.jar@start,\n    Jersey1.2/jersey-servlet-1.12.jar@start,\n    Jersey1.2/jersey-server-1.12.jar@start,\n    Jersey1.2/jersey-core-1.12.jar@start,\nJersey1.2/jsr311-api-1.1.1.jar@start
```

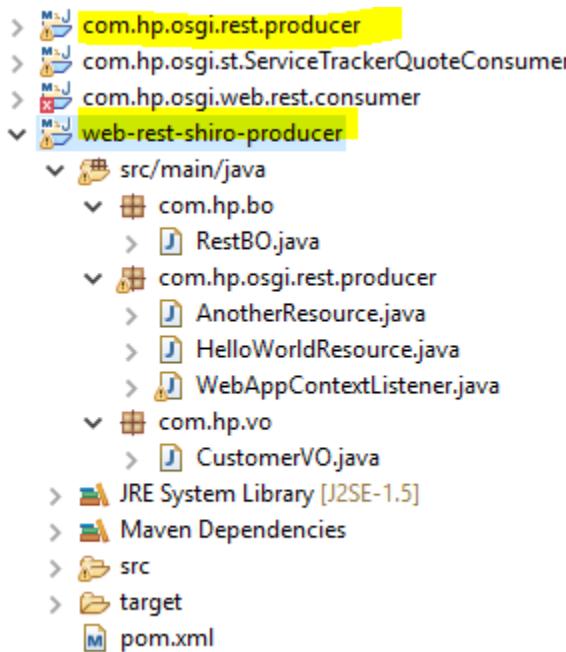
```
java -jar org.eclipse.osgi-3.6.0.v20100517.jar -console
```

```
install file:D:/OSGI/userBundles/simple-web-rest-consumer-1.0.1-SNAPSHOT.jar
```

Enabling REST API Security with Shiro

In this lab we will enable security to REST web services, we will use the earlier REST web services to enable security using Shiro framework.

Copy the com.hp.osgi.rest.producer project by right click on the project → copy → Paste. Name the new project as “web-rest-shiro-producer”. At the end of this step, you should have project view as shown below:



Update pom.xml to include the following in it.

- Artifact Id of the Project so that it doesn't conflict with the earlier project

```
<groupId>com.hp.spring-web-rest</groupId>
<artifactId>shiro-web-rest</artifactId>
<version>2.0.1</version>
<packaging>war</packaging>

<name>shiro-web-rest</name>
```

- Include shiro dependency

```
<!-- START FOR SHIRO -->
<dependency>
    <groupId>org.apache.shiro</groupId>
    <artifactId>shiro-core</artifactId>
    <version>1.1.0</version>
</dependency>
<dependency>
    <groupId>org.apache.shiro</groupId>
    <artifactId>shiro-web</artifactId>
    <version>1.1.0</version>
</dependency>
<!-- END FOR SHIRO -->
```

- Import jersey packages
- Update webcontext with SecureWorld

```

</executions>
<configuration>
    <manifestLocation>${project.build.directory}/META-INF</manifestLocation>
    <supportedProjectTypes>
        <supportedProjectType>bundle</supportedProjectType>
        <supportedProjectType>war</supportedProjectType>
    </supportedProjectTypes>
    <instructions>
        <Import-Package>com.sun.jersey.api.core,
            com.sun.jersey.spi.container.servlet,javax.servlet;
            version="2.5",javax.servlet.http;version="2.5",
            javax.ws.rs,javax.ws.rs.core,org.osgi.framework;version="1.5",org.osgi.service.event;version="1.2"<
        <Exclude-Package>org.apache.*</Exclude-Package>
        <Include-Resource>src/main/webapp</Include-Resource>
        <Webapp-Context>SecureWorld</Webapp-Context>
        <Web-ContextPath>SecureWorld</Web-ContextPath>
        <Bundle-Activator>com.hp.osgi.rest.producer.WebApplicationContextListener</Bundle-Activator>
    </instructions>
</configuration>

```

- Include shiro jar from the bundle classpath.

```

18<configuration>
19    <attachClasses>true</attachClasses>
20    <archive>
21        <manifestFile>${project.build.directory}/META-INF/MANIFEST.MF</manifestFile>
22        <manifestEntries>
23            <Bundle-ClassPath>WEB-INF/classes,WEB-INF/lib/shiro-all-1.2.0.jar,WEB-INF/lib/slf4j-api-1.6.1.jar</Bundle-ClassPath>
24        </manifestEntries>
25    </archive>
26</configuration>
27<gin>
28
29<in>

```

You can simply copy the following in the existing pom.xml instead of typing.

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.hp.spring-web-rest</groupId>
  <artifactId>shiro-web-rest</artifactId>
  <version>2.0.1</version>
  <packaging>war</packaging>

  <name>shiro-web-rest</name>
  <url>http://maven.apache.org</url>

  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <jersey.version>1.18.1</jersey.version>
  </properties>

  <dependencies>
    <dependency>
      <groupId>com.sun.jersey</groupId>
      <artifactId>jersey-servlet</artifactId>
      <version>${jersey.version}</version>
      <scope>provided</scope>
    </dependency>
    <dependency>
      <groupId>com.sun.jersey</groupId>
      <artifactId>jersey-json</artifactId>
      <version>${jersey.version}</version>
      <scope>provided</scope>
    </dependency>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>3.8.1</version>
      <scope>test</scope>
    </dependency>
    <dependency>
      <groupId>javax.servlet</groupId>
      <artifactId>servlet-api</artifactId>
```

```
<version>2.5</version>
    <scope>provided</scope>
</dependency>

<dependency>
    <groupId>org.osgi</groupId>
    <artifactId>org.osgi.core</artifactId>
    <version>4.2.0</version>
    <scope>provided</scope>
</dependency>
<dependency>
    <groupId>org.osgi</groupId>
    <artifactId>org.osgi.compendium</artifactId>
    <version>4.2.0</version>
    <scope>provided</scope>
</dependency>
<dependency>
    <groupId>org.apache.felix</groupId>
    <artifactId>org.apache.felix.http.bundle</artifactId>
    <version>2.0.4</version>
    <scope>provided</scope>
</dependency>
<!-- START FOR SHIRO -->
<dependency>
    <groupId>org.apache.shiro</groupId>
    <artifactId>shiro-core</artifactId>
    <version>1.1.0</version>
</dependency>
<dependency>
    <groupId>org.apache.shiro</groupId>
    <artifactId>shiro-web</artifactId>
    <version>1.1.0</version>
</dependency>
<!-- END FOR SHIRO -->
</dependencies>

<build>
    <!-- to generate the MANIFEST-FILE required by the bundle -->
    <plugins>
```

```
<!-- to generate the MANIFEST-FILE required by the bundle -->
<plugin>
    <groupId>org.apache.felix</groupId>
    <artifactId>maven-bundle-plugin</artifactId>
    <version>2.0.0</version>
    <extensions>true</extensions>
    <executions>
        <execution>
            <id>bundle-manifest</id>
            <phase>process-classes</phase>
            <goals>
                <goal>manifest</goal>
            </goals>
        </execution>
    </executions>
    <configuration>
        <manifestLocation>${project.build.directory}/META-INF</manifestLocation>
        <supportedProjectTypes>
            <supportedProjectType>bundle</supportedProjectType>
            <supportedProjectType>war</supportedProjectType>
        </supportedProjectTypes>
        <instructions>
            <Import-Package>com.sun.jersey.api.core,
                com.sun.jersey.spi.container.servlet,javax.servlet;
                version="2.5",javax.servlet.http;version="2.5",
                javax.ws.rs,javax.ws.rs.core,org.osgi.framework;version="1.5",org.osgi.service.event;version="1.2"<!--Import-Package--&gt;
            &lt;Exclude-Package&gt;org.apache.*&lt;/Exclude-Package&gt;
            &lt;Include-Resource&gt;src/main/webapp&lt;/Include-Resource&gt;
            &lt;Webapp-Context&gt;SecureWorld&lt;/Webapp-Context&gt;
            &lt;Web-ContextPath&gt;SecureWorld&lt;/Web-ContextPath&gt;
            &lt;Bundle-Activator&gt;com.hp.osgi.rest.producer.WebAppContextListener&lt;/Bundle-Activator&gt;
        &lt;/instructions&gt;
    &lt;/configuration&gt;
&lt;/plugin&gt;</pre>
```

```
<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-war-plugin</artifactId>
    <configuration>
        <attachClasses>true</attachClasses>
        <archive>
            <manifestFile>${project.build.directory}/META-INF/MANIFEST.MF</manifestFile>
            <manifestEntries>
                <Bundle-ClassPath>WEB-INF/classes,WEB-INF/lib/shiro-all-1.2.0.jar,WEB-
INF/lib/slf4j-api-1.6.1.jar</Bundle-ClassPath>
            </manifestEntries>
        </archive>
    </configuration>
</plugin>

<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-compiler-plugin</artifactId>
    <version>2.1</version>
    <configuration>
        <source>1.5</source>
        <target>1.5</target>
    </configuration>
</plugin>

</plugins>

</build>

<repositories>
    <repository>
        <id>com.springsource.repository.bundles.release</id>
        <name>SpringSource EBR - SpringSource Bundle Releases</name>
        <url>http://repository.springsource.com/maven/bundles/release</url>
        <releases>
            <enabled>true</enabled>
            <updatePolicy>always</updatePolicy>
        </releases>
        <snapshots>
```

```
        <enabled>true</enabled>
        <updatePolicy>always</updatePolicy>
    </snapshots>
</repository>

<repository>
    <id>com.springsource.repository.bundles.external</id>
    <name>SpringSource EBR - External Bundle Releases</name>
    <url>http://repository.springsource.com/maven/bundles/external</url>
    <releases>
        <enabled>true</enabled>
        <updatePolicy>always</updatePolicy>
    </releases>
    <snapshots>
        <enabled>true</enabled>
        <updatePolicy>always</updatePolicy>
    </snapshots>
</repository>

<repository>
    <id>spring-maven-milestone</id>
    <name>Springframework Maven Repository</name>
    <releases>
        <enabled>true</enabled>
        <updatePolicy>always</updatePolicy>
    </releases>
    <snapshots>
        <enabled>true</enabled>
        <updatePolicy>always</updatePolicy>
    </snapshots>
    <url>http://s3.amazonaws.com/maven.springframework.org/milestone</url>
</repository>

<repository>
    <id>maven2-repository.dev.java.net</id>
    <name>Java.net Repository for Maven</name>
    <url>http://download.java.net/maven/2/</url>
    <releases>
        <enabled>true</enabled>
```

```
        <updatePolicy>always</updatePolicy>
    </releases>
    <snapshots>
        <enabled>true</enabled>
        <updatePolicy>always</updatePolicy>
    </snapshots>
    <layout>default</layout>
</repository>

<repository>
    <id>jersey-bundle</id>
    <releases>
        <enabled>true</enabled>
        <updatePolicy>always</updatePolicy>
    </releases>
    <snapshots>
        <enabled>true</enabled>
        <updatePolicy>always</updatePolicy>
    </snapshots>
    <name>Springframework Maven OSGified Artifacts Repository</name>
    <url>http://maven.java.net</url>
</repository>
</repositories>
</project>
```

Add two secure methods in the AnotherResoure class as shown below

```
import org.apache.shiro.SecurityUtils;
import org.apache.shiro.subject.Subject;

@Path("Secure")
public String getAnotherMessageShiro() {
    String state = "Henry";
    if (SecurityUtils.getSubject().hasRole("vip")) {
        state = "authorized";
    } else {
        state = "authenticated";
    }

    return state;
}

@Path("NSecure")
public String getAnotherMessageN() {

    Subject sub = SecurityUtils.getSubject();
    System.out.println(sub.getPrincipal());
    String state;
    state = "Not secure Access: - authorized" + sub;
    return state;
}
```

The screenshot shows the Eclipse IDE interface with the following details:

- Package Explorer:** Shows the project structure with packages like com.hp.osgi.bundle, com.hp.osgi.declarative.consumer, com.hp.osgi.declarative.services, com.hp.osgi.firstbundle, com.hp.osgi.httpservice, com.hp.osgi.quote, com.hp.osgi.quoteconsumer, com.hp.osgi.quoteservice, and com.hp.osgi.rest.producer.
- Java Editor:** Displays the content of `HelloWorldResource.java`. The code includes annotations for REST endpoints and security logic using Shiro:

```
23     format(System.currentTimeMillis());
24 }
25
26 @GET @Produces("text/plain")
27 @Path("Secure")
28 public String getAnotherMessageShiro() {
29     String state = "Henry";
30     if (SecurityUtils.getSubject().hasRole("vip")) {
31         state = "authorized";
32     } else {
33         state = "authenticated";
34     }
35
36     return state;
37 }
38
39 @GET @Produces("text/plain")
40 @Path("NSecure")
41 public String getAnotherMessageN() {
42
43     Subject sub = SecurityUtils.getSubject();
44     System.out.println(sub.getPrincipal());
45     String state;
46     state = "Not secure Access: - authorized" + sub;
47     return state;
48 }
```

Finally, AnotherResource.java file will have the code as listed below

```
package com.hp.osgi.rest.producer;

import java.text.SimpleDateFormat;

import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.Produces;

import org.apache.shiro.SecurityUtils;
import org.apache.shiro.subject.Subject;

@Path("/another")
public class AnotherResource {

    @GET @Produces("text/plain")
    public String getAnotherMessage() {
        return "Another";
    }
    @Path("/time")
    @GET @Produces("text/plain")
    public String getWhatTime() {
        return new SimpleDateFormat("dd/mm/yyyy hh:mm:ss").
            format(System.currentTimeMillis());
    }

    @GET @Produces("text/plain")
    @Path("Secure")
    public String getAnotherMessageShiro() {
        String state ="Henry";
        if (SecurityUtils.getSubject().hasRole("vip")) {
            state = "authorized";
        } else {
            state = "Un authorized";
        }

        return state;
    }
}
```

```
@GET @Produces("text/plain")
@Path("NSecure")
public String getAnotherMessageN() {

    Subject sub = SecurityUtils.getSubject();
    System.out.println(sub.getPrincipal());
    String state;
    state = "Not secure Access: - authorized" + sub;
    return state;
}

}
```

When **secure** path is access with user Henry user having vip role, it will display “authorized”

We will also update HelloWorldResource class to enable security using Shiro framework

You need to include the following packages for the shiro api.

```
import org.apache.shiro.SecurityUtils;  
import org.apache.shiro.subject.Subject;  
import javax.ws.rs.core.Response.Status;
```

Update the findCustomer method with Shiro Security API as shown below; i.e we are allowing to access these methods when only the user has vip role.

```
@GET  
{@Produces("text/json")  
@Path("Find/{userName}{path:.*}")  
public Response findCustomer(@PathParam("userName") String name) {  
  
    if(SecurityUtils.getSubject().hasRole("vip")) {  
        System.out.print("-----" + name);  
        CustomerVO vo = RestBO.findCustomer(name);  
        String json = "{ 'user' : { 'name' : " + name + " , 'city' :'" + vo.getCity() + " , 'Profession' :'" + vo.getProfession()  
        +" }}";  
        System.out.print("-----" + json);  
        return Response.status(200).type("application/json").entity(json).build();  
    } else {  
        return Response.status(Status.FORBIDDEN).build();  
    }  
}
```

Finally the complete code of `HelloWorldResource` java file is as below:

```
package com.hp.osgi.rest.producer;

import javax.ws.rs.Consumes;
import javax.ws.rs.FormParam;
import javax.ws.rs.GET;
import javax.ws.rs.POST;
import javax.ws.rs.PUT;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.core.Response;
import javax.ws.rs.core.Response.Status;

import org.apache.shiro.SecurityUtils;

import com.hp.bo.RestBO;
import com.hp.vo.CustomerVO;

@Path("/ShiroWorld")
public class HelloWorldResource {

    @GET @Produces("text/plain")
    @Path("Default")
    public String getClickedMessage() {
        return "Hello World From REST - JERSEY";
    }

    @POST
    @Produces("text/json")
    // @Consumes("text/json")
    // @Consumes(MediaType.APPLICATION_JSON)
    @Consumes("application/x-www-form-urlencoded")
    @Path("Create")
    public Response createCustomer(@FormParam("name") String name){
        //,@FormParam("city") String city, @FormParam("professional") String pro) {
```

```

//System.out.print("-----" + name);
CustomerVO vo = new CustomerVO(name, "", "");
System.out.print("--- RS SERVICES-----" + vo);
String json = "{" + RestBO.createCustomer(vo) + "}";
System.out.print("----RS SERVICES 1-----" + json);
return Response.status(200).type("application/json").entity(json).build();
}

@PUT
@Produces("text/json")
@Path("Update/{userName}{path:.*}")
@Consumes("application/x-www-form-urlencoded")
public Response updateCustomer(@PathParam("name") String name,@FormParam("profession") String
profession,@FormParam("city") String city) {
    CustomerVO vo = new CustomerVO(name, city, profession);
    System.out.print("-----" + vo);
    String json = "{" + RestBO.updateCustomer(vo)+";"+ vo.toString() +"}";
    return Response.status(200).type("application/json").entity(json).build();
}

@POST
@Produces("text/json")
@Path("Delete/{userName}")
public Response deleteCustomer(@PathParam("userName") String name) {
    String json = "{" + RestBO.deleteCustomer(name) +"}";
    return Response.status(200).type("application/json").entity(json).build();
}

@GET
@Produces("text/json")
@Path("Find/{userName}{path:.*}")
public Response findCustomer(@PathParam("userName") String name) {

    if(SecurityUtils.getSubject().hasRole("vip")) {
        System.out.print("-----" + name);
        CustomerVO vo = RestBO.findCustomer(name);
        String json = "{ 'user' : { 'name' : '" + name + "' , 'city' :'" + vo.getCity() + "' , 'Profession' :'" +
vo.getProfession() + "' }}";
        System.out.print("-----" + json);
    }
}

```

```
        return Response.status(200).type("application/json").entity(json).build();
    } else {
        return Response.status(Status.FORBIDDEN).build();
    }
}
```

Let us create a file name shiro.ini in src/main/resources. Add the following in the file, here we are defining users and roles for our application. You can also fetch such details from LDAP or database. Each line specifies the user name, password and role respectively.

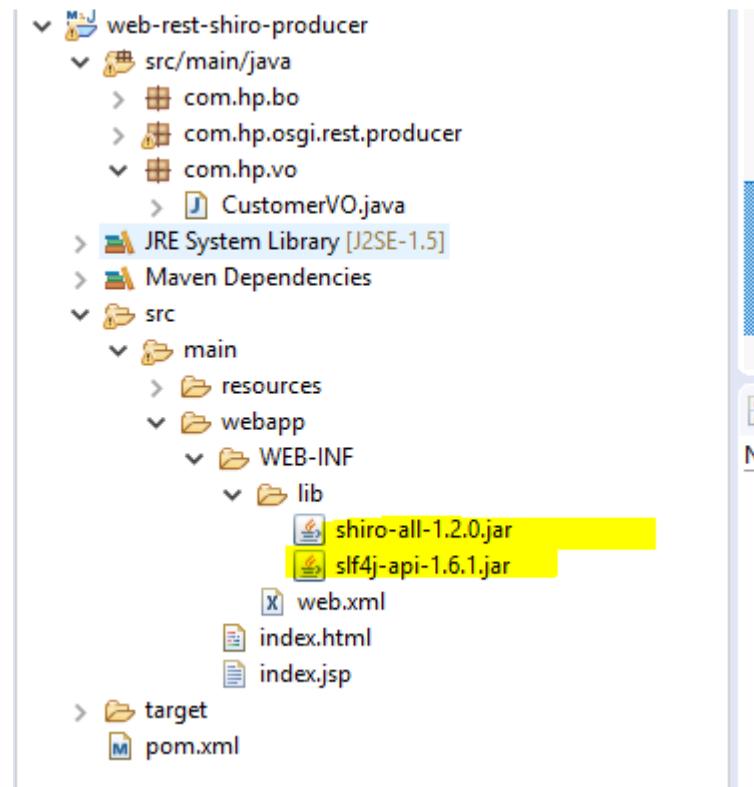
```
[main]
[users]
root = secret,admin
henry = henry,vip
[roles]
admin =
vip =
common =
[url]
/** = authcBasic
```

The screenshot shows the Eclipse IDE interface with the following details:

- Package Explorer:** Shows the project structure. The `src/main/java` folder contains packages like `com.hp.osgi.bundle`, `com.hp.osgi.declarative.consumer`, `com.hp.osgi.declarative.services`, `com.hp.osgi.firstbundle`, `com.hp.osgi.httpservice`, `com.hp.osgi.quote`, `com.hp.osgi.quoteconsumer`, `com.hp.osgi.quoteservice`, and `com.hp.osgi.rest.producer`. It also includes `RestBO.java`, `AnotherResource.java`, and `HelloWorldResource.java` under `com.hp.osgi.rest.producer`.
- Code Editor:** Displays the `shiro.ini` configuration file content:

```
1 [main]
2 [users]
3 root = secret,admin
4 henry = henry,vip,
5 [roles]
6 admin = *
7 vip = *
8 common = *
9 [urls]
10 /** = authcBasic
11 |
```
- Project Navigator:** Shows the `src` folder containing `main` and `webapp`. The `main/resources` folder contains the `shiro.ini` file.
- Bottom Bar:** Includes tabs for `Problems`, `Javadoc`, `Declaration`, `Console`, and `Progress`. The `Progress` tab displays the message: "No operations to display at this time."

Create a folder WEB-INF/lib and dump the two librarys in the lib folder. You can get these library from the software folder, if its not there contact instructor for it.



Update web.xml to include Shiro Filter class so that all requests to this web application are passed through Shiro Servlet.

```
<filter>
    <filter-name>ShiroFilter</filter-name>
    <filter-class>org.apache.shiro.web.servlet.IniShiroFilter</filter-class>
</filter>

<filter-mapping>
    <filter-name>ShiroFilter</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
```



```
3     xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
4     http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
5     version="2.4">
6     <display-name>Spring DM web sample</display-name>
7     <description>Spring DM web sample</description>
8     <welcome-file-list>
9         <welcome-file>index.html</welcome-file>
10    </welcome-file-list>
11    <listener>
12        <listener-class>com.hp.osgi.rest.producer.WebAppContextListener</listener-class>
13    </listener>
14    <filter>
15        <filter-name>ShiroFilter</filter-name>
16        <filter-class>org.apache.shiro.web.servlet.IniShiroFilter</filter-class>
17    </filter>
18    <filter-mapping>
19        <filter-name>ShiroFilter</filter-name>
20        <url-pattern>/*</url-pattern>
21    </filter-mapping>
22
23    <servlet>
24        <servlet-name>Jersey Web Application</servlet-name>
25        <servlet-class>com.sun.jersey.spi.container.servlet.ServletContainer</servlet-class>
26        <init-param>
27            <param-name>com.sun.jersey.config.property.resourceConfigClass</param-name>
28            <param-value>com.sun.jersey.api.core.ClassNamesResourceConfig</param-value>
```

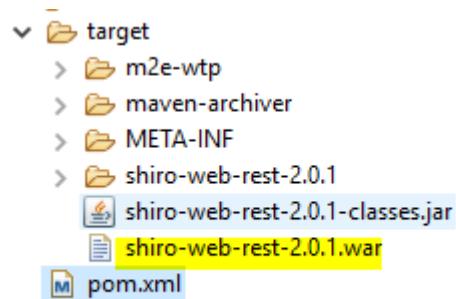
Final web.xml is listed for your reference:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<web-app xmlns="http://java.sun.com/xml/ns/j2ee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
  http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
  version="2.4">
  <display-name>Shiro web sample</display-name>
  <description>Shiro web sample</description>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>
  <listener>
    <listener-class>org.apache.shiro.web.env.EnvironmentLoaderListener</listener-class>
  </listener>
  <filter>
    <filter-name>ShiroFilter</filter-name>
    <filter-class>org.apache.shiro.web.servlet.ShiroFilter</filter-class>
  </filter>
  <filter-mapping>
    <filter-name>ShiroFilter</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>
  <listener>
    <listener-class>com.hp.osgi.rest.producer.WebAppContextListener</listener-class>
  </listener>
  <servlet>
    <servlet-name>Jersey Web Application</servlet-name>
    <servlet-class>com.sun.jersey.spi.container.servlet.ServletContainer</servlet-class>
    <init-param>
      <param-name>com.sun.jersey.config.property.resourceConfigClass</param-name>
      <param-value>com.sun.jersey.api.core.ClassNamesResourceConfig</param-value>
    </init-param>
    <init-param>
      <param-name>com.sun.jersey.config.property.classnames</param-name>
      <param-
value>com.hp.osgi.rest.producer.HelloWorldResource;com.hp.osgi.rest.producer.AnotherResource</param-value>
    </init-param>
    <init-param>
```

```
<param-name>com.sun.jersey.api.json.POJOMappingFeature</param-name>
<param-value>true</param-value>
</init-param>
<load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>Jersey Web Application</servlet-name>
    <url-pattern>/webresources/*</url-pattern>
</servlet-mapping>
</web-app>
```

Now, we have updated all relevant code related to enabling security using Shiro framework. Let us compile, package and install it in karaf container.

Perform maven clean then install. It should generate the war file in the target folder, that needs to be installed in the container.



We need to install shiro bundles in the karaf container. For this command to work, you need to copy the war file in the temp folder.

```
# bundle:install -s file:///d:/temp/shiro-web-rest-2.0.1.war
```

```
163 | Active   | 80 | 1.18.1      | jersey-client
165 | Active   | 80 | 1.0.0       | simple-web
karaf@root(bundle)> bundle:install -s file:///d://temp/shiro-web-rest-2.0.1.war
Bundle ID: 205
karaf@root(bundle)> SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.

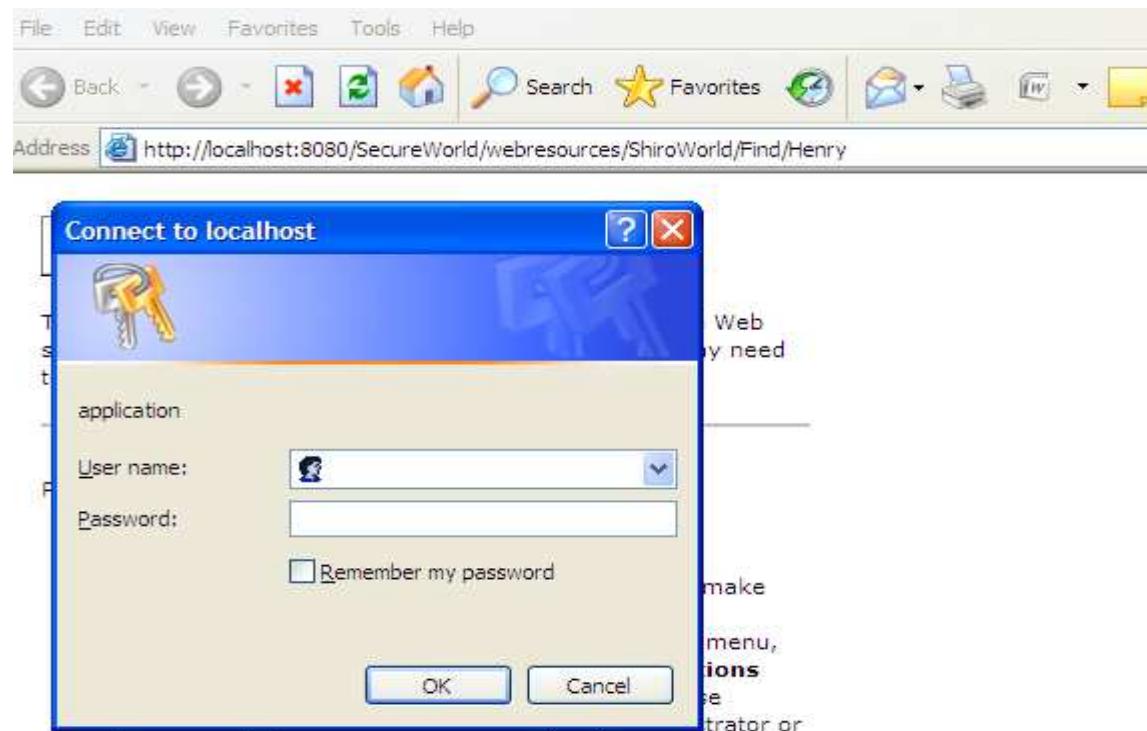
karaf@root(bundle)> list | grep shiro
205 | Active   | 80 | 2.0.1      | shiro-web-rest
karaf@root(bundle)>
```

Once you have installed, the particular bundle will be in the started mode, you can confirm using `list | grep shiro` command.

Let us test the web application now.

Only a role of vip can access the link i.e henry user id

<http://localhost/SecureWorld/webresources/ShiroWorld/Find/henry>



Enter root/secret , It should fail the access as shown below:



HTTP ERROR 403

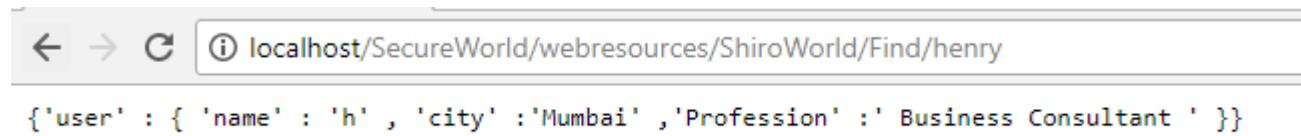
Problem accessing /SecureWorld/webresources/ShiroWorld/Find/Henry. Reason:

Forbidden

Powered by Jetty://

Again access the URL and pass the following credentials:

henry/henry



Try the following URL: (Try with different combination of root/henry logon)

<http://localhost/SecureWorld/webresources/another/time>

<http://localhost/SecureWorld/webresources/another/Secure>

You should get the following text, if you are able to authenticate with the credentials.



Testing – Mock & Pax Exam

We will perform testing of bundle using Pax tools.

Create one maven project using the following details: com.hp.osgi.test

New Maven project

Specify Archetype parameters

Group Id: com.hp.osgi.test

Artifact Id: com.hp.osgi.test.pax

Version: 1.0.0

Package: com.hp.osgi.test

Properties available from archetype:

| Name | Value |
|---------|------------------|
| package | com.hp.osgi.test |

Add... Remove

Click Finish

Update the Activator.java class with the following code.

```
package com.hp.osgi.test;

import org.osgi.framework.BundleActivator;
import org.osgi.framework.BundleContext;
import org.osgi.framework.ServiceReference;
import org.osgi.service.log.LogService;

public class MyMockActivator implements BundleActivator {
    BundleContext m_context;

    public void start(BundleContext bc) throws Exception {
        System.out.println(" Your Maven Bundle Activated!");
        startTestThread();
    }

    public void stop(BundleContext arg0) throws Exception {
        System.out.println(" Your Maven Bundle DeActivated!");
        stopTestThread();
    }

    // Test LogService by periodically sending a message
    class LogServiceTest implements Runnable {
        public void run() {

            while (Thread.currentThread() == m_logTestThread) {

                // lookup the current "best" LogService each time, just before we need to use it
                ServiceReference logServiceRef = m_context.getServiceReference(LogService.class.getName());

                // if the service reference is null then we know there's no log service available
                if (logServiceRef != null) {
                    // because we have a valid reference we know the service is there... or do we?
```

```
        ((LogService) m_context.getService(logServiceRef)).log(LogService.LOG_INFO, "ping");
    } else {
        alternativeLog("LogService has gone");
    }

    pauseTestThread();
}
}

//-----
// The rest of this is just support code, not meant to show any particular best practices
//-----

volatile Thread m_logTestThread;

void startTestThread() {
    // start separate worker thread to run the actual tests, managed by the bundle lifecycle
    m_logTestThread = new Thread(new LogServiceTest(), "LogService Tester");
    m_logTestThread.setDaemon(true);
    m_logTestThread.start();
}

void stopTestThread() {
    // thread should cooperatively shutdown on the next iteration, because field is now null
    Thread testThread = m_logTestThread;
    m_logTestThread = null;
    if (testThread != null) {
        testThread.interrupt();
        try {testThread.join();} catch (InterruptedException e) {}
    }
}

protected void pauseTestThread() {
    try {
        // sleep for a bit
        Thread.sleep(5000);
    } catch (InterruptedException e) {}
}
```

```
}

void alternativeLog(String message) {
    // this provides similar style debug logging output for when the LogService disappears
    String tid = "thread=\"" + Thread.currentThread().getName() + "\"";
    String bid = "bundle= " + m_context.getBundle().getBundleId();
    System.out.println("<--> " + tid + ", " + bid + " : " + message);
}
}
```

Pom.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

  <!-- Licensed to the Apache Software Foundation (ASF) under one or more
      contributor license agreements. See the NOTICE file distributed with this
      work for additional information regarding copyright ownership. The ASF licenses
      this file to you under the Apache License, Version 2.0 (the "License"); you
      may not use this file except in compliance with the License. You may obtain
      a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless
      required by applicable law or agreed to in writing, software distributed
      under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES
      OR CONDITIONS OF ANY KIND, either express or implied. See the License for
      the specific language governing permissions and limitations under the License. -->

  <modelVersion>4.0.0</modelVersion>

  <groupId>com.hp.osgi.test</groupId>
  <artifactId>com.hp.osgi.test.pax</artifactId>
  <version>1.0.0</version>
  <packaging>bundle</packaging>

  <name>com.hp.osgi.test.pax Bundle</name>
  <description>
    com.hp.osgi.test.pax OSGi bundle project.
  </description>

  <properties>
    <maven-bundle-plugin.version>2.5.4</maven-bundle-plugin.version>
    <osgi.version>6.0.0</osgi.version>
  </properties>

  <dependencies>
    <dependency>
      <groupId>org.osgi</groupId>
      <artifactId>org.osgi.core</artifactId>
      <version>${osgi.version}</version>
      <scope>provided</scope>
```

```
</dependency>
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>3.8.1</version>
    <scope>test</scope>
</dependency>
<dependency>
    <groupId>org.apache.felix</groupId>
    <artifactId>org.osgi.core</artifactId>
    <version>1.0.0</version>
</dependency>
<dependency>
    <groupId>org.osgi</groupId>
    <artifactId>org.osgi.compendium</artifactId>
    <version>4.2.0</version>
    <scope>provided</scope>
</dependency>

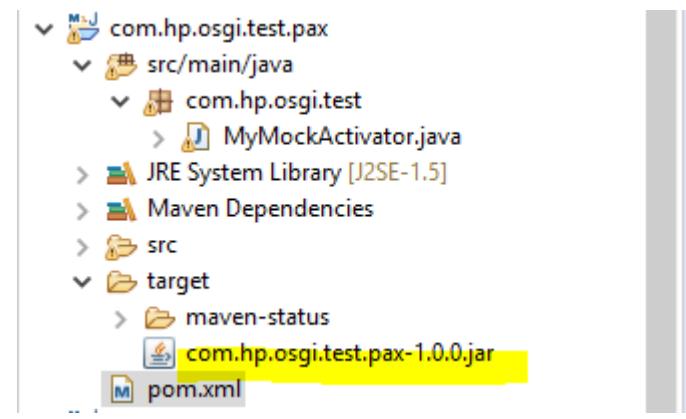
</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.felix</groupId>
            <artifactId>maven-bundle-plugin</artifactId>
            <version>${maven-bundle-plugin.version}</version>
            <extensions>true</extensions>
            <configuration>
                <instructions>
                    <Bundle-SymbolicName>${project.artifactId}</Bundle-SymbolicName>
                    <Bundle-Version>${project.version}</Bundle-Version>
                    <Bundle-Activator>com.hp.osgi.test.MyMockActivator</Bundle-Activator>
                    <Export-Package>
                        com.hp.osgi.test*;version=${project.version}
                    </Export-Package>
                    <Import-Package>
                        org.osgi.framework,
                        *;resolution:=optional,
```

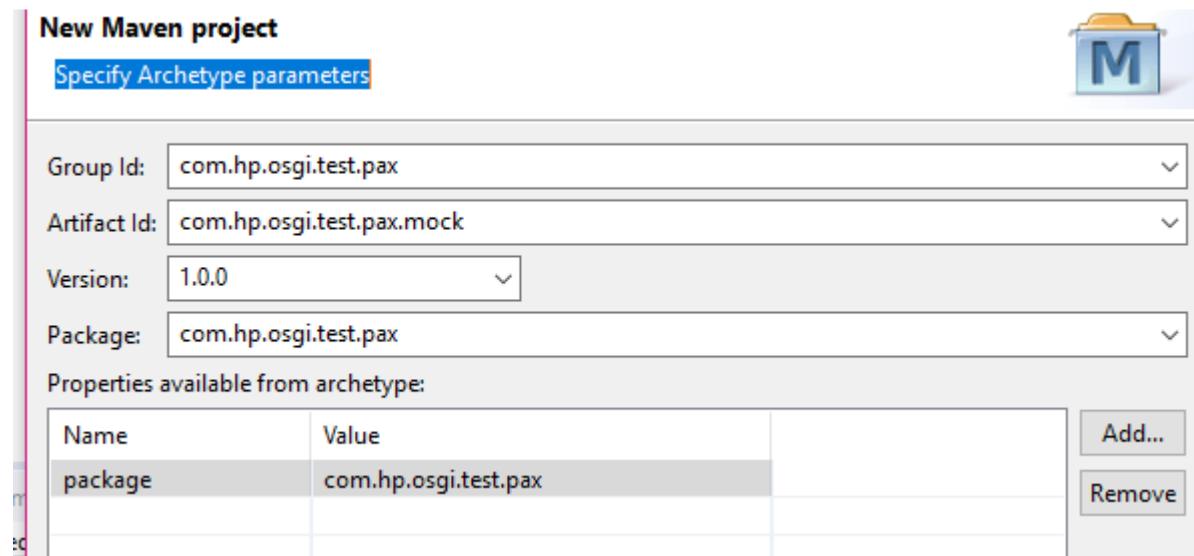
```
        </Import-Package>
    </instructions>
</configuration>
</plugin>
</plugins>
</build>

</project>
```

Compile and install the bundle

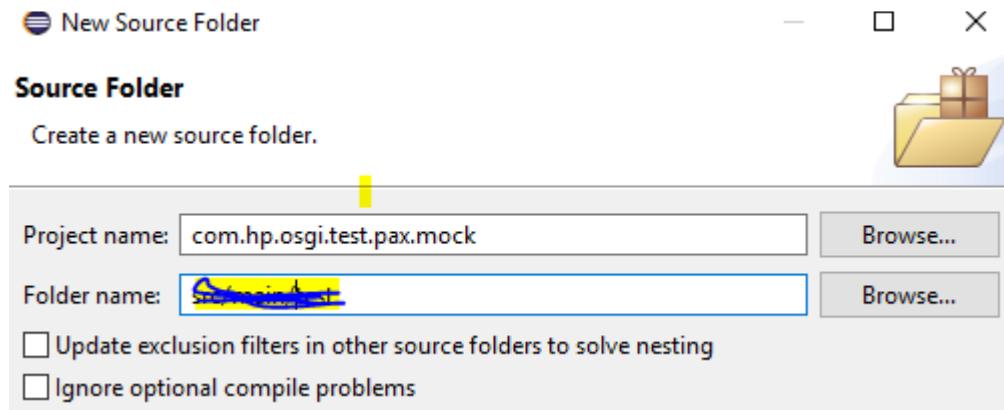


Create an another maven project: com.hp.test.mock , that will be the test bundle for the earlier bundle.



Finish

Create the following Test class in test package/folder [src/test/java]



Finish

Let us two test class for this lab.

Create COnsoleMockTest class as shown below, that will use the mock class for mocking the OSGI container.

```
package com.hp.osgi.test.pax.mock;

import static org.easymock.EasyMock.and;
import static org.easymock.EasyMock.expect;
import static org.easymock.EasyMock.geq;
import static org.easymock.EasyMock.isA;
import static org.easymock.EasyMock.leq;
import static org.easymock.EasyMock.replay;
import static org.easymock.EasyMock.verify;

import org.easymock.EasyMock;
import org.junit.Test;
import org.osgi.framework.Bundle;
import org.osgi.framework.BundleContext;
import org.osgi.framework.ServiceReference;
import org.osgi.service.log.LogService;

import com.hp.osgi.test.MyMockActivator;

public class ContainerMockTest {

    @Test
    public void testLogClientBehaviour()
        throws Exception {

        // MOCK - create prototype mock objects
        // =====

        // we want a strict mock context so we can test the ordering
        BundleContext context = EasyMock.strictMock(BundleContext.class);
        ServiceReference serviceRef = EasyMock.strictMock(ServiceReference.class);
    }
}
```

```
LogService logService = EasyMock.strictMock(LogService.class);

// nice mocks return reasonable defaults
Bundle bundle = EasyMock.strictMock(Bundle.class);

// EXPECT - script the expected behavior
// =====

// expected behaviour when log service is available
expect(context.getServiceReference(LogService.class.getName())).andReturn(serviceRef);
expect(context.getService(serviceRef)).andReturn(logService);
logService.log(and(geq(LogService.LOG_ERROR), leq(LogService.LOG_DEBUG)), isA(String.class));

// REPLAY - prepare the mock objects
// =====

replay(context, serviceRef, logService, bundle);

// TEST - run code using the mock objects
// =====

// override pause method to allow test synchronization
MyMockActivator logClientActivator = new MyMockActivator();

logClientActivator.start(context);
try {
    Thread.sleep(1000);
} catch (InterruptedException e) {}

logClientActivator.stop(context);

// VERIFY - check the behavior matches
// =====

verify(context, serviceRef, logService);
}
```

```
}
```

Create ContainerTest class as shown below, that will use the pax with junit class for unit testing the bundle in the OSGI container.

```
package com.hp.osgi.test.pax.mock;
import static org.ops4j.pax.exam.CoreOptions.mavenBundle;

import java.util.Arrays;
import java.util.Optional;

import javax.inject.Inject;

import org.junit.Assert;
import org.junit.Test;
import org.junit.runner.RunWith;
import org.ops4j.pax.exam.Configuration;
import org.ops4j.pax.exam.Option;
import org.ops4j.pax.exam.junit.PaxExam;
import org.ops4j.pax.exam.karaf.options.KarafDistributionKitConfigurationOption;
import org.ops4j.pax.exam.karaf.options.KarafDistributionKitConfigurationOption.Platform;
import org.ops4j.pax.exam.spi.reactors.ExamReactorStrategy;
import org.ops4j.pax.exam.spi.reactors.PerClass;
import org.osgi.framework.Bundle;
import org.osgi.framework.BundleContext;

@RunWith(PaxExam.class)
@ExamReactorStrategy(PerClass.class)
public class ContainerTest {

    @Inject
    private BundleContext bc;

    @Configuration
```

```

public Option[] config() {

    return new Option[]{
        //mavenBundle().groupId("com.hp.osgi.test").artifactId("com.hp.osgi.test.pax");
        new KarafDistributionKitConfigurationOption("mvn:org.apache.karaf/apache-karaf/4.0.9/zip",
            "karaf", "4.0.9", Platform.WINDOWS).executable("bin\\karaf.bat")
            .filesToMakeExecutable("bin\\admin.bat"),
    }

    mavenBundle().groupId("com.hp.osgi.test").artifactId("com.hp.osgi.test.pax").versionAsInProject().start()

    };
}

@Test
public void testBC() throws Exception {
    Assert.assertNotNull(bc);
}

@Test
public void testExistBundle() throws Exception {

    Optional<Bundle> b = Arrays.asList(bc.getBundles())
        .stream()
        .filter(x ->
x.getSymbolicName().equalsIgnoreCase("com.hp.osgi.test.pax")).findFirst();

    System.out.println("Bundle Context " + bc + " >>> " + b.isPresent());
    Assert.assertTrue(b.isPresent());
}

}

```

pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.hp.osgi.test.pax</groupId>
  <artifactId>com.hp.osgi.test.pax.mock</artifactId>
  <version>1.0.0</version>
  <packaging>jar</packaging>

  <name>com.hp.test.pax</name>
  <url>http://maven.apache.org</url>

  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <junit-version>4.5</junit-version>
    <pax-exam-version>4.7.0</pax-exam-version>
    <osgi-version>4.2.0</osgi-version>
    <jetty-version>0.7.1</jetty-version>
    <slf4j-log4j-binding-impl-version>1.6.4</slf4j-log4j-binding-impl-version>
    <pax.url.aether.version>2.4.1</pax.url.aether.version>
  </properties>

  <dependencies>

    <dependency>
      <groupId>com.hp.osgi.test</groupId>
      <artifactId>com.hp.osgi.test.pax</artifactId>
      <version>1.0.0</version>
      <scope>test</scope>
    </dependency>

    <dependency>
      <groupId>org.slf4j</groupId>
      <artifactId>slf4j-log4j12</artifactId>
      <version>${slf4j-log4j-binding-impl-version}</version>
      <scope>test</scope>
    </dependency>
  
```

```
</dependency>
<!-- https://mvnrepository.com/artifact/org.ops4j.pax.exam/pax-exam-junit-extender-impl -->
<dependency>
    <groupId>org.ops4j.pax.exam</groupId>
    <artifactId>pax-exam-junit-extender-impl</artifactId>
    <version>1.2.4</version>
    <scope>test</scope>
</dependency>

<dependency>
    <groupId>org.ops4j.pax.exam</groupId>
    <artifactId>pax-exam</artifactId>
    <version>${pax-exam-version}</version>
    <scope>test</scope>
</dependency>
<dependency>
    <groupId>org.ops4j.pax.exam</groupId>
    <artifactId>pax-exam-junit4</artifactId>
    <version>${pax-exam-version}</version>
    <scope>test</scope>
</dependency>
<dependency>
    <groupId>org.ops4j.pax.url</groupId>
    <artifactId>pax-url-aether</artifactId>
    <version>${pax.url.aether.version}</version>
    <scope>test</scope>
</dependency>
<dependency>
    <groupId>javax.inject</groupId>
    <artifactId>javax.inject</artifactId>
    <version>1</version>
    <scope>test</scope>
</dependency>
<dependency>
    <groupId>org.ops4j.pax.exam</groupId>
    <artifactId>pax-exam-container-karaf</artifactId>
    <version>${pax-exam-version}</version>
    <scope>test</scope>
</dependency>
```

```
<dependency>
    <groupId>org.ops4j.pax.web</groupId>
    <artifactId>pax-web-jetty-bundle</artifactId>
    <version>${jetty-version}</version>
    <scope>test</scope>
</dependency>
<dependency>
    <groupId>org.ops4j.pax.web</groupId>
    <artifactId>pax-web-extender-war</artifactId>
    <version>${jetty-version}</version>
    <scope>test</scope>
</dependency>

<!-- https://mvnrepository.com/artifact/org.ops4j.pax.exam/pax-exam-junit -->
<dependency>
    <groupId>org.ops4j.pax.exam</groupId>
    <artifactId>pax-exam-junit</artifactId>
    <version>1.2.4</version>
    <scope>test</scope>
</dependency>
<!-- https://mvnrepository.com/artifact/org.apache.geronimo.specs/geronimo-atinject_1.0_spec -->
<dependency>
    <groupId>org.apache.geronimo.specs</groupId>
    <artifactId>geronimo-atinject_1.0_spec</artifactId>
    <version>1.0</version>
    <scope>provided</scope>
</dependency>

<dependency>
    <groupId>org.apache.karaf</groupId>
    <artifactId>apache-karaf</artifactId>
    <version>4.0.0</version>
    <type>tar.gz</type>
    <scope>test</scope>
</dependency>
<dependency>
```

```
<groupId>junit</groupId>
<artifactId>junit</artifactId>
<version>4.11</version>
<scope>test</scope>
</dependency>
<!-- https://mvnrepository.com/artifact/org.apache.servicemix.tooling/depends-maven-plugin -->
<dependency>
    <groupId>org.apache.servicemix.tooling</groupId>
    <artifactId>depends-maven-plugin</artifactId>
    <version>1.4.0</version>
</dependency>

<!-- https://mvnrepository.com/artifact/org.ops4j.pax.runner/pax-runner-platform-equinox -->
<dependency>
    <groupId>org.ops4j.pax.runner</groupId>
    <artifactId>pax-runner-platform-equinox</artifactId>
    <version>1.9.0</version>
</dependency>

<dependency>
    <groupId>org.apache.sling</groupId>
    <artifactId>org.apache.sling.testing.osgi-mock</artifactId>
    <version>2.3.0</version>
</dependency>
<dependency>
    <groupId>org.easymock</groupId>
    <artifactId>easymock</artifactId>
    <version>3.4</version>
    <scope>test</scope>
</dependency>
</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.servicemix.tooling</groupId>
```

```
<artifactId>depends-maven-plugin</artifactId>
<version>1.4.0</version>
<executions>
    <execution>
        <id>generate-dependencies</id>
        <goals>
            <goal>generate-dependencies</goal>
        </goals>
    </execution>
</executions>
</plugin>

<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-compiler-plugin</artifactId>
    <version>3.6.1</version>
    <configuration>
        <source>1.8</source>
        <target>1.8</target>
    </configuration>
    <executions>
        <execution>
            <id>test-compile</id>
            <phase>test-compile</phase>
            <goals>
                <goal>testCompile</goal>
            </goals>
        </execution>
    </executions>
</plugin>
<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-resources-plugin</artifactId>
    <version>3.0.2</version>
    <executions>
        <execution>
            <phase>process-test-resources</phase>
            <goals>
                <goal>testResources</goal>
            </goals>
        </execution>
    </executions>
</plugin>
```

```
        </goals>
    </execution>
</executions>
</plugin>
<!-- to generate the MANIFEST-FILE required by the bundle -->
<plugin>
    <groupId>org.apache.felix</groupId>
    <artifactId>maven-bundle-plugin</artifactId>
    <version>2.0.0</version>
    <extensions>true</extensions>
</plugin>
</plugins>
</build>

</project>
```

At this point, you can perform the maven test.

TESTS

```
Running com.hp.osgi.test.pax.mock.ContainerMockTest
Your Maven Bundle Activated!
Your Maven Bundle DeActivated!
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 1.235 sec
```

Test :

- Install the com.hp.osgi.bundle using pom.xml
- Run the test in com.hp.test.mock, pom.xml

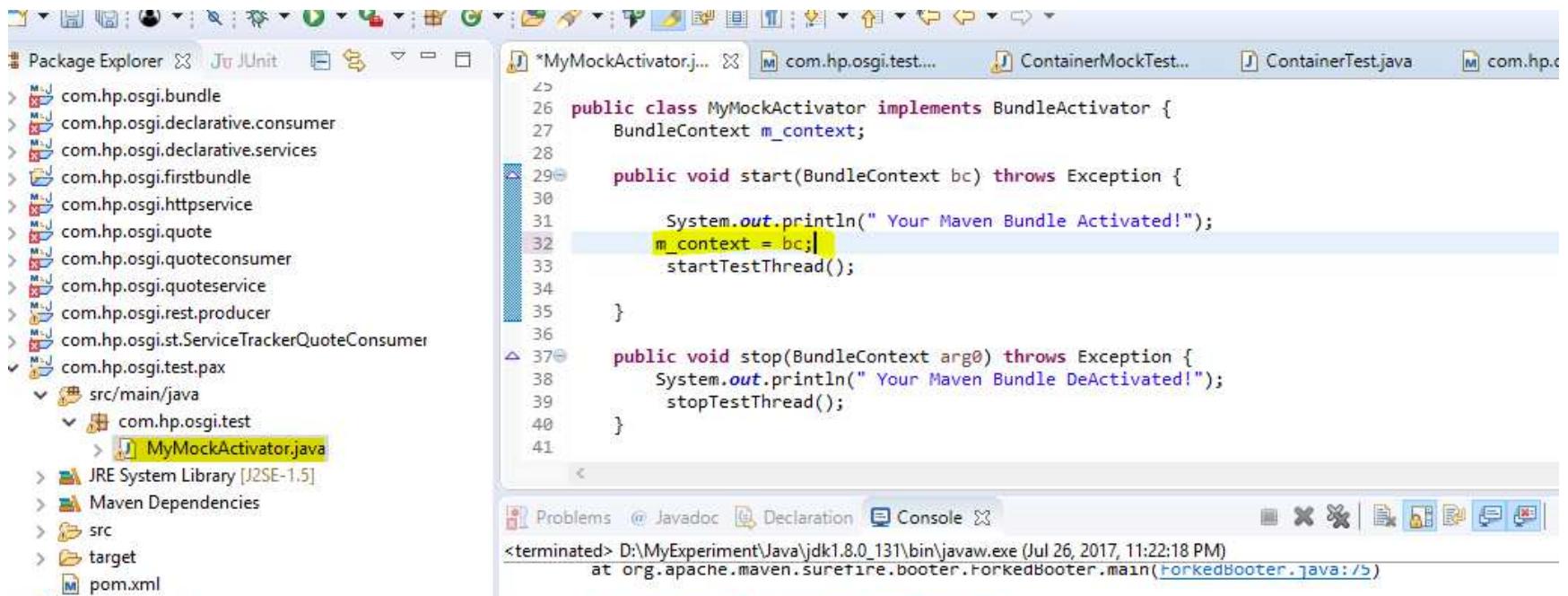
Case I:

Execute maven test ;it will give the following errors, why is it? We have not initialized the bundle; hence it throws Null Pointer exception, which is very common in development phase.

```
-----  
T E S T S  
-----  
Running com.hp.osgi.test.pax.mock.ContainerMockTest  
Your Maven Bundle Activated!  
Exception in thread "LogService Tester" java.lang.NullPointerException  
    at com.hp.osgi.test.MyMockActivator$LogServiceTest.run(MyMockActivator.java:50)  
    at java.lang.Thread.run(Thread.java:748)  
Your Maven Bundle DeActivated!  
Tests run: 1, Failures: 1, Errors: 0, Skipped: 0, Time elapsed: 1.322 sec <<< FAILURE!  
testLogClientBehaviour(com.hp.osgi.test.pax.mock.ContainerMockTest) Time elapsed: 1.228 sec <<< FAILURE!  
java.lang.AssertionError:  
    Expectation failure on verify:  
        BundleContext.getServiceReference("org.osgi.service.log.LogService"): expected: 1, actual: 0  
        BundleContext.getService(EasyMock for interface org.osgi.framework.ServiceReference): expected: 1, actual: 0  
            at org.easymock.internal.MocksControl.verify(MocksControl.java:225)
```

Case II :

Initialize the bundle context in the bundle, for which we want to perform unit testing as shown below;



```

public void start(BundleContext bc) throws Exception {

    System.out.println(" Your Maven Bundle Activated!");
    m_context = bc; // [Include this line in the Bundle and install and test again]
    startTestThread();

}

```

Perform, Maven clean → Install on the testing Bundle,

After Initializing the bundle context [], all the test cases works well. This way we can test the code before deploying the bundle in the container.

Then, you can execute; Maven clean → Test on the Mock bundle.

```

-----  

T E S T S  

-----  

Running com.hp.osgi.test.pax.mock.ContainerMockTest  

Your Maven Bundle Activated!  

Your Maven Bundle DeActivated!  

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 1.204 sec  

-----  

Running com.hp.osgi.test.pax.mock.ContainerTest  

SLF4J: Class path contains multiple SLF4J bindings.  

SLF4J: Found binding in [jar:file:/C:/Users/Henry/.m2/repository/org/slf4j/slf4j-log4j12/1.6.4/slf4j-log4j12-1.6.4.jar!/org/slf4j/impl/StaticLoggerBinder.class]  

SLF4J: Found binding in [jar:file:/C:/Users/Henry/.m2/repository/org/apache/karaf/org.apache.karaf.client/4.0.0/org.apache.karaf.client-4.0.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]  

SLF4J: Found binding in [jar:file:/C:/Users/Henry/.m2/repository/org/ops4j/pax/logging-api/1.8.3/pax-logging-api-1.8.3.jar!/org/slf4j/impl/StaticLoggerBinder.class]  

SLF4J: See http://www.slf4j.org/codes.html#multiple\_bindings for an explanation.  

[ DefaultExamSystem] - Pax Exam System (Version: 4.7.0) created.  

[ ProbeRunner] - creating PaxExam runner for class com.hp.osgi.test.pax.mock.ContainerTest  

[ ProbeRunner] - running test class com.hp.osgi.test.pax.mock.ContainerTest  

[ KarafTestContainer] - Wait for test container to finish its initialization [ RelativeTimeout value = 180000 ]  

[ RemoteBundleContextClient] - Waiting for remote bundle context.. on 21000 name: 5d26b1d8-0ac7-4480-981e-fc7d1bcf08e timeout: [ RelativeTimeout value = 180000 ]  

Your Maven Bundle Activated!  

-----  

/ / / /  

/ , < / / / / / / / / / / / / / / / / / / / /  

/ / / / / / / / / / / / / / / / / / / / / / / / / /  

Apache Karaf (4.0.9)  

Hit '<tab>' for a list of available commands  

and '[cmd] --help' for help on a specific command.  

Hit '<ctrl-d>' or type 'system:shutdown' or 'logout' to shutdown Karaf.  

karaf@root()> Bundle Context org.apache.felix.framework.BundleContextImpl@67ed242e>>> true  

Your Maven Bundle DeActivated!  

-----  

2017-07-26 23:18:18,526 | WARN  | FelixStartLevel | core | 17 - org.apache.aries.jmx.core - 1.1.7 | Task rejected for JMX Notification dispatch of event  

2017-07-26 23:18:18,527 | WARN  | lixDispatchQueue | core | 17 - org.apache.aries.jmx.core - 1.1.7 | Task rejected for JMX Notification dispatch of event  

[ DefaultExamSystem] - Option org.ops4j.pax.exam.karaf.options.KarafDistributionKitConfigurationOption has not been recognized.  

[ DefaultExamSystem] - Option org.ops4j.pax.exam.options.FrameworkStartLevelOption has not been recognized.  

[ ReactorManager] - suite finished  

Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 15.894 sec

```

Enabling Proxy in karaf

Need to install maven for this.

In Apache karaf folder there is a etc folder. Edit the org.ops4j.pax.url.mvn.cfg file. You need to perform two steps for it to work.

- Point karaf to your maven installation: find the following string in your cfg file org.ops4j.pax.url.mvn.settings uncomment it and add your maven home path i.e. org.ops4j.pax.url.mvn.settings= /maven/conf/settings.xml
- Tell karaf to use the maven proxy settings: find the following string in your cfg file org.ops4j.pax.url.mvn.proxySupport uncomment it and set it to true if needs be i.e. org.ops4j.pax.url.mvn.proxySupport=true

I restarted Karaf and I can now download/install features

C:\Users\henryp\.m2\setting.xml

```
<proxies>
<proxy>
  <id>HT</id>
  <active>true</active>
  <protocol>http</protocol>
  <username>domain\henryp</username>
  <password>####</password>
  <host>##.##.15.60</host>
  <port>80</port>
</proxy>
</proxies>
```

Configuration

For instance, to set the minimum and maximum memory size for the JVM, you can define the following values:

```
rem Content of bin\setenv.bat  
set JAVA_MIN_MEM=256M  
set JAVA_MAX_MEM=1024M
```

Security

`jaas:realm-list` command list the current defined realms:

```
karaf@root>> jaas:realm-list
Index : Realm Name : Login Module Class Name
-----
1 | karaf      | org.apache.karaf.jaas.modules.properties.PropertiesLoginModule
2 | karaf      | org.apache.karaf.jaas.modules.publickey.PublickeyLoginModule
3 | karaf      | org.apache.karaf.jaas.modules.audit.FileAuditLoginModule
4 | karaf      | org.apache.karaf.jaas.modules.audit.LogAuditLoginModule
5 | karaf      | org.apache.karaf.jaas.modules.audit.EventAdminAuditLoginModule
karaf@root>>
```

To identify the realm and login module that you want to manage:

```
#jaas:realm-manage --index 1
```

```
#jaas:user-list
```

```
karaf@root>> jaas:realm-manage --index 1
karaf@root>> jaas:user-list
User Name : Group      : Role
-----
karaf    | admingroup | admin
karaf    | admingroup | manager
karaf    | admingroup | viewer
karaf    | admingroup | systembundles
karaf@root>>
```

adds a new user (and the password) in the currently edited login module:

```
jaas:user-add henry henry
```

To "commit" your change (here the user addition), you have to execute the `jaas:update` command:

```
#jaas:update
```

```
# jaas:realm-manage --index 1
```

```
# jaas:user-list
```

```
karaf@root>> jaas:user-list
No JAAS Realm/Login Module has been selected
karaf@root>> jaas:realm-manage --index 1
karaf@root>> jaas:user-list
User Name | Group      | Role
-----
karaf    | admingroup  | admin
karaf    | admingroup  | manager
karaf    | admingroup  | viewer
karaf    | admingroup  | systembundles
henry   |             |
karaf@root>> jaas:user-add henry henry
karaf@root>> jaas:user-list
User Name | Group      | Role
-----
karaf    | admingroup  | admin
karaf    | admingroup  | manager
karaf    | admingroup  | viewer
karaf    | admingroup  | systembundles
henry   |             |
karaf@root>>
```

Assigns a group (and eventually creates the group) to an user in the currently edited login module:

```
#jaas:group-add henry admingroup  
#jaas:update  
#jaas:realm-manage --index 1  
#jaas:user-list
```

```
karaf@root<*> jaas:user-list  
User Name | Group      | Role  
-----  
karaf    | admingroup | admin  
karaf    | admingroup | manager  
karaf    | admingroup | viewer  
karaf    | admingroup | systembundles  
henry   | admingroup |  
karaf@root<*> jaas:update  
karaf@root<*> jaas:user-list  
No JAAS Realm/Login Module has been selected  
karaf@root<*> jaas:realm-manage --index 1  
karaf@root<*> jaas:user-list  
User Name | Group      | Role  
-----  
karaf    | admingroup | admin  
karaf    | admingroup | manager  
karaf    | admingroup | viewer  
karaf    | admingroup | systembundles  
henry   | admingroup | admin  
henry   | admingroup | manager  
henry   | admingroup | viewer  
henry   | admingroup | systembundles  
karaf@root<*>
```

Troubleshooting Issues

It's not always easy for the developers to understand why a bundle is not active.

It could be because the Activator failed, the Blueprint container start failed, etc.

The `bundle:diag` command gives you details about a bundle is not active:

```
#bundle:diag
```

The `shell:stack-traces-print` command prints the full stack trace when the execution of a command throws an exception.

```
shell:stack-traces-print {true/false}
```

The `bundle:tree-show` command shows the bundle dependency tree based on the wiring information of a given single bundle ID.

List all the bundle installed in the server and find the dependency tree for that particular bundle camel-karaf-commands using bundle id.

```
#bundle:list
```

```
# bundle:tree-show 56
```

```
karaf@root>> bundle:list
START LEVEL 100 , List Threshold: 50
ID | State | Lvl | Version | Name
--+
52 | Active | 50 | 2.15.2 | camel-catalog
53 | Active | 50 | 2.15.2 | camel-commands-core
54 | Active | 50 | 2.15.2 | camel-core
55 | Active | 50 | 2.15.2 | camel-spring
56 | Active | 50 | 2.15.2 | camel-karaf-commands
57 | Active | 50 | 1.1.1 | geronimo-jta_1.1_spec
karaf@root>> bundle:tree-show 56
Bundle org.apache.camel.karaf.camel-karaf-commands [56] is currently ACTIVE

org.apache.camel.karaf.camel-karaf-commands [56]
+- org.apache.aries.blueprint.core [12]
|   +- org.ops4j.pax.logging.pax-logging-api [1]
|   |   +- org.apache.karaf.services.eventadmin [6]
|   |   |   +- org.apache.felix.metatype [5]
|   |   |   +- org.apache.felix.configadmin [3]
|   +- org.apache.aries.blueprint.api [10]
|   +- org.apache.aries.proxy.api [19]
|   |   +- org.apache.aries.util [21]
|   |   |   +- org.ops4j.pax.logging.pax-logging-api [1]
|   +- org.apache.karaf.services.eventadmin [6]
|   +- org.apache.aries.util [21]
+- org.ops4j.pax.logging.pax-logging-api [1]
+- org.apache.camel.camel-core [54]
|   +- org.ops4j.pax.logging.pax-logging-api [1]
+- org.apache.camel.camel-commands-core [53]
|   +- org.apache.camel.camel-core [54]
|   +- org.apache.camel.camel-catalog [52]
+- jline [9]
+- org.apache.karaf.shell.core [43]
|   +- org.apache.aries.blueprint.core [12]
|   |   +- org.apache.felix.configadmin [3]
|   |   +- org.ops4j.pax.logging.pax-logging-api [1]
|   |   +- org.apache.aries.blueprint.api [10]
|   |   +- org.apache.karaf.services.eventadmin [6]
|   |   +- jline [9]
karaf@root>>
```

Use log:display or log:display-exception to display the log

log:display | more | grep ERROR

```
karaf@root>>> log:display | more | grep ERROR
2017-07-07 14:33:41,396 | ERROR | nsole user karaf | ShellUtil
| 43 - org.apache.karaf.shell.core - 4.0.9 | Exception caught while execut
ing command
2017-07-07 14:34:08,351 | ERROR | nsole user karaf | ShellUtil
| 43 - org.apache.karaf.shell.core - 4.0.9 | Exception caught while execut
ing command
karaf@root>>>
```

log:display-exception

```
karaf@root>>> log:exception-display
java.lang.IllegalArgumentException: No matching bundles
    at org.apache.karaf.bundle.command.BundlesCommand.doExecute(BundlesComma
nd.java:59)[23:org.apache.karaf.bundle.core:4.0.9]
    at org.apache.karaf.bundle.command.BundlesCommand.execute(BundlesCommand
.java:54)[23:org.apache.karaf.bundle.core:4.0.9]
    at org.apache.karaf.shell.impl.action.command.ActionCommand.execute(Acti
onCommand.java:83)[43:org.apache.karaf.shell.core:4.0.9]
    at org.apache.karaf.shell.impl.console.osgi.secured.SecuredCommand.execu
te(SecuredCommand.java:67)[43:org.apache.karaf.shell.core:4.0.9]
    at org.apache.karaf.shell.impl.console.osgi.secured.SecuredCommand.execu
te(SecuredCommand.java:87)[43:org.apache.karaf.shell.core:4.0.9]
    at org.apache.felix.gogo.runtime.Closure.executeCmd(Closure.java:480)[43
:org.apache.karaf.shell.core:4.0.9]
    at org.apache.felix.gogo.runtime.Closure.executeStatement(Closure.java:4
06)[43:org.apache.karaf.shell.core:4.0.9]
    at org.apache.felix.gogo.runtime.Pipe.run(Pipe.java:108)[43:org.apache.k
araf.shell.core:4.0.9]
    at org.apache.felix.gogo.runtime.Closure.execute(Closure.java:182)[43:or
g.apache.karaf.shell.core:4.0.9]
    at org.apache.felix.gogo.runtime.Closure.execute(Closure.java:119)[43:or
g.apache.karaf.shell.core:4.0.9]
    at org.apache.felix.gogo.runtime.CommandSessionImpl.execute(CommandSessi
onImpl.java:94)[43:org.apache.karaf.shell.core:4.0.9]
    at org.apache.karaf.shell.impl.console.ConsoleSessionImpl.run(ConsoleSes
sionImpl.java:274)[43:org.apache.karaf.shell.core:4.0.9]
    at java.lang.Thread.run(Thread.java:745)[:1.8.0_45]
```

Performance Tuning

If you encounter issues like performance degradations, weird behaviour, it could be helpful to have a kind of snapshot about the current activity of the container.

```
#dev:dump-create
```

```
[root@karaf ~]# dev:dump-create  
Created dump zip: 2017-07-07_142643.zip
```

Monitoring

Karaf uses JMX for monitoring and management of all Karaf components.

The JMX connection could be:

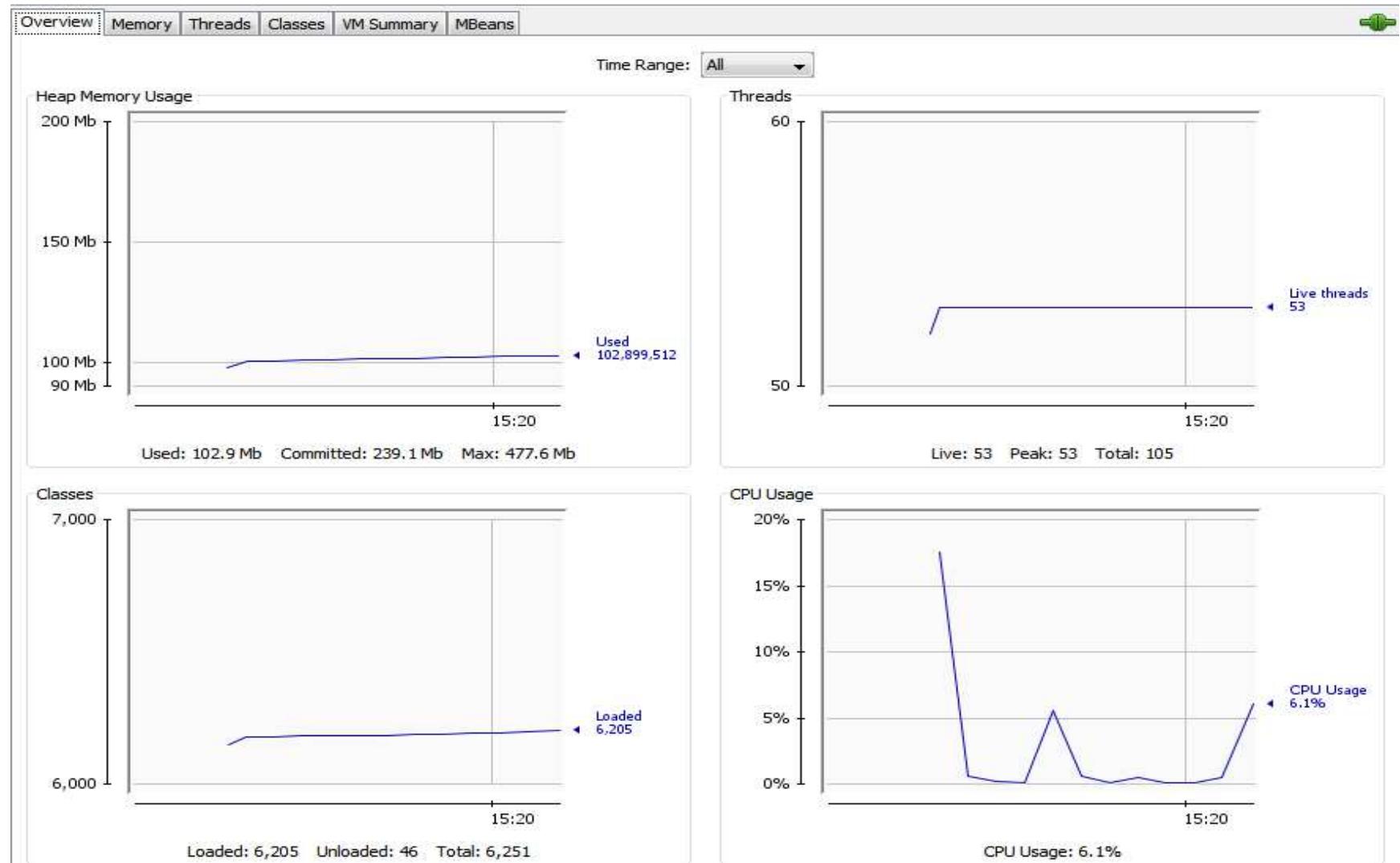
- local using the process id

#jconsole

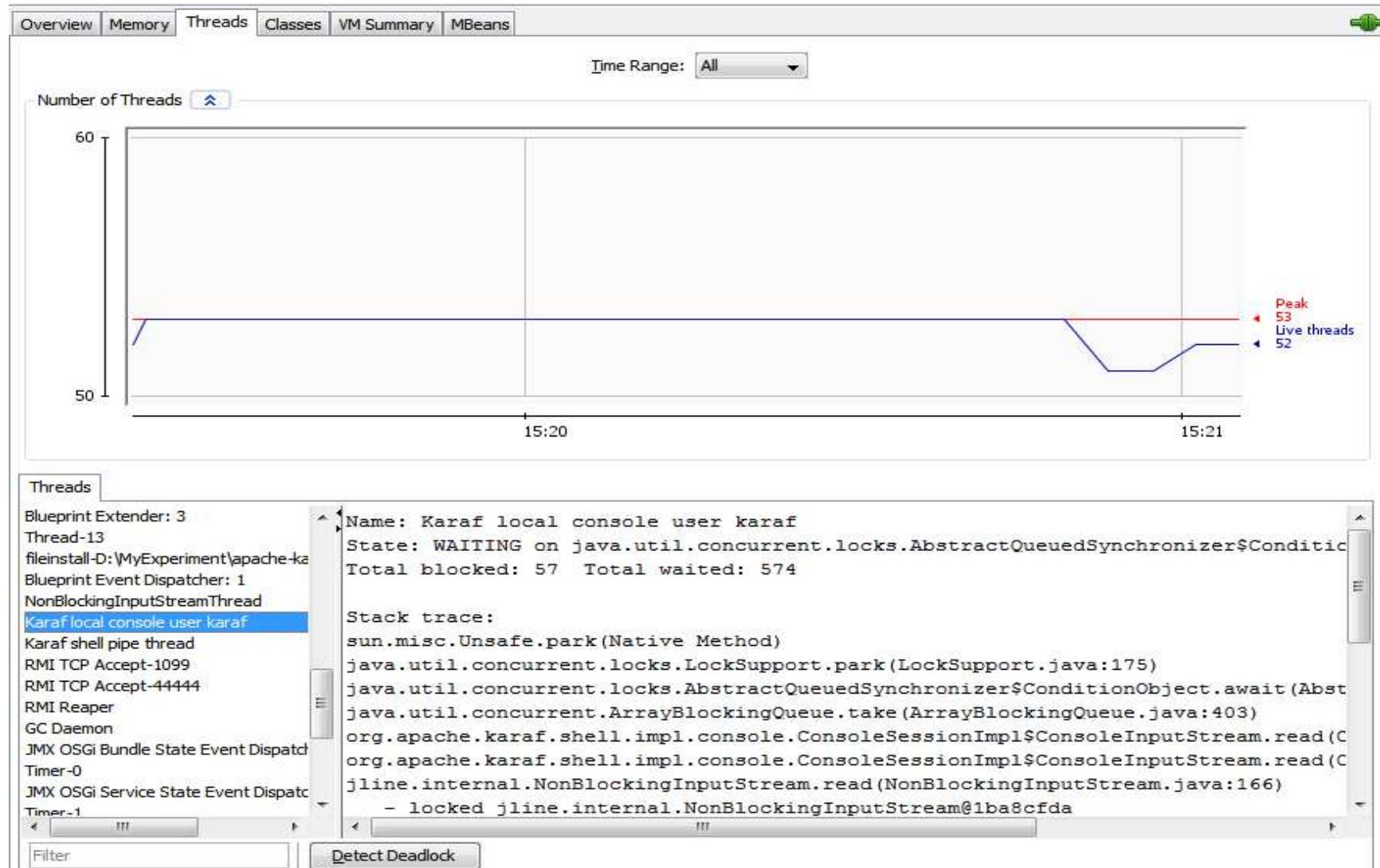


Using JMX, you can have a clean overview of the running Karaf instance:

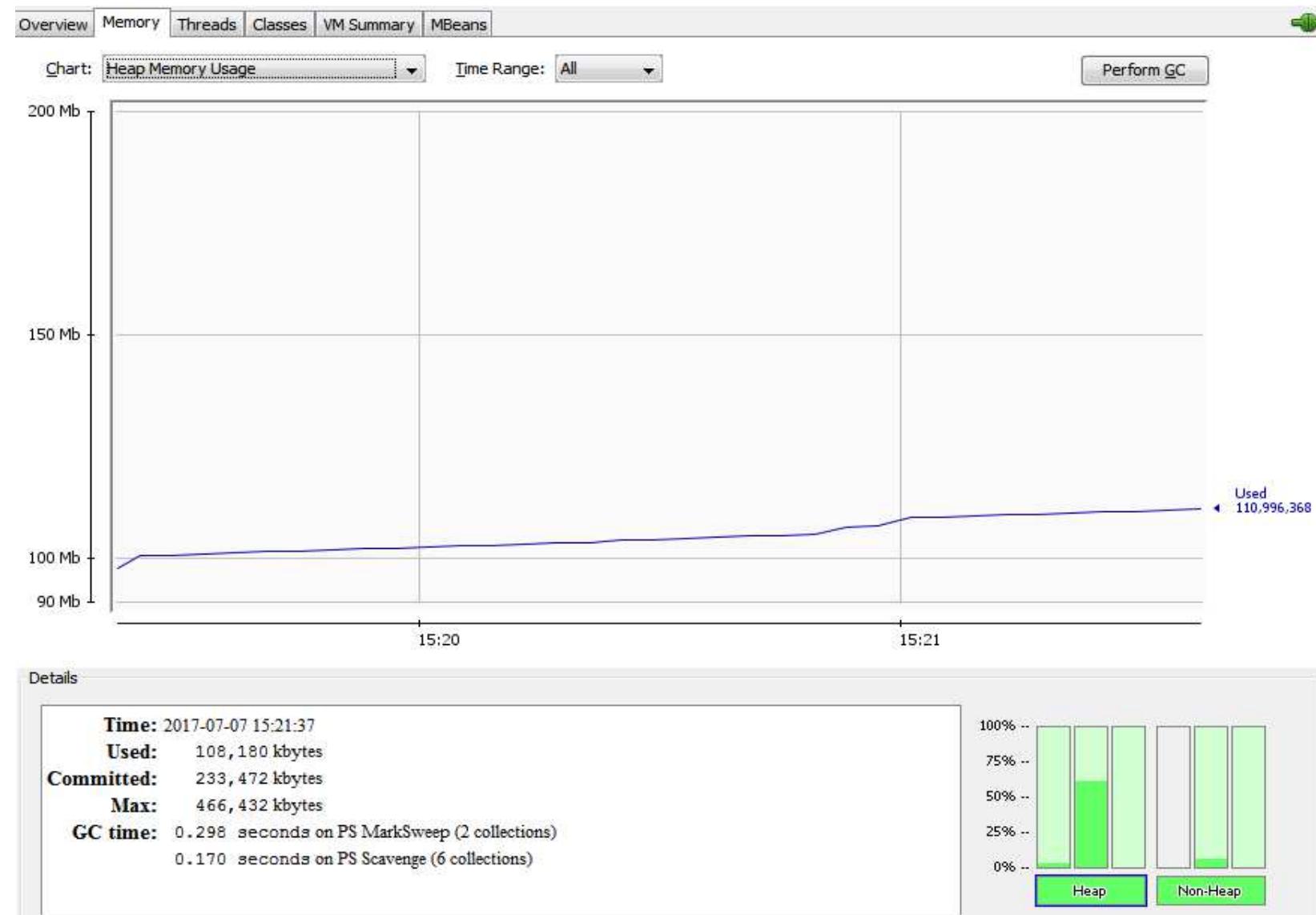
- A overview with graphics displaying the load in terms of thread, heap/GC, etc:



- A thread overview:



- A memory heap consumption, including "Perform GC" button:



- A complete JVM summary, with all number of threads, etc:

| Overview | Memory | Threads | Classes | VM Summary | MBeans | | | |
|----------------------------------------------------------------------------------------------------|--------|---------|----------------------------------------|------------|------------------------------------------|--|--|--|
| VM Summary | | | | | | | | |
| Friday, July 7, 2017 3:22:17 PM IST | | | | | | | | |
| Connection name: pid: 9952 org.apache.karaf.main.Main | | | | | Uptime: 1 hour 11 min | | | |
| Virtual Machine: Java HotSpot(TM) 64-Bit Server VM version 25.45-b02 | | | | | Process CPU time: 46.628 second | | | |
| Vendor: Oracle Corporation | | | | | JIT compiler: HotSpot 64-B | | | |
| Name: 9952@LP000007011370 | | | | | Total compile time: 29.109 second | | | |
| Live threads: 51 | | | Current classes loaded: 6,232 | | | | | |
| Peak: 53 | | | Total classes loaded: 6,278 | | | | | |
| Daemon threads: 46 | | | Total classes unloaded: 46 | | | | | |
| Total threads started: 106 | | | | | | | | |
| Current heap size: 110,350 kbytes | | | Committed memory: 233,472 k | | | | | |
| Maximum heap size: 466,432 kbytes | | | Pending finalization: 0 objects | | | | | |
| Garbage collector: Name = 'PS MarkSweep', Collections = 2, Total time spent = 0.298 seconds | | | | | | | | |
| Garbage collector: Name = 'PS Scavenge', Collections = 6, Total time spent = 0.170 seconds | | | | | | | | |
| Operating System: Windows 7 6.1 | | | Total physical memory: 4,007 | | | | | |
| Architecture: amd64 | | | Free physical memory: 1,008 | | | | | |
| Number of processors: 4 | | | Total swap space: 10,821 | | | | | |
| Committed virtual memory: 386,544 kbytes | | | Free swap space: 6,245 | | | | | |

You can manage Karaf features like you are in the shell. For example, you have access to the Admin service MBean, allowing you to create, rename, destroy, change SSH port, etc. Karaf instances: You can also manage Karaf features MBean to list, install, and uninstall Karaf features:

Creating bundles for non OSGi third party dependencies

Command

bundle:headers 183

Information about the bundles