Lab 4 – Consuming your own builds

The goal of this lab is to build to a p2 repository that is then used for other downstream builds.

Start by pointing Eclipse at the `webapps/root/labs/lab-4` folder contained in the tutorial root. Import the projects into the workspace.

In this lab you will do the following:

1. Add a dependency on a non-visual feature.

2. Build the non-visual feature and publish it to a p2 repository.

3. Build the visual feature that consumes the new p2 repository.

Add a dependency on a non-visual feature.

A new “quad” has been added to the projects for this lab. The following projects make up the quad:

```
com.example.app.utility
com.example.app.utility.feature
com.example.app.utility.p2
com.example.app.utility.releng
```

These projects contain sample code and are set up to build correctly using Maven Tycho. We’ll now add a dependency from the `com.example.app.perspectives` quad to this one.

1. Open the manifest for the `com.example.app.perspectives` bundle. On the **Dependencies** tab, add `com.example.app.utility` to this list or imported packages.

2. Open the `View` class in the `com.example.app.perspectives` bundle. In the `createPartControl` method, insert this code at the beginning of the method:

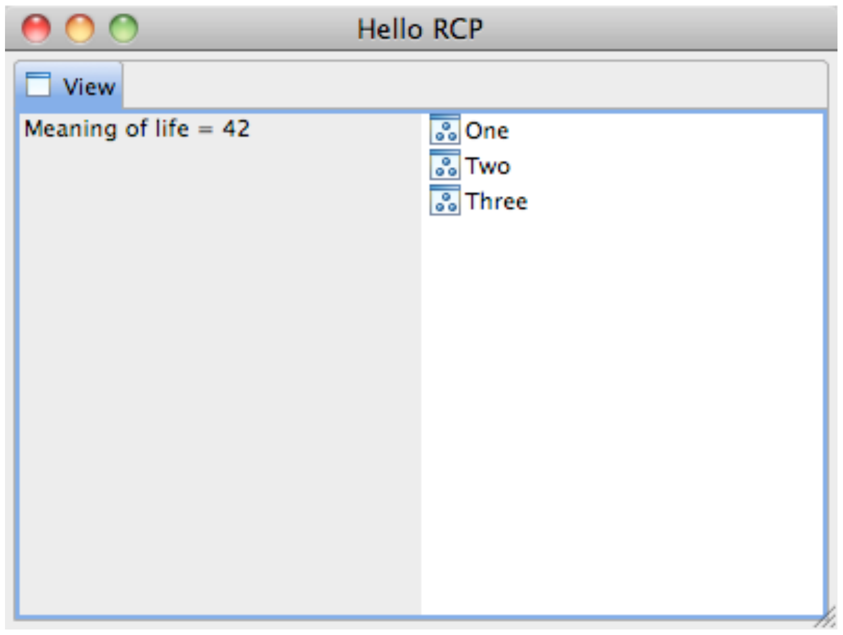
```
Label label = new Label(parent, SWT.NONE);
int meaningOfLife = new MeaningFactory().getMeaningOfLife();
label.setText("Meaning of life = " + meaningOfLife);
```

3. Test the code by clicking the link in the `example.product` file. You will get an error because the perspective and utility features are not included in the run configuration.

4. Select **Run > Run Configurations…** from the main menu. Select the `example.product` run configuration and switch to the **Plug-ins** tab.

5. Place a check next to all of the bundles in the workspace. Then click the **Add Required Plug-ins** button. Click **Run**.

   The application should run successfully and you should see the view with the new label. You will get an exception in the console related to the p2 auto-update process. This is ok.



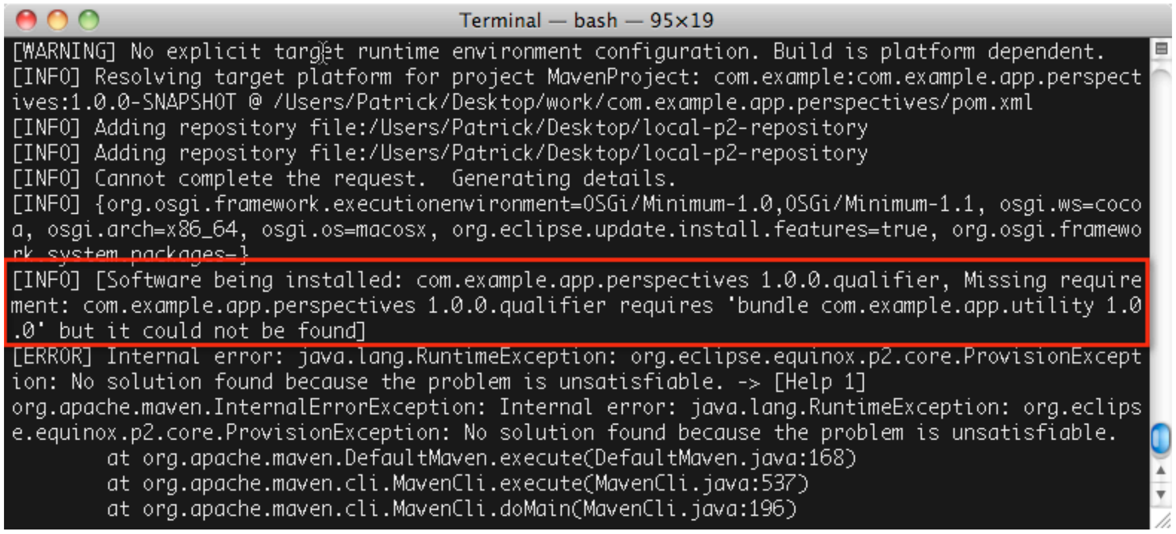
Build the non-visual feature and publish it to a p2 repository.

1. From a terminal window, move to the `com.example.app.utility.releng` project. Do a `mvn clean package` to build the non-visual feature.

2. Refresh the `com.example.app.utility.p2` project. The p2 repository can be found under the `target/site` folder.

Build the visual feature that consumes the new p2 repository.

1. Move to the `com.example.app.perspectives.releng` project. Do a `mvn package` and notice that the build fails because the utility feature cannot be found.



```
[WARNING] No explicit target runtime environment configuration. Build is platform dependent.
[INFO] Resolving target platform for project MavenProject: com.example:com.example.app.perspect
ives:1.0.0-SNAPSHOT @ /Users/Patrick/Desktop/work/com.example.app.perspectives/pom.xml
[INFO] Adding repository file:/Users/Patrick/Desktop/local-p2-repository
[INFO] Adding repository file:/Users/Patrick/Desktop/local-p2-repository
[INFO] Cannot complete the request.  Generating details.
[INFO] {org.osgi.framework.executionenvironment=OSGi/Minimum-1.0,OSGi/Minimum-1.1, osgi.ws=coco
a, osgi.arch=x86_64, osgi.os=macosx, org.eclipse.update.install.features=true, org.osgi.framewo
rk.system.packages=}
[INFO] [Software being installed: com.example.app.perspectives 1.0.0.qualifier, Missing require
ment: com.example.app.perspectives 1.0.0.qualifier requires 'bundle com.example.app.utility 1.0
.0' but it could not be found]
[ERROR] Internal error: java.lang.RuntimeException: org.eclipse.equinox.p2.core.ProvisionExcept
ion: No solution found because the problem is unsatisfiable. -> [Help 1]
org.apache.maven.InternalErrorException: Internal error: java.lang.RuntimeException: org.eclips
e.equinox.p2.core.ProvisionException: No solution found because the problem is unsatisfiable.
        at org.apache.maven.DefaultMaven.execute(DefaultMaven.java:168)
        at org.apache.maven.cli.MavenCli.execute(MavenCli.java:537)
        at org.apache.maven.cli.MavenCli.doMain(MavenCli.java:196)
```

2. Open the `pom.xml` file in the `com.example.app.perspectives.releng` project. Add a repository element that points to the update site we generated earlier, like this:

```
        <repository>
            <id>utility-repository</id>
            <layout>p2</layout>
            <url>http://localhost:8080/labs/lab-
4/com.example.app.utility.p2/target/site</url>
        </repository>
```

3. Re-run the build and it should now complete successfully. This lab is now complete.