# Introduction to Mapping Lab

## Goals

- Import and examine a Red Hat JBoss Fuse project with a mapping activity that converts XML to a Java object
- Build and test the mapping

# 1. Update JBoss Developer Studio to Install the New Data Transformation Tool

1. Open JBoss Developer Studio.
2. Click the **JBoss Central** tab. You should see **Welcome to JBoss** in the upper-left corner of the screen.
3. Click the **Software/Update** tab at the bottom of the screen.

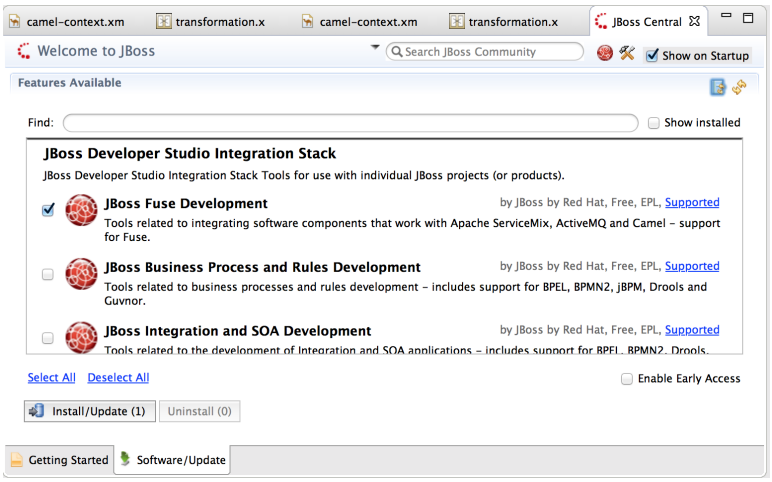


**Figure 1. Software/Update tab**

4. Check the **Enable Early Access** box in the lower-right corner.
5. Click **Yes** to install the updates.

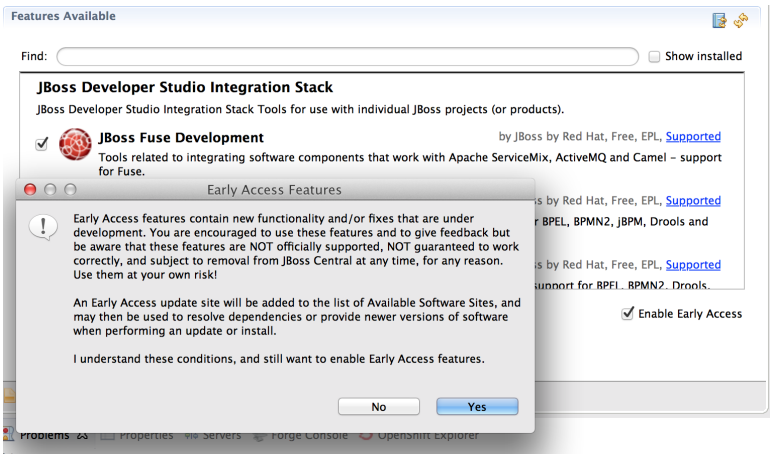


**Figure 2. Install updates confirmation**

# 2. Import the Lab Project

1. In JBoss Developer Studio, switch to the **Git** perspective.
2. Click the icon at the top to clone a Git repository and add the clone to this view.
3. In the **URI** text box, copy and paste the following:

```
https://github.com/gpe-mw-training/fuse-mapper-labs.git
```

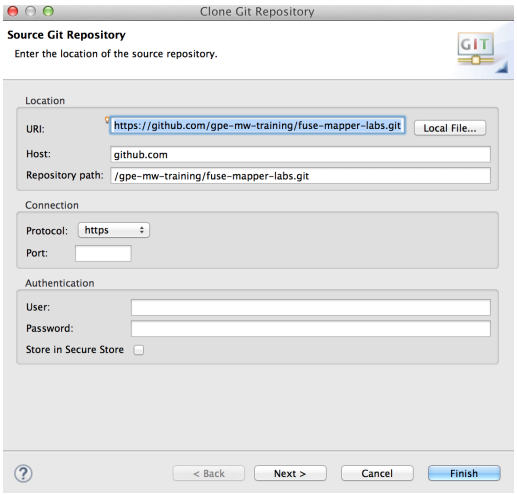


**Figure 3. Select location**

4. Click **Next**.
5. On the screens that appear, click **Next** until the **Local Destination** screen appears.
6. In the **Directory** field, enter the directory path to which you want to clone the Switchyard Git projects.
7. Check the **Import all existing projects after clone finishes** box and click **Next**.

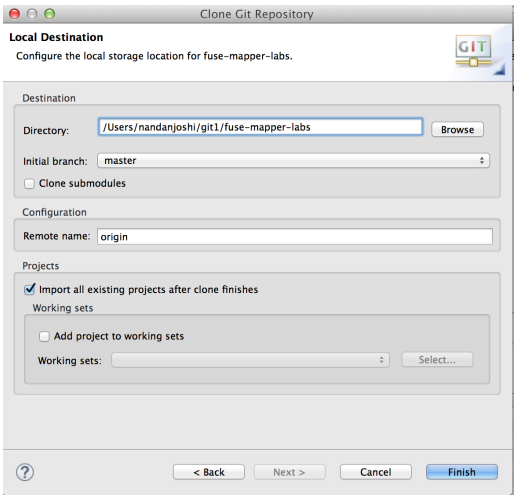


**Figure 4. Import project**

8. The **Git Repositories** panel should now list the local copy of the cloned repositories.
   - This step imports all the relevant repositories for this lab.
   - Make note of the path on your local file system.

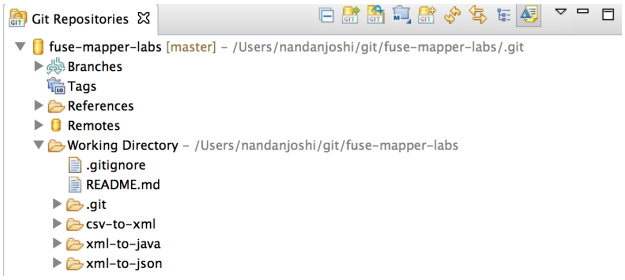


**Figure 5. Git repositories**

9. Switch to **Project Explorer** for the **JBoss** perspective.
10. Import a new Maven project by selecting **File** → **Import** → **Maven** → **Existing Maven Projects** → **Next**.

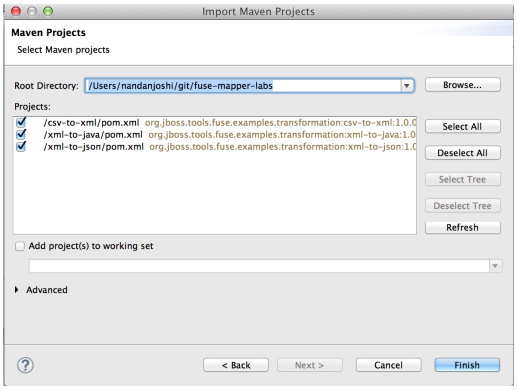


**Figure 6. Import Maven project**

11. Navigate to the location of the Git projects for the Data Transformation tool and click **Finish**.
12. Click the **Project Explorer** tab and expand the `xml-to-java` project node.

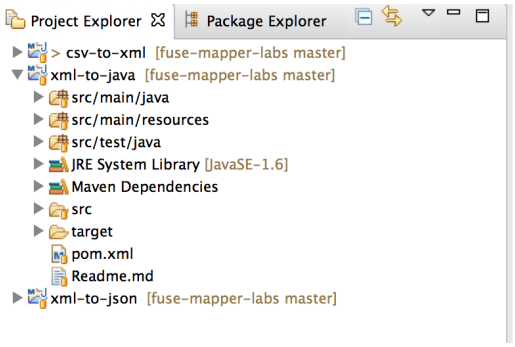


**Figure 7. Application structure**

13. Examine the following files and folders that appear in the expanded view:
    - `src/main/java`: Contains the Java files that provide your application's logic. These files are packaged with your application for deployment.

      Examples of these types of files:

      - Java classes generated for the input mapping type
      - Java classes generated for the output mapping type

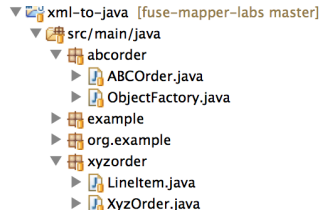


**Figure 8. src/main/java example**

- `src/main/resources`: Contains non-Java resources packaged with your application.

  Examples of these types of files:

  - Camel route defined for Spring runtime (`META-INF/spring/camel-context.xml`)
  - Camel route defined for OSGI runtime (`OSGI-INF/blueprint/camel-context.xml`)
  - Input schema or document
  - Output schema or document
  - XML file where the transformation is stored (`transformation.xml`)
  - Features defined for OSGI (`features.xml`)

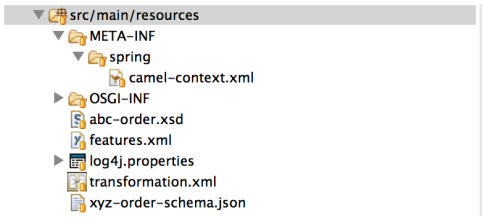


**Figure 9. src/main/resources example**

- `src/test/java`: Contains all Java classes related to testing your application.

  Examples of these types of files:

  - Transformation tests created by the mapping wizard

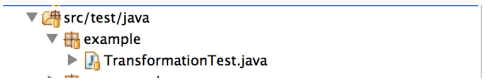


**Figure 10. src/test/java example**

# 3. Learn the JBoss Fuse Application

Here you walk through the various components of the JBoss Fuse application, with special focus on mapping activity.

## 3.1. Examine the Camel Route and Data Transformation Tool Activity

1. Click the **Project Explorer** tab.
2. Expand the `src/main/resources` node.
3. Under the `META-INF/spring` directory, double-click `camel-context.xml`.
    - This opens the Camel route visual application editor, which includes the following:

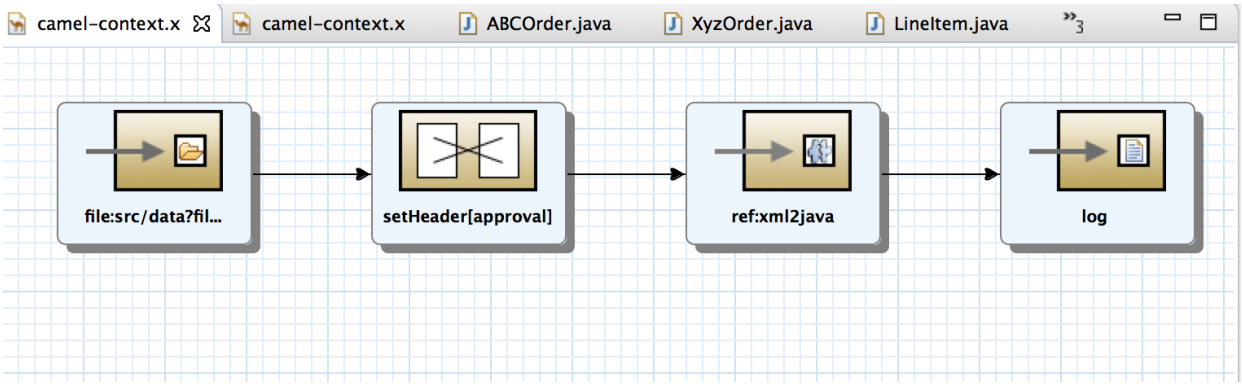


**Figure 11. Camel route example**

    - **#1**: The *canvas* provides a visual representation of the components and sequencing in a Camel route.
    - **#2**: The *palette* provides a set of application building blocks that you can drag onto the canvas to create a route.

## 3.2. Build the `xml-to-java` Project

Here you build the `xml-to-java` project using the `m2e` Maven plug-in in JBoss Developer Studio.

1. In **JBoss Developer Studio**, click the **Package Explorer** tab.
2. Right-click `xml-to-java`.
3. Navigate to **Run As** → **Maven build**.

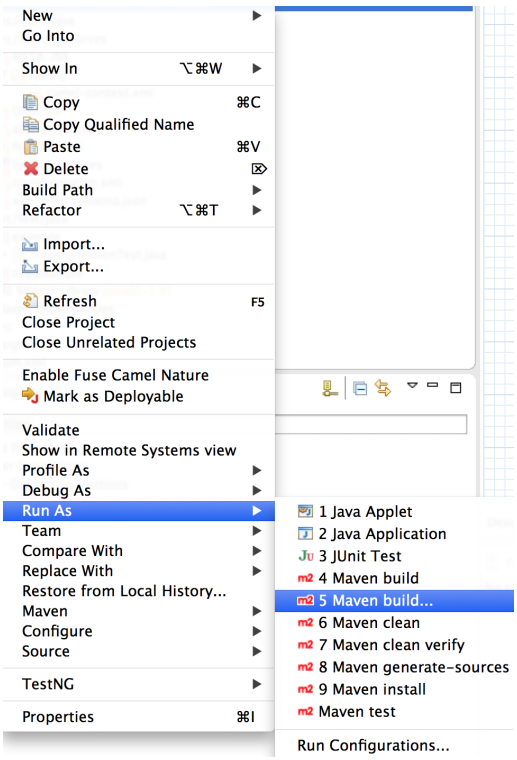


**Figure 12. Run as Maven build**

4. On the **Edit configuration and launch** screen, enter the following:
   - **Name**: `xml-to-java`
   - **Goals**: `clean install`
5. Click **Run**.

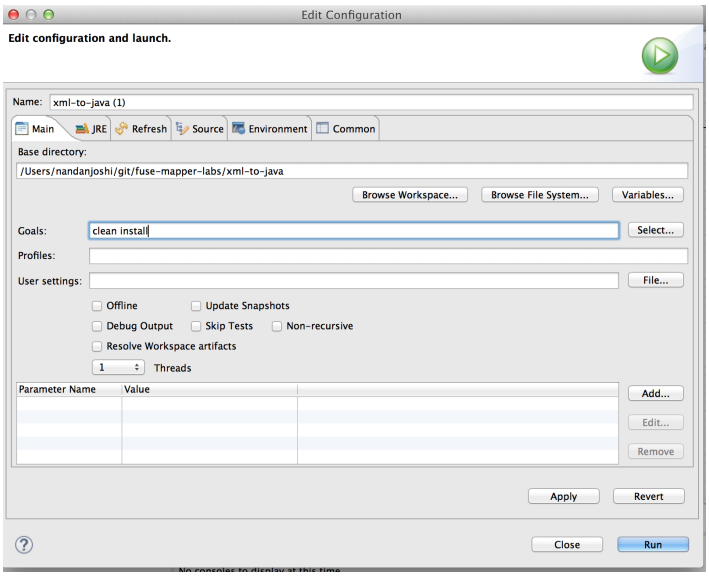


**Figure 13. Run Maven build**

   - You should see log statements similar to the following in the console of your JBoss Developer Studio:

```
-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running example.TransformationTest
```

```
[                          main] TransformationTest             INFO
********************************************************************************
[                          main] TransformationTest             INFO  Testing:
transform(example.TransformationTest)
[                          main] TransformationTest             INFO
********************************************************************************
[                          main] DefaultTypeConverter           INFO  Loaded 183
type converters
[                          main] SpringCamelContext             INFO  Apache
Camel 2.15.1 (CamelContext: camel-1) is starting
[                          main] ManagedManagementStrategy      INFO  JMX is
enabled
[                          main] GlobalSettings                 INFO  Trying to
find Dozer configuration file: dozer.properties
[                          main] GlobalSettings                 WARN  Dozer
configuration file not found: dozer.properties.  Using defaults for all Dozer
global properties.
[                          main] DozerComponent                 INFO
Configuring GlobalSettings to enable EL
[                          main] DozerInitializer               INFO
Initializing Dozer. Version: 5.5.0, Thread Name: main
[                          main] JMXPlatformImpl                INFO  Dozer JMX
MBean [org.dozer.jmx:type=DozerStatisticsController] auto registered with the
Platform MBean Server
[                          main] JMXPlatformImpl                INFO  Dozer JMX
MBean [org.dozer.jmx:type=DozerAdminController] auto registered with the
Platform MBean Server
[                          main] DozerBeanMapper                INFO
Initializing a new instance of dozer bean mapper.
[                          main] DozerEndpoint                  INFO  Loading
Dozer mapping file transformation.xml.
[                          main] SpringCamelContext             INFO
AllowUseOriginalMessage is enabled. If access to the original message is not
needed, then its recommended to turn this option off as it may improve
performance.
[                          main] SpringCamelContext             INFO
StreamCaching is not in use. If using streams then its recommended to enable
stream caching. See more details at http://camel.apache.org/stream-caching.html
[                          main] FileEndpoint                   INFO  Endpoint
is configured with noop=true so forcing endpoint to be idempotent as well
[                          main] FileEndpoint                   INFO  Using
default memory based idempotent repository with cache max size: 1000
[                          main] SpringCamelContext             INFO  Route:
route1 started and consuming from: Endpoint[file://src/data?fileName=abc-
order.xml&noop=true]
[                          main] SpringCamelContext             INFO  Route:
route2 started and consuming from: Endpoint[direct://xml2java-test-input]
[                          main] SpringCamelContext             INFO  Total 2
routes, of which 2 is started.
[                          main] SpringCamelContext             INFO  Apache
Camel 2.15.1 (CamelContext: camel-1) started in 0.438 seconds
[                          main] route2                         INFO  Before
transformation:
 <?xml version="1.0" encoding="UTF-8"?>
```

```
<ABCOrder xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:java="http://java.sun.com">
    <header>
        <status>GOLD</status>
        <customer-num>ACME-123</customer-num>
        <order-num>ORDER1</order-num>
    </header>
    <order-items>
        <item id="PICKLE">
            <price>2.25</price>
            <quantity>1000</quantity>
        </item>
        <item id="BANANA">
            <price>1.25</price>
            <quantity>400</quantity>
        </item>
    </order-items>
</ABCOrder>
[                          main] JaxbDataFormat                INFO  Creating
JAXBContext with contextPath: abcorder and ApplicationContextClassLoader:
sun.misc.Launcher$AppClassLoader@6e0be858
[                          main] StaxConverter                 INFO  Created
XMLInputFactory: com.sun.xml.internal.stream.XMLInputFactoryImpl@4089713.
DOMSource/DOMResult may have issues with
com.sun.xml.internal.stream.XMLInputFactoryImpl@4089713. We suggest using
Woodstox.
[                          main] route2                        INFO  After
transformation:
 custId:ACME-123
 priority:GOLD
 orderId:ORDER1
 origin:
 approvalCode:AUTO_OK
 lineItems:[xyzorder.LineItem@2cd2c8fe, xyzorder.LineItem@1a6f2363]
[                          main] TransformationTest            INFO
********************************************************************************
[                          main] TransformationTest            INFO  Testing
done: transform(example.TransformationTest)
[                          main] TransformationTest            INFO  Took:
0.156 seconds (156 millis)
[                          main] TransformationTest            INFO
********************************************************************************
[                          main] SpringCamelContext            INFO  Apache
Camel 2.15.1 (CamelContext: camel-1) is shutting down
[                          main] DefaultShutdownStrategy       INFO  Starting
to graceful shutdown 2 routes (timeout 10 seconds)
[el-1) thread #1 - ShutdownTask] DefaultShutdownStrategy       INFO  Route:
route2 shutdown complete, was consuming from: Endpoint[direct://xml2java-test-
input]
[el-1) thread #1 - ShutdownTask] DefaultShutdownStrategy       INFO  Route:
route1 shutdown complete, was consuming from: Endpoint[file://src/data?
fileName=abc-order.xml&noop=true]
[                          main] DefaultShutdownStrategy       INFO  Graceful
```

```
shutdown of 2 routes completed in 0 seconds
[                          main] SpringCamelContext             INFO  Apache
Camel 2.15.1 (CamelContext: camel-1) uptime 0.611 seconds
[                          main] SpringCamelContext             INFO  Apache
Camel 2.15.1 (CamelContext: camel-1) is shutdown in 0.006 seconds
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 2.031 sec - in
example.TransformationTest
```

Last updated 2015-10-05 15:09:58 EDT