JSXM MAVEN PLUGIN
Getting started

Konstantinos Margaritis

Computer Science Department
CITY College
An International Faculty of the
University of Sheffield

Version: v1.2
Last update: 21 March 2014

Contents

1 Requirements

In order to successfully deploy the examples and the JSXM Maven plugin, the environment must be equipped with JDK 1.7.

Download the appropriate JDK 1.7 for your operating system from http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html and follow the installation dialogs.

After the successful installation of the JDK, download the Eclipse Standard (Kepler) from https://www.eclipse.org/downloads/ and extract the compressed folder.

2 Maven Installation

In order to build and run the Maven projects the Eclipse Maven plugin must be installed prior to any other operation.

1. In Eclipse: Go to Help and click install new software. Make sure you have the box "Hide items that are already installed" ticked.
   (a) **Kepler**:
      i. Enter the url http://download.eclipse.org/releases/kepler and type the word Maven in the text filter.
2. In case Maven is not installed in your Eclipse application you should see 2 results from 2 different sections. Collaboration and general purpose tools. Expand the General Purpose Tools and click the m2e-maven Integration for Eclipse.
3. Click Next(2 times) and accept the license agreement. Finally click Finish.
4. You will be prompted with a window asking you to either restart Eclipse or Apply Changes now. Click restart now and wait until your Eclipse resumes.
5. To check that your installation was successful you can go to About Eclipse –> Installation details –> Installed software tab. Check if m2e-Maven integration in Eclipse is present.

3 Eclipse Setup

In order to use the JDK within the Eclipse IDE some additional configuration is required.

1. In Eclipse: Go to Preferences and expand Java tree.
2. Click Installed JREs and use the search option of Eclipse to locate the JDK 1.7.
3. As long as Eclipse locates the JDK 1.7 you can select it as default and click OK to apply your settings.

4 Downloading and importing Examples

1. Go to http://www.jsxm.org/jsxm-maven-plugin/ and locate the example that you want to download. The download link is located in the end of each examples' page.
2. Download the example and extract the compressed folder.
3. Go to Eclipse: File –> Import –>Existing Maven Project
4. Browse to the extracted folder of the downloaded example, select the folder and click Finish to import the JSXM example.

5 Using JSXM examples

Before running the examples: if any of the examples have an error indicator right click on the example project –>Maven–> Update Project.

1. Right click on any of the examples that you have downloaded Run As–> Maven Build ...–> enter "clean jsxm:classpath" in the goals section and click Run. Now the specification folder is treated as a source folder and allows easier navigation.

2. After modelling your system, you can use a graphical user interface for animating your specification and ensuring that the correct functionality is modelled. To use the above feature you should click Run As–> Maven Build ... –> enter "clean jsxm:animate-gui" in the goals section and click Run.

3. After animating your specification Right click on any of the examples that you have downloaded Run As–> Maven Build ...–> enter "clean jsxm:generate test" in the goals section and click Run.

4. You should see the test generation process and the execution of the tests. In case you want to change the k you can enter in the pom.xml and in the configuration section you can add <kInput></kInput> and set you desired input. The default k is 2.

5. After the successful build you can expand the **src/test/java directory** inside the example you are building and you will see the generated tests along with the adapter. In case you want to run the tests manually you can right click on **[specName]AdapterTest.java** and click Run as–>JUnit test. You should see all the tests running successfully.

*Examples eShop and Cart do not have implementation thus there are not Junit tests generated.

6 Creating your own examples (jsxm-archetypes)

6.1 jsxm-quickstart-archetype

The archetype provides the pom.xml settings for the successful execution of JSXM maven plugin. Furthermore the archetype generates the folder structure and an abstract example in order to provide a good insight of how a Stream X machine can be described using the JSXM language. The spec.xml and sets.xml are located inside the src/spec/your-package-name/specification. The spec folder is automatically added in the classpath as a classpathentry and is treated as a source folder in order to provide quick navigation in the example.

- In order to use the jsxm-quickstart-archetype you should follow the guide available at http://www.jsxm.org/jsxm-maven-plugin/archetypes/eclipse.html with the difference of selecting the jsxm-quickstart-archetype instead of the jsxm-example-archteype that is used by the guide.

7 General Settings

1. For viewing the line numbers: Preferences–>General–>Editors–>TextEditors–>Show line numbers

2. For changing the limited console output. Preferences–>Run/Debug–>Console–>Limit Console Output(uncheck)