Instructions to install and run the OSLC Simulink Adapter (Matlab R2013b)

By Axel Reichwein, March 19 2014

1. Installing Eclipse Lyo

Follow the instructions in the document named “Instructions to install Eclipse Lyo”. The document also contains instructions on how to use a proxy server with Maven and Eclipse.

2. Downloading oslc4jsimulink repository

1. Open the Git Repositories View (Window -> Show View -> type “Git Repositories” in the search field)
2. Click on the Clone Repository icon 
3. In the URI field, paste the following URL: git://git.eclipse.org/gitroot/lyo/org.eclipse.lyo.adapter.simulink.git
4. The Host and Repository fields will autofill. Leave the Username and Password fields empty.
5. Click Next until Finish.

3. Importing projects into the Eclipse workspace

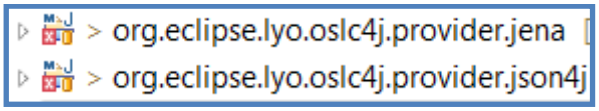
1. In the Git repositories view, right-click oslc4jsimulink and select “Import Projects”. Click Next until Finish
2. The 3 oslc4jsimulink projects are in the Eclipse workspace

4. Building the oslc4jsimulink projects

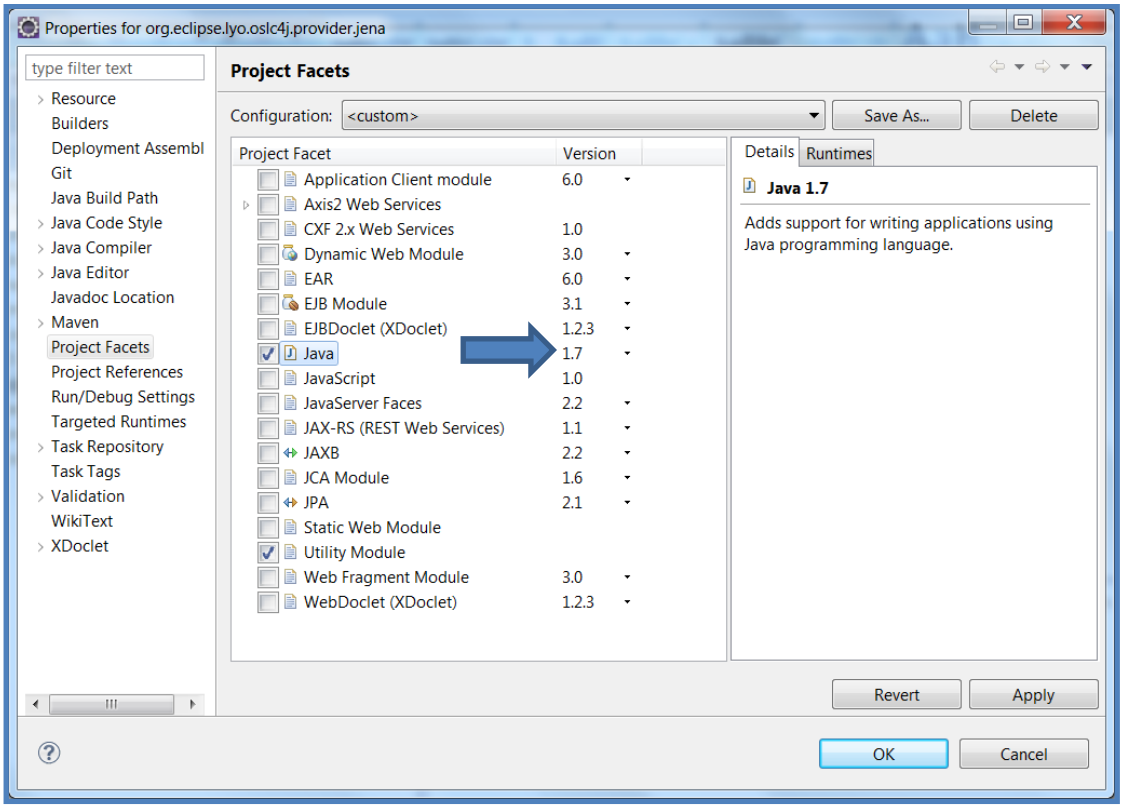
1. In Eclipse, open the Project Explorer view. (Window → Show View → Project Explorer)
2. Expand the **org.eclipse.lyo.adapter.simulink.ecore** project
3. Right click pom.xml -> Run As -> Maven clean
4. Right click pom.xml -> Run As -> Maven install
5. Expand the **org.eclipse.lyo.adapter.simulink.resources** project
6. Right click pom.xml -> Run As -> Maven clean
7. Right click pom.xml -> Run As -> Maven install
8. Expand the **org.eclipse.lyo.adapter.simulink** project
9. Right click pom.xml -> Run As -> Maven clean
10. Right click pom.xml -> Run As -> Maven install

If there is no error mark next to any project, you can skip the next steps.

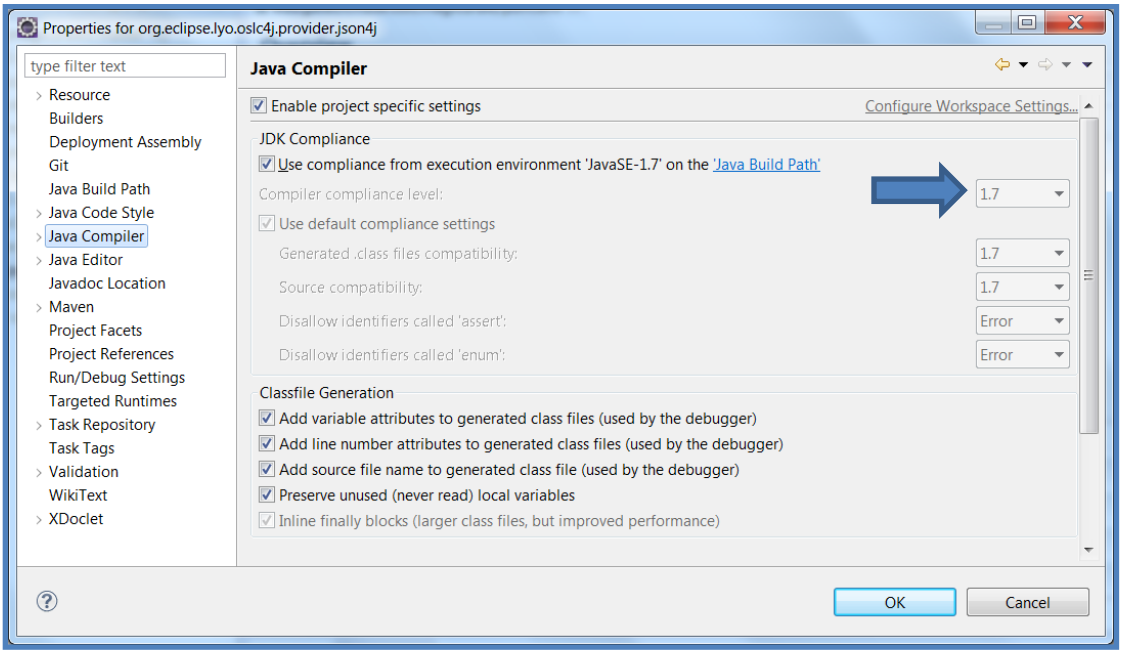
11. If there is a red error mark next to any project, select the project. Right-click->Maven->Update Project… and click OK
12. Make sure that the Eclipse projects displayed in the project explorer view do not contain any error icons displayed next to the project names as for example displayed below.



If a project has an error icon, then select the project and open its properties view (Project->right click->Properties). Under the Projects Facet tab, make sure that 1.7 is selected as Java version as shown below.



If a project still shows an error, then change its **JDK compliance** to 1.7. Select the project, right-click -> Properties. Select Java Compiler and select **1.7** in the drop down menu next to the JDK compliance setting as highlighted below.



5. Manual configuration

The OSLC Simulink adapter currently supports the retrieval of Simulink models within a specific directory. The location of the directory containing the Simulink projects is currently hard coded in a configuration file. Several Simulink models are already located in the ***matlab*** folder in the **org.eclipse.lyo.adapter.simulink** project.

1. Specify the **location of the folder containing Simulink models** which will be considered by the OSLC Simulink SysML adapter in the config.properties file under *org.eclipse.lyo.adapter.simulink /configuration*. As an example displayed below, the location of the folder containing Simulink models for the OSLC adapter is specified to **C:/Users/…/git/oslc4jsimulink / org.eclipse.lyo.adapter.simulink /matlab/**

   Note: The file path can contain backslashes

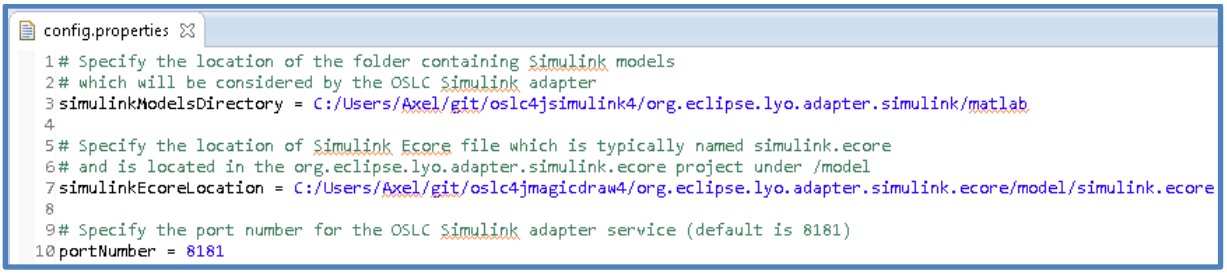
   Warning: Do not put quotes around the file path!

2. Specify the **location of Simulink Ecore file** in the config.properties file under *org.eclipse.lyo.adapter.simulink/configuration*. The location of the Simulink ecore file named simulink.ecore is in the **org.eclipse.lyo.adapter.simulink.ecore** project under /model/simulink.ecore. As an example displayed below, the location of the simulink.ecore file is specified to **C:/Users/…/git/oslc4jmagicdraw/ org.eclipse.lyo.adapter.simulink.ecore /model/simulink.ecore**

   Note: The file path can contain backslashes

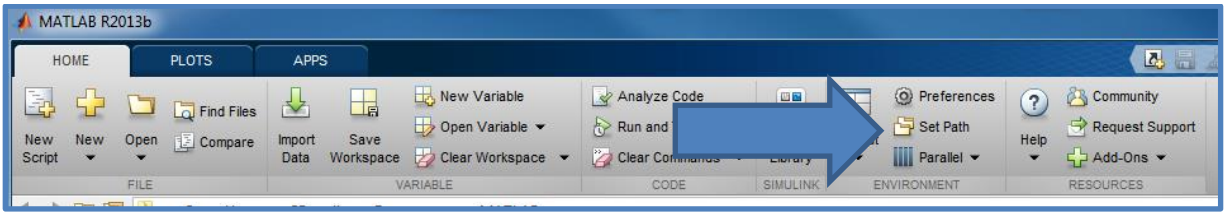
   Warning: Do not put quotes around the file path and add nothing at the end!

3. Specify the **port number** of the OSLC Simulink adapter service of in the config.properties file under *org.eclipse.lyo.adapter.simulink/configuration*. By default, port 8181 will be used. As an example displayed below, the port number is set to **8181**.

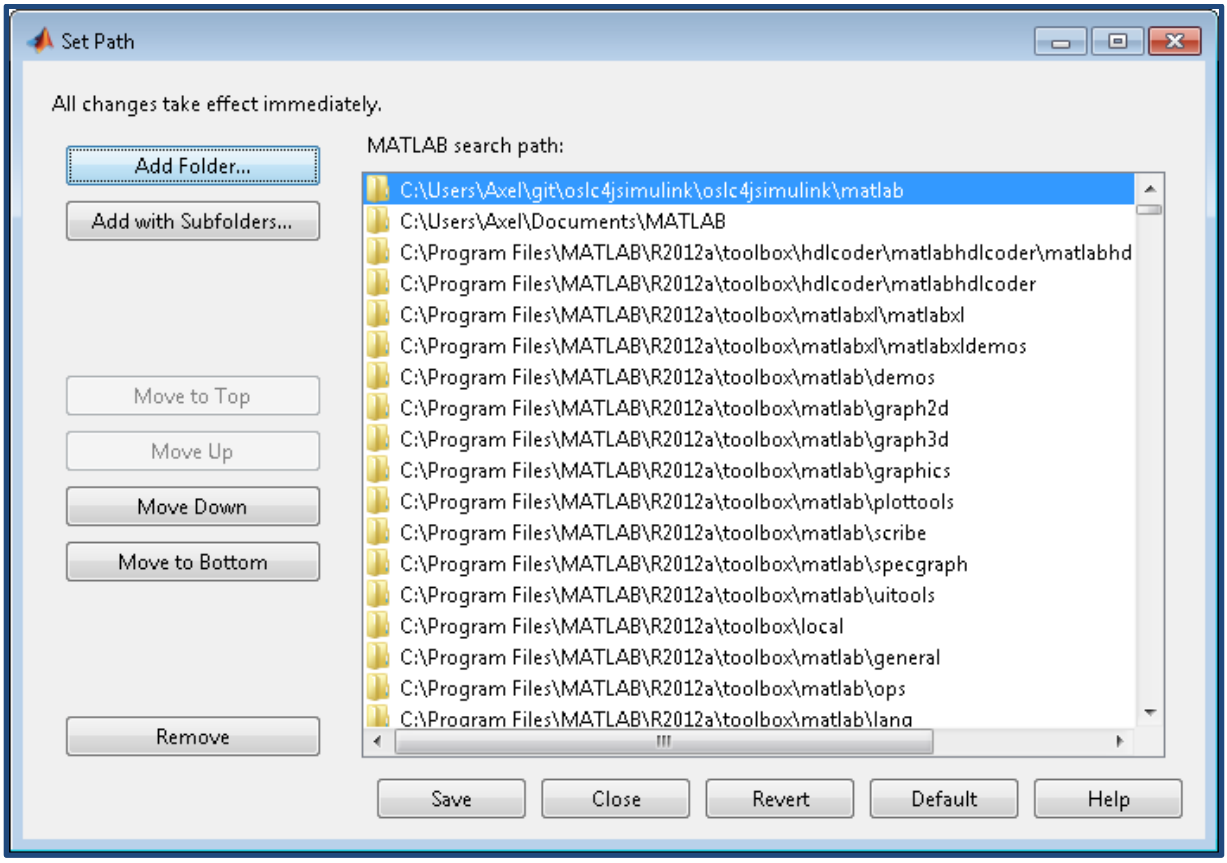


6. Setting up the Matlab workspace

1. Launch Matlab
2. From the File menu, select *Set Path…(*if your Matlab version has a ribbon-based user interface such as inR2013, choose *Set Path* in the *Home* ribbon in the *Environment* section*)*



3. Use the Add Folder… command to add the *matlab* folder of the *oslc4jsimulink* project to the Matlab search path as shown below based on the location of your local git repository
4. Click *Save* and then *Close*

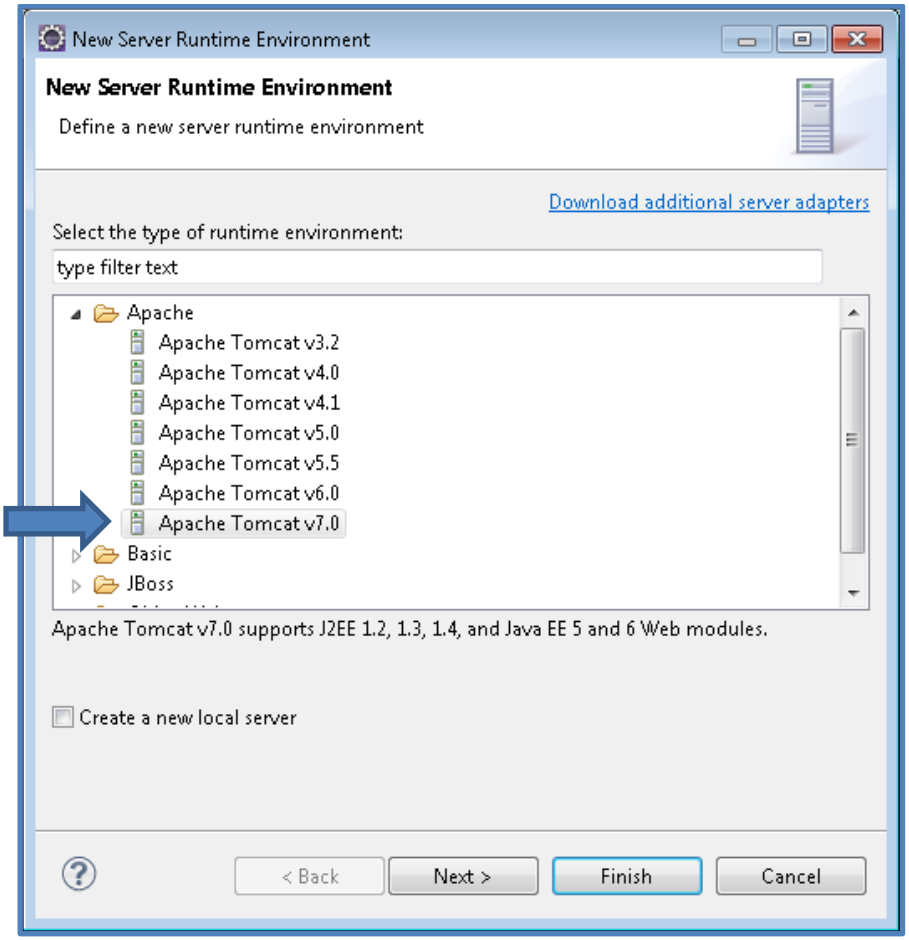


7. Installing Apache Tomcat

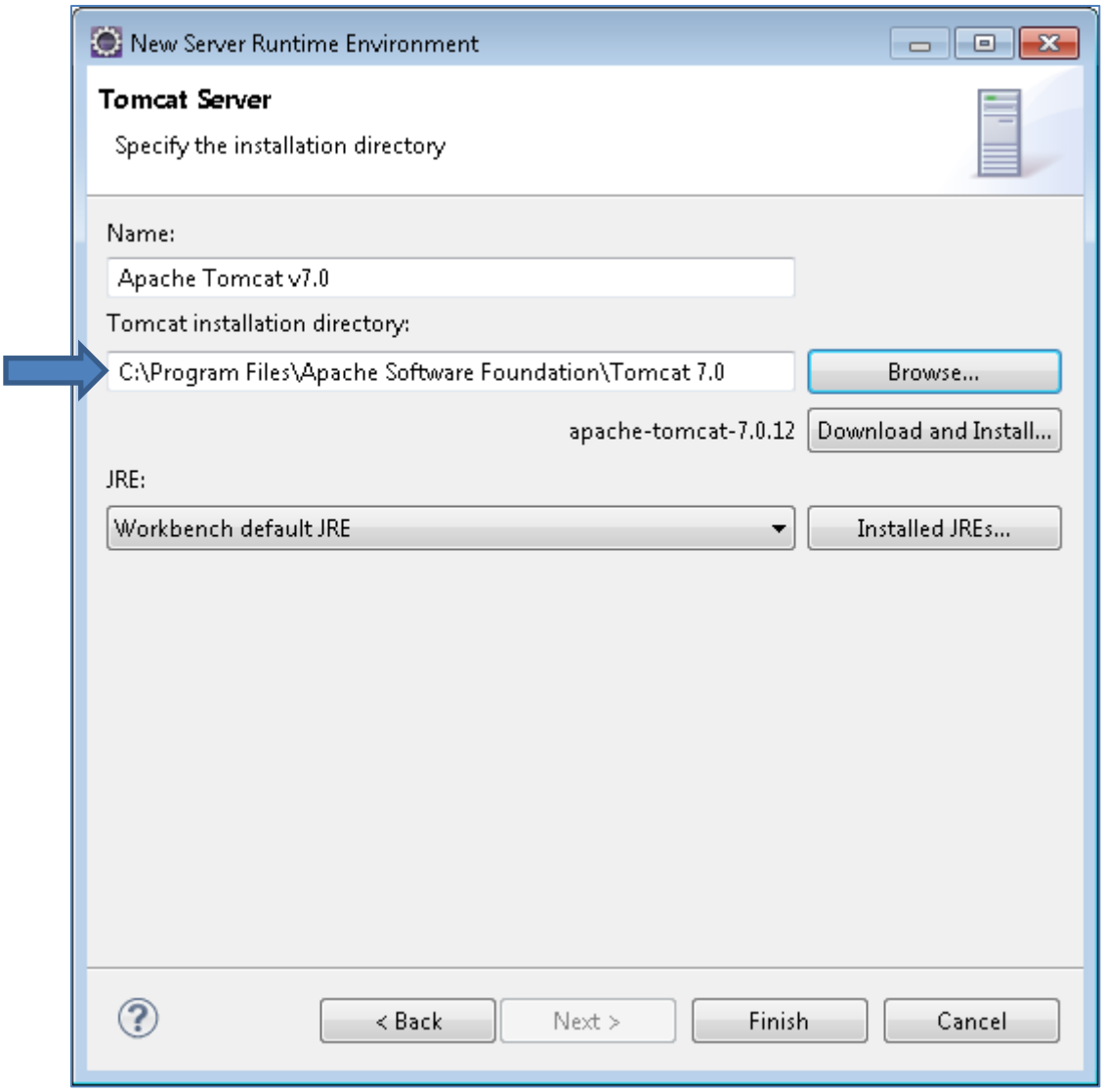
1. Go on the web page http://tomcat.apache.org/download-70.cgi
2. Download the apache 7 distribution for your operating system. If you are using Windows, the easiest is to select 32-bit/64-bit Windows Service Installer (http://tomcat.apache.org/download-70.cgi) and run the Apache Tomcat 7 installer

8. Adding Server Runtime Environment in Eclipse

1. In Eclipse. Open **Window -> Preferences -> Server -> Runtime Environments** to create a Tomcat installed runtime.
2. Click on **Add...** to open the New Server Runtime dialog,
3. From the drop down menu, select Tomcat 7.0 as shown below. Click Next.



4. Enter the **Tomcat 7.0 installation directory** (not the Apache installation directory!) as highlighted below.
5. Click on Finish.


New Server Runtime Environment
Tomcat Server
Specify the installation directory
Name:
Apache Tomcat v7.0
Tomcat installation directory:
C:\Program Files\Apache Software Foundation\Tomcat 7.0
Browse...
apache-tomcat-7.0.12
Download and Install...
JRE:
Workbench default JRE
Installed JREs...
< Back
Next >
Finish
Cancel

9. Enabling PUT on Apache Tomcat

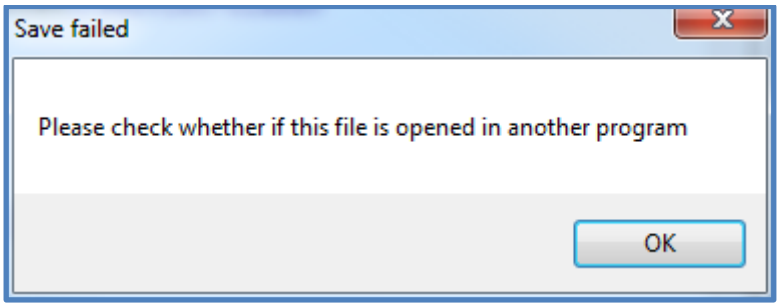
Tomcat by default is not enabled for HTTP PUT command. But, it can easily be configured to support it.

1. In your Apache Tomcat 7 installation directory, open **/conf/web.xml**
2. Add the ***readonly*** **init param** to the web.xml file as shown below and save the file

<init-param>
<param-name>readonly</param-name>
<param-value>false</param-value>
</init-param>

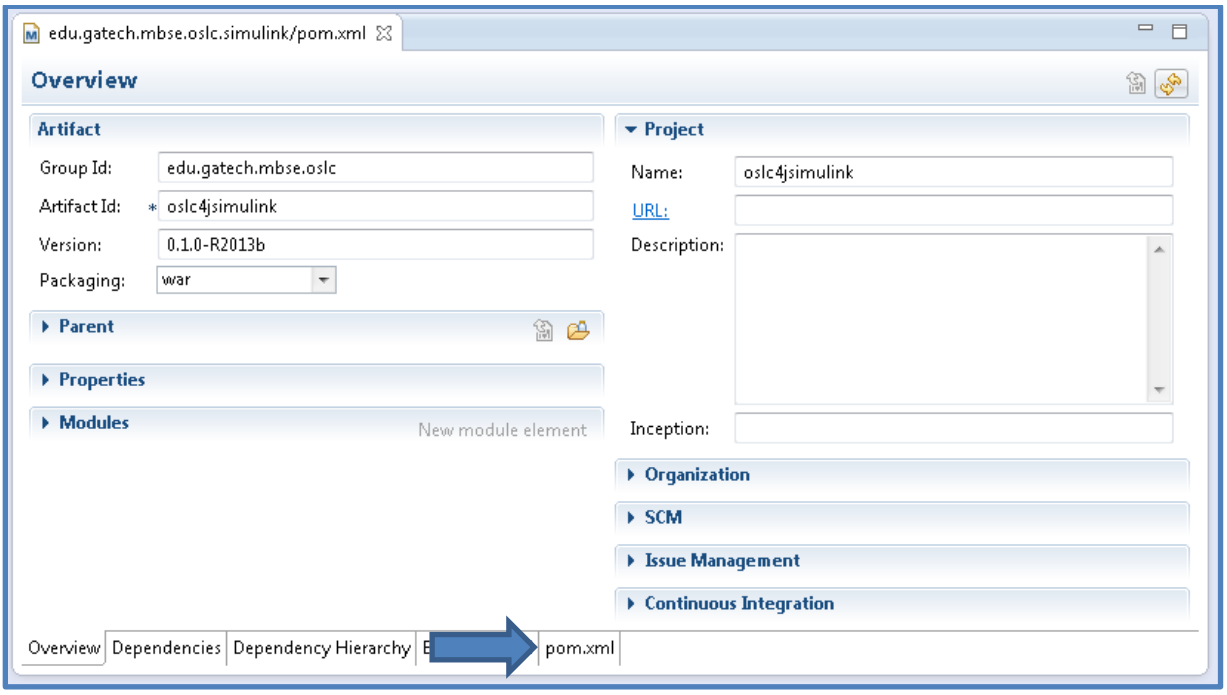
```
<servlet>
    <servlet-name>default</servlet-name>
    <servlet-class>org.apache.catalina.servlets.DefaultServlet</servlet-class>
    <init-param>
        <param-name>debug</param-name>
        <param-value>0</param-value>
    </init-param>
    <init-param>
        <param-name>listings</param-name>
        <param-value>false</param-value>
    </init-param>
    <init-param>
        <param-name>readonly</param-name>
        <param-value>false</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
</servlet>
```

Note: If you get the warning shown below while trying to save the file, then copy the web.xml file into another location, modify it, and then replace the original web.xml file by the modified web.xml file.



10. Setting the Apache Tomcat server port

1. By default, the OSLC Simulink adapter service will run on port 8181. Change the port of the oslc4jsimulink service only if you need to avoid a conflict with another service already running on port 8181. Skip the next steps if you do not need to change the port.
2. In Eclipse, open the Project Explorer view. (Window → Show View → Project Explorer)
3. Expand the **org.eclipse.lyo.adapter.simulink** project
4. Select and open the maven **pom.xml** file through double-click
5. The pom.xml file contains several tabs. By default, the overview tab will be displayed. The various available tabs are displayed at the bottom of the editor window. Click on the **pom.xml tab** of the pom.xml file as highlighted below.

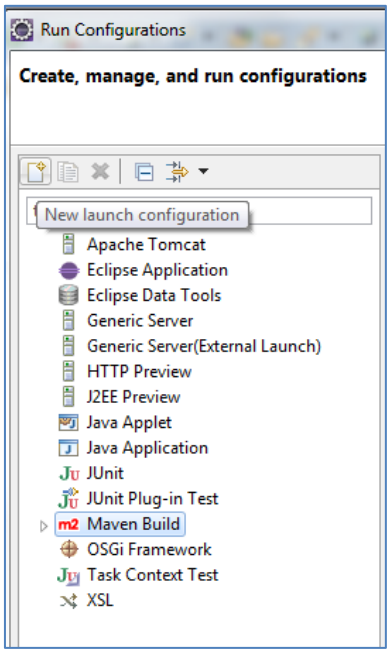


6. In the pom.xml tab of the pom.xml file, specify the **port of the OSLC Simulink adapter service** in the Maven tomcat plugin configuration found at the **bottom of the pom.xml** tab of the pom.xml file as highlighted below. Enter the port number in the configuration section as shown below.

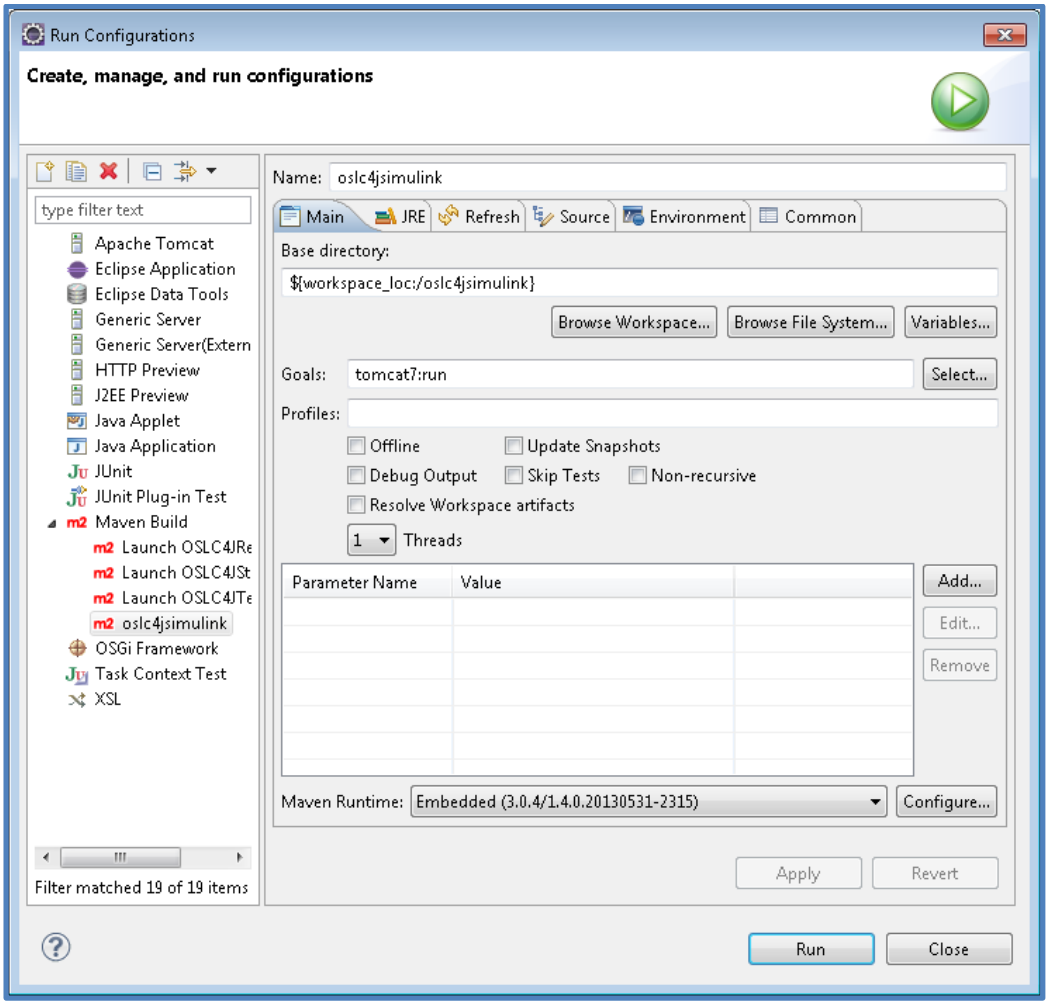
```
<build>
    <plugins>
        <plugin>
            <groupId>org.apache.tomcat.maven</groupId>
            <artifactId>tomcat7-maven-plugin</artifactId>
            <version>2.0</version>

            <configuration>
                <server>tomcat-development-server</server>
                <port>8181</port>
            </configuration>

        </plugin>
    </plugins>
</build>
```

11. Creating a launch configuration for oslc4jsimulink

1. Open the launch configuration window (Run -> Run Configurations…)
2. Select the “Maven Build category” and click on “New Launch Configuration” (leftmost icon at the top left, right above the searh field)

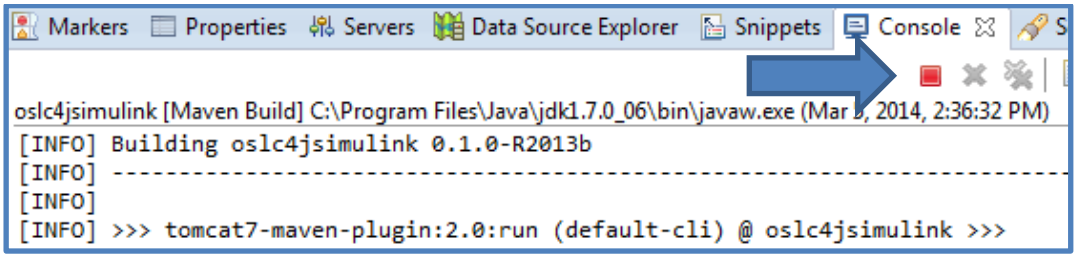


3. In the launch configuration window, edit the name field and set it to **oslc4jsimulink**.
4. Under “Base Directory”, select Browse Workspace and select the **org.eclipse.lyo.adapter.simulink** project
5. In the “Goals” field, set **tomcat7:run** as shown below



Note: Only run a single instance of the oslc4jsimulink web application at a time.

Note: In order to stop a running oslc4jsimulink web application, click Terminate in the Console window, as shown below, or in the toolbar of the debug perspective.



12. Installing the Chrome/Firefox Postman plugin (or any REST client)

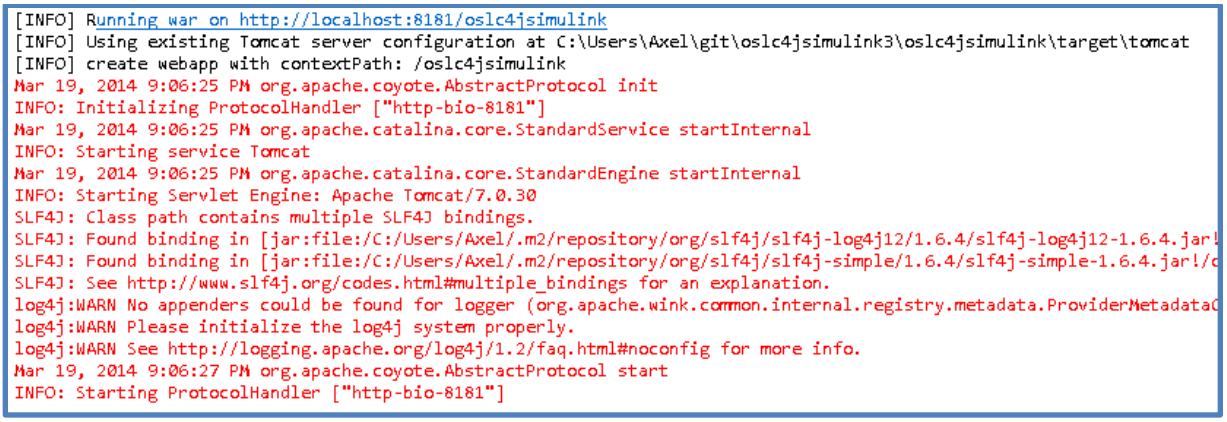
1. For Google Chrome, add the Postman REST client to your browser: https://chrome.google.com/webstore/detail/postman-rest-client/fdmmgilgnpjigdojojpjoooidkmcomcm?hl=en
2. And the Postman launcher: https://chrome.google.com/webstore/detail/postman-launcher/igofndmniooofoabmmpfonmdnhgchoka?hl=en

13. Launching oslc4jsimulink (OSLC Simulink Adapter)

Select the oslc4jsimulink launch configuration (Run -> Run Configurations… and select in the Maven build category the launch configuration named **oslc4jsimulink**) and click Run.

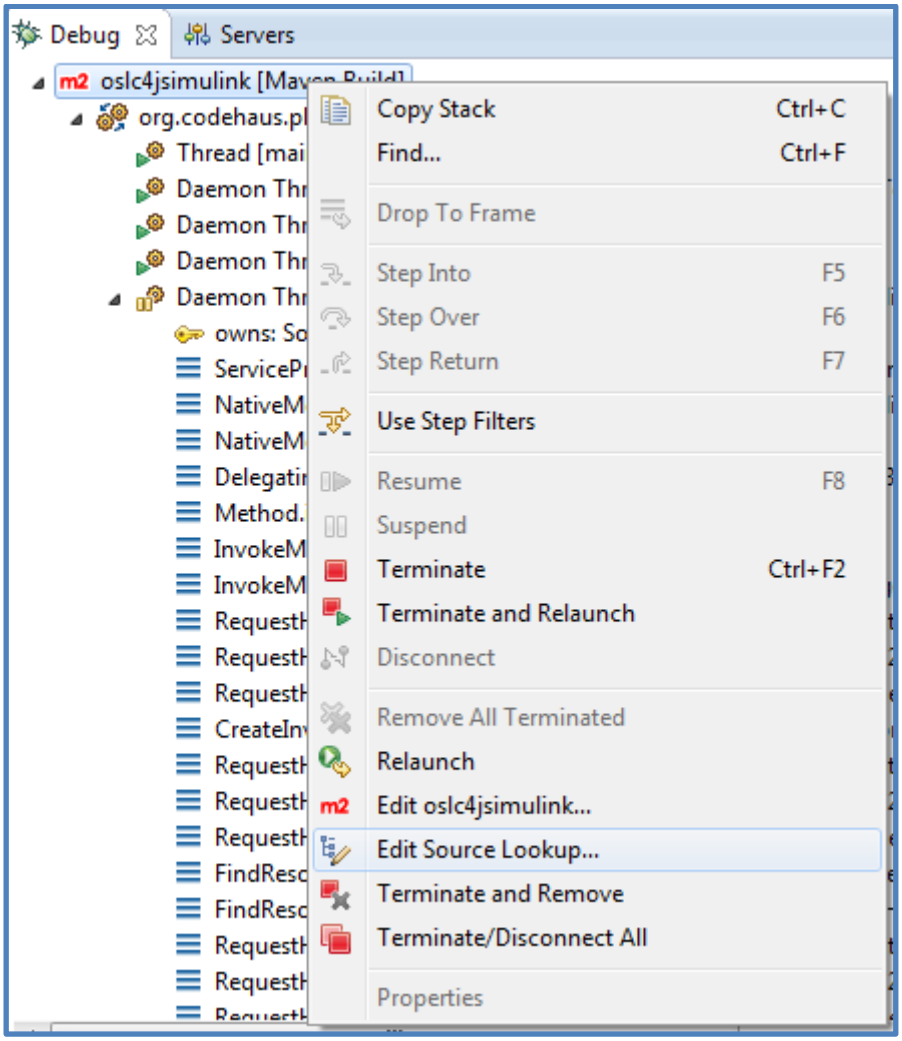
In the console window, several logging related exceptions will appear (SLF4J and log4j). This is not critical.

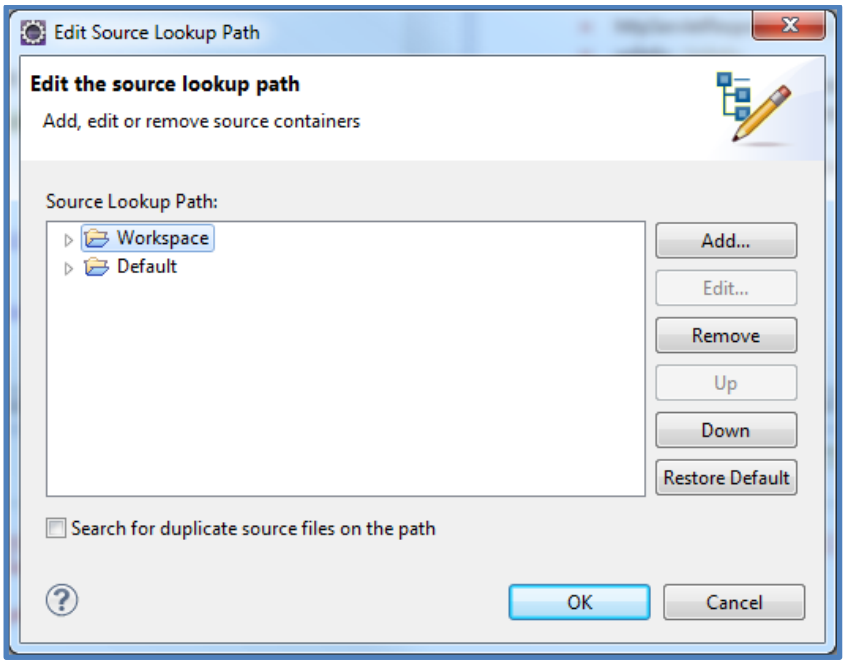
The OSLC Simulink adapter is running and following statements can be seen in the Eclipse Console windows displayed below



Warning: If the OSLC Simulink adapter service fails to launch due to a **java.net.BindException**, a different port for the OSLC Simulink adapter needs to be used since there is a conflict with another service using the same port. By default, the OSLC Simulink adapter uses port 8181. A **java.net.BindException** means that a different service is already using this port. Go back to Steps #5 and #10 to change the port number.

Note: If you launch the Maven launch configuration (OSLC Simulink adapter) in **debug mode**, and do not see the Java code when the application hits a breakpoint, then you need to add the Eclipse workspace to the source lookup path. In the Debug view, right click on the running thread (in threads tab), or on the application as shown below and select **Edit Source Lookup**, and add the workspace. Re-launch the Maven launch configuration and the code should be visible in the editor when the application hits a breakpoint.

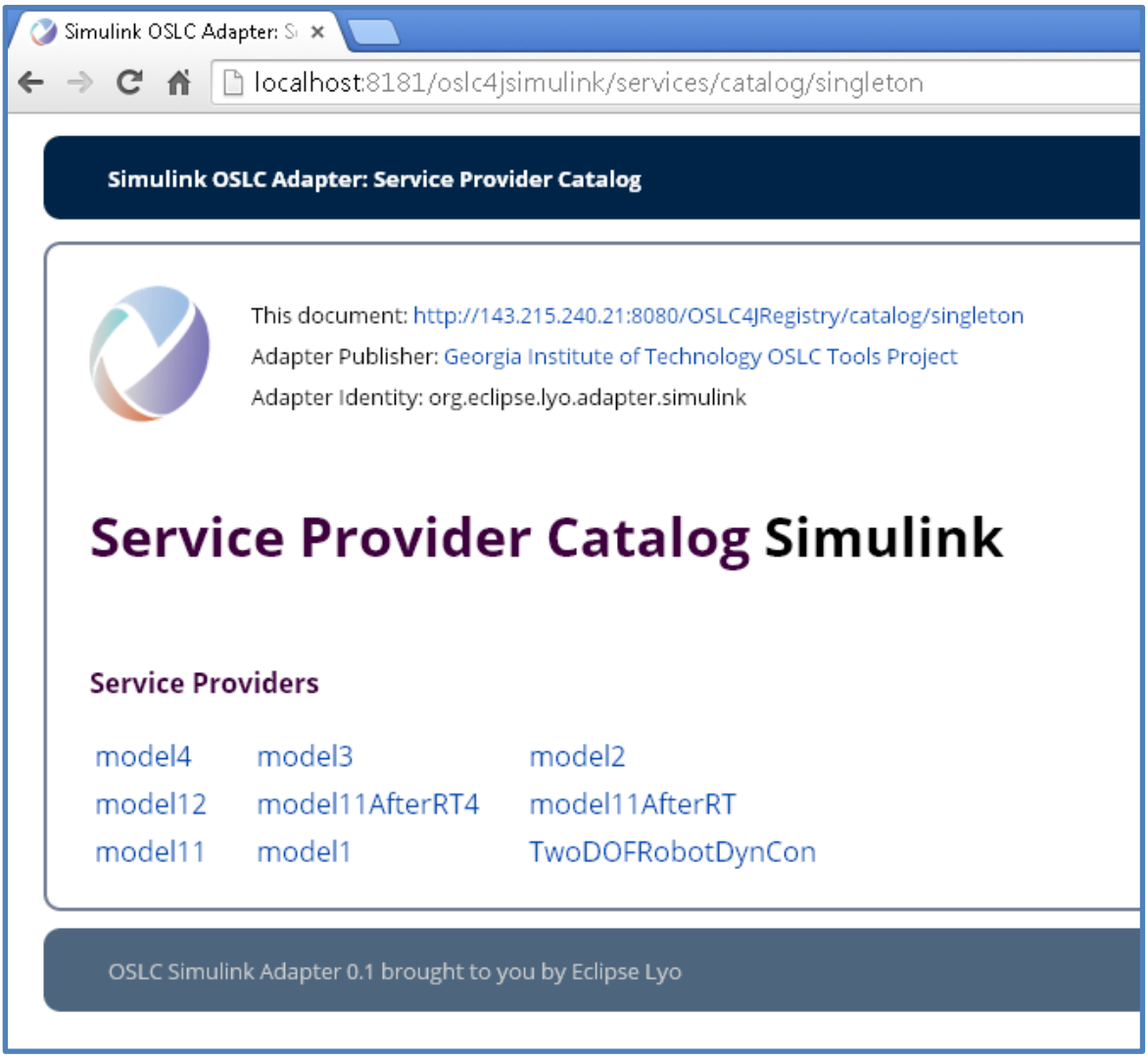

Debug
Servers
Copy Stack Ctrl+C
Find... Ctrl+F
Drop To Frame
Step Into F5
Step Over F6
Step Return F7
Use Step Filters
Resume F8
Suspend
Terminate Ctrl+F2
Terminate and Relaunch
Disconnect
Remove All Terminated
Relaunch
Edit oslc4jsimulink...
Edit Source Lookup...
Terminate and Remove
Terminate/Disconnect All
Properties


Edit Source Lookup Path
Edit the source lookup path
Add, edit or remove source containers
Source Lookup Path:
Workspace
Default
Add...
Edit...
Remove
Up
Down
Restore Default
Search for duplicate source files on the path
OK
Cancel

14. Testing the OSLC Simulink Adapter

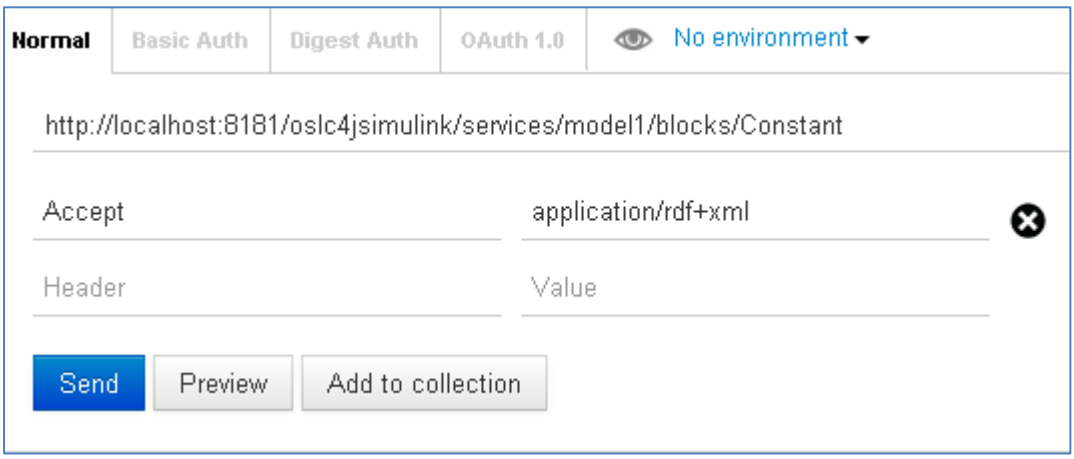
Testing the retrieval of OSLC resources in HTML

1. Launch Google Chrome
2. In the URL field, type for test purposes: http://localhost:8181/oslc4jsimulink/services/catalog/singleton. This will send a HTTP GET request to retrieve the HTML representation of the Simulink Service Provider Catalog. This will launch a Matlab command window which will close automatically. The Matlab command window may display warnings if it is an older version than R2013b.
3. You will then see an HTML page showing you the list of Service Providers. **You can browse from the Service Providers (e.g. for model11) to the Services and ultimately to the OSLC Simulink resources.**



Testing the retrieval of OSLC resources in RDF

1. Click on the Postman icon at the top right of the Chrome browser  . A new tab will open.
2. In the URL field, type for test purposes: http://localhost:8181/oslc4jsimulink/services/model1/blocks/Constant.
3. Click on the *Headers* field to the right of the URL field
4. Enter *Accept* in the Header field and *application/rdf+xml* in the value field as shown below
5. Click Send



6. This will send a HTTP GET request to retrieve the RDF/XML representation of the Simulink block named “Constant”.

The Postman REST client will display the RDF/XML representation of the Simulink block named “constant”. Other HTTP requests to retrieve other SysML elements can be sent.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF

    xmlns:simulink_line="http://mathworks.com/simulink/rdf#Line/"

    xmlns:simulink="http://mathworks.com/simulink/rdf#"

    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"

    xmlns:oslc_rm="http://open-services.net/ns/rm#"

    xmlns:oslc="http://open-services.net/ns/core#"

    xmlns:oslc_data="http://open-services.net/ns/servicemanagement/1.0/"

    xmlns:foaf="http://xmlns.com/foaf/0.1/"

    xmlns:simulink_parameter="http://mathworks.com/simulink/rdf#Parameter/"

    xmlns:simulink_outputport="http://mathworks.com/simulink/rdf#Outputport/"

    xmlns:dcterms="http://purl.org/dc/terms/"

    xmlns:simulink_block="http://mathworks.com/simulink/rdf#Block/"

    xmlns:simulink_model="http://mathworks.com/simulink/rdf#Model/"

    xmlns:simulink_inputport="http://mathworks.com/simulink/rdf#Inputport/"

    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#" >
    <rdf:Description rdf:about="http://localhost:8181/oslc4jsimulink/services/model1/blocks/Constant">
        <simulink_block:parameter rdf:resource="http://localhost:8181/oslc4jsimulink/services/model1/parameters/Constant::SamplingMode"/>
        <simulink_block:parameter rdf:resource="http://localhost:8181/oslc4jsimulink/services/model1/parameters/Constant::FramePeriod"/>
        <simulink_block:parameter rdf:resource="http://localhost:8181/oslc4jsimulink/services/model1/parameters/Constant::OutDataTypeStr"/>
        <simulink_block:parameter rdf:resource="http://localhost:8181/oslc4jsimulink/services/model1/parameters/Constant::SampleTime"/>
        <simulink_block:parameter rdf:resource="http://localhost:8181/oslc4jsimulink/services/model1/parameters/Constant::OutMin"/>
        <simulink_block:outputPort rdf:resource="http://localhost:8181/oslc4jsimulink/services/model1/outputports/Constant::outport::1"/>
        <simulink_block:parameter rdf:resource="http://localhost:8181/oslc4jsimulink/services/model1/parameters/Constant::Value"/>
        <rdf:type rdf:resource="http://mathworks.com/simulink/rdf#Block"/>
        <simulink_block:parameter rdf:resource="http://localhost:8181/oslc4jsimulink/services/model1/parameters/Constant::VectorParams1D"/>
        <simulink_block:name rdf:datatype="http://www.w3.org/1999/02/22-rdf-syntax-ns#XMLLiteral">Constant</simulink_block:name>
        <simulink_block:parameter rdf:resource="http://localhost:8181/oslc4jsimulink/services/model1/parameters/Constant::LockScale"/>
        <simulink_block:type rdf:datatype="http://www.w3.org/1999/02/22-rdf-syntax-ns#XMLLiteral">Constant</simulink_block:type>
        <simulink_block:parameter rdf:resource="http://localhost:8181/oslc4jsimulink/services/model1/parameters/Constant::OutMax"/>
    </rdf:Description>
```

15. Testing the OSLC Simulink Adapter through example Simulink model

The org.eclipse.lyo.adapter.simulink project contains example Simulink models containing different types of Simulink elements. The example models are located in the folder named **matlab**.

model11 contains blocks, subsystems, model reference blocks, ports, lines, and lines with multiple target ports