Setup for CTS2 Framework Plugin:

Required:

Java 1.6 or newer

MySQL 5.16

Maven 2.2.1

Eclipse (Instructions for Intellij Community Edition are also included)

Gene Ontology Data Base dump

Download and install MySQL

<http://www.mysql.com/downloads/mysql/5.1.html>

I used all the defaults except I pushed the character encoding to UTF-8 and included the option for the command line client

Download and install Maven:

<http://maven.apache.org/download.html>

We used version 2.2.1 (compatible with Intellij 11.1’s plugin).

This note on Maven’s install instructions in Windows 7: You may need to create the environmental variables in the system, rather than user, variables in order to get Maven working from the command line. To be sure your maven install works create all the recommended environmental variables including JAVA\_HOME.

We’ll be using eclipse, but Included are instructions for Intellij

Intellij or Eclipse:

<http://www.jetbrains.com/idea/download/>

<http://www.eclipse.org/downloads/>

Intellij’s Maven integration works better for some than Eclipse. You can install Eclipse’s M2e plugin for maven if you don’t mind ignoring the errors it generates. We’ll be working with Eclipse for the most part during the tutorial. We’ll also use the M2e maven plugin to add dependencies. Code snippets and other materials for the tutorial are available in a zip file here:

From gitHub:

<https://github.com/hsbauer/cts2-example-service/archive/master.zip>

Take note that this file downloads as cts2-example-service-master.zip

Gene Ontology database import (from the zip file):

This is an ontology chosen for it’s small size and relatively rich ontology metadata table. It’s not the full Gene Ontology. You’ll need to create a new database named “godata” and import it into MySQL;

From the MySQL command line client you can execute:

Create database godata;

And then:

Mysql –u “your username” –p”your password “godata < database.dump.name

Download and copy to a convenient directory the cts2 standalone framework server:

http://informatics.mayo.edu/cts2/framework/downloads/cts2framework-standalone.jar

**Eclipse instructions:**

Install Eclipse and its Maven plugin.

Extract the example-service zip to the Eclipse workspace folder

Create a new project named example-service and you should see the project structure of the new project displayed in the package explorer.

Right click on the project and select Configure/Convert to Maven Project

Your pom file will have errors but they can be safely ignored.

If any Java files still have errors, Right click the project and try running Maven => update project

Otherwise

Right click the project:

Choose run as => Maven Clean

Then run as => Maven Install

**Follow the Slides from Here**

From the cts2-example-service/example service zip, get the JDBCConnection.java file and place it in the source folder.

If you run the main class here it will fail. Pull in a connector with adjustments to the pom file.

Run the main class in the JDBC connector again and it should give you metadata about the GO ontology.

Now we want to add this to the build. Using additions to the pom file we'll add the connector to a maven build.

Do a clean install sequence

from whatever directory you choose to install the cts2 framework standalone server, run java -jar cts2framework-standalone.jar

And login as admin/admin

Choose install update from the bar at the top of the list of plugins and find the jar constructed in the maven build target folder.

Refresh the browser

Click the play/stop button next to the example service plugin.

Type localhost:8080/codesystem/GO

This should give you a hard coded response

replace the ExampleCodeSystemCatalogReadService method read with the one in the code fragments:

Resolve any new dependencies if needed.

run clean install and upload changes to the framework.

Once again we'll type localhost:8080/codesystem/GO

This time the service goes to the database and returns GO metadata.

The slides will provide a step through of the Eclipse process. If you like you can use Intellij for the same implementation.

**Intellij:**

Create a new project:

* add a java SDK to the project
* copy source (src) directory and pom files to the project
* Right click on src directory and go to Mark Directory and choose Root Source.
* Right click on the pom.xml file and select add the project as Maven Project

Using the Maven Project view, do a clean install.

* Both should return a value of 0.
* You should see a small green popup in the upper right corner.
* Click on the upload link to upload all dependencies to maven so the application can be run.

From the cts2-example-service/example service zip, get the JDBCConnection.java file and place it in the source folder.

* If you run the main class here it will fail.
* Pull in a connector with adjustments to the pom file.
* Run the main class in the JDBC connector again and it should give you metadata about the GO ontology.

Add the MySQL connector to the build.

* Using additions to the pom file, add the connector to a maven build.

Do a clean install

Start up the stand alone server.

* from whatever directory you choose to install cts2 standalone framework server, run java -jar cts2framework-standalone.jar
* login as admin/admin
* Choose install update from the bar at the top of the list of plugins and find the jar constructed in the maven build target folder.
* Refresh the browser
* Click the play/stop button next to the example service plugin.

If there are no errors returned to the terminal window then the plugin has successfully installed. Time to test the plugin:

* Type localhost:8080/codesystem/GO
* This should give you a hard coded response

We’ll want a more dynamic interaction with a data source:

* replace the ExampleCodeSystemCatalogReadService method read with the one in the code fragments:
* Use alt enter to resolve dependencies.
* run clean install and upload changes to the framework.
* Once again we'll type localhost:8080/codesystem/GO
* This time the service goes to the database and returns GO metadata.