Development Support:Getting Started

From pitawiki

This is a guide for setting up standard development tools, and also the specific setup instructions for SLP.

Essentially, it is a guide for anyone trying to use a pre-configured Eclipse bundle (Eclipse + plugins + basic configuration).

Note for Non-developers: Non-developers using Eclipse mainly for SVN access only need to follow the sections indicated with an asterisk (*).

Most of these steps require copying/extracting files using Explorer (Window's native file explorer.) This can be opened by opening any folder or pressing **<Windows key>+E**

Contents

Installing Firefox

If you prefer using Firefox over IE, Firefox and other programs authorized to be installed on DevNet machines (e.g. 7z) are available at

```
http://devcas024/dnsr/
```

Installing Java*

Make sure you have a Java JDK installed. At the time of this writing, the latest recommended version is 1.6.27. I suggest installing only the 64 Bit version. Use the default install locations.

- \\devslo020\data\devsupport\installers\java\jdk-6u27-windows-x64.exe

Add a new *system* variable: **JAVA_HOME** set to **C:\Program Files\Java\jdk1.6.0_27**

You could also add (append) **%JAVA_HOME%\bin** to the **PATH** variable so you could run java, javac, and keytool, etc. from the commandline.

```
Checkpoint...
CDM -- At this point you should have:
* C:\Program Files\Java\jdk1.6.0_27
* C:\Program Files\Java\jre6
```

Installing and Configuring Maven

Software developers should install Maven. Other users (like MUS group or individuals using eclipse primarily for SVN access) do not need to install Maven.

Maven builds the project (handles compiling, generating the final packages, etc) and handles dependencies.

Installing Maven

- Create a directory named "maven" on your C drive (ex -- C:\maven)
- Extract this Maven zip file into the directory you created.

  Type this into explorer.

```
\\devslo020\data\devsupport\installers\maven\apache-maven-3.0.3-bin.zip
```

  Note: If you want to be able to run mvn outside of eclipse (i.e. from the commandline), add the maven install location to you PATH environment variable.

Configuring Maven

- Create the directory **c:\users\<user_name>\.m2**, where <user_name> is your username on the machine.
- Copy the file **settings.xml** from *\\devslo020\data\devsupport\settings.xml* to *c:\users\<user_nam>\.m2\settings.xml*

- Add another new system variable in the **Environment Variables** dialog
  - Set the variable name to **MAVEN_OPTS**
  - Set the variable value to **-Xms256m -Xmx1024m -XX:MaxPermSize=256m**

```
Checkpoint...
CDM -- At this point you should have:
* C:\
   |- maven
      |- apache-maven-3.0.3
         |- bin
         |- boot
         |- conf
         |- lib
         |- LICENSE.txt
         |- NOTICE.txt
         |- README.txt
```

Maven Proxy Fix

The cert for the nexus server needs to be added to Java's accepted certificates.

1. If you are using Java 6
   1. Copy the directory \\devslo020\data\devsupport\PROXY_FIX to your local machine.
   2. Run the installcert.bat file. A file named “cacerts” will be generated in the “PROXY_FIX” directory.
2. If you are using Java 7
   1. Copy the directory \\devslo020\data\devsupport\PROXY_FIX_JAVA7 to your local machine.
   2. Run the installcert.bat file. A file named “cacerts” will be generated in the “PROXY_FIX_JAVA7” directory.
3. Replace the “cacerts” file found in your <jdk home>/jre/lib/security with the newly generated file.

   Note: Replacing cacerts may require an admin privilage if Java was already installed on the new machine.

Note: <jdk home> is the JDK being used by your Eclipse and/or Maven. If you use multiple JDKs (i.e. one 32-bit and one 64-bit or just varying versions) and you will be executing maven builds with those JDKs, the new "cacerts" file needs to be placed in each.

Additional Java 7 VM Argument

1. Open Eclipse.
2. Go to "Windows"->"Preferences".
3. Go to the "Java"->"Installed JREs" node.
4. Select your Java 7 jdk and click "Edit". Note: If it does not appear, you will need to add it first.
5. In the "Default VM Arguments" field, add "-Djsse.enableSNIExtension=false" (without quotes).
6. Click "Finish and then "OK" to exit the dialogs.

Installing Eclipse*

For ease of installation, you just have to extract a bundled zip file. These bundles include an Eclipse installation, a default Eclipse workspace, and a shortcut to start up Eclipse.

To install Eclipse:

1. The bundle **eclipse-bundle.zip** is located at \\devslo020\data\devsupport\
2. Extract the bundle into c:\. The bundle contains a topmost directory named **eclipse**, so you should end up with the bundle under c:\eclipse

Checkpoint... At this point you should have:

```
* C:\
  |- eclipse
     |- eclipse-installations
     |- eclipse-shortcuts
     |- eclipse-workspaces
```

Under each directory you should have either a directory or shortcut with the name of the bundle which you extracted. For example, if you extracted the bundle named 'eclipse-jee-helios-subv', your directory would look like this:

```
* C:\
  |- eclipse
     |- eclipse-installations
        |- eclipse-jee-helios-subv
     |- eclipse-shortcuts
        |- eclipse-jee-helios-subv workspace (Note: This is not a directory, it is a shortcut).
     |- eclipse-workspaces
        |- eclipse-jee-helios-subv
```

You should now be able to start using your Eclipse. Navigate into the

```
C:\eclipse\eclipse-shortcuts
```

directory and double click the shortcut. Your Eclipse should start up and be pre-configured with your Java, Maven (if applicable), and Subversion.

You may copy the shortcut to a more convenient location.

This ends the instructions for the basic set up of eclipse for non-developers. Developers should continue on to the next section.

Installing Tomcat

Most developers will also need to have Tomcat installed.

1. Download the tomcat zip 'apache-tomcat-6.0.33-windows-x64.zip' from **http://devcas024/dnsr/**
2. Extract to the desired directory. Ex: c:\

```
Checkpoint...
--At this point you should have:
* C:\
  |- apache-tomcat-6.0.33
```

Setup Tomcat within Eclipse

If you plan to use the Tomcat plugin in eclipse, you will need to configure Eclipse to point to this installation directory. Once you have completed the step to install Eclipse:

1. Go to 'Window'->'Preferences'

2. Select 'Tomcat' from the left-hand pane of the preferences dialog.
3. Under 'Tomcat version', select the 'Version 6.x' (or the version you used) radio button.
4. Set the 'Tomcat home' to point to your Tomcat installation (c:\apache-tomcat-6.0.33\).
5. Check "Contex files" and put **c:\apache-tomcat-6.0.33\conf\Catalina\localhost** in **Context Directory**
6. Expand the 'Tomcat' node in the left-hand pane.
7. Select 'JVM Settings'.
8. Make sure jdk1.6... (the version installed) is selected under JRE.
9. Copy and paste the the following parameters (line by line) into the 'Append to JVM Parameters' field:

TODO - Note: A file with these are also available at \\devslo020\data\devsupport\

```
-Djava.endorsed.dirs=${tomcat_home}\common\endorsed
-Djava.io.tmpdir=${tomcat_home}\temp
-Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
-Djava.util.logging.config.file=${tomcat_home}\conf\logging.properties
-Djavax.net.ssl.trustStorePassword=changeit
-Djavax.net.ssl.trustStore=conf\truststore.jks
-XX:MaxPermSize=384m
-Xmx1024m
-Dgs.home=${tomcat_home}\temp
-Dciw.url=https://fusion:443
```

1. Expand the 'Run/Debug' node in the left-hand pane.
2. Select 'String Substitution'
3. Create a new variable 'tomcat_home' pointing towards to tomcat install directory. Ex: c:\apache-tomcat-6.0.33

Setup Tomcat for SSL

Most CDM applications require an https connections. The following will describe how to create a local keystore and configure tomcat to allow https connections. Note: The following commands assume {tomcat.home} is the tomcat home directory.

- Create the .keystore file by executing the following command:

```
%JAVA_HOME%\bin\keytool -genkey -alias tomcat -keyalg RSA -keystore {tomcat.home}\conf\keystore.jks
```

  - Set the keystore password to 'changeit'
  - For "What is your first and last name?" enter `localhost`.
  - You can use the defaults for the remaining prompts.

- Copy \\devslo020\data\devsupport\Certificates\Servers\truststore.jks into {tomcat.home}\conf\

- Edit tomcat server config file server.xml (located in {tomcat.home}\conf\)

  - Ensure that a connector is setup for your keystore. You should have a connector that includes the following information: port="443" keystoreFile="conf/keystore.jks" keystorePass="changeit"

- Edit database login info in server.xml

  - Enter your username in **https://fusion/ciw-sso-tools** -> **Encrypt Message** to generate a database password.

Verify you now have a working https tomcat server:

1. Start tomcat via the eclipse tomcat plugin in Eclipse.
2. Open the url: *https://localhost/* and you should see an apache tomcat homepage.

Enable Tomcat DevLoader (Optional)

Do this after checking out the code from SVN and importing the maven project in eclipse.

1. To enable the eclipse tomcat plugin's DevLoader (useful for debugging and hotswapping), select open the project's properties.
2. Select the "Tomcat" category on the left.
3. Select the "DevLoader Classpath" tab.
4. Check "Activate DevLoader".
5. Check any project directories or jar files to specifically enable hotswapping.
6. Copy the DevLoader jar to your '${tomcat_home}/lib' as the standard tomcat installation does not contain it.

The DevLoader jar can can be found in your tomcat plugin:

```
${eclipse_installation}/plugins/com.sysdeo.eclipse.tomcat_3.3.0/DevloaderTomcat7.jar
```

Installing Flash and Flashbuilder

Flex developers will need to install the stand-alone Flash Builder 4.5. An installer may be available on your local machine under C:\installers. If that is not present or does not work, go to **\\devslo020\data\devsupport \installers\Adobe Flash Builder 4.5**.

If you are not developing in Flex, you still need Flash Player. You can download and install the Flashplayer 10.2 Plugins for both Firefox (with "pl" in name) and IE (with "ax" in name) from **\\devslo020\data\devsupport \installers\Flash 10.2**.

Update FlashBuilder.ini

- Open up FlashBuilder.ini from the C:\Program Files(x86)\adobe\Adobe Flash Builder 4.5 directory
- Modify the -Xmx512m entry to -Xmx718m
- Modify the -XX:MaxPermSize=256m entry to -XX:MaxPermSize=512m
- Modify the -XX:PermSize=64m entry to -XX:PermSize=128m
- Save your changes

NOTE: You may need to save your changes to a different location then copy the file over the existing file so that you are prompted for you DevNet Admin password.

Installing maven plugin for FlashBuilder

1. Run FlashBuilder as Administrator.
2. Under the Help menu, click 'Install New Software...'
3. Type this URL in the "Work with" box and press enter: `http://sloeclipseupdates/flashbuilder`

4. Select the *Maven Integration* category to install all 3 items: *m2e - Maven Integration for Eclipse*, *flexmojos connector for m2e*, and *cdm-version connector for m2e*.
5. Click 'Next>' on this and the next page to get to the license agreements page.
6. Accept and click 'Finish'
7. Click 'OK' through a security warning about unsigned content and restart Flash Builder
8. Under 'Window'->'Preferences'->'Maven' un-check 'Download repository index updates on startup'
9. Expand the 'Maven' node. Under Installations, make sure that Flash Builder is using the version of Maven installed in a previous step (Maven 3.0.3 as of this writing)
10. Under the 'User Settings' node, make sure the settings file is found.
11. Click 'OK'

Stop here if you only want a standard development environment.

Checking out, importing, building and running the SLP project

Follow these instructions for setting up SLP.

Checking out the code from SVN

One way to checkout the code from the repository is using TortoiseSVN. The other way is from Eclipse (This is left as an excercise.)

Create a directory for the code, for example, **c:\repo**. In explorer, right click on the directory and click "SVN Checkout..." The repositary is

```
https://devslo038/massive/Trunk
```

Assuming your SVN account has been set up, use the same credidentials as your devnet machine.

The code should download after a short pause.

Importing the project in Eclipse

In the left pane of Eclipse (Project or Package Explorer) right click then choose Import->Import... Chose Maven -> Existing Maven Project.

Navigate to the directory where you checked out your code. e.g. **c:\repo**. You don't need all the projects, but you will need **slp-war/pom.xml**, **slp-db/pom.xml** and **SLP/pom.xml** under SLP (near the end.) Ask your mentor what project(s) to import.

Install Browser User Certificates

The user certificates are for the web browser (IE or Firefox).

```
\\devslo020\data\user_certs\
```

If you don't find yours ask your mentor which ones you may use.

Configure Tomcat for SLP

In Eclipse, assuming the **slp-war** project is imported:

- Right click on it and select **Properties**.
- Under "Tomcat", check "Is a Tomcat Project."
- Enter **SLP** in the "Context Name" field.
- Enter **/target/SLP** in the "Subdirectory to set as web application root (optional)" field.
- In the "Extra Information" block, add the configuration from here: (and replace the ServerName value with your machine name (get it from rightlicking on start->Computer))

TODO - have these settings emailed

- Click "OK"
- Richt-click on **slp-war**. Click "Tomcat project" -> "Update context Defenition

Install Tomcat dependencies

Four more jars are required by to be in tomcat's lib/ folder:

```
ojdbc14
slf4j-api
log4j.jar
ciw-crypto
```

TODO - for now ask James to email them to you.

Creating a Build Configuration in Eclipse

In Eclipse, click Run -> Run Configurations -> Maven Build.

Create a new configuration by clicking on the New button in the upper left, and fill out these:

- Base Directory: ${project_loc}
- Goals: clean install
- Maven Runtime: External ...
- Check only **Skip tests** for now

Compiling and Deploying

The above step should have created a build target. Click on it and if everything is configured correctly, it should build the SLP web server and, thanks to Maven, retrieve all the dependencies.

If no errors were raised, the SLP flash interface should be accessable from:

```
https://localhost/SLP/SLP_Driver.html
```

If you see a loading screen followed by an empty interface, congratulations, you have set up, compiled, deployed and ran the SLP server and client!

Running SLP Client

If you are using Firefox to run SLP, you may have trouble with the Flash plugin being stopped by Firefox. To prevent this, type in the address bar in Firefox "about:config" and hit enter. The page you get should be a long table of variable names and values. Search for the setting called "dom.ipc.plugins.timeoutSecs" using the filter bar. For that setting change the value to "-1".