How to Run the Testbed in Eclipse with Jetty 9.4

- How To Run the Testbed in Eclipse with Jetty 9.4
- Issues
  - Logging
- Why is Jetty 9.4 different than 9.3 ?
- Example Eclipse Launcher

Works with Eclipse 2019-06

How To Run the Testbed in Eclipse with Jetty 9.4

1. Install Jetty (download and unpack `jetty-distribution`)

   The version of Jetty should match the `jetty.version` property in the testbed POM (`java-idp-testbed/pom.xml`)

   The Jetty installation directory should be set as the `jetty.home` system property in the Eclipse Launcher when running the testbed's Main.java, see Program Arguments below.

2. Clone the `java-idp-testbed`, `java-identity-provider`, and `java-idp-jetty-base` source code repositories as peers in the same directory :

   ```
   git clone git@git.shibboleth.net:java-idp-testbed
   git clone git@git.shibboleth.net:java-idp-jetty-base
   git clone git@git.shibboleth.net:java-identity-provider
   ```

   Optionally, clone additional source code repositories :

   ```
   git clone git@git.shibboleth.net:java-support
   git clone git@git.shibboleth.net:spring-extensions
   git clone git@git.shibboleth.net:java-opensaml
   ```

3. Import projects into Eclipse : `File > Import > Maven > Existing Maven Projects`

4. Checkout the `9.4-testbed-eclipse` branch of `java-idp-jetty-base`

5. Assemble third party dependencies by building the IdP WAR and removing IdP JARs :

   ```
   cd java-identity-provider/idp-war
   mvn clean package -DskipTests
   rm target/idp-war-*/WEB-INF/lib/idp-*.jar
   ```

   Optionally remove additional JARs :

   ```
   rm target/idp-war-*/WEB-INF/lib/java-support-*.jar
   rm target/idp-war-*/WEB-INF/lib/spring-extensions-*.jar
   rm target/idp-war-*/WEB-INF/lib/opensaml-*.jar
   ```

6. Download logging and testbed dependencies and add to the Jetty server classpath :

   ```
   cd java-idp-jetty-base/src/main/resources/jetty-base
   java -jar $jetty.home/start.jar --create-files
   ```

7. Run the testbed in Eclipse as a Java Application :

   In the `idp-testbed` project, right-click on `idp-testbed-jetty-9.4.launch > Run As > Java Application`

8. Go to the testbed webapp :

   https://localhost:8443/index.html

Issues

Logging

You will probably want to change the root logging level from `DEBUG` to `INFO` in `idp-conf/src/test/resources/logback-test.xml`

Why is Jetty 9.4 different than 9.3 ?

Running the IdP testbed in Eclipse with Jetty version 9.4 requires classpath changes.

I believe this is because :

> *The code that was changed in 9.4, is that we no long look to load classes/resources from the same classloader that loader jetty. That was really just encouraged code to not properly set the thread context classloader.*
>
> *For this method to work for you, then the class you are looking for needs to either be on the system classpath or visible to a Classloader that is set as the current thread context classloader. If executed from within a webapp, you will have a classloader that is able to see WEB-INF classes, but jetty and some system classes are hidden from it.*
>
> *This is a deliberate change, which I think is the correct way to go. However, we are open to understand your use-case of why we may need to search a bit harder for classes, so please describe your classloader setup and classpath.*

Example Eclipse Launcher

To create an Eclipse Launcher manually :

> Use Eclipse Launcher "Program arguments" not "VM arguments"

Right-click on `Main.java > Run As > Java Application`

Eclipse Launcher program arguments :

Program Arguments

```
-Djetty.home=<path to jetty-distribution> -Didp.home=classpath: -Didp.webflows=classpath*:/flows -Djava.io.
tmpdir=tmp
```

Eclipse Launcher working directory :

Working Directory

```
${workspace_loc:idp-jetty-base/src/main/resources/jetty-base}
```

Example Eclipse Launcher :

idp-testbed.launch

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<launchConfiguration type="org.eclipse.jdt.launching.localJavaApplication">
    <listAttribute key="org.eclipse.debug.core.MAPPED_RESOURCE_PATHS">
        <listEntry value="/idp-testbed/src/main/java/Main.java"/>
    </listAttribute>
    <listAttribute key="org.eclipse.debug.core.MAPPED_RESOURCE_TYPES">
        <listEntry value="1"/>
    </listAttribute>
    <booleanAttribute key="org.eclipse.jdt.launching.ATTR_EXCLUDE_TEST_CODE" value="true"/>
    <booleanAttribute key="org.eclipse.jdt.launching.ATTR_USE_CLASSPATH_ONLY_JAR" value="false"/>
    <booleanAttribute key="org.eclipse.jdt.launching.ATTR_USE_START_ON_FIRST_THREAD" value="true"/>
    <stringAttribute key="org.eclipse.jdt.launching.CLASSPATH_PROVIDER" value="org.eclipse.m2e.launchconfig.
classpathProvider"/>
```

```
    <stringAttribute key="org.eclipse.jdt.launching.MAIN_TYPE" value="Main"/>
    <stringAttribute key="org.eclipse.jdt.launching.PROGRAM_ARGUMENTS" value="-Djetty.home=/workspaces/jetty
/jetty-9.4 -Didp.home=classpath: -Didp.webflows=classpath*:/flows -Djava.io.tmpdir=tmp" />
    <stringAttribute key="org.eclipse.jdt.launching.PROJECT_ATTR" value="idp-testbed"/>
    <stringAttribute key="org.eclipse.jdt.launching.SOURCE_PATH_PROVIDER" value="org.eclipse.m2e.launchconfig.
sourcepathProvider"/>
    <stringAttribute key="org.eclipse.jdt.launching.WORKING_DIRECTORY" value="${workspace_loc:idp-jetty-base/src
/main/resources/jetty-base}"/>
</launchConfiguration>
```