|  |
| --- |
| Eclipse Lyo project |
| **How to Run Lyo OSLC Provider Tests and Generate Assessment Report** |
|  |
|  |
| Created 2014-01-15 by Yuhong Yin and Kohji Ohsawa, IBM |
| Updated 2020-12-18 by Andrew Berezovskyi, KTH |

Contents

[1. Introduction to the OSLC Test Suite 4](#_Toc59213087)

[2. Building a RIO Running Environment 5](#_Toc59213088)

[3. Check out the OSLC Test Suite Code 5](#_Toc59213089)

[4. Run Tests in the OSLC Test Suite 6](#_Toc59213090)

[4.1 Run Provider Tests against RIO-CM 6](#_Toc59213091)

[4.2 Run provider test against RIO-AM 9](#_Toc59213092)

[4.3 Run provider test against RIO-RM 9](#_Toc59213093)

[4.4 Run provider test against RTC (a CM Provider) 9](#_Toc59213094)

[4.5 Run provider test against ClearQuest (a CM Provider) 11](#_Toc59213095)

[4.6 Run provider test against RQM (a QM Provider) 12](#_Toc59213096)

[5. Run Tests and Generate OSLC Assessment Report 14](#_Toc59213097)

[5.1 Set up assessment report working environment 15](#_Toc59213098)

[5.2 Run the Provider Test using the corresponding xml file 15](#_Toc59213099)

[5.3 Generate the OSLC Assessment Report 18](#_Toc59213100)

[5.4 Interpret the OSLC Assessment Report 19](#_Toc59213101)

[6. Trouble Shooting 20](#_Toc59213102)

[6.1 Compilation errors in the RIO 20](#_Toc59213103)

[6.2 Compilation errors in the Test Suite 21](#_Toc59213104)

[6.3 Run-time errors in the OSLC Test Suite 21](#_Toc59213105)

[7. How to Search or Report a Test Suite Bug 25](#_Toc59213106)

[8. How to Contribute to the OSLC Test Suite? 26](#_Toc59213107)

The Open Services for Lifecycle Collaboration (OSLC) is a new approach to solving the problem of integrating lifecycle tools to share data (for example, requirements, defects, test cases, plans, or code) with one another. With more teams within IBM and more companies outside IBM choose to adopt OSLC for integration (either an OSLC service provider or an OSLC service consumer), it is highly desired to have a standard testing solution to assess an OSLC service provider implementation, validate its quality and accelerate the adoption.

This documentation provides detailed instructions on how to run the OSLC provider JUnit test suite and generate OSLC assessment report.

This documentation will evolve over time.

1. Introduction to the OSLC Test Suite

The OSLC test suite is a set of JUnit tests that made available through the Eclipse Lyo project. It is designed to test against an OSLC service provider (in specific domains, such as CM, QM and RM) for its implementation of the OSLC core specification and the corresponding domain specifications.

The roles of the OSLC test suite include:

**As an OSLC assessment assessor**

In this role, the test suite will run again a particular OSLC provider and as minimal, make sure the provider passes the OSLC core spec test and the OSLC domain spec test. It provides an assessment report delivering clear and actionable information to be eligible for each OSLC spec.

**As an OSLC quality validator**

In this role, the test suite provides functional capabilities to help test an OSLC based solutions and evolve new or existing OSLC embracing solutions into high quality offerings. It provides reusable test cases to reduce the effort improving the quality of OSLC solution.

**As an OSLC adoption accelerator**

In this role, the test suite will consider building canned images for learning OSLC, adding inline comments to test cases, and providing additional documentation.

The latest Test Suite can be found under <https://github.com/eclipse/lyo.testsuite>

1. Building a RIO Running Environment

**TODO 2020-12-18 update for** [**https://github.com/oslc-op/refimpl**](https://github.com/oslc-op/refimpl)

We recommend that you have a running Reference Implementation (RIO) environment.

Follow the instructions in this wiki page to build a RIO Running Environment: [**http://wiki.eclipse.org/Lyo/BuildRIO**](http://wiki.eclipse.org/Lyo/BuildRIO)

Please note that you need to set these Workspace | Preference

Java --> Complier assessment level: set to 1.6

Java --> Installed JRE: point to a Java 1.6 SDK

**TODO 2020-12-18 update for Java 8**

Please check the [**Troubleshooting**](#_6.1_Compilation_errors) section if you run into compilation errors in any of the RIO projects.

1. Check out the OSLC Test Suite Code

Clone the <https://github.com/eclipse/lyo.testsuite> repository.

Please check the [**Troubleshooting**](#_6.1_Compilation_errors) section if you run into compilation errors in the test suite project.

1. Run Tests in the OSLC Test Suite

Now you are ready to run the OSLC provider tests. Here are some general instructions on running provider tests after you have chosen a provider (domain and test suite).

**[1] Make sure the server of that provider is running**

**[2] Check and verify the parameters specified in the test setup.properties file**

**[3] Run the tests**

**[4] Examine the results**

It is recommended that you start with running an OSLC provider test against the RIO implementation, with an OSLC domain (CM, QM or RM) that you are interested in.

The following sections start on how to run the RIO provider tests, and then cover basic information about running the provider tests against some Rational Products, such as Rational Team Concert (RTC), ClearQuest, Rational Quality Manager (RQM) and Rational Requirements Composer (RRC).

## Run Provider Tests against RIO-CM

This section provides instructions on how to run the provider tests against RIO-CM.

### Start the RIO-CM Server

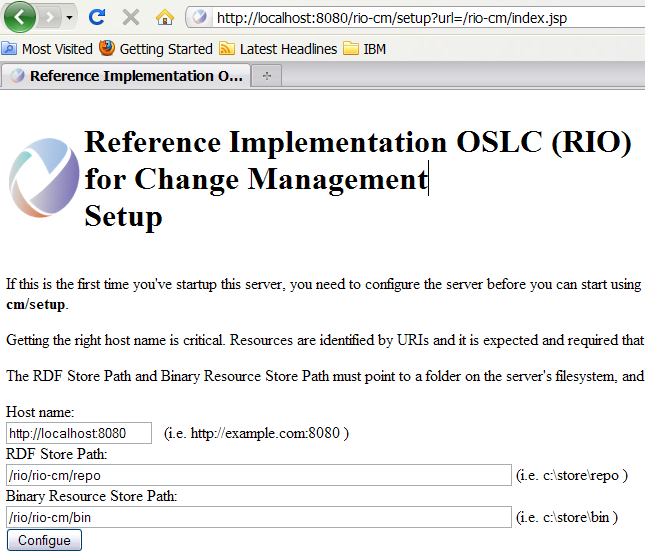
Before running any provider test against the RIO implementation, you must first start the corresponding RIO server.

To start the RIO-CM Server, from “**Run**” 🡪 “**Run Configurations …**”, find **Maven Build** | **Run RIO-CM** and click “**Run**”

After the jetty server started, confirm that you can access Rio-CM server via

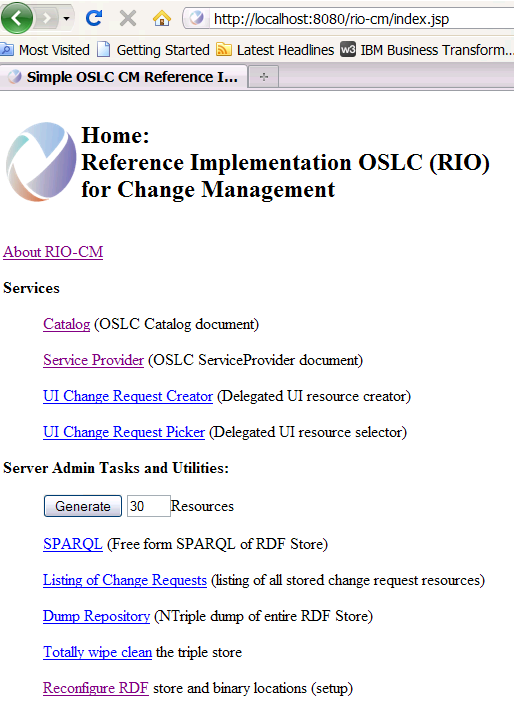
<http://localhost:8080/rio-cm>

The very first time you run this on your machine, you will see the setup page like this:



You can enter the defaults for RDF Store Path and Binary Resource Store Path as suggested, and click the “**Configure**” button.

You will then see a page like the following:



You will use this URL (<http://localhost:8080/rio-cm>) in OSLC provider test setting later.

### Check and verify parameters in the setup.properties file

Refer to the latest update on

<http://wiki.eclipse.org/Lyo/BuildTestSuite#Configure_the_OSLC_Test_Suites>

### Run the Provider Test against RIO-CM Server

Refer to the latest update on

<http://wiki.eclipse.org/Lyo/BuildTestSuite#Configure_the_OSLC_Test_Suites>

### Examine the Test Results

Refer to the latest update on

<http://wiki.eclipse.org/Lyo/BuildTestSuite#Configure_the_OSLC_Test_Suites>

## Run provider test against RIO-AM

To be added.

## Run provider test against RIO-RM

To be added.

## Run provider test against RTC (a CM Provider)

This section provides instructions on how to run the provider tests against a RTC Server.

### Start a RTC Server

Identify an RTC server to run your tests against.

### Check and verify parameters in the setup.properties file

If you want to test out RTC as an OSLC v1 spec provider,

locate config/rtc/rtc-setupv1.properties file and adjust accordingly

If you want to test out RTC as an OSLC v2 spec provider,

locate config/rtc/rtc-setupv2.properties file and adjust accordingly

You will need to change parameters in various config files or template files for the server you are testing.

Follow these steps:

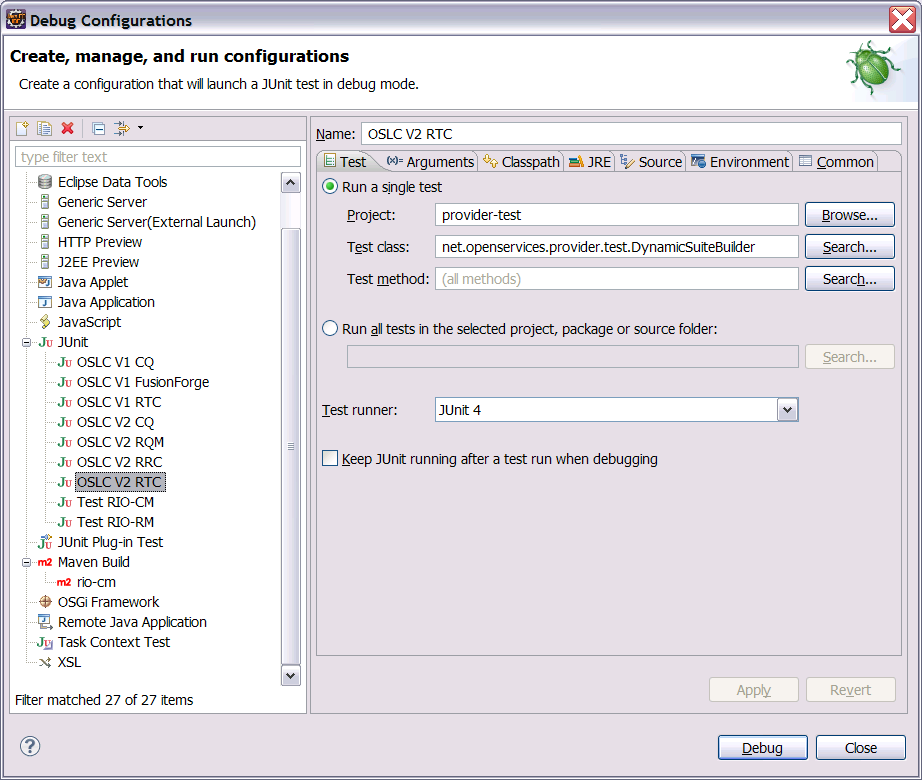
1. Submit a defect type work item called “templatedDefect”.
2. Adjust these properties in rtc-setupv2.properties:

|  |  |
| --- | --- |
| Parameters | Example |
| baseUri | <https://quagmire.rtp.raleigh.ibm.com:14444/ccm/oslc/workitems/catalog> |
| formUri | <https://quagmire.rtp.raleigh.ibm.com:14444/ccm/authenticated/j_security_check> |
| userId | ADMIN |
| Pw | ADMIN |

### Run the Test Suite

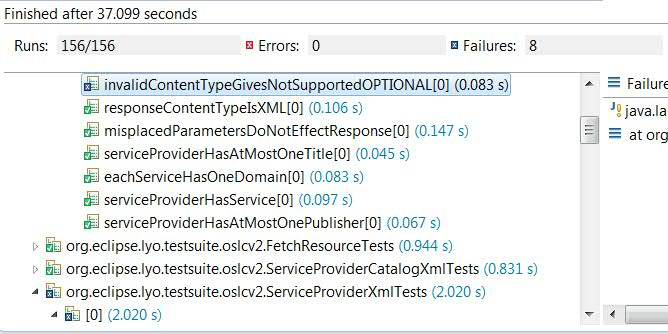
If you want to test out RTC as an OSLC v1 spec provider, find and run the **OSLC V1 RTC.launch**; If you want to test out RTC as an OSLC v2 spec provider, find and run the **OSLC V2 RTC.launch**.

Here is a screenshot of the “OSLC V2 RTC” junit test launcher.



### Examine the Test Results

You should see the test results like shown below.



If your result is very different (with more errors or failures) from what is shown above, you need to go back and check your RTC related config files and template files.

## Run provider test against ClearQuest (a CM Provider)

This section provides instructions on how to run the provider tests against a ClearQuest Server.

### Start a ClearQuest Web Server

Make sure your ClearQuest server is running.

### CQ as v1 provider vs. as v2 provider

If you want to test out ClearQuest as an OSLC v1 spec provider,

locate **config/cq/OSLC V1 CQ.launch** file and run it.

If you want to test out ClearQuset as an OSLC v2 spec provider,

locate **config/cq/OSLC V2 CQ.launch** file and run it.

### CQ Configurations Files

All the CQ configuration files are located off directory **config/cq**

Config files when running CQ as an OSLC v2 provider.

|  |  |
| --- | --- |
| **Config file** | **Description** |
| cq-setupv2.properties | Main configuration file for v2 testing |
| cq-template2.xml | Template used by CreateAndUpdateRdfXmlTests.java  and CreateAndUpdateXmlTests.java  (in package org.eclipse.lyo.testsuite.oslcv2)  to create a new change request |
| cq-json-template2.json | Template used by CreateAndUpdateJsonTests.java  (in package org.eclipse.lyo.testsuite.oslcv2)  to update a newly created change request |
| cq-update2.xml | Template used by CreateAndUpdateJsonTests.java  (in package org.eclipse.lyo.testsuite.oslcv2)  to create a new change request |
| cq-json-update2.json | Template used by CreateAndUpdateJsonTests.java  (in package org.eclipse.lyo.testsuite.oslcv2)  to update a newly created change request |

Config files when running CQ as an OSLC v1 provider.

|  |  |
| --- | --- |
| **Config file** | **Description** |
| cq-setupv1.properties | Main configuration file for v1 testing |
| cq-template1.xml  cq-json-template1.json | Template used by CreateAndUpdateTests.java  (in package org.eclipse.lyo.testsuite.server.oslcv1tests)  to create a new change request |
| cq-update1.xml  cq-json-update1.json | Template used by CreateAndUpdateTests.java  (in package org.eclipse.lyo.testsuite.server.oslcv1tests)  to update a newly created change request |

## Run provider test against RQM (a QM Provider)

This section provides instructions on how to run the provider tests against RQM.

You need to change the following configuration files and contents if you run against your own RQM server.

RQM configuration files: config/rqm/rqm-setupv2.properties

config/rqm/rqm-templates2.xml

config/rqm/rqm-update2.xml

### 4.6.1 rqm-setupv2.properties

This is the main configuration file that specifies base URI of the RQM server and other information needed by the tests. Please follow the comments embedded inside the file to understand parameters and their values.

### 4.6.2 rqm-templates2.xml

This configuration file is used by CreateAndUpdate\*Tests.java

The default file is a template for creating a Test Script RQM resource.

### 4.6.3 rqm-update2.xml

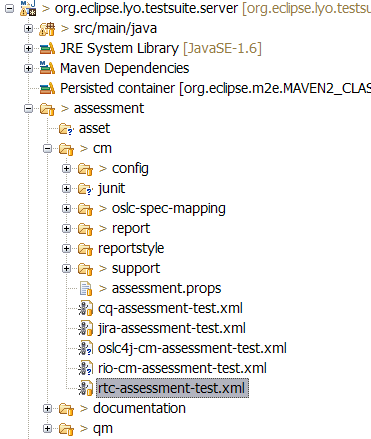
This configuration file is used by CreateAndUpdate\*Tests.java

The default file is a template for updating a Test Script RQM resource.

1. Run Tests and Generate OSLC Assessment Report

This section provides information about running the provider tests with a goal to generate an OSLC assessment report for a particular OSLC provider.

The assessment toolkit is now available in the Lyo TestSuite project. You should be seeing the assessment directory like this:



Right now, the assessment toolkit has support for OSLC CM providers (off and OSLC QM providers. Support for other OSLC domains will be added.

Here is a quick walk-through of the assessment toolkit folder structure.

***assessment/cm***

The “cm” folder contains mappings, report templates and properties files for OSLC CM providers.

***assessment/qm***

The “qm” folder contains mappings, report templates and properties files for OSLC CM providers.

***assessment/documentation***

The “documentation” folder contains instructions on how to run the Lyo TestSuite and generate assessment report.

## Set up assessment report working environment

Locate the **assessment.props** file from /assessment/cm

Edit it to provide the location of your eclipse\_home directory and the location of your home directory (top). For examples,

**eclipse\_home**=C:/Eclipse36/eclipse

This is your eclipse home directory.

**top**=C:/Documents and Settings/yyin

This is the directory that contains the ***.m2*** folder.

The assessment tool references jars off the ***.m2*** folder.

**provider**=RTC3.0.1.1

Use this parameter to specify the name and version of the OSLC provider you test against. It is for reporting and categorizing purpose. In this example, a folder called “RTC3.0.1.1” will be created to gather the Junit test results and report.

## Run the Provider Test using the corresponding xml file

**Prerequisite:** check to make sure that the OSLC service provider server is running.

These are several assessment test xml files off the /assessment/cm directory.

For instances,

***rio\_cm-assessment-tests.xml***

This is for testing against the RIO CM server.

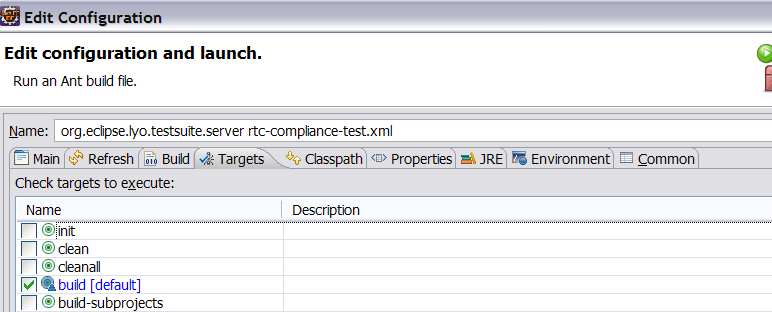
***rtc-assessment-test.xml***

This is for testing against a RTC server.

Choose one that corresponds to your test interest.

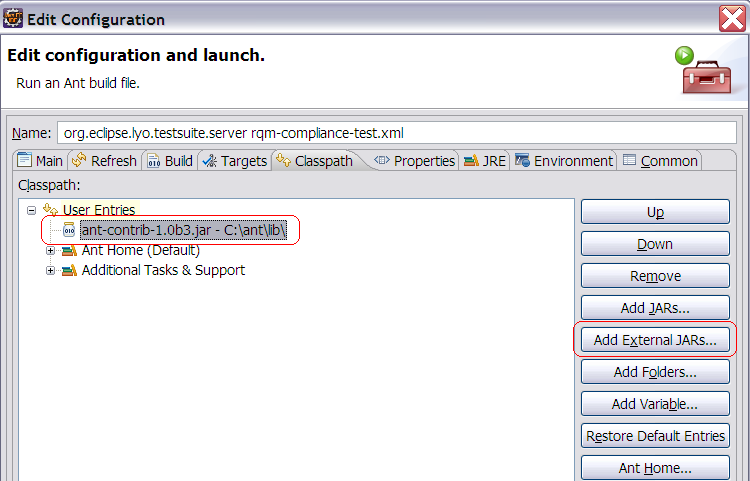
Let’s use the ***rtc-assessment-test.xml*** as an example here.

Find the ***rtc-assessment-test.xml***, right click it, from **Run As … |Ant Build …** to bring up the “Edit configuration and launch” dialog

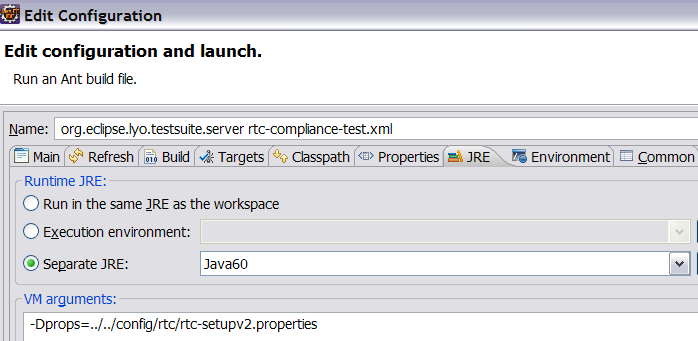


Click the “Classpath” tab, add “ant-contrib-1.0b3.jar” via the “Add External JARs…” button. You can download the library from

<http://sourceforge.net/projects/ant-contrib/files/ant-contrib/1.0b3/>

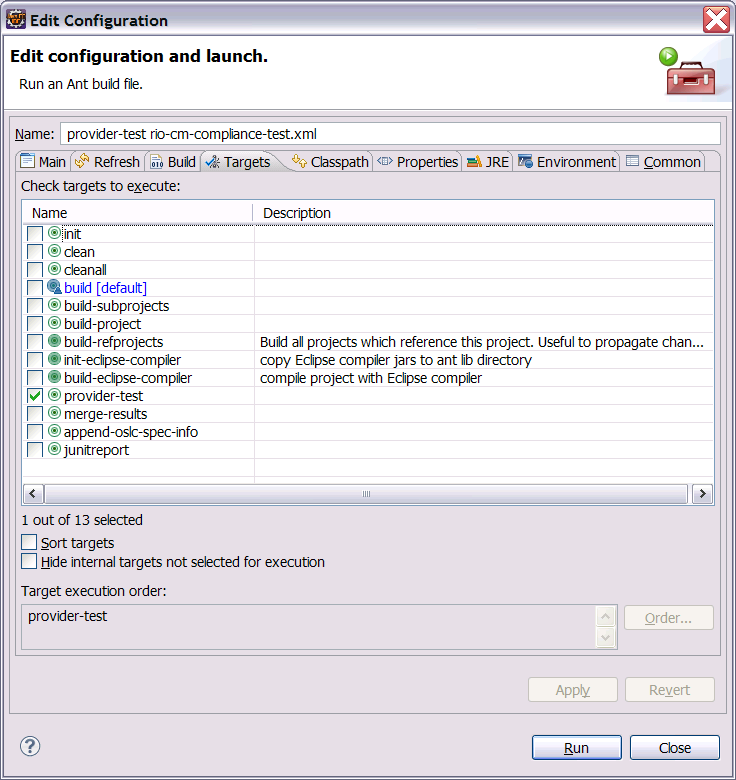


Click the “JRE” tag, in the JVM argument, enter “-Dprops=../../config/rtc/rtc-setupv2.properties” if this is not there.



Go to the Target tab, choose the build target of “Build” and click “Run”.

This build should be successful. If not, check the “[Trouble Shooting](#_6.2_Compilation_errors)” chapter.



The Junit test will start.

## Generate the OSLC Assessment Report

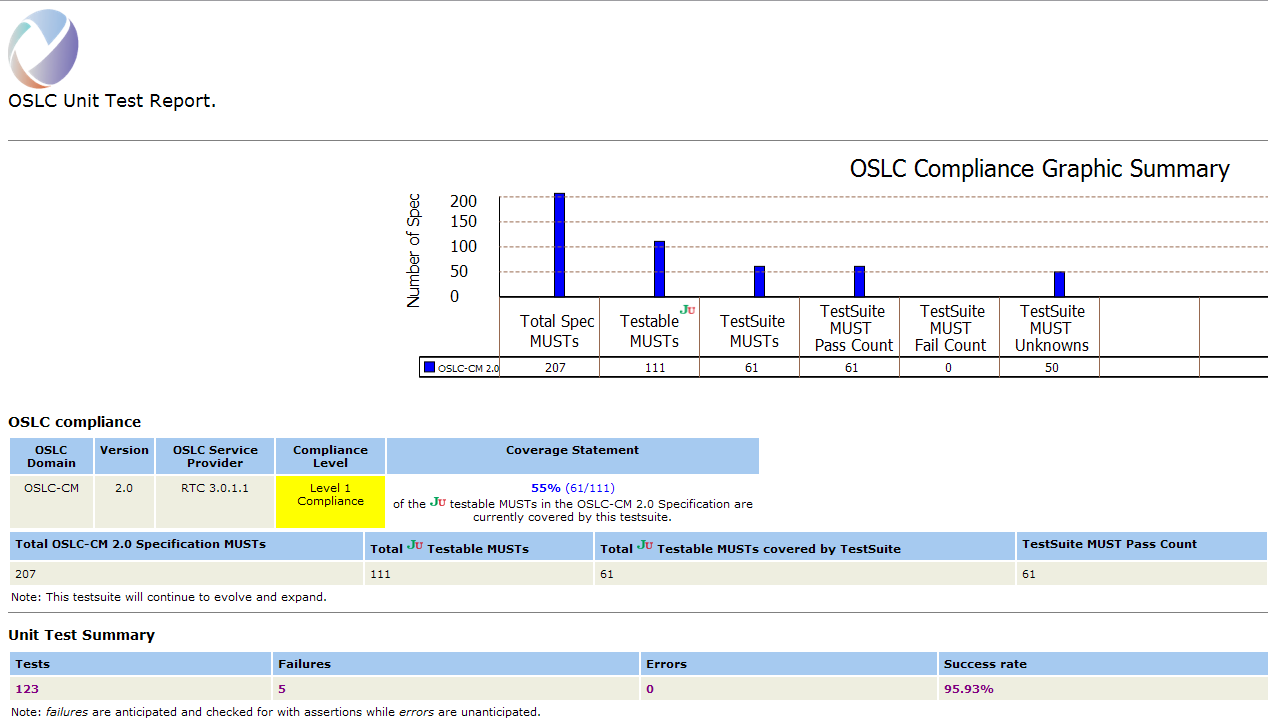
Run the build target “**junitreport**” (which in turn runs “**merge-results**” and “**append-oslc-spec-info**” to generate an OSLC assessment report based on the provider test you run in previous chapter.

Now you can view the generated OSLC assessment report ***OSLCAssessmentReport.html*** file from the proper report tree hierarchy. For instance,

assessment/cm/report/rtc/ has the report for RTC

assessment/cm/report/rio-cm/ has the report for RIO-CM

Here is a screenshot of the assessment report when running against a RTC 3.0.1.1 server.



## Interpret the OSLC Assessment Report

The assessment report HTML page currently contains the following useful information:

* Total number of the MUST requirements in the an OSLC specification
* Total number of the JUnit Testable MUST requirements in an OSLC specification
* Total number of JUnit Testable MUST requirements covered by the OSLC test suite
* Statement about the test coverage calculated using above numbers.
* Total passed tests for MUST requirements
* Total failed tests for MUST requirements
* Details information about what tests passed or failed

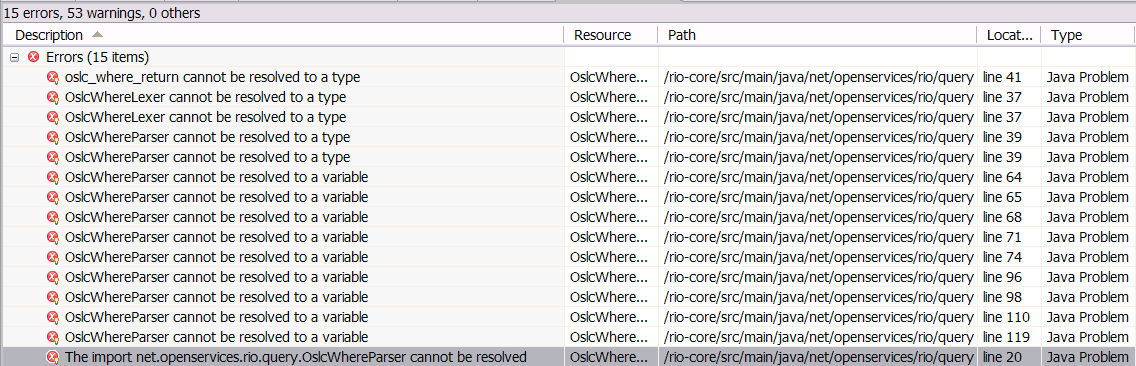
1. Troubleshooting

This section serves a trouble shooting guidance to resolve compilation or run-time errors.

## Compilation errors in the RIO

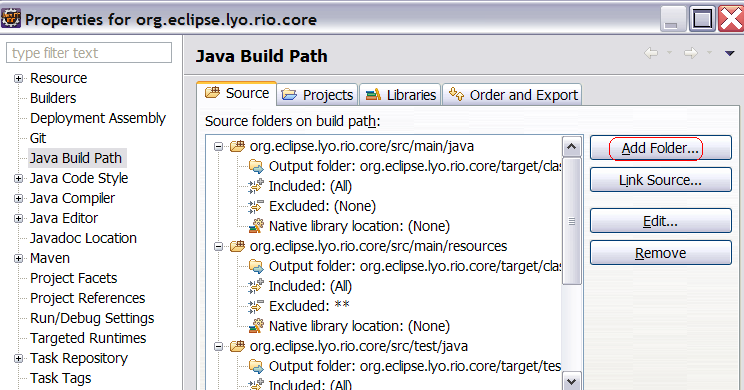
Here are some common errors for the RIO projects.

### Error: Import xxx cannot be resolved



**Solution**: You can resolve these errors by adding a generated code folder to the project classpath.

For instance, bring up the “***Java Build Path***” properties page for the project in question.



Click the “***Add Folder …***” button and select the target | generated-sources | antlr3



Click “Ok” and rebuild this project.

## Compilation errors in the Test Suite

Here are some compilation errors and solutions for the OSLC Test Suite.

### 6.2.1 Error: xxx

To be added.

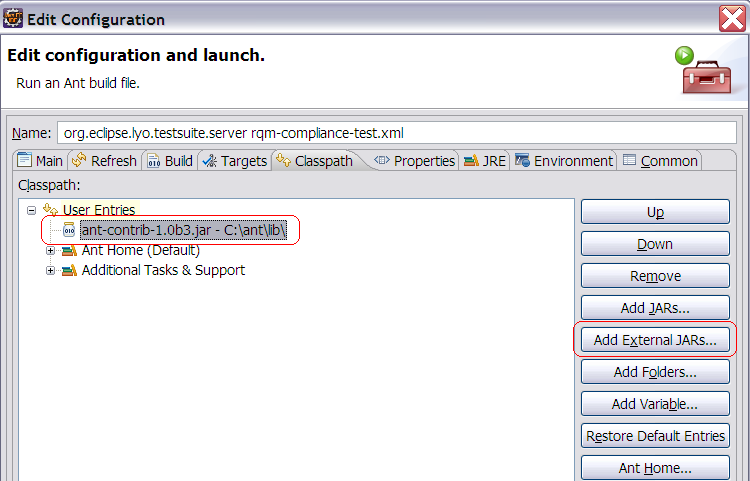
## Run-time errors in the OSLC Test Suite

Here are some run-time errors and solutions for the OSLC Test Suite.

### Error: net/sf/antcontrib/antcontrib.properties.

Error: “[taskdef] Could not load definitions from resource net/sf/antcontrib/antcontrib.properties. It could not be found. It could not be found.”

You can fix this by adding “ant-contrib-1.0b3.jar” via the “Add External JARs…”.



### Error: NullPointerException

This type of errors is normally caused by inconsistent contents in your workspace, or by incorrect entries in test config files or the template files, or other issues.

**Solution**: Start by refreshing to the Test Suite project, running a Maven Clean, and then a Maven Install.

If the NPE still exists, go back read carefully about guidance in previous chapters on how to supply the entries in the config files and the template files.

### Error: initializationError

This type of errors is normally caused by incorrect entries in test config files or the template files.

**Solution**: Go back read carefully about guidance in previous chapters on how to supply the entries in the config files and the template files.

If you still have issues, follow instructions in the next chapter to submit requests on extending the trouble shooting guidance.

### Error: The first argument to the non-static Java function 'replace' is not a valid object reference.

Error:

[xslt] Loading stylesheet /Users/sam/git/org.eclipse.lyo.testsuite/org.eclipse.lyo.testsuite.server/assessment/cm/reportstyle/junit-noframes.xsl

     [xslt] : Error! The first argument to the non-static Java function 'replace' is not a valid object reference.

     [xslt] : Error! Could not compile stylesheet

     [xslt] : Fatal Error! Could not compile stylesheet Cause: Cannot convert data-type 'void' to 'reference'.

     [xslt] Failed to process

This is a bug introduced in Java 6 build 32. It is worked around in ant-1.9.1, planned for Eclipse 4.4. In the meantime, you might need to run the final junitreport goal from the command-line. Here is an example. (Commands on Windows will be different.)

One time setup:

$ mkdir -p ~/.ant/lib

$ cp /path/to/ant-contrib-1.0b3.jar ~/.ant/lib

Then run:

$ cd org.eclipse.lyo.testsuite.server/assessment/cm/

$ ant -f rtc-assessment-test.xml -Dprops=../../config/rtc/rtc-setupv2.properties junitreport

See discussion at

http://stackoverflow.com/questions/10536095/ant-junit-build-error-inside-eclipse

### 

### Error: Use of the extension function '{xalan://java.io.File}exists' is not allowed when Java security is enabled

Error:

[xslt] : Fatal Error! [ERR 0663] Use of the extension function '{xalan://java.io.File}exists' is not allowed when Java security is enabled. To override this, set the com.ibm.xtq.processor.overrideSecureProcessing property to true. This override only affects XSLT processing.

Add -Dcom.ibm.xtq.processor.overrideSecureProcessing=true to the VM arguments in launch for your Ant build.

