RESO RETS Certification Server Testing Tool

Instruction Manual

Version: 02

June 2014

Table of Contents

1 Introduction

This document is to help organizations submit their RETS servers to RESO for certification. A process walk through is found below.

The following are links to resources that will help your technology to become RESO certified. Explanations for how these fit within the certification process will be included below.

1.1 RESO Server Testing Tool (Download)

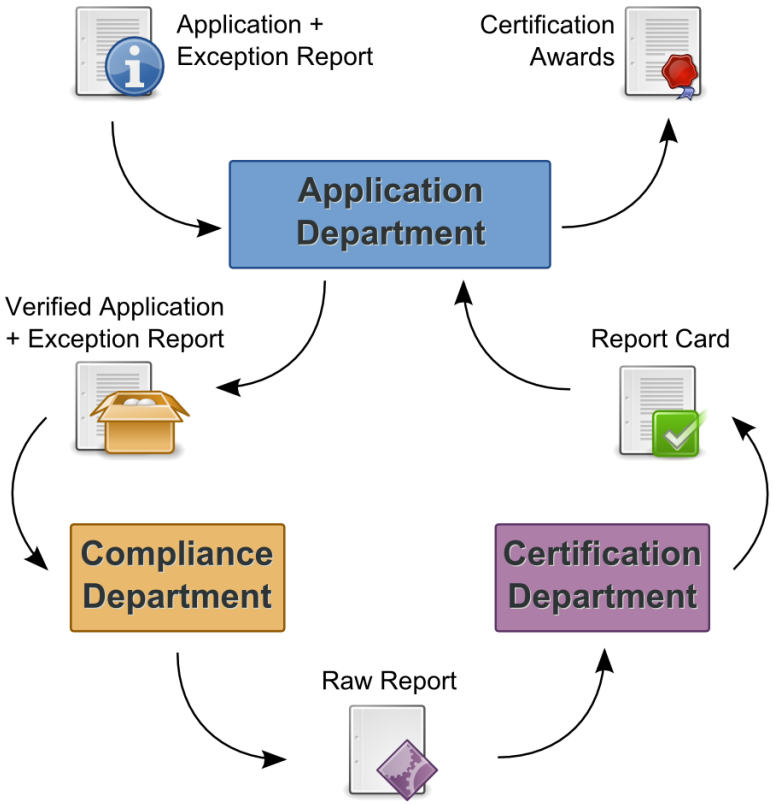
URL: https://github.com/RESO-RETS/RESO-Server-Compliance-Tester

The RESO Server Testing Tool is a Java-based application that may be installed on your local computer. It can be configured to query your server, like a RETS client. During these queries, the server responses will be analyzed and test results will be provided.

These transactions will be compared with the RETS 1.8 Standard. The results of the comparison will be provide.

2 Before Certification

2.1 Certification Flow (Summary)



Certification Flow	Group	Action	Output
1: Application Processing (Pre-Certification)	**Application Department**	Accept and Verify Applicant's ***'Certification Application'*** and ***'Exception Report'*** via reso.org.	Prepare for Compliance Testing. Pass application and ***'Exception Report'*** to Compliance Department.
2: Compliance Testing	**Compliance Department**	Test applicant's metadata against well-defined Compliance Rules as set forth by the RESO RETX 1x Workgroup.	Testing results formatted in ***'Raw Report'*** package. Pass ***'Raw Report'*** to Certification Department.
3: Certification Analysis	**Certification Department**	Analyze ***'Raw Report'*** to determine if applicant qualifies for a certificate. Create a ***'Report Card'*** with findings.	Pass analysis results and ***'Report Card'*** back to Application Processing.
4: Application Processing (Post-Certification)	**Application Department**	Act on Certification Department recommendation	Notify applicant of Certificate Pass/Fail. Send notification and ***'Report Card'*** back to Applicant.

2.2 Getting Ready – Using the RESO Client Testing Tool

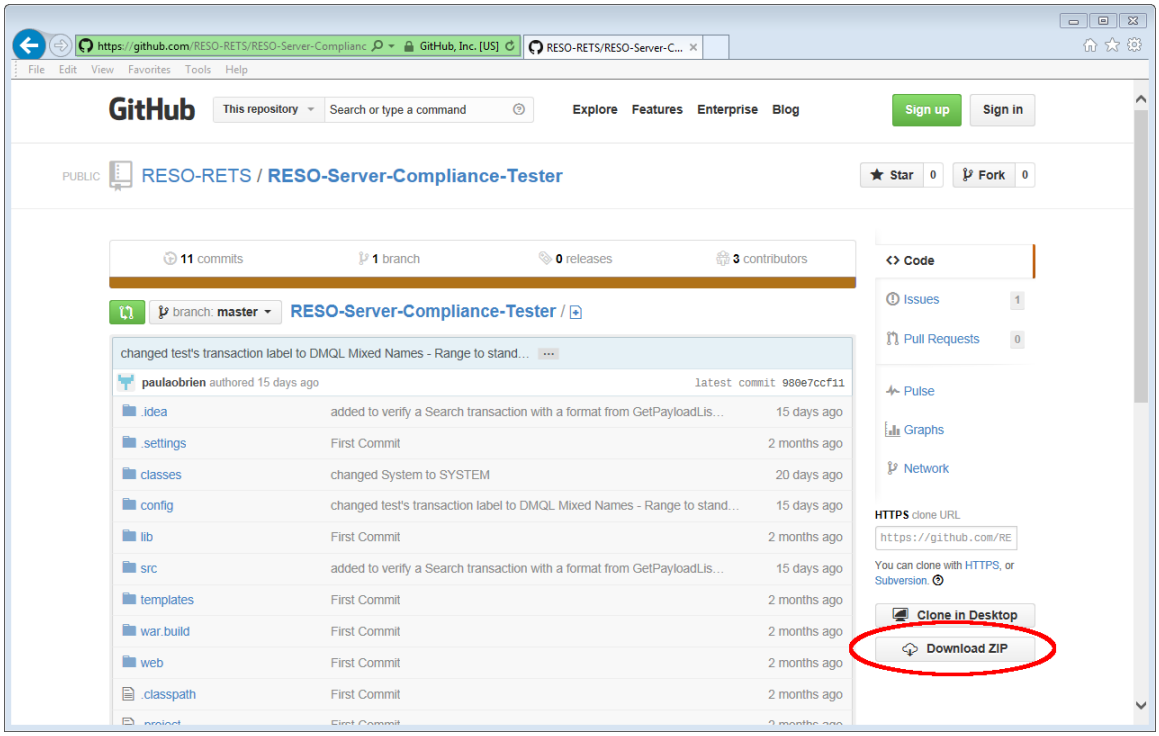
2.2.1 System Requirements

The RESO Server Compliance Checker requires Java 1.7. If Java is already on your computer, it may already be configured correctly. The following instructions are provided if the Testing Tool does not work with current settings.

- You can find instructions for installing Java 1.7 on Windows here: http://www.java.com/en/download/help/windows_manual_download.xml
- Once Java 1.7 has been installed, you will want to set up your environment variables.
- To set the PATH variable, follow the instructions here: http://www.java.com/en/download/help/path.xml
- To set the JAVA_HOME variable and test your Java configuration, you can follow the instructions here: http://zombiebrainzjuice.fr/installing-java-on-windows-7-and-setting-up-the-java_home/

2.2.2 Downloading the RETS Server Testing Tool files

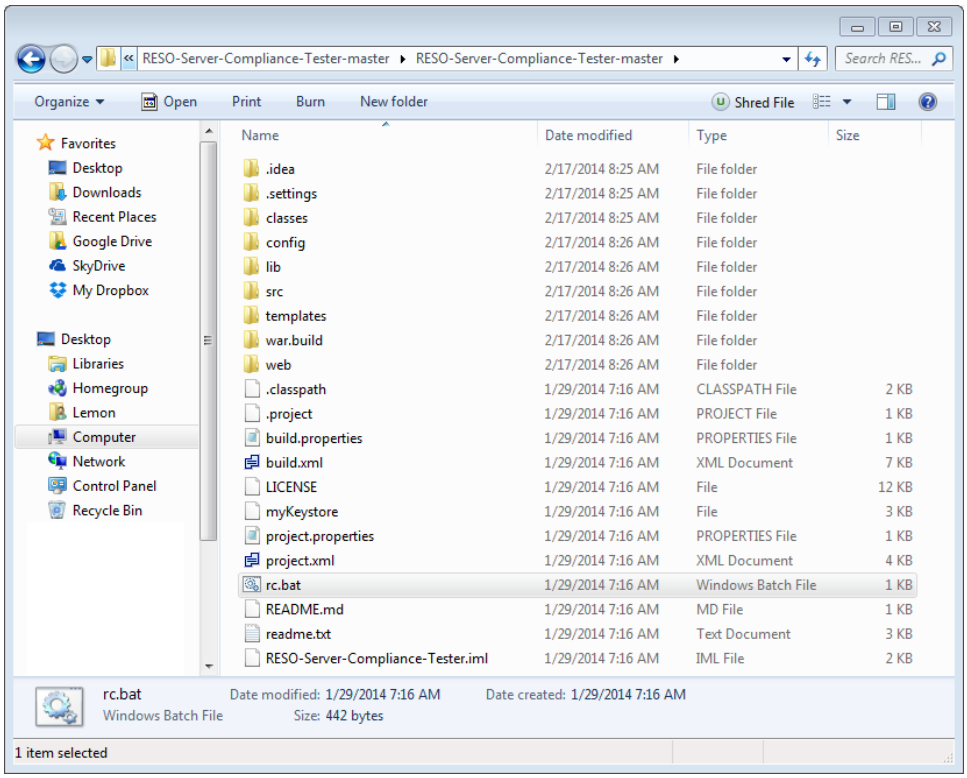
- Visit https://github.com/RESO-RETS/RESO-Server-Compliance-Tester



- Click the "Download ZIP" link and save the file anywhere on your computer.
- Extract the files from the ZIP to a convenient folder.

2.2.3 Launching the Server Testing Tool

The extracted folder “RESO-Server-Compliance-Tester-master” contains all files for the Testing Tool. All instructions will be relative to this directory.



To launch the Testing Tool:

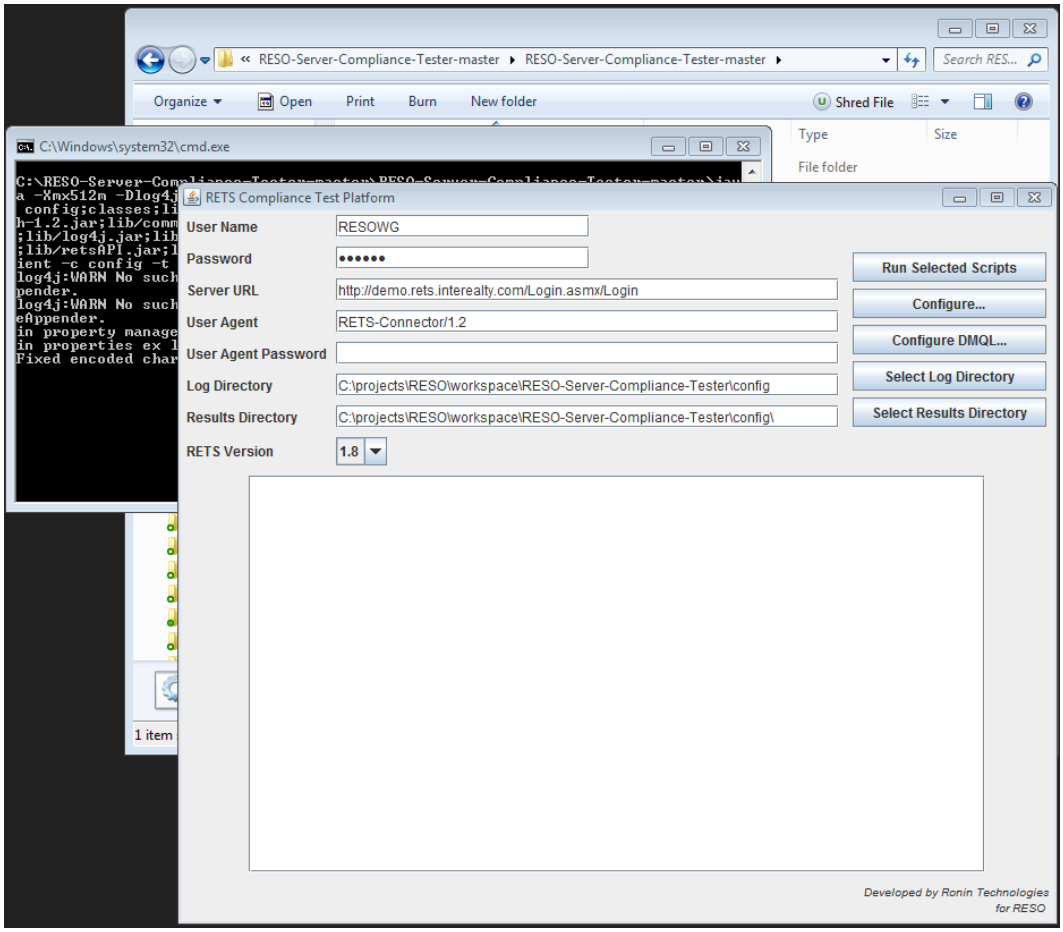
- Navigate to the extracted “RESO-Server-Compliance-Tester-master” folder
- Double-Click “rc.bat” to start the Testing Tool

This can done through the command line as well.

Launching the Testing Tools will create a Windows’s command window and the Testing Tool.

The command window will display information as it is transmitted between the server and Testing Tool.

2.2.4 Server Testing Tool: Main Window – Connect to Server



The Server Testing Tool main window will collect the information required to connect to the server being tested. The buttons on the right allow for additional configuration and running the test scripts.

Server Connection Fields	Description
User Name	The RETS user name or ID provided by the server vendor as used in the RETS login transaction (required).
Password	The RETS password provided by the server vendor as used in the RETS login transaction (required).
Server URL	The URL for RETS access provided by the server vendor. This is the absolute path for the RETS login transaction (required).
User Agent	The RETS user-agent value provided by the server vendor as used in the RETS login transaction. Some server vendors will only allow access to certain user-agents. If the vendor provides a specific user-agent value, enter it here. If not, leave this entry blank (may be required by vendor).

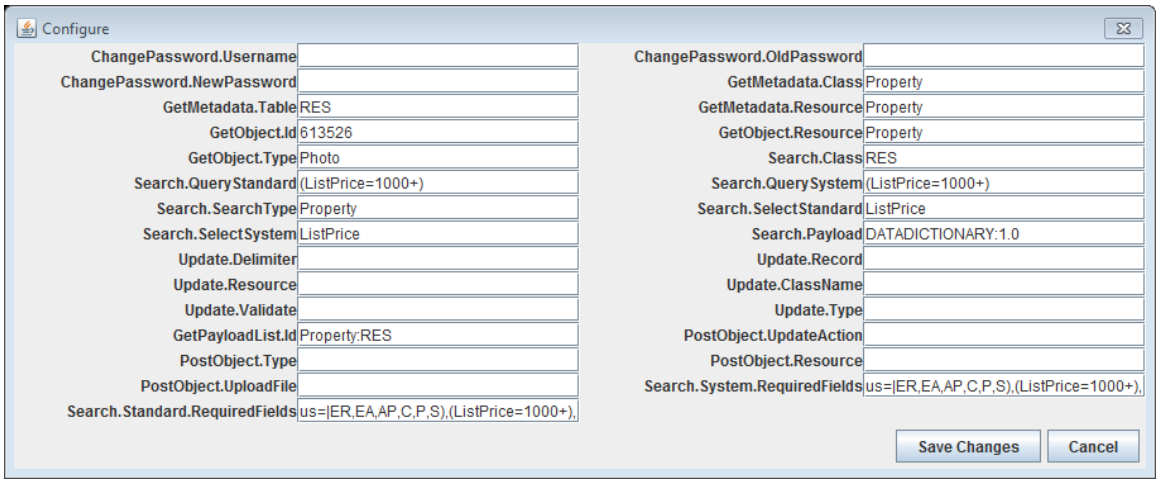
Server Connection Fields	Description
User Agent Password	The RETS UA password (commonly referred to as the "client password") value as used in the RETS login transaction to calculation UA authentication header. Some server vendors may require UA authentication. If so, they will provide a User Agent Password. If not, leave this entry blank (may be required by vendor).
Log Directory	The local computer folder where the application will persist the HTTP request and response information for each RETS transaction executed. The folder location is changed by clicking "Select Log Directory" on the main window. Use the dialog box to select or create a directory.
Results Directory	The local computer folder where the application will export Test Results for each RETS transaction saved. The folder location is changed by clicking "Select Results Directory" on the main window. Use the dialog box to select or create a directory.
RETS Version	The Server Testing Tool tests against the RETS 1.8 Standard.

The functionality of the Server Testing Tool is found in the buttons to the right of the Server Connection Fields.

Testing Buttons	Description
Run Selected Scripts	Launches a dialog box to select the server test scripts. These scripts take the format of XML files (more details below the descriptions of the other buttons).
Configure…	Launches the "Configure" dialog box. This will set basic parameters for testing across all transactions (more details below).
Configure DMQL…	Launches the "Configure DMQL" dialog box. This will set basic parameters for testing across all transactions (more details below).
Select Log Directory	A button to select the directory that is displayed in the "Log Directory" text field.
Select Results Directory	A button to select the directory that is displayed in the "Results Directory" text field.

2.2.5 "Configure…" Dialog Box

The Testing Tool must be configured before running any of the Test Scripts. This dialog box focuses on standard server functionality.



Each set of parameters are used by the various tests run against the server being tested. If the server being tested does not support a certain transaction, those entries may be left blank.

> NOTE: There are no parameters on this screen for login. Those were entered on the main screen, as Login is required in order for any other RETS transactions to run.
>
> NOTE: For testing purposes, the server we are using does not support ChangePassword, Update or PostObject. You will notice that all values pertaining to those transactions are left blank.

It is important to provide statements that would return listings or objects on the server being tested.

RETS Parameter Group	Parameter Fields	Descriptions
ChangePassword	Username	The RETS user id whose password is being changed. May or may not be the same as that used to perform the initial login.
	OldPassword	The existing password for the RETS user whose password is being changed.
	NewPassword	The new password for the RETS user whose password is being changed. Can be any value you choose that meets the guidelines provided by the server vendor (length, types of characters, etc) EXCEPT that it MUST NOT be the same value as the existing password for that user.
GetMetadata	Resource	StandardName to use in the ID for the METADATA-RESOURCE request. GetMetadata.Table and GetMetadata.Class must be values that are associate with this Resource.
	Class	ClassName to use in the ID for the METADATA-CLASS request. Used with the GetMetadata.Resource parameter.

RETS Parameter Group	Parameter Fields	Descriptions
	Table	ClassName to use in the ID for the METADATA-TABLE request. Used in conjunction with the GetMetadata.Resource parameter.
GetObject	Id	The keyfield value for an object. The keyfield can be found in the METADATA-RESOURCE. You can look at the search results from a search transaction to determine a keyfield value to use for testing. (In most servers the keyfield for property is the mls number or listing id or sysid for a record).
	Resource	The resource associated with the object being downloaded; normally, “Property”.
	Type	Default this to “Photo” for testing, as it is always a supported media type.
Search	Class	The SearchClass corresponds to the ClassName from the METADATA-CLASS. For testing purposes, if you have a server vendor’s metadata, you can find this value there. Must be from a Class that is related to the Resource you used for Search.SearchType.
	QueryStandard	The DMQL query used for static testing where StandardNames=1. You can form this from StandardNames in the METADATA-TABLE for a Class/Resource for testing. Must be formatted as proper DMQL with the parenthesis around each subquery and commas etc between subqueries.
	QuerySystem	The DMQL query used for static testing where SysytemNames=0. You can form this from SystemNames in the METADATA-TABLE for a Class/Resource for testing. Must be formatted as proper DMQL with the parentheses around each subquery and commas etc between subqueries.
	SearchType	The SearchType corresponds to the METADATA-RESOURCE. For testing purposes, if you have a server vendor’s metadata, you can find this value there. Must be related to Search.Class. Most commonly defaults to “Property”.
	SelectStandard	A single StandardName used in the RETS Select argument where StandardNames=1. Again, you can find this in the METADATA-TABLE as a StandardName.
	SelectSystem	A single SystemName used in the RETS Select argument where StandardNames=0. Again, you can find this in the METADATA-TABLE as a SystemName.

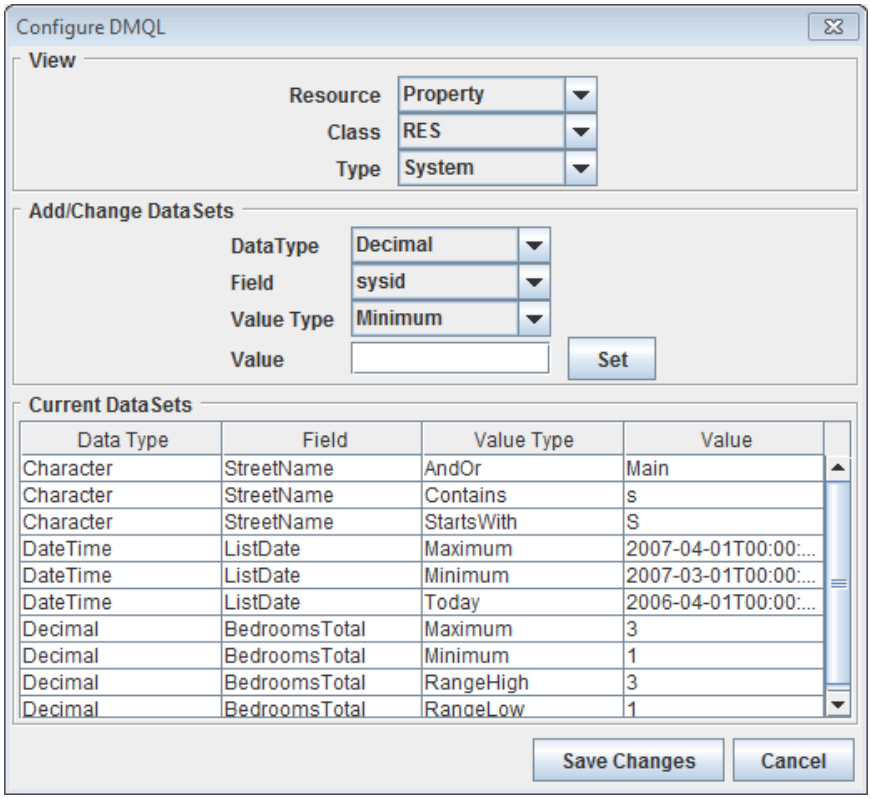
RETS Parameter Group	Parameter Fields	Descriptions
	Payload	This can be derived from the GetPayloadList transaction making sure it corresponds to the Search.Class and Search.SearchType.
	System. RequiredFields	Some RETS Servers require certain fields to be included in every search. This field contains the DMQL Statements required to perform searches that return results. Each DMQL statement must end with a comma. These statements are prepended to the test scripts. Failure to provide the separating and ending commas will cause errors in the DMQL testing. These DMQL statements require System Names from the Metadata to be used. (See an example below).
	Standard. RequiredFields	These DMQL statements required Standard Names from the Metadata to be used. (See the "RequiredFields" definition above). For example: (ListingStatus=|ER,EA,AP,C,P,S),(ListPrice=1000+),
Update	Delimiter	The record delimiter; leave this blank for the ASCII HT character.
	Record	The fieldname=fieldvalue pairs separated by the delimiter for the record you wish to update, formatted as in section 10.
	Resource	The Resource's StandardName from the Metadata.
	ClassName	The ClassName for the record you are updating from the Update section of the Metadata.
	Validate	0, 1, or 2 depending upon if the record will be stored or not stored on the server and if it is full or partial.
	Type	The supported update action from the Metadata. ("Add", "Change" etc)
GetPayloadList	Id	Either the Resource (ie "Property") or Resource:Class combination (ie: "Property:RES") from the Metadata.
PostObject	UpdateAction	The supported update action from the Metadata ("Add", etc).
	Type	The Object Type, i.e. "Photo".
	Resource	The Resource from the Metadata (i.e., "Property").
	UploadFile	The location of the file to upload (i.e."c:\tmp\example.jpg").

The settings of the Configure dialog box is persisted in the “TestParameters.properties” file in the “config” folder when the “Save Changes” button is pressed.

2.2.6 “Configure DMQL…” Dialog Box

The Testing Tool must be configured before running any of the Test Scripts. This provides the tests scripts with server-specific fields and values needed to fully test the server’s different DMQL data.

This dialog box may take some time before appearing. The RETS Server Testing Tool will download a copy of the server’s metadata to populate the dropdown fields.



Configuring both System and Standard Names for DMQL Tests:

DMQL Test are performed with the metadata’s “System” names or “Standard” names. Both System and Standard names need to be updated the match the fields used by the RETS server. Changing the “Type” dropdown will change the contents of the “Current Data Sets” panel.

Changing Current Data Sets:

1. Click the row in the “Current Data Sets” panel that needs to be changed. The values in the row will be placed in the “Add/Change Data Sets” fields.
2. Change the values in the “Add/Change Data Sets” fields as needed.
3. Click “Set” to update the values into the “Current Data Sets” panel.
4. Repeat steps 1-3 as needed. Change “Type” dropdown as needed to update System or Standard data sets.
5. Click “Save Changes” to close the dialog box and save the data sets.

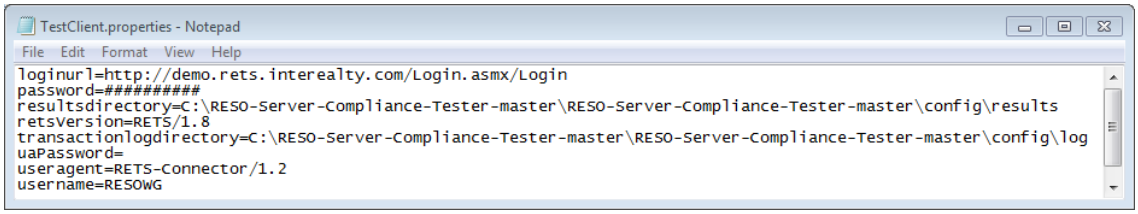
It is highly recommended that the “Data Type” and “Value Type” values are NOT changed. These cover all of the required test. Please change the “Field” and “Value” columns to values that would return records on the server.

The following tables provide additional information on how this dialog box functions.

Configure DMQL Panels	Panel Fields	Descriptions
View	Resource	Select the resource from the server’s metadata that the tool should test against.
	Class	Select the class from the server’s metadata that the tool should test against.
	Type	Select “Standard” or “System” to change the “Current Data Sets” that are viewed and can be changed.
Add/Change Data Sets	DataType	**Please do not change these values.** Specifies the type of field being tested.
	Field	The “Standard” or “System” name found in the RETS server’s metadata.
	Value Type	**Please do not change these values.** Specified the DMQL feature being tested. This changes based on Data Type.
	Value	The value used to search within the specified field. This must be a value that would return a valid record from the server.
	“Select” Button	Updates the values into the “Current Data Sets” panel.
Current Data Sets	Data Type	(See “Changing Current Data Sets” description)
	Field	(See “Changing Current Data Sets” description)
	Value Type	(See “Changing Current Data Sets” description)
	Value	(See “Changing Current Data Sets” description)

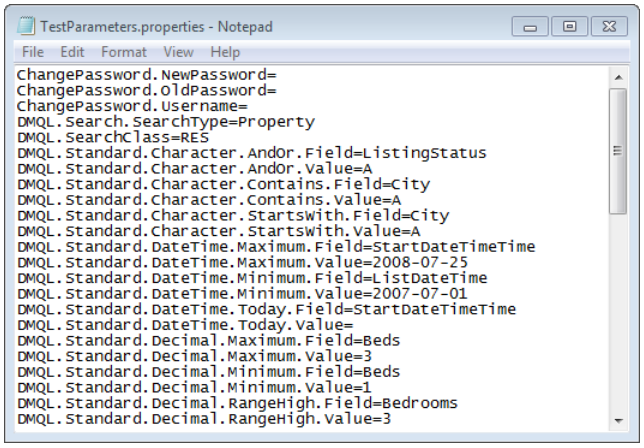
2.2.7 “TestClient.properties” Configuration File

The Testing Tool configurations from the main window is persisted in the “TestClient.properties” configuration file. The text file is made up of key-value pairs used by the test scripts. **THIS FILE WILL BE PROVIDED TO RESO STAFF FOR CERTIFICATION PURPOSES.**

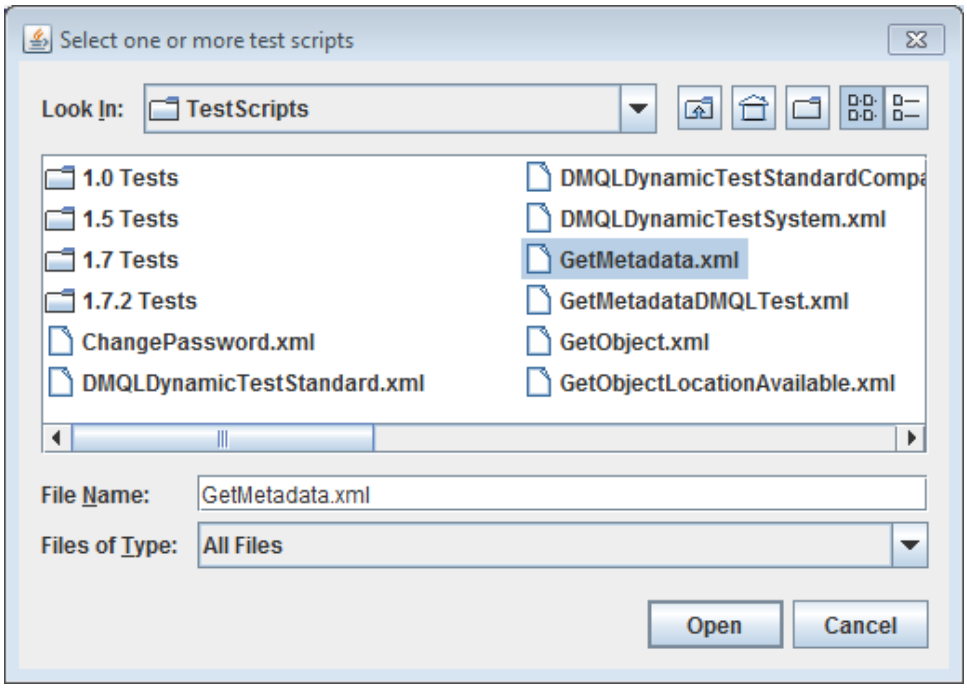


2.2.8 "TestParameters.properties" Configuration File

The Testing Tool configurations from the "Configure" and "Configure DMQL" Dialog boxes are persisted in the "TestParameters.properties" configuration file. The text file is made up of key-value pairs used by the test scripts. **THIS FILE WILL BE PROVIDED TO RESO STAFF FOR CERTIFICATION PURPOSES.**



2.2.9 "Run Selected Scripts" Dialog Box



Clicking the "Run Selected Scripts" button will activate this dialog box. One or more tests can be selected by ctrl-clicking the different test.

Clicking "Open" after selecting test(s) will run them against the server.

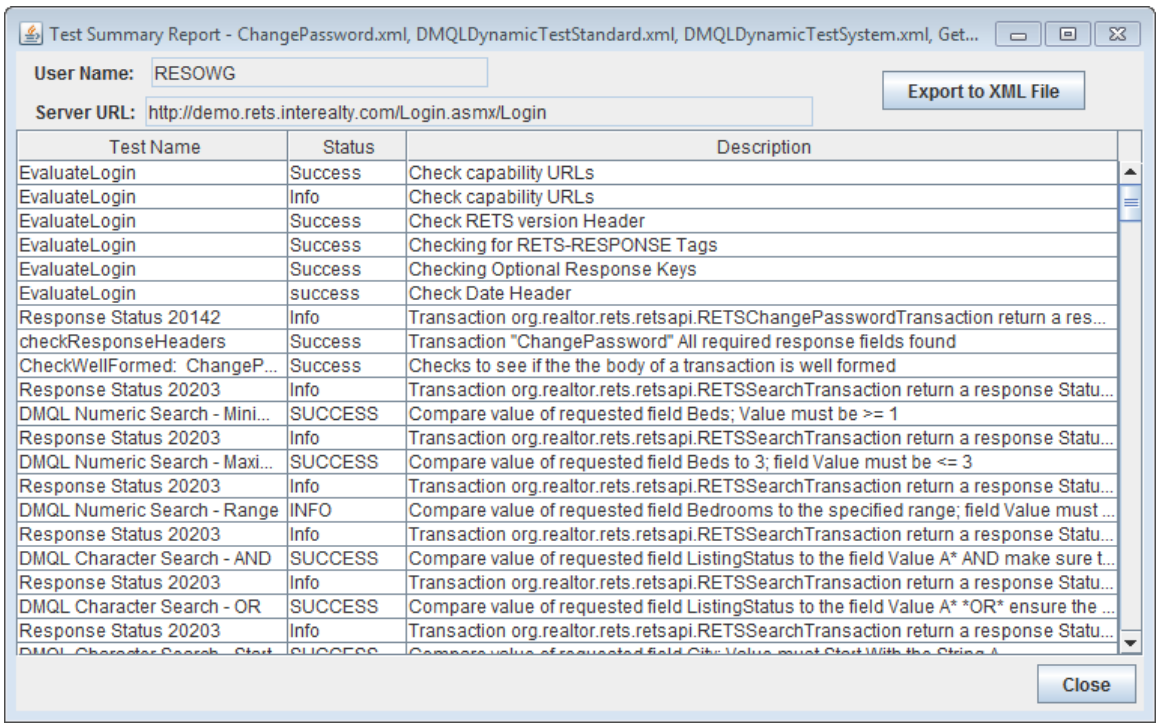
2.2.10 RETS Transaction/Test Script Table

The following table shows which XML test scripts are used to test the different server function groups.

RETS Transaction	Required Test Scripts	Optional Test Scripts
ChangePassword	ChangePassword.xml	n/a
Logout	Logout.xml	n/a
GetObject	GetObject.xml, NegativeTestingGetObject.xml	GetObjectLocationAvailable.xml, GetObjectObjectData.xml
GetMetadata	GetMetadata.xml, NegativeTestingGetMetadata.xml	n/a
Search	Search.xml, NegativeTestingSearch.xml, DMQLDynamicTestStandard.xml, DMQLDynamicTestSystem.xml	SearchWithOffset.xml, SearchWithNoQuery.xml
GetPayloadList	GetPayloadList.xml, NegativeGetPayloadList.xml	n/a
Update	Update.xml, NegativeTestingUpdate.xml	n/a
PostObject	PostObject.xml, NegativePostObject.xml	n/a

2.2.11 Test Summary Report

The "Test Summary Report" dialog box will automatically display when the selected test(s) finish.

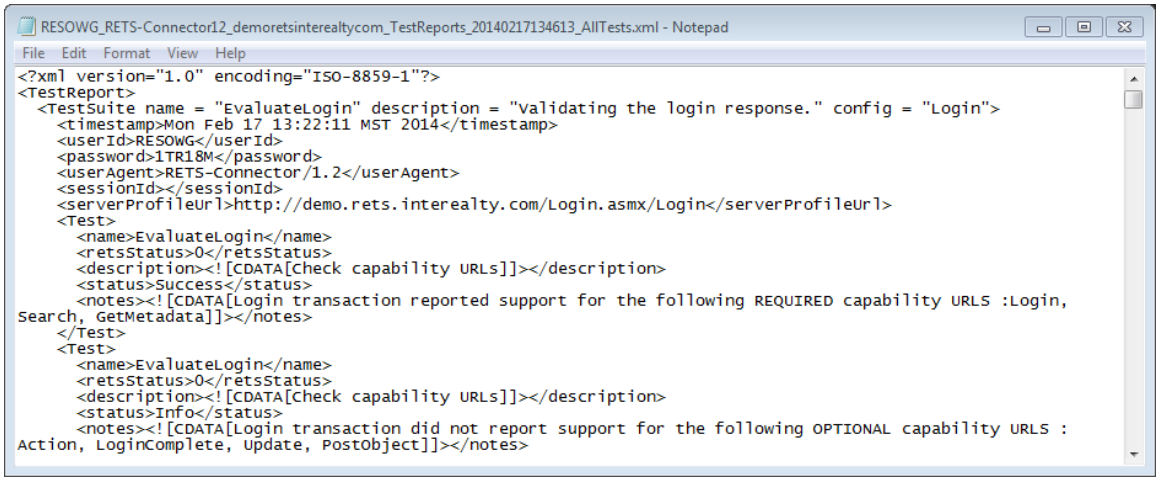


Test Summary Report Fields	Description
Test Name	The name of the test performed. This also could be a Response Status description when a non-zero response is received.

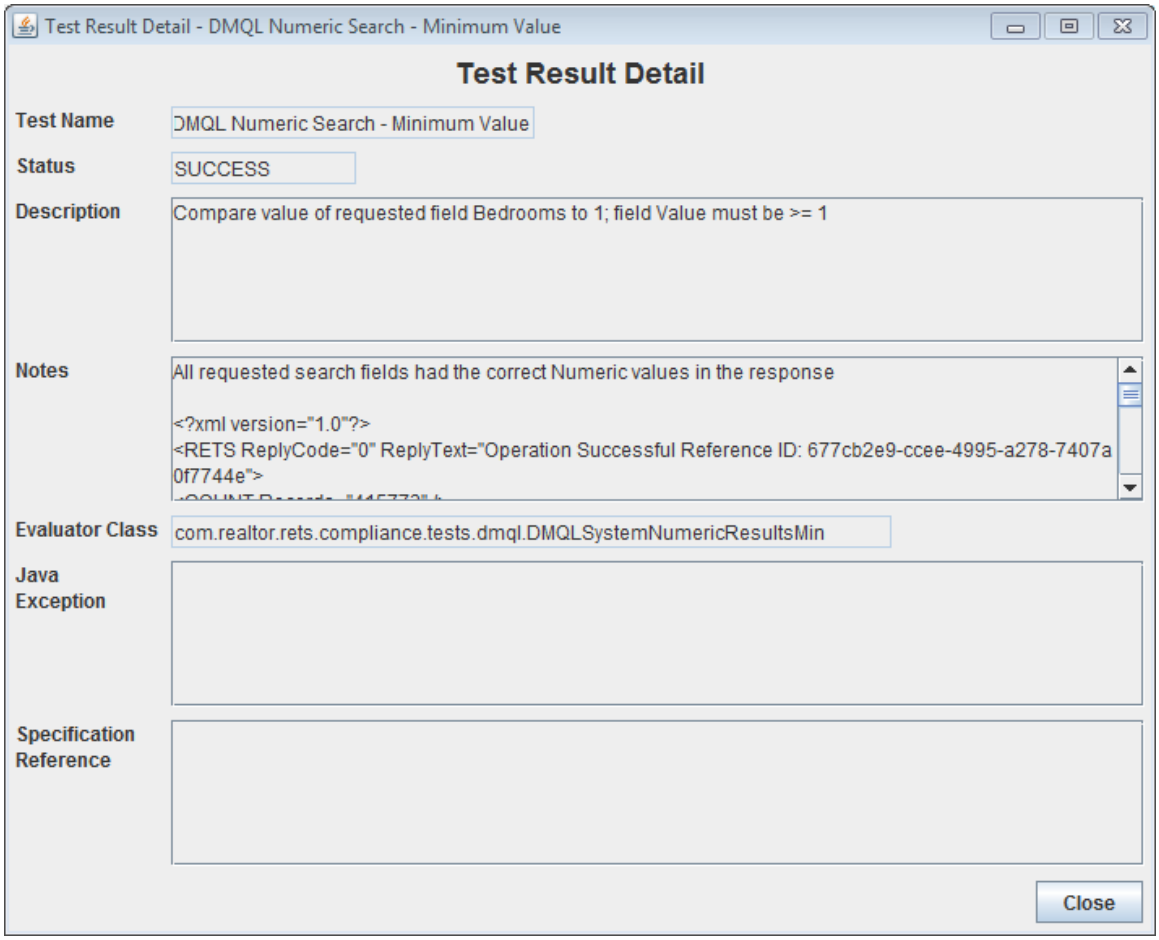
Test Summary Report Fields	Description
Status	The status of the test results. Often the tests end in “Success” or “Failure”. Some “Info” records are provided when additional information is provided.
Description	Details of the test

2.2.12 “Export to XML File”

The only way to save the test results performed is to click the “Export to XML File”. This is a view of the XML text. It contains all information found in the Test Summary and Details.



2.2.13 Test Results Detail Report



Test Report Detail Fields	Description
Test Name	Same as "Test Summary Report"
Status	Same as "Test Summary Report"
Description	Same as "Test Summary Report"
Notes	Notes on the test, including the RETS Transactions.
Evaluator Class	The internal java class used to run the test.
Java Exception	Notes on java exceptions that may have been thrown during the test.
Specification Reference	The section of the RETS specification related to this test.