

|  |  |  |
| --- | --- | --- |
| Asynchronous Software Ltd. | | |
| User Guide | | |
|  | Version | 1.0 |
|  | Date | 01-June-2012 |

Copyright Notice © AsyncSoft, (2012). All Rights Reserved

The information contained in this document is the property of AsyncSoft. No part of this document may be reproduced, stored in a retrieval system, or transmitted in any form, or by any means; mechanical, photocopying, recording, or otherwise, without the prior written consent of AsyncSoft. Under the law, copying includes translating into another language or format. Legal action will be taken against any infringement.

The information contained in this document is subject to change without notice and does not carry any contractual obligation for AsyncSoft. AsyncSoft reserves the right to make changes to any products or services described in this document at any time without notice. AsyncSoft shall not be held responsible for the direct or indirect consequences of the use of the information contained in this document.

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 01-06-2012 | 1.0 | Initial Draft | AsyncSoft |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Contents

[Copyright Notice © AsyncSoft, (2012). All Rights Reserved 0](#_Toc326532351)

[Overview 3](#_Toc326532352)

[Prerequisites 3](#_Toc326532353)

[Creating a New Project 3](#_Toc326532354)

[Saving Project 6](#_Toc326532355)

[Opening a Saved Project 6](#_Toc326532356)

[Updating a Project’s Properties 8](#_Toc326532357)

[Adding a Class\Report 9](#_Toc326532358)

[Deleting a Class 11](#_Toc326532359)

[Update a Class’s Properties 12](#_Toc326532360)

[Property Panel 13](#_Toc326532361)

[Verify reporting Server & Service URL’s 14](#_Toc326532362)

[Get a Report’s Parameters 15](#_Toc326532363)

[Call the Report 16](#_Toc326532364)

[Add an Assertion 17](#_Toc326532365)

[Update an Assertion 18](#_Toc326532366)

[Delete an Assertion 18](#_Toc326532367)

[Generate Test Unit Code 19](#_Toc326532368)

[Print Generated Unit Code 20](#_Toc326532369)

[Copy Generated Unit Code to Clipboard 20](#_Toc326532370)

[About 21](#_Toc326532371)

[Registering 21](#_Toc326532372)

[Miscellaneous 22](#_Toc326532373)

[Application Restrictions 22](#_Toc326532374)

[Localisation Constraints 22](#_Toc326532375)

[Demo Restrictions 22](#_Toc326532376)

[Requirements 23](#_Toc326532377)

# Overview

The purpose of this document is to provide the user with a guided view on creating unit test code for Microsoft’s Report Service Reports, code that they can copy to their existing visual studio test projects.

This is classic ‘Black Box’ testing, were the user will pre-populate the database with scripts in their unit test code and then run the generated code from this application. So, the user must run the database scripts that they are going to use with their tests first, and then run this application to generate tests against the same data.

# Prerequisites

The only prerequisites needed by the ‘Reporting Service Test Generator’ to function correctly are;

1. To be able to connect to the Microsoft Reporting Server, to gather information on the reports
2. Microsoft Word is installed

# Creating a New Project

To create a new project, click on the ‘New Project’ button within the home ribbon bar menu tab (fig 1). This will bring up the new project dialog (fig 2).

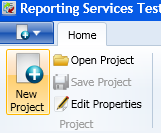


Fig 1

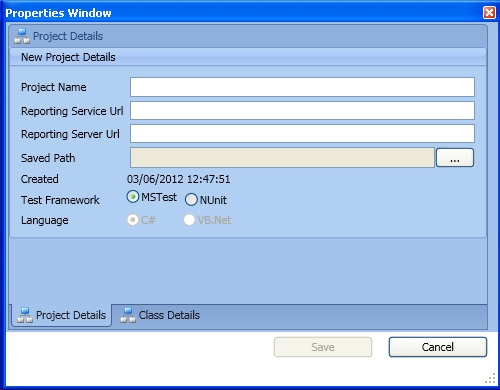


Fig 2

The user will have to provide the URL’s to the report service and the report server, as the application will try and verify these URLs when the save button is clicked. The save button will not be enabled until all the textboxes have been populated (Fig 3)

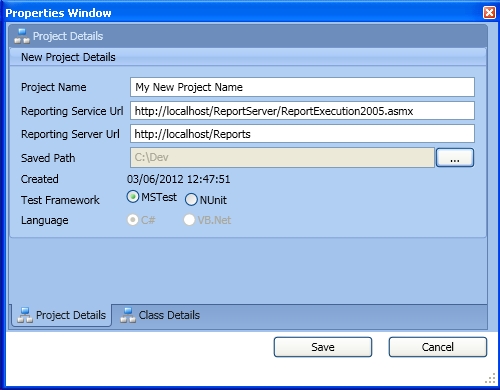


Fig 3

You will notice that the language radio button option has been disabled and defaulted to C#. In this initial version of the application, the only code generated will be C#. To get around this, the user can copy the generated C# code and run it through an online code conversion site, to generate the VB.Net equivalent code (this will be rectified in future releases).

In the above screenshot, the user has elected to use the MSTest framework, this is an option that cannot be changed when the project is created.

If any of the URLs are non-contactable (or the syntax is incorrect), the warning message will be displayed as in fig 4. The user has the option to continue and correct at a later date, or to remain within the new project dialog and resolve the URL in question. In fig 5, you will see the newly created project, with the tree structure on the left displaying the new project (currently with no associated class\reports).

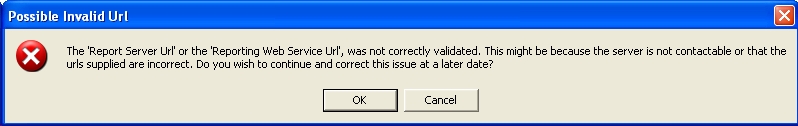


Fig 4

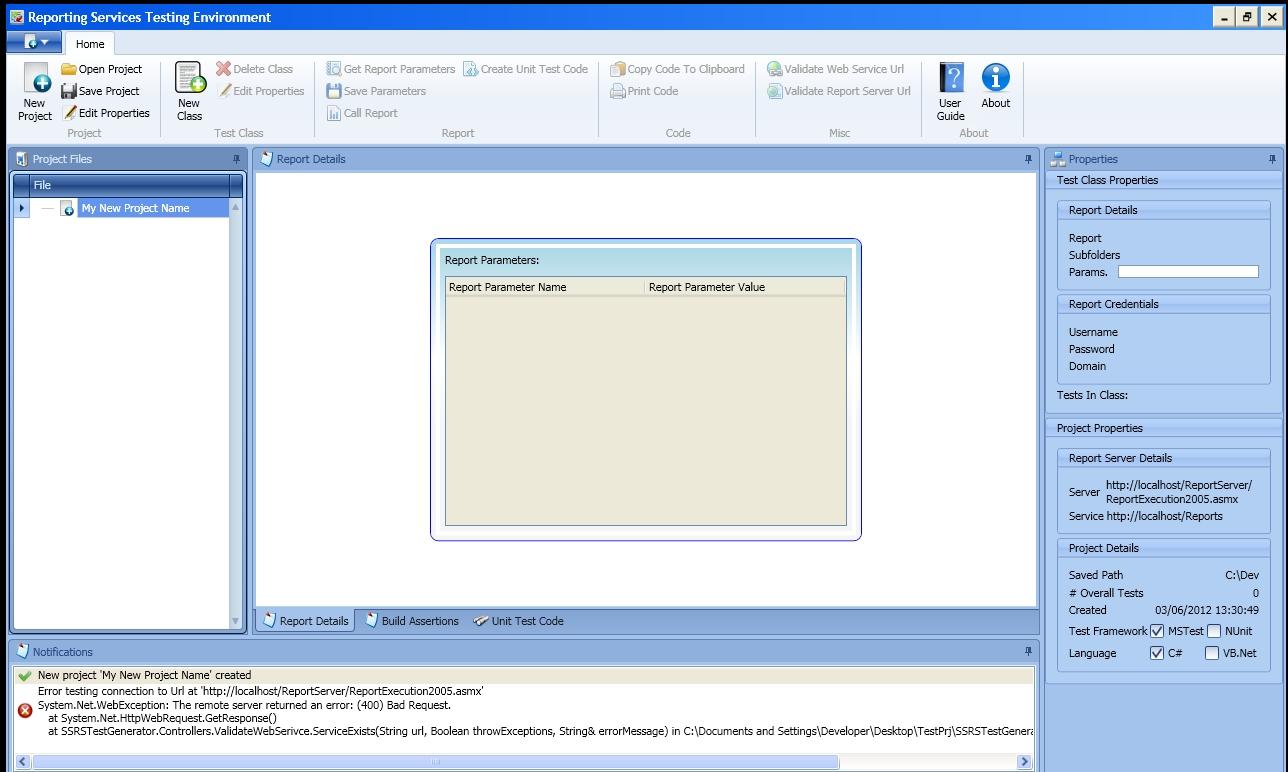


Fig 5

You can also see in the notifications tab (fig 5) at the bottom, the error message that was generated when the reporting service was un-contactable – this is noticeable by its (red) error icon.

# Saving Project

To save the project, simply click on the “Save Project” button (fig 6). A notification will be displayed in the notifications tab, to signify the status of the save event (fig 7).

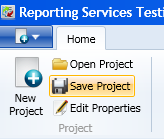


Fig 6



Fig 7

# Opening a Saved Project

To test that your project was saved successfully, you can reopen the saved project by clicking on the ‘Open Project’ button (fig 8) and browse to the saved path (which you will find in the properties panel on the right hand side fig 9).

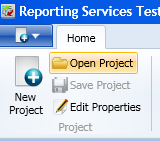


Fig 8

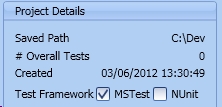


Fig 9

Once the ‘Open Project’ button has been clicked, the project browser dialog is displayed (fig 10), then click on the navigation button and browse to an ‘.rstp’ file (fig 11). Which will populate the project browser dialog as in fig 12. Then, click on the ‘Open’ button to reload the project.

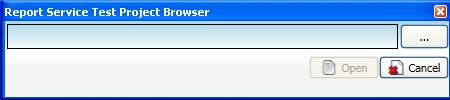


Fig 10

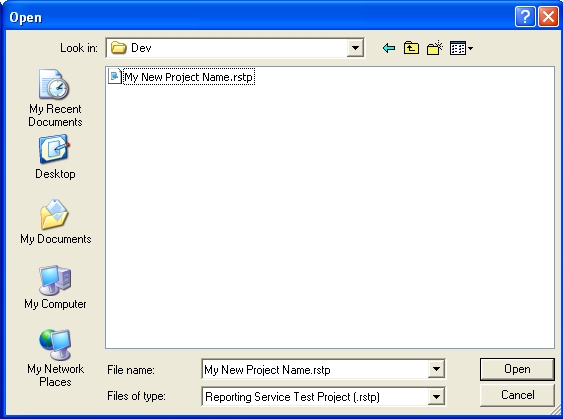


Fig 11

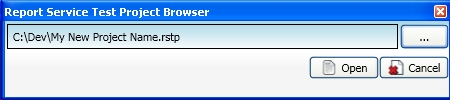


Fig 12

# Updating a Project’s Properties

If you wish to edit a project’s reporting service URL or its name, simply click on the ‘Edit Properties’ button within the ribbon bar (fig 13).

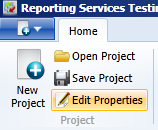


Fig 13

As in fig 14, you will be able to change the project name, reporting service URL, reporting server URL or the path to where the project has been saved to.

The language and test framework radio options are not editable. When you are happy with your changes, click on the save button, again the URL’s will be validated before being saved.

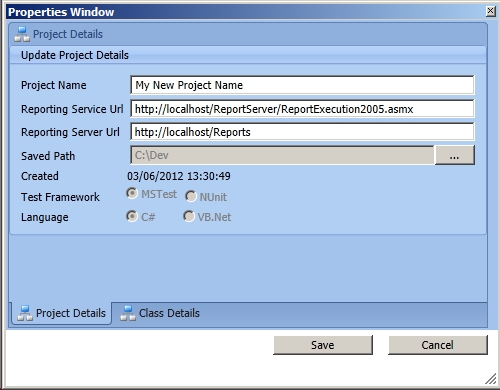


Fig 14

# Adding a Class\Report

The classes are what bind the code to the actual report that you want to create the unit tests for. To add a class click the ‘New Class’ button as in fig 15.

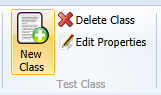


Fig 15

The new class dialog will open (fig 16), and the user must complete the class name and the report name (fig 17). The report path is the folder path (not the URL path) within the report server that the report has been deployed to (if deployed to the root of the report server, leave blank). The report credentials are optional, but if they are needed to view the report, they must be entered, so that the report can be retrieved by the application.

NB. The password will be displayed in clear text within the properties panel (fig 18), and the tests should be generated against a test server not a production server.

When you are happy with the new class, click on the save button. The new class will be displayed in the project file tree control on the left hand side (fig 19).

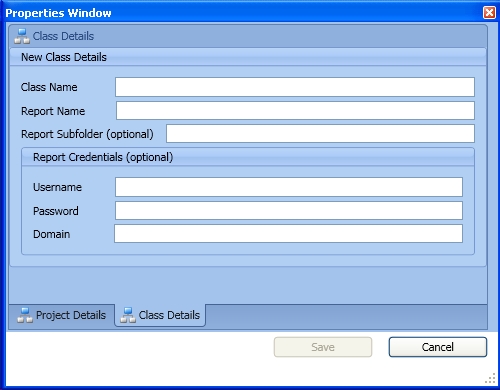


Fig 16

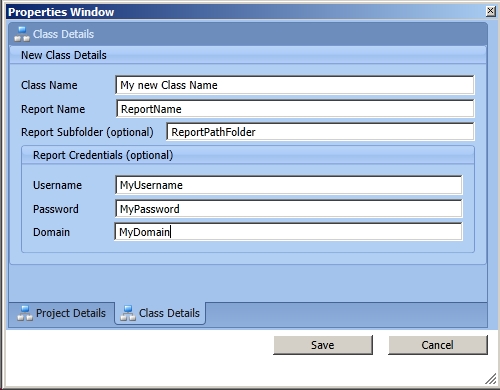


Fig 17

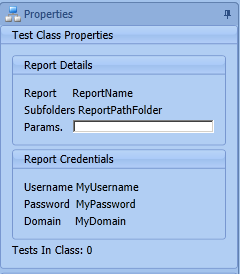


Fig 18

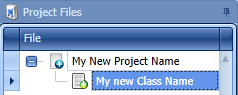


Fig 19

# Deleting a Class

If you need to remove a class from the project, select the class from the project tree and click the “Delete Class” button (fig 20). The user will be prompted (fig 21) before the class is deleted.

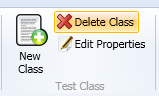


Fig 20

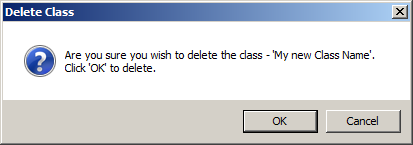


Fig 21

# Update a Class’s Properties

Select the class that you wish to update, and then click on the “Edit Properties” button (fig 22) within the test class tab, to bring up the edit dialog (fig 23). When you are happy with your changes, click on the save button.

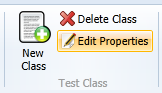


Fig 22

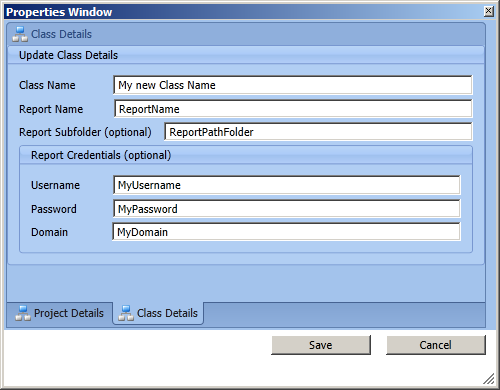


Fig 23

# Property Panel

This tab (fig 24) will keep the user informed about the currently selected class and the project in general. There are a couple things of interest within the tab, notably the references to the number of tests in a class and the overall number of tests for the project.

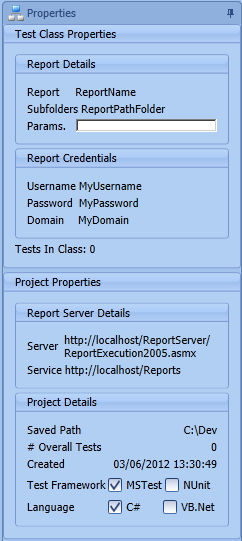


Fig 24

# Verify reporting Server & Service URL’s

The project’s reporting service and report server URL’s can be validated at any time by clicking on the respective button within the ribbon bar (fig 25\27), and any notifications will be displayed (fig 26\28).

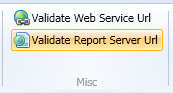


Fig 25



Fig 26

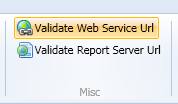


Fig 27



Fig 28

# Get a Report’s Parameters

If your report uses parameters to filter out data, then you test will also need to provide these parameters (unless you wish to use the report default parameters). Click on the “Get Report Parameters” button within the ribbon bar (fig 29).

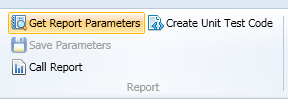


Fig 29

This will display the parameter entry grid (fig 30) that the user must save (fig 31) when the parameters have been entered, along with then saving the project. A notification will be displayed as to the success of the save event.

Double click the grid’s value cell, to enter edit mode and type in your value that you wish to be passed to the report in your tests.

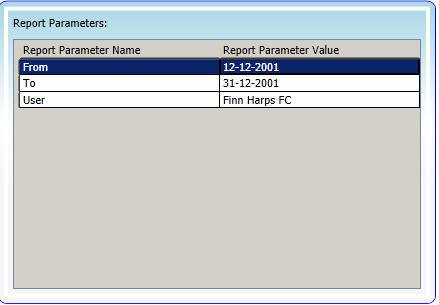


Fig 30

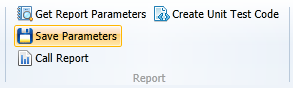


Fig 31

# Call the Report

To get the report, click the “Call Report” button (fig 32) within the ribbon bar. So that you can assign assertions against it, that will inturn be turned into C# test assertion statements.

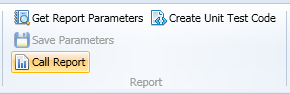


Fig 32

Once the report has been called, the build assertion screen will be displayed. The report is transformed into an xml document that is bound to tree control. Only items that are in the body of the report are able to be tested and thus are displayed within the (xml) tree control.

The user can find sub-reports within sub-reports in the tree control, and also report controls like graphs are also testable.

It is important to name your controls within your Microsoft Report appropriately, so that they can be found within the xml tree.

If the user selects a tree node, which in the case of the screenshot below (fig 33) is a tablix report control displaying grid like information, the first combo box (fig 34) will contain the tablix grid cells that you can test.

Report Tablix Data



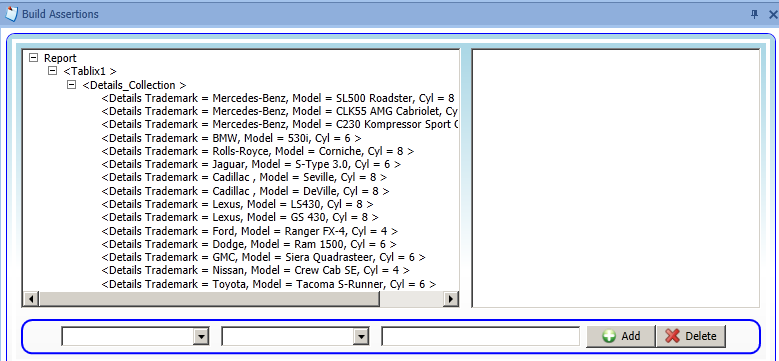


Fig 33

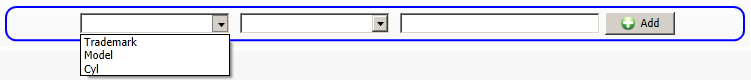


Fig 34Add an Assertion

Select the tree node that you wish to test against and then select the node’s element from the combo box (fig 34). Then, select the appropriate assertion statement from the second combo box (fig 35).

NB. That the assertions will reflect the test framework initially selected (MSTest or NUnit)

NB. Some MSTest assertions (DoesNotMatch and Matches) use the Regex object, but you only have to enter the matching string and the application will create the Regex assertion code for you, see fig 38 for example.

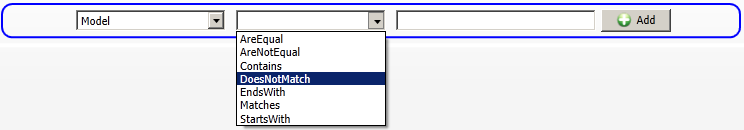


Fig 35

The next step is to add a value that you wish to test the element’s value against (fig 36), then click the “Add” button. The newly added assertion will be displayed in the listbox (fig 37).



Fig 36

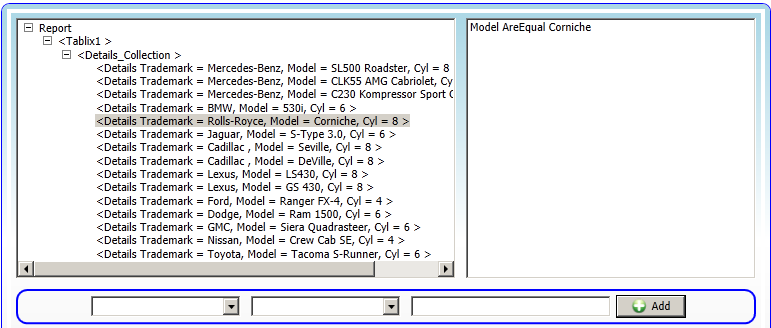


Fig 37

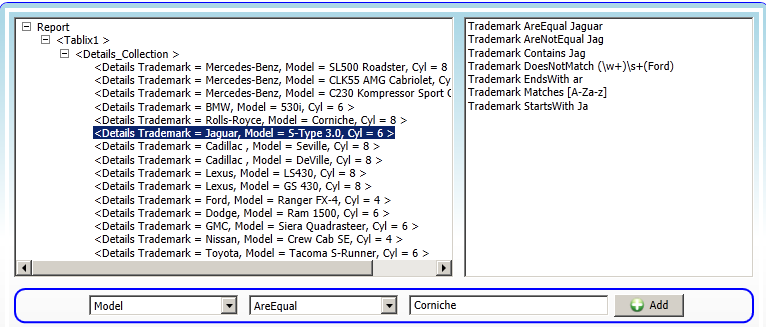


Fig 38

# Update an Assertion

To alter an assertion, simply click on the assertion within the listbox on the right hand side (fig 39). By clicking on the assertion, the combo boxes will be populated with the appropriate information, and then the user can update the assertion and update the assertion by clicking the update button.

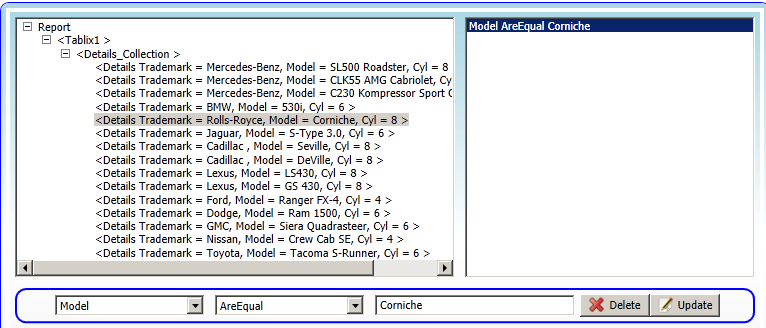


Fig 39

# Delete an Assertion

To remove an assertion, select the assertion (fig 39) from the listbox on the right hand side and click on the delete button.

# Generate Test Unit Code

Once the user has created the assertions, to then generate the unit test code, simply click on the “Create Unit Test Code button (fig 40). This in turn will automatically generate the test code fig 41

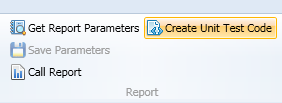


Fig 40

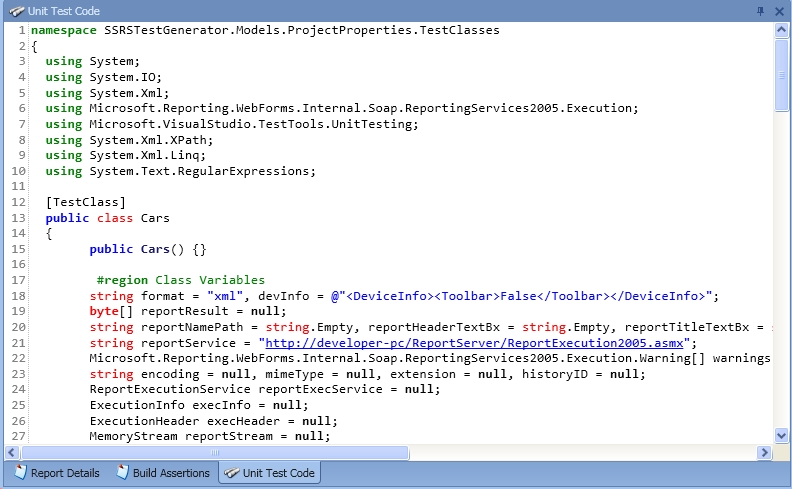


Fig 41

The code within this screen is made up of plumbing code, to call the report with or without parameters, process the returned bytes into xml and create the unit code tests. The start of the code is made up with the plumbing code, and if the user scrolls down they will see the code for their assertions (fig 42).

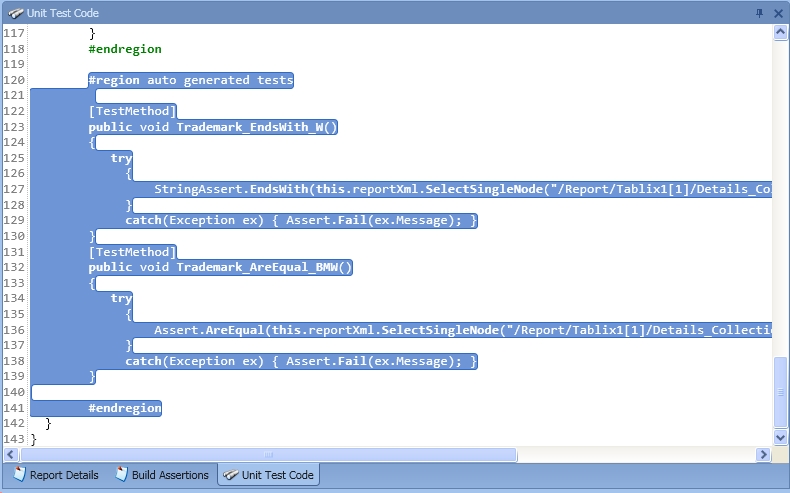


Fig 42

# Print Generated Unit Code

Once the code has been generated, the user can print the code to the default printer by clicking the “Print Code” (fig 43) button within the ribbon bar.

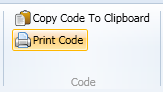


Fig 43

# Copy Generated Unit Code to Clipboard

The user can copy all the generated code to the clipboard for use within other application by clicking the “Copy Code To Clipboard” (fig 44) button within the ribbon bar.

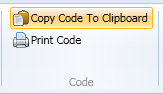


Fig 44

# About

To get information about the application, click on the “About” button (fig 45). This will display the dialog (fig 46) providing information about the application version, test framework versions along with contactable details.

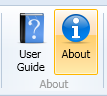


Fig 45

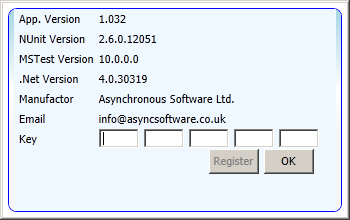


Fig 46

# Registering

When you purchase a key, it will be emailed to you. This can then be entered into the key boxes (fig 46). Once all the key boxes have been populated the registration button will be enabled. Once clicking this, you will be prompted to restart the application for the changes to take effect.

# Miscellaneous

## Application Restrictions

The application has the following restrictions:

1. Only able to create C# unit tests currently
2. Generated test framework code is either MSTest or NUuit
3. By default, the reporting service does not cater for localisation – the service will only allow English(US) - this is a Microsoft restriction
4. Not able to create exception based assertions – but to get around this, you can use the application to generate the assertion and copy the code into your project, where you can convert the assertion into an exception based assertion.
5. String based assertions – because the data is xml based, only string comparisons can be performed

## Localisation Constraints

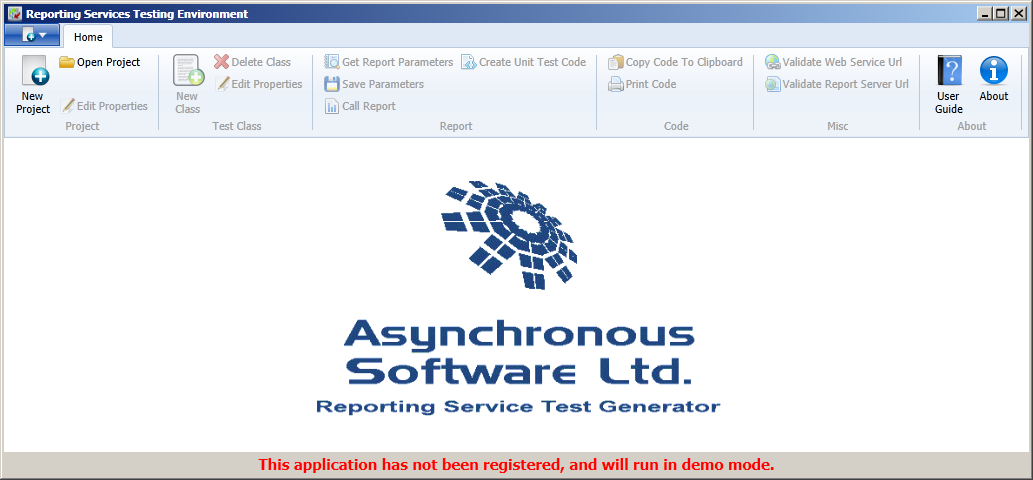
By default (mentioned in point 3 above), the reporting service will only handle English (US) localisation (Microsoft service restriction).

## Demo Restrictions

The following features will not be available in the demo\trial version:

* Saving project button has been removed
* Not able to add more than one class
* Not able to delete a class

There is also a banner displayed at the bottom of the screen in red – stating that this is a demo version (fig xx)



## Requirements

* Operating System – Windows XP or above (x86-bit)
* Microsoft Word (for viewing help document)