**MCFTestLib User Guide**

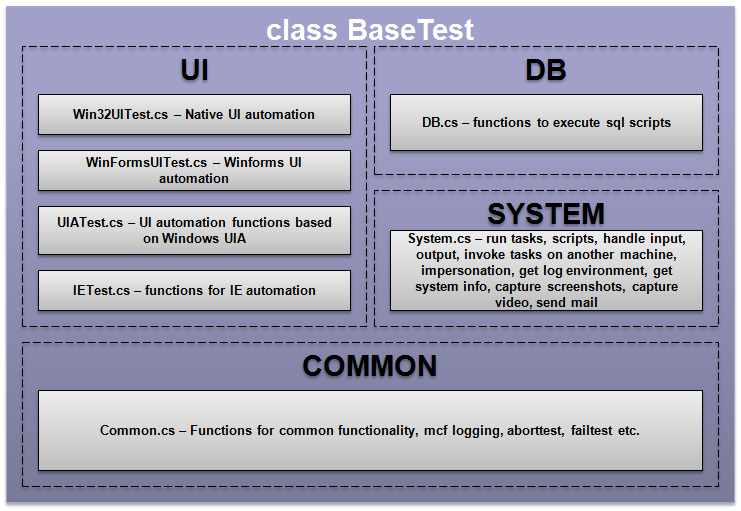
**What's new?**

Version 2.4 of the MCFTestLib adds support for screen capture and recording. You can use the new functions in the library to record screen while your tests are running and then watch the stored video for analysis. See [Screen Capture and Recording](http://codebox/mcf/Wiki/MCFTestLib+User+Guide/Screen+Capture+and+Recording) for more information.

**What is MCFTestLib?**

MCFTestLib is a managed test automation library that contains functions for various common automation and verification tasks such as

* WPF & SilverLight UI Automation
* Win32 UI automation
* WinForms UI automation
* IE automation
* DB testing
* Screen Capture and Recording
* Local and remote task invocation
* User impersonation

The following diagram shows the areas of functionality that the library offers as well as its architecture.  
There are two flavors of the library, one that can be used in standalone mode and another (integrated) that can be invoked from MCF scenario file (AKA varmap). See examples below.  
One can either compile with the library's source code or reference the prebuilt dll/assembly. The library depends on MAUI for ui automation and RE.exe for remote process invocation. All the dependencies are included in the release files.  
  


**Sample Code and Usage**

The following samples shows how to use the library from your test code with or without MCF...

**With MCF**

* Create a new managed class library in Visual Studio
* Add a reference to mcftestlib.dll and a "Using MCF" directive to your class file
* Make your class inherit from MCF.BaseTest class. This will make all the functions in the library available to your test dll.
* You can add custom functions to your class or override the ones in MCF.BaseTest and build your class library dll.

The following example shows a simple IE test written in MCF variation map (assumes you have a SampleTest.SampleTest class that inherits from MCF.BaseTest):

<var set="1" lvl="2" vid="1000" cls="SampleTest.SampleTest" dsc="Sample IE Test">

<fnc>OpenIE;url=www.microsoft.com</fnc>

<fnc>VerifyHtml;html\_rx=microsoft</fnc>

<fnc>Neg\_VerifyHtml;html\_rx=google</fnc>

<fnc>CloseIE</fnc>

</var>

The test case opens IE, navigates to microsoft.com home page, verifies that the page includes the word microsoft and that it doesn't include the word google. If all these verification function pass, it closes Internet Explorer window and test case is marked as passed. If any of the functions fail, MCF will report the test case as failed.

**Without MCF**

* Create a new managed console application in Visual Studio.
* Add a reference to mcftestlib.dll and a "Using MCF" directive to your program.cs file.
* Add the following code to your program.cs file

class Program : BaseTest

{

static void Main(string[] args)

{

Program program = new Program();

try

{

program.OpenCloseIETest();

Console.WriteLine("IE Test passed.");

}

catch (TestFailedException e)

{

Console.WriteLine("IE Test failed. Reason = " + e.ToString());

}

}

public void OpenCloseIETest()

{

OpenIE("www.microsoft.com");

VerifyHtml("microsoft");

Neg\_VerifyHtml("google");

CloseIE();

}

}

The sample code opens Internet Explorer, goes to microsoft.com page, verifies the the text 'microsoft' is present on the page and verifies that the text 'google' doesn't exist on the page. If all these verification function pass, it closes Internet Explorer window and returns without any errors. If any of these verification functions fail, a TestFailedException is thrown which the caller catches and reports the test case failure.   
  
For more information about the available functions in different areas of the library, see the following pages:

* [UI Automation](http://codebox/mcf/Wiki/MCFTestLib+User+Guide/UI+Automation)
* [DB functions](http://codebox/mcf/Wiki/MCFTestLib+User+Guide/DB+functions)
* [System functions](http://codebox/mcf/Wiki/MCFTestLib+User+Guide/System+functions)
* [Screen Capture and Recording](http://codebox/mcf/Wiki/MCFTestLib+User+Guide/Screen+Capture+and+Recording)