Rapise® | Using Rapise with MbUnit

Inflectra Corporation

Table of Contents

About MbUnit Integration

SeSMbUnit is a sample of using MbUnit. We provide special attribute to help executing Rapise GUI tests from within MbUnit tests.

Standard MbUnit test looks like this:

```
using System;
using MbUnit.Framework;

[TestFixture]
public class MyTests
{
    [Test] – THIS ATTRIBUTE FOR STANDARD MbUnit Test
    public void MyTest1()
    {
        Assert.AreEqual(1, 2, "Check equality");
    }
}
```

Each test case is a function with a special attribute `[Test]`. MbUnit uses it to find test cases, then collects cases in sets and so on.

Rapise integration makes execution of Rapise tests as simple as execution of normal MbUnit tests:

```
using System;

using MbUnit.Framework;
using SeSMbUnit; – We include Rapise helper class

[TestFixture]
public class PlayerTesting
{
    – Next line means:
    “this is an MbUnit test that executes Cross Browser.sstest from Rapise”
    [SeSMbUnitTest(@"T:\Samples\Cross Browser\CrossBrowser.sstest")]
    public void TestIEandFirefox()
    {
        int exitCode = SeSMbUnitHelper.TestExecute();
        Assert.AreEqual(0, exitCode);
    }
}
```

Now we use another attribute to mark the test:

```
[SeSMbUnitTest(@"<path to .sstest>")]
```

We just mark this test as a wrapper for concrete Rapise test instance.

Installing MbUnit

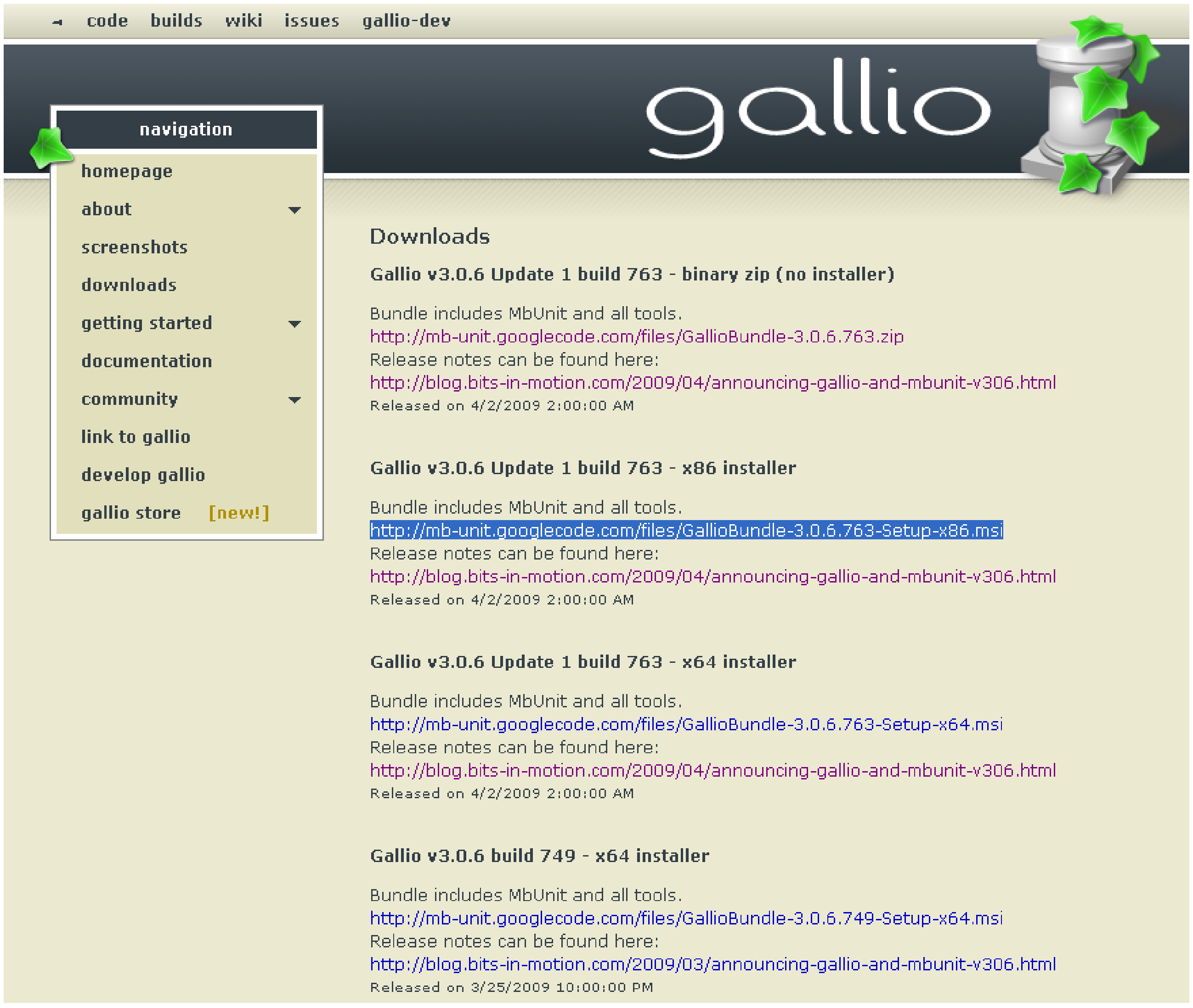
SeSMbUnit is a sample of using MbUnit with costume attribute, `[SeSMbUnitTestAttribute("…\Project.sstest")].` This attribute applies one parameter, path of the test, which must be checked.

To see what is doing this attribute first of all we need MbUnit and Gallio.

MbUnit is a unit testing framework in the tradition of xUnit frameworks such as JUnit. In addition, MbUnit includes a rich suite of features designed to simplify other automation tasks that arise during integration testing.

The Gallio Automation Platform is an open, extensible, and neutral system for .NET that provides a common object model, runtime services and tools (such as test runners) that may be leveraged by any number of test frameworks. MbUnit v3.is a default test framework for Gallio.

We need MbUnit v3 and because it is bundled with Gallio you need just to download Gallio latest release. It is available on http://www.gallio.org/Downloads.aspx :



Downloads

Gallio v3.0.6 Update 1 build 763 - binary zip (no installer)

Bundle includes MbUnit and all tools.
http://mb-unit.googlecode.com/files/GallioBundle-3.0.6.763.zip
Release notes can be found here:
http://blog.bits-in-motion.com/2009/04/announcing-gallio-and-mbunit-v306.html
Released on 4/2/2009 2:00:00 AM

Gallio v3.0.6 Update 1 build 763 - x86 installer

Bundle includes MbUnit and all tools.
http://mb-unit.googlecode.com/files/GallioBundle-3.0.6.763-Setup-x86.msi
Release notes can be found here:
http://blog.bits-in-motion.com/2009/04/announcing-gallio-and-mbunit-v306.html
Released on 4/2/2009 2:00:00 AM

Gallio v3.0.6 Update 1 build 763 - x64 installer

Bundle includes MbUnit and all tools.
http://mb-unit.googlecode.com/files/GallioBundle-3.0.6.763-Setup-x64.msi
Release notes can be found here:
http://blog.bits-in-motion.com/2009/04/announcing-gallio-and-mbunit-v306.html
Released on 4/2/2009 2:00:00 AM

Gallio v3.0.6 build 749 - x64 installer

Bundle includes MbUnit and all tools.
http://mb-unit.googlecode.com/files/GallioBundle-3.0.6.749-Setup-x64.msi
Release notes can be found here:
http://blog.bits-in-motion.com/2009/03/announcing-gallio-and-mbunit-v306.html
Released on 3/25/2009 10:00:00 PM

Figure 1

On the site is also available .msi file, so if download it you can easily install gallio and MbUnit on your computer.

Running MbUnit test

After Gallio is installed we can execute our test with costume attribute. Gallio comes bundled with many different runners, that is, programs or plug-ins for other programs that allows you to execute tests. This time we will use Icarus, the graphical runner, because it's a standalone application (meaning that you don't need to install anything else to run it). From “Start” select Gallio and from it Icarus GUI Test Runner:

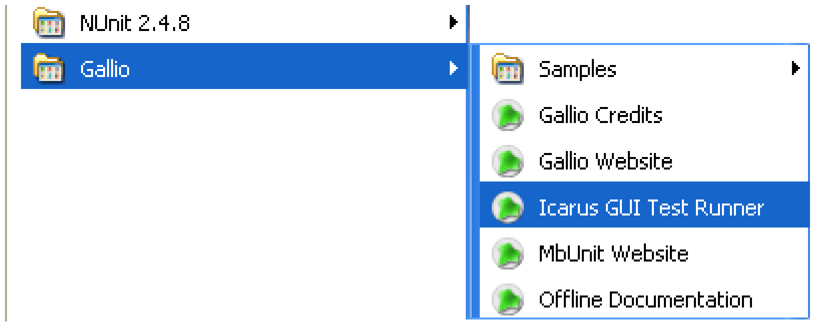


Figure 2

Once Icarus is running, go to Assemblies->Add Assemblies... in the menu:

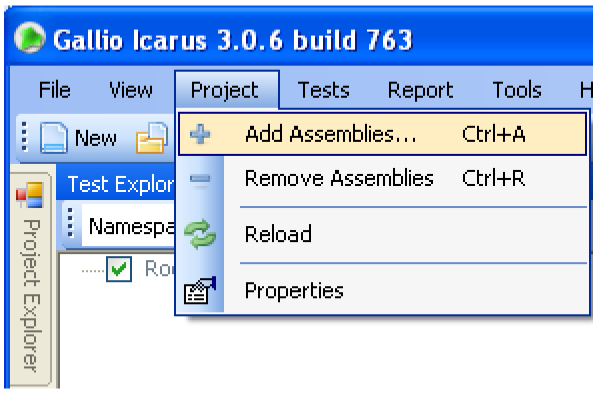
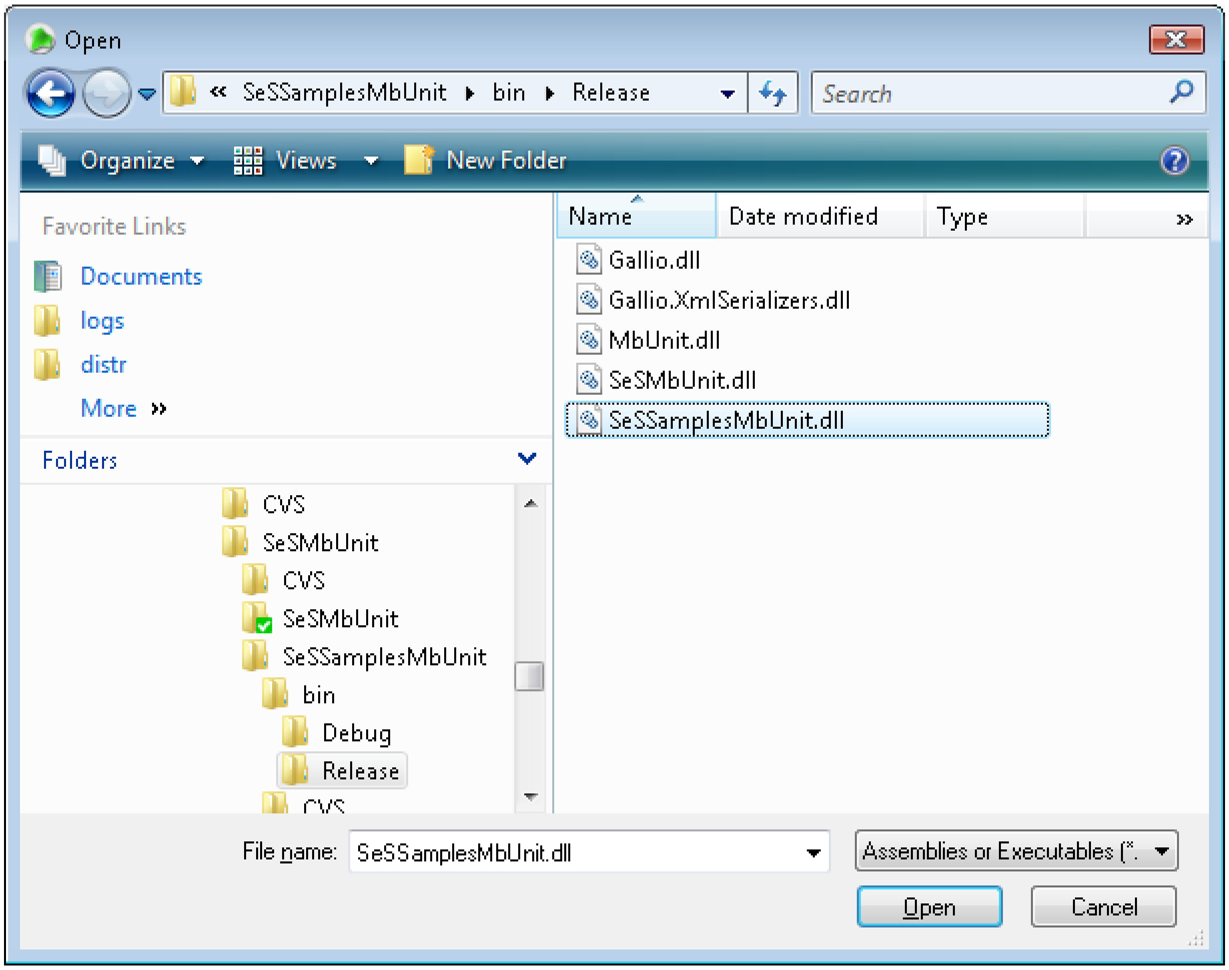


Figure 3

In opened window find and select
 <Rapise Setup
Folder>\Extensions\UnitTesting\MBUnit\SeSMbUnit\SeSSamplesMbUnit\bin\Debug\SeSSamplesMbUnit.dll
”. Press “Open”:



File name: SeSSamplesMbUnit.dll

Figure 4

The next step is to execute the test, for which you only need to press the Start button:

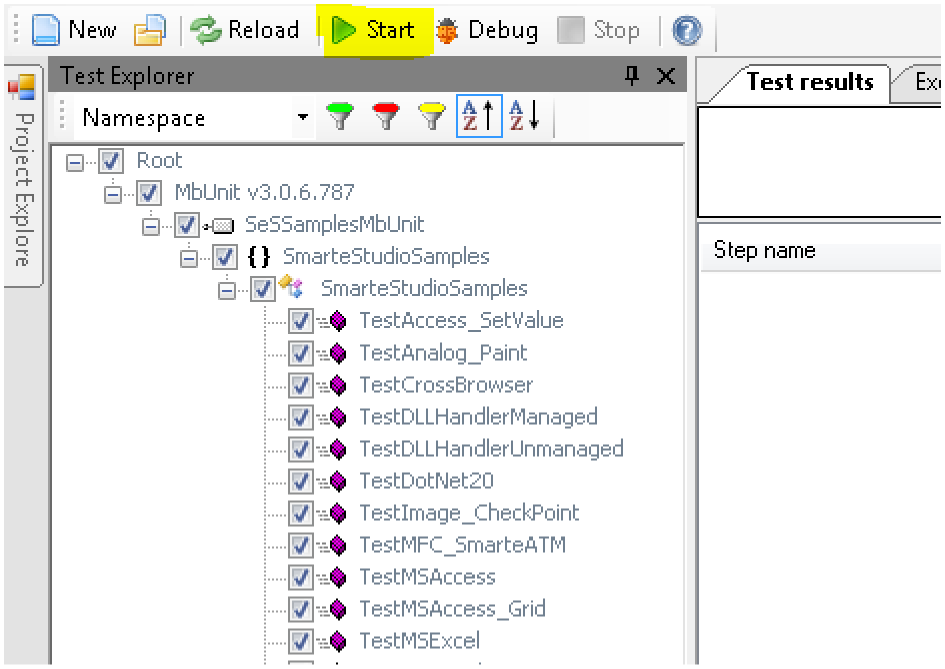
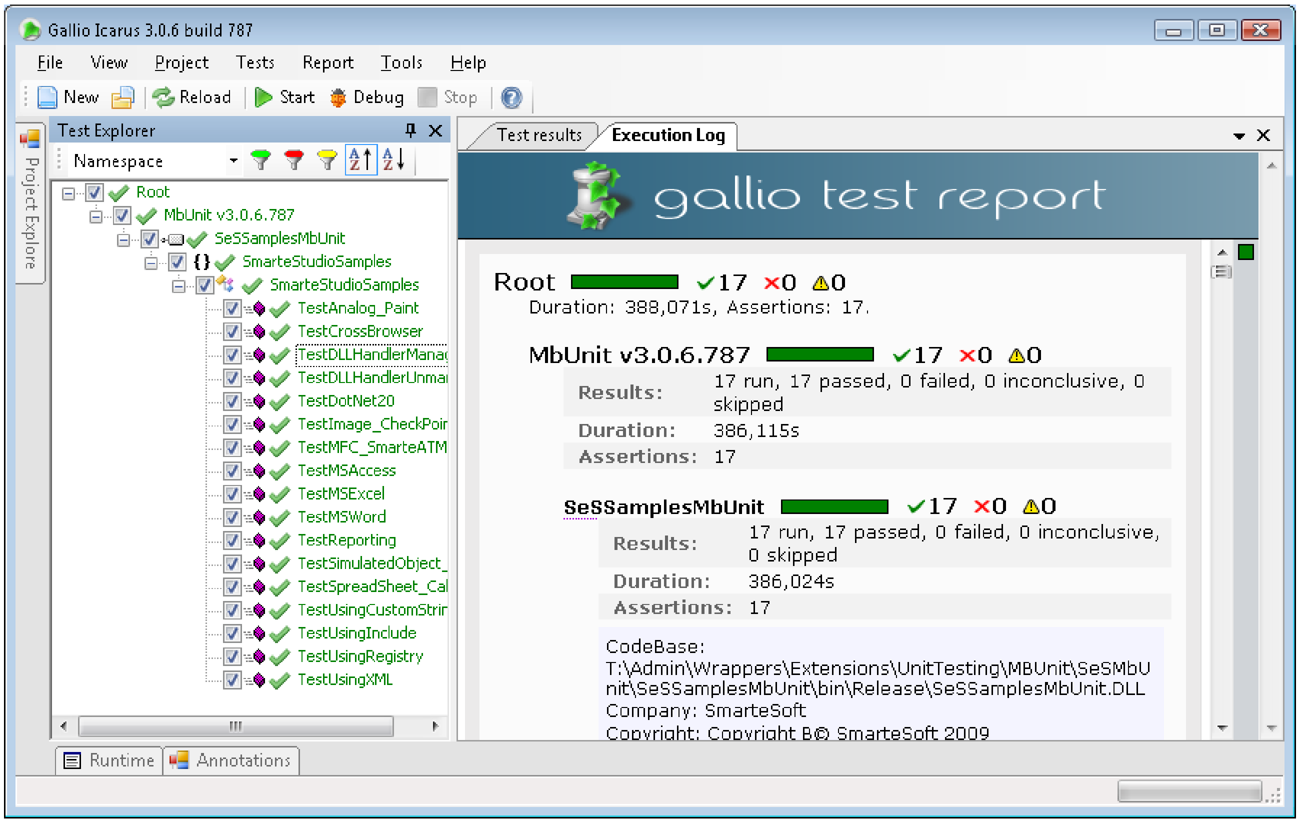


Figure 5

After doing so we see that it passes:



```
Root ✓17 ×0 ⚠0
  Duration: 388,071s, Assertions: 17.

  MbUnit v3.0.6.787 ✓17 ×0 ⚠0
    Results:     17 run, 17 passed, 0 failed, 0 inconclusive, 0
                 skipped
    Duration:    386,115s
    Assertions:  17

    SeSSamplesMbUnit ✓17 ×0 ⚠0
      Results:     17 run, 17 passed, 0 failed, 0 inconclusive,
                   0 skipped
      Duration:    386,024s
      Assertions:  17

      CodeBase:
      T:\Admin\Wrappers\Extensions\UnitTesting\MBUnit\SeSMbU
      nit\SeSSamplesMbUnit\bin\Release\SeSSamplesMbUnit.DLL
      Company: SmarteSoft
      Copyright: Copyright B© SmarteSoft 2009
```

Figure 6

But what does it mean for a test to "pass"? It means that it was executed, all its assertions were true and no exception was thrown. This is the basic structure of a test in the so called state-based testing: you create one on more objects, call a method and assert over the state of it after doing it.

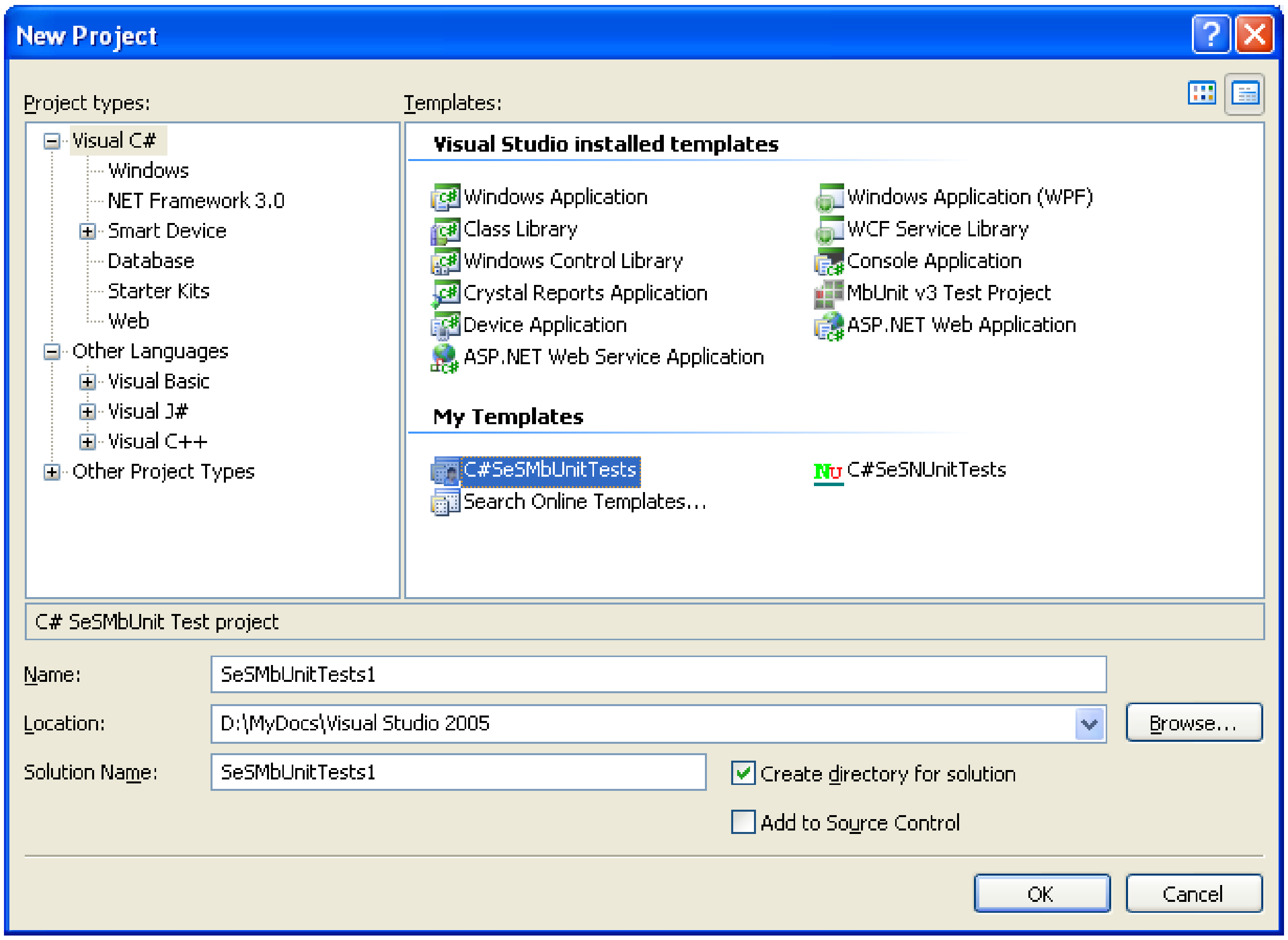
Visual Studio Integration

We provide code snippets and templates for smooth Visual Studio Integration. You may install them by double-clicking the file <Rapise Setup Folder>\Extensions\UnitTesting\MBUnit\SeSMbUnit.vsi

Creating SeSMbUnit test

Maybe you will want to write your own SeSMbUnit test. We have special template which will help you to do that. In this part we'll explain how to use it on C#.

The only thing you need to do is just to create "SeSMbUnitTests" type project. For that open VS2005, on "Start Page" click on "Project.."(in the right side of "Create:") and in opened window from "My Templates" part select "SeSMbUnitTests". If you want you can change name of dll. By default it is "SeSMbUnitTests1":



C# SeSMbUnit Test project

Name: SeSMbUnitTests1

Location: D:\MyDocs\Visual Studio 2005

Solution Name: SeSMbUnitTests1

[✓] Create directory for solution

Figure 7

In the created project open “Fixture1.cs” file. All necessary references are already added:

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Diagnostics;
using MbUnit.Framework;
using SeSMbUnit;
```

File also contains `Fixture1`class with `[TestFixture]attribute:`

```
[TestFixture]
public class Fixture1
{
    ...
}
```

TextFixtureAttribute specifies that a class represents a test fixture. This attribute is optional.

This attribute may be omitted whenever a test fixture class contains at least one test method or test parameter or when other MbUnit attributes are applied to the test fixture class. This is almost always the case unless for some reason you have an empty fixture.

The class must have a public default constructor. The class may not be static.

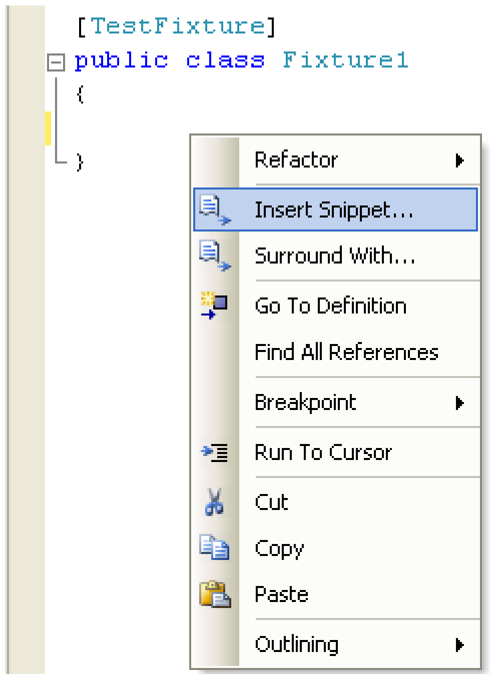
In class we have SetUp() and TearDown() methods , and one more test method:

```
    [SetUp()]
    public void SetUp()
    {
        //TODO - Setup your test objects here
    }

    [TearDown()]
    public void TearDown()
    {
        //TODO - Tidy up your test objects here
    }

    [SeSMbUnitTest("")]
    public void TestSeS()
    {
        int exitCode = SeSMbUnitHelper.TestExecute();
        Assert.AreEqual(0, exitCode);
    }
```

Now you also have a snippet, by which you can easily add "TestSeS" method with `[SeSMbUnitTest(@"<path to .sstest>")] attribute.`Right click in class body and from opened context menu select "Insert Snippet…":

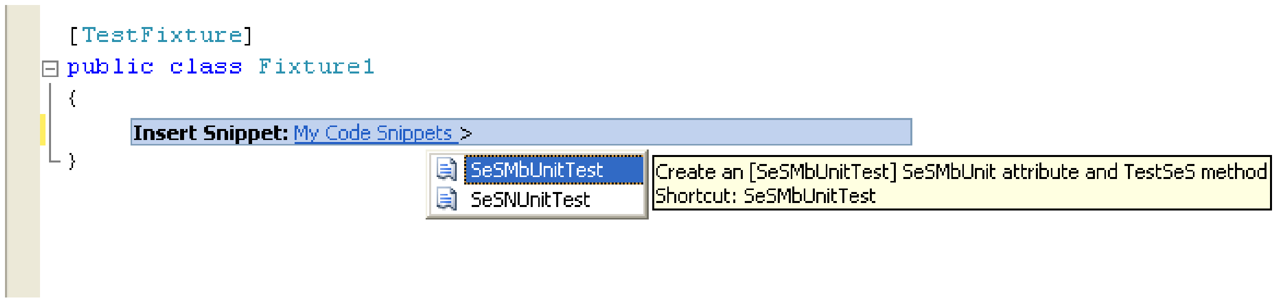


```
[TestFixture]
public class Fixture1
{

}
```

Figure 8

From it select "My Code Snippets"(if your snippets are in "My Code Snippets" folder , otherwise select proper folder), and then "SeSMbUnitTest":



```
[TestFixture]
public class Fixture1
{

}
```

Create an [SeSMbUnitTest] SeSMbUnit attribute and TestSeS method
Shortcut: SeSMbUnitTest

Figure 9

This code will be added:

```
[SeSMbUnitTest(/*Insert path to .sstest file which must be run.*/)]
public void TestSeS()
{
    int exitCode = SeSMbUnitHelper.TestExecute();
    Assert.AreEqual(0, exitCode);
}
```

You just need to add path of .sstest file to `SeSMbUnitTest` attribute. If you add code via snippet in standard SeSMbUnit Test project you will have two “TestSeS” methods, so don’t forget to change the name of one of them.