NUnit-Testlink-Adapter

Authors: A.A. Momeni Vesalian, A. Nikooee, S. Mokari
Date: June 4th, 2016

Copyright Notice

About

The NUnit-Testlink-Adapter (NTA) allows the automated declaring and exporting of test run results into Testlink.

NUnit is a unit-testing framework for all .Net languages. Initially ported from JUnit, the current production release, version 2.6.4. It is written entirely in C# and has been completely redesigned to take advantage of many .NET language features, for example custom attributes and other reflection related capabilities. NUnit brings xUnit to all .NET languages.

NUnit also includes its own command-line runner, Echo, and a Windows GUI.

To find out more about NUnit go to http://www.NUnit.org

TestLink enables easily to create and manage Test cases as well as organize them into Test plans. These Test plans allow team members to execute Test cases and track test results dynamically, generate reports, trace software requirements, prioritize and assign tasks.

To find out more about Testlink go to www.teamst.org

A Quick Run Through

(this assumes you are comfortable with unit test harnesses).

To make use of the NTA, you must first add a new attribute to each TestFixture. This attribute, called **NUnit.TestLink.TestLinkFixture** takes parameters that tell the adapter information about test cases and use **NUnit.TestLink.TLAssert** instead of **NUnit.Framework.Assert** to record test results.

Typically the tests are run as part of a build script or a batch file. The GTA is an extension to the command line runner.

When the tests have been run, the extension is called to record the results.

The extension decorates **NUnit.Framework.Assert** functionalities, parses the results, loads the test assemblies that have been run and extracts the test link fixture. It then contacts TestLink to look for test cases with the same name as they test methods in the unit test.

If one of the test project, test plan, build, test suite or test case does not exist, GTA will create them and assign them to the test plan. Because of the lack of API for creating platform, after the first run, user must create platform manually, assign it to test plan and run the tests again.

It then records the test result and send them to TestLink.

Installation

Run Time Files

Copy TestLinkAdapter.dll into project library folder and then install these packages by NuGet Package Manager :

- NUnit 2.6.4 (http://www.nuget.org/packages?q=xmlrpcnet)
- TestLinkFixture 1.0.0 (http://www.nuget.org/packages?q=testlink)
- NUnitTestLinkAddOn 1.0.0 (http://www.nuget.org/packages?q=testlink)
- XML-RPC.NET-Client 3.0.0.266 (http://www.nuget.org/packages?q=xmlrpcnet)
- log4net 2.0.5 (https://www.nuget.org/packages?q=log4net)

Invocation

In this example the console runner is used:

NUnit-console.exe *[unitTestAssembly]*

TestLink Setup

In order for this to work you need to have in testlink:

a) Enabled the API (see installation manual)
b) Created an account with admin permission for testing
c) Created an ApiKey for this account
d) After first run, create platform and assign it to test plan

Unit Test setup

Write your test cases as you are used to and use **NUnit.TestLink.TLAssert** instead of **NUnit.Framework.Assert**.

Add the **NUnit.TestLink.TestLinkFixture** to your testfixture.

The parameters are:

- **Url:** URL of the TestLink Api
- **ProjectName:** The name of the test project in TestLink
- **ProjectPrefix:** The prefix of the test project in TestLink. If the ProjectName is not defined in TestLink and this property is not set, the first letter of the ProjectName will be used.
- **ProjectDescription:** The description of the test project in TestLink. If the ProjectName is not defined in TestLink and this property is not set, ‘Automated Test Project’ will be used.
- **UserId:** UserName under which the test cases are authored
- **DevKey:** An ApiKey (provided by the TestLink application)
- **TestPlanName:** The test plan name in the test project. If the TestPlanName is not defined in TestLink and this property is not set, ‘Default-Test-Plan’ will be used.
- **TestPlanDescription:** The test plan description in the test project. If the TestPlanName is not defined in TestLink and this property is not set, ‘Automated Test Plan’ will be used.
- **BuildName:** The name of build for test plan in TestLink. If this property is not set, the latest build name will be used and if there is not any build, 'Default-Build' will be used.
- **BuildDescription:** The description of build for test plan in TestLink. If BuildName is not set, the latest build will be used and if there is not any build, 'Automated Build' will be used.
- **TestSuiteName:** The name of the top level test suite where the test cases are expected. If this property is not set, The name of test class will be used.
- **TestSuiteDescription:** The description of the top level test suite where the test cases are expected. If this property is not set, at first description of TestFixtureAttribute will be used and if that property is not set, 'Automated Test Suite' will be used.
- **PlatformName:** The name of the platform that is linked to the test plan in TestLink. If this property is not set or the platform is not linked to test plan, all test cases will be created but none of them will be executed.

Example

```
using NUnit.Framework;
using NUnit.TestLink;

namespace TestLinkAdapter.Test
{
    [TestFixture(Description = "According to goal of the suite ...")]
    [TestLinkFixture(
        Url = "http://localhost/testlink/lib/api/xmlrpc/v1/xmlrpc.php",
        ProjectName = "NUnit-TestLink-Adapter-Test",
        ProjectPrefix = "NTAT",
        ProjectDescription = "Test Project that try to test ...",
        UserId = "your-admin-user",
        DevKey = "67aa0e00f605bfd00f78a31c9ed96fb8",
        TestPlanName = "NUnit-TestLink-Adapter-Test-Plan",
        TestPlanDescription = "In this plan we try to test all ...",
        BuildName = "NUnit-TestLink-Adapter-Build",
        BuildDescription = "This is default build.",
        TestSuiteName = "TLAssert-Conditional-Expresion-Test",
        TestSuiteDescription = "This Suite is collection of tests that try to ...",
        PlatformName = "NUnit-TestLink-Adapter-Platform")]
    public class TLAssertConditionalExpresionTest
    {
        [Test(Description = "Test of TLAssert.IsTrue() method by ...")]
        public void IsTrueByTrueArgumentTest()
        {
            TLAssert.IsTrue(true);
        }
    }
}
```

Also you can use Description property of TestFixture and Test attribute, for using in test suites and test cases descriptions.

References

To implement this project, https://github.com/mathume/nunit_testlink_adaptor source code have been used.