# C# TDD with xUnit.net

**Suthep Sangvirotjanaphat**

Facebook.com/groups/GreatFriends.Biz

Facebook.com/suthep

# Agenda

- Introduction to TDD Concept
- xUnit.NET Unit Testing Framework
- Running Tests in Console and Visual Studio
- xUnit Assertions
- Should Assertion Library
- Test data from CSV and Excel Files
- Test Output
- Exercises!

# Unit Testing

- Function-Level Testing
- Automated Testing
- Run by Unit Testing Framework's Test Runner
  - MS Test (http://bit.ly/ms-testfx)
  - NUnit (http://nunit.org)
  - xUnit (http://xunit.github.io)

# Test that Drives

- Test First, Code After. Huh!?
- Test Drives for Requirement Understanding, Analysis, and Design.
- Test Drives for Working Code.
- Test is a live documentation.

- **RED:** Write a Test that failed
- **GREEN:** Write just enough code that passed all tests
- **REFACTORING:** Test & Code for better readability and maintainability

# A unit test method

- Arrange / Act / Assert
- SUT (System Under Test), CUT (Class..), MUT (Method..)

- class **Foo** { method **Bar**; method **Boo**; }
- class **FooFact** {
  class **Bar** { method Test1; method Test2; }
  class **Boo** { method Test3; method Test4; }
  }

# Using xUnit.NET

- Project **Demo**
- Project **Demo.Facts**
  - Install **xunit**
  - Install **xunit.runner.visualstudio**
  - Install **xunit.runner.console**
  - Add References to SUT (Project Demo)

# Run xUnit Testing in Console

```
@echo off
Packages\xunit.runner.console.2.0.0\tools\xunit.console ^
  GFDN.ThaiBahtTextFacts\bin\Release\GreatFriends.ThaiBahtTextFacts.dll ^
  -parallel all ^
  -html Result.html ^
  -nologo -quiet
@echo on
```

# Compare to other frameworks (1)

| NUnit 2.2 | MSTest 2005 | xUnit.net 2.x | Comments |
|---|---|---|---|
| `[Test]` | `[TestMethod]` | `[Fact]` | Marks a test method. |
| `[TestFixture]` | `[TestClass]` | *n/a* | xUnit.net does not require an attribute for a test class; it looks for all test methods in all public (exported) classes in the assembly. |
| `[ExpectedException]` | `[ExpectedException]` | `Assert.Throws` `Record.Exception` | xUnit.net has done away with the ExpectedException attribute in favor of `Assert.Throws` . See Note 1 |
| `[SetUp]` | `[TestInitialize]` | Constructor | We believe that use of `[SetUp]` is generally bad. However, you can implement a parameterless constructor as a direct replacement. See Note 2 |

# Compare to other frameworks (2)

| | | | |
|---|---|---|---|
| `[TearDown]` | `[TestCleanup]` | `IDisposable.Dispose` | We believe that use of `[TearDown]` is generally bad. However, you can implement `IDisposable.Dispose` as a direct replacement. See Note 2 |
| `[TestFixtureSetUp]` | `[ClassInitialize]` | `IClassFixture<T>` | To get per-class fixture setup, implement `IClassFixture<T>` on your test class. See Note 3 |
| `[TestFixtureTearDown]` | `[ClassCleanup]` | `IClassFixture<T>` | To get per-class fixture teardown, implement `IClassFixture<T>` on your test class. See Note 3 |

# Compare to other frameworks (3)

| | | | |
|---|---|---|---|
| *n/a* | *n/a* | `ICollectionFixture<T>` | To get per-collection fixture setup and teardown, implement `ICollectionFixture<T>` on your test collection. See Note 3 |
| `[Ignore]` | `[Ignore]` | `[Fact(Skip="reason")]` | Set the Skip parameter on the `[Fact]` attribute to temporarily skip a test. |
| `[Property]` | `[TestProperty]` | `[Trait]` | Set arbitrary metadata on a test |
| *n/a* | `[DataSource]` | `[Theory]` `[XxxData]` | Theory (data-driven test). See Note 4 |

# Assertion (1)

| NUnit 2.2 | MSTest 2005 | xUnit.net 1.x | Comments |
|---|---|---|---|
| `AreEqual` | `AreEqual` | `Equal` | MSTest and xUnit.net support generic versions of this method |
| `AreNotEqual` | `AreNotEqual` | `NotEqual` | MSTest and xUnit.net support generic versions of this method |
| `AreNotSame` | `AreNotSame` | `NotSame` | |
| `AreSame` | `AreSame` | `Same` | |
| `Contains` | `Contains` | `Contains` | |
| `DoAssert` | *n/a* | *n/a* | |
| *n/a* | `DoesNotContain` | `DoesNotContain` | |
| *n/a* | *n/a* | `DoesNotThrow` | Ensures that the code does not throw any exceptions |

# Assertion (2)

| | | | |
|---|---|---|---|
| `Fail` | `Fail` | *n/a* | xUnit.net alternative: `Assert.True(false, "message")` |
| `Greater` | *n/a* | *n/a* | xUnit.net alternative: `Assert.True(x > y)` |
| `Ignore` | `Inconclusive` | *n/a* | |
| *n/a* | *n/a* | `InRange` | Ensures that a value is in a given inclusive range (note: NUnit and MSTest have limited support for `InRange` on their `AreEqual` methods) |
| `IsAssignableFrom` | *n/a* | `IsAssignableFrom` | |
| `IsEmpty` | *n/a* | `Empty` | |
| `IsFalse` | `IsFalse` | `False` | |

# Assertion (3)

| | | | |
|---|---|---|---|
| `IsInstanceOfType` | `IsInstanceOfType` | `IsType` | |
| `IsNaN` | *n/a* | *n/a* | xUnit.net alternative: `Assert.True(double.IsNaN(x))` |
| `IsNotAssignableFrom` | *n/a* | *n/a* | xUnit.net alternative: `Assert.False(obj is Type)` |
| `IsNotEmpty` | *n/a* | `NotEmpty` | |
| `IsNotInstanceOfType` | `IsNotInstanceOfType` | `IsNotType` | |
| `IsNotNull` | `IsNotNull` | `NotNull` | |
| `IsNull` | `IsNull` | `Null` | |

# Assertion (4)

| | | | |
|---|---|---|---|
| `Less` | *n/a* | *n/a* | xUnit.net alternative: `Assert.True(x < y)` |
| *n/a* | *n/a* | `NotInRange` | Ensures that a value is not in a given inclusive range |
| *n/a* | *n/a* | `Throws` | Ensures that the code throws an exact exception |

# Test Output

```
using Xunit;
using Xunit.Abstractions;

public class MyTestClass
{
    private readonly ITestOutputHelper output;

    public MyTestClass(ITestOutputHelper output)
    {
        this.output = output;
    }

    [Fact]
    public void MyTest()
    {
        var temp = "my class!";
        output.WriteLine("This is output from {0}", temp);
    }
}
```