NUnit and Mono

Cross-Platform Testing

Agenda

- Where NUnit came from
- Recent Developments
- What’s Coming Next

The Past

Programmers Have Always Written Tests

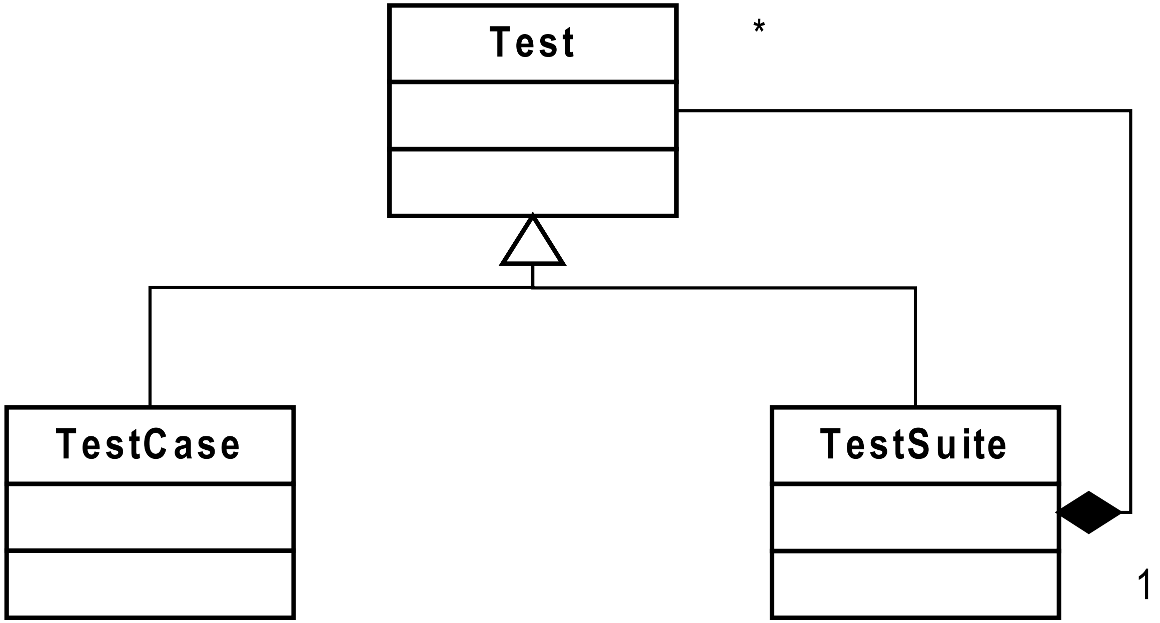
```
Class Calculator
{
  public double calc( double x, double y )
  {  …     }

#if TESTING
  static void Main()
  {
     double result = calc( 5.2, 7.3 );
     Console.WriteLine(
           “calc(5.2, 7.3) returns {0}”,
           result );
  }
#endif
}
```

We need a framework

sUnit

Kent Beck & Eric Gamma


Test
*
TestCase
TestSuite
1

SUnit
JUnit
CppUnit
CppUnitLite
DUnit
PerlUnit
RUnit
PyUnit
VBUnit
PhpUnit
HtmlUnit
HttpUnit

NUnit

NUnit Releases

• 1.x	JUnit-Like
• 2.0	Attributes, Gui, .Net 1.0
• 2.1	Gui Improvements, .Net 1.1 support, Mono support
• 2.2	More Gui Stuff, Attributes, .Net 2.0
• 2.2.4	Partial 2.4 Feature release

```
[TestFixture]
public class MoneyTest
{
    …
    [Test]
    public void SimpleAdd()
    {
       Money m12CHF = new Money(12, "CHF");
       Money m14CHF = new Money(14, "CHF");
       Money expected = new Money(26, "CHF");
       Money result = m12CHF.Add( m14CHF );
       Assert.AreEqual( expected, result );
    }
    …
}
```

The Present

Current Releases

- 2.2.8 Latest Bug fixes on 2.2.4 Feature Set
- 2.3.6293 Latest Development release (2.4 Beta 2)
- 2.2.0 Release used by Mono and MonoDevelop

NUnit 2.4 Features

- nunit.core.interfaces assembly
- WiX-based Windows installer
- .Net 2.0 and Mono support

NUnit 2.4 Features

- AreEqual/AreNotEqual array support
- CollectionAssert
- FileAssert
- PropertyAttribute
- DescriptionAttribute
- ExpectedException enhancements

NUnit 2.4 Features

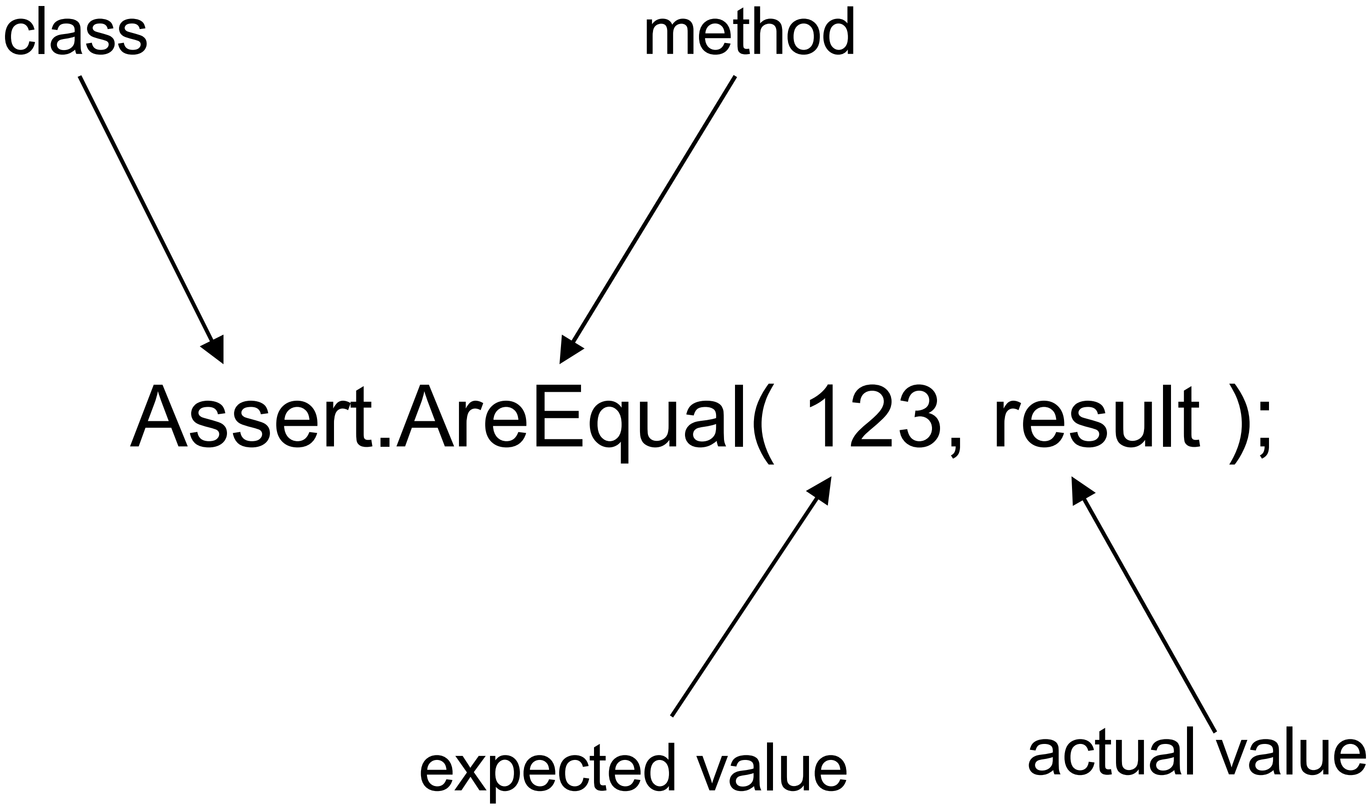
- Prior version compatibility
- Load in multiple AppDomains
- Flat list of test fixtures
- Merge tests across assemblies
- Parallel execution of tests (disabled)

NUnit 2.4 Features

- Mini-Gui
- Gui Improvements
  - Icons, Font Changes, Menu Items
- Extensibility
  - Addin Architecture
  - Core, Client and Gui
  - Only Core is implemented

NUnit 2.4 Demo

The Future



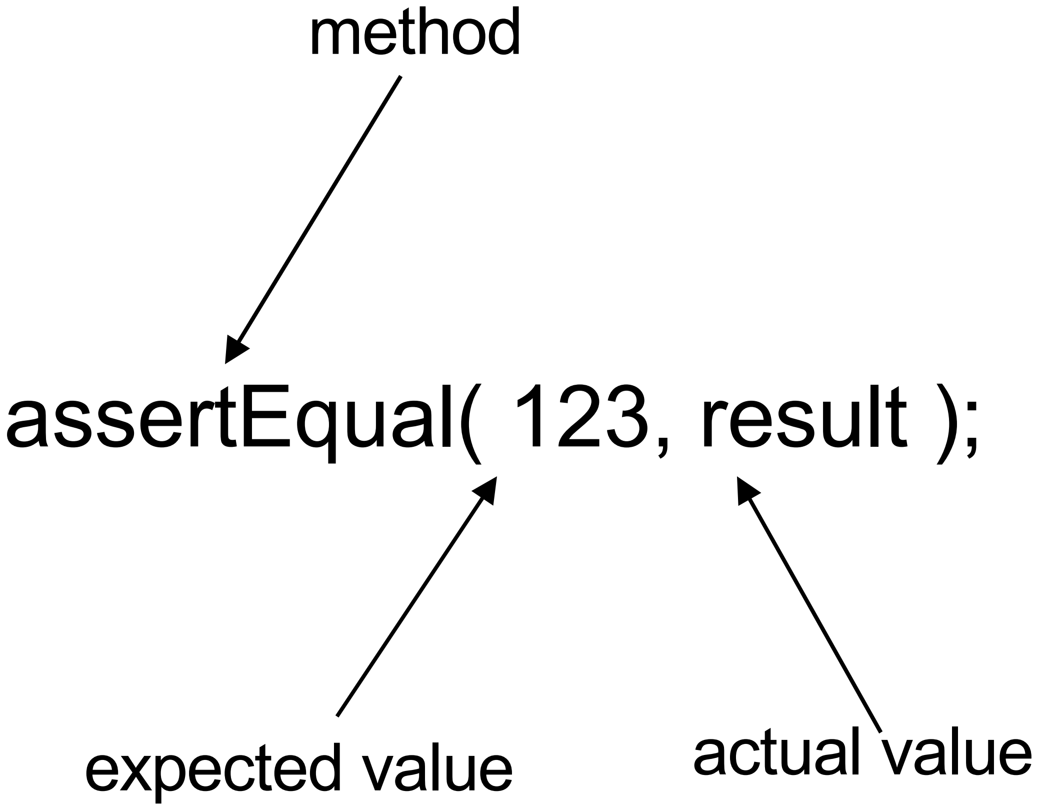
class

method

```
Assert.AreEqual( 123, result );
```

expected value

actual value

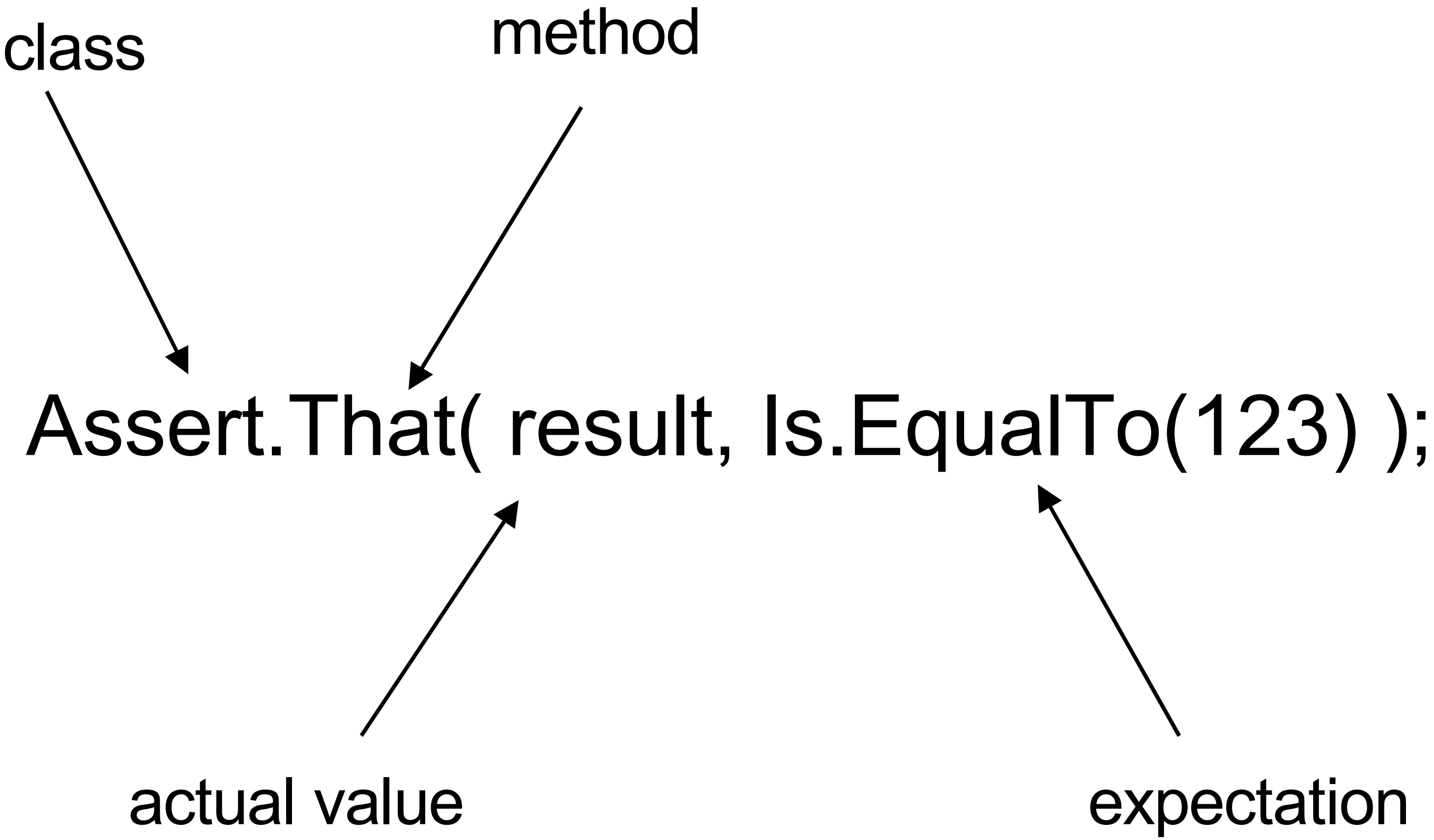


method

```
assertEqual( 123, result );
```

expected value

actual value



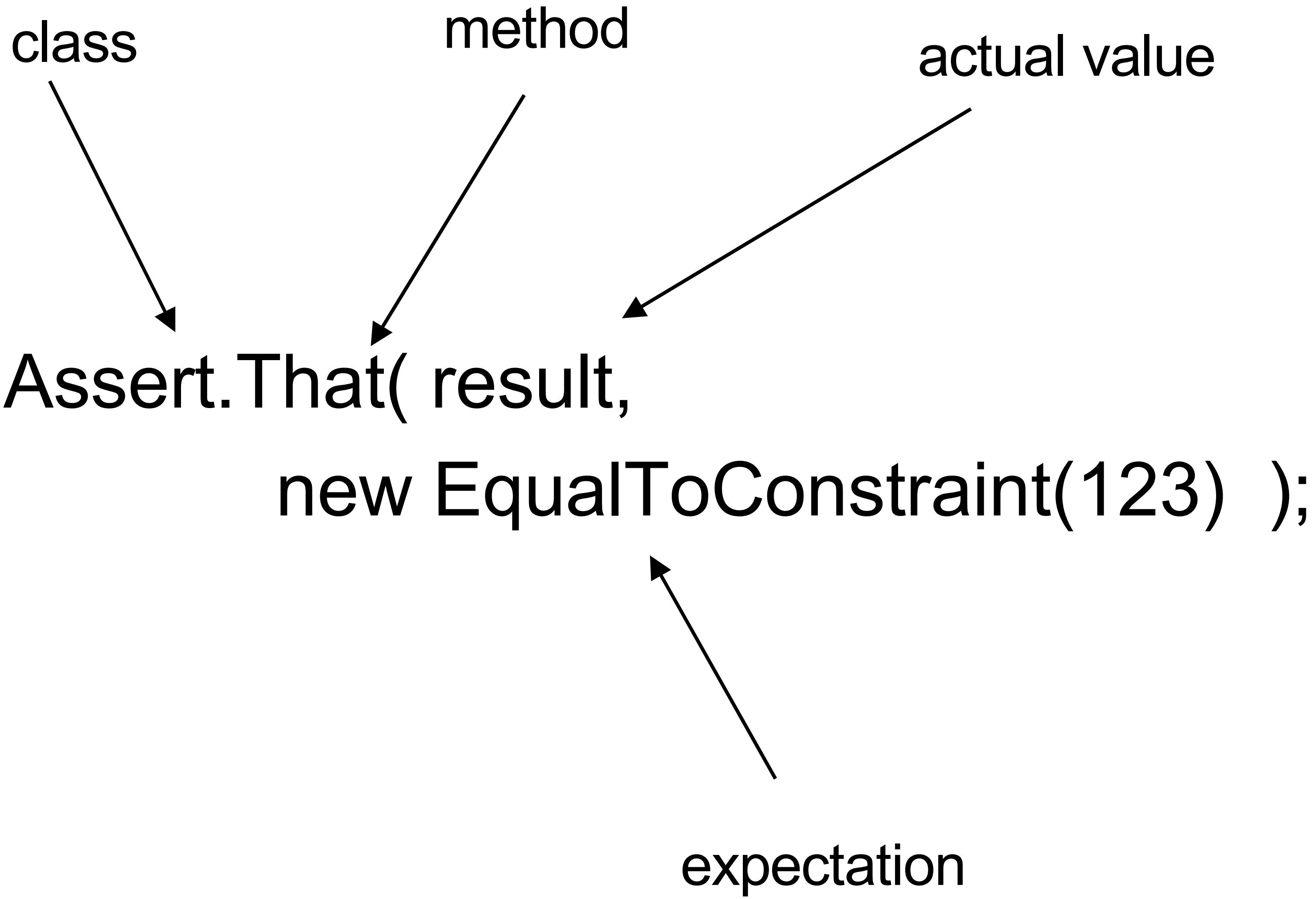
class

method

```
Assert.That( result, Is.EqualTo(123) );
```

actual value

expectation



class

method

actual value

```
Assert.That( result,
         new EqualToConstraint(123)  );
```

expectation

```
MyObject obj;
…
string expected = “hello”;

Assert.That( obj.Message,
  Is.EqualTo(“hello”) );
```

```
MyObject obj;
…
string expected = “hello”;

Assert.That( obj.Message,
  Is.EqualTo(“hello”).IgnoringCase );
```

```
string[] array = new string[]
  { “Hello”, “XP”, “User” “Group” };

Assert.That( array,
     Contains.Item( “XP”) );
```

```
string[] array = new string[]
  { “Hello”, “XP”, “User” “Group” };

Assert.That( array,
  Contains.Item( “xp”).IgnoringCase );
```

```
object[] array = new object[]
  { “Hello”, “XP”, “User” “Group” };

Assert.That( array,
     Is.All.Not.Null &
     Is.All.Type( typeof(string) );
```

```
int[] a = new int[]
  { 1, 2, 3, 4, 5, 6 };
int[,] b = new int[,]
  { ( 1, 2 }, {3, 4 ), { 5, 6 } };
Assert.That( b, Is.Not.EqualTo( a ) );
Assert.That( b,
     Is.EqualTo( a ).AsCollection );
Assert.That( b,
     Is.All.GreaterThan( 0 ) &
     Is.All.LessThan( 10 ) );
```

```
Expected is System.Int32[], actual is System Int32[,]
Arrays differ at expected index [3], actual index [1,1]
Expected string is length 5, actual is length 6
Strings differ at index 5
Expected: “Hello”
But was:  “Hello!”
----------------^
```

NUnit 3.0 Features?

- Separation of NUnit backend from Runners?
- Shims for client code and addins?
- Leverage .Net 2.0 and 3.0 features?
- Run unmanaged tests?
- Run Java tests?
- Scripting?
- GUI Testing?
- Web Testing?