Improving Unit Tests with Fluent Assertions

Jason Roberts

.NET DEVELOPER

@robertsjason   dontcodetired.com

Suggested prerequisite courses:

Introduction to .NET Testing with NUnit 3

http://bit.ly/psnunit3

Mocking with Moq and NUnit

http://bit.ly/psmoqnunit

A Brief Introduction to Fluent Assertions

Clearer explanations about why a test failed

Improved readability of test source code

An Example of Improved Test Code Readability

```
var loanTerm = new LoanTerm(1);


// MSTest
Assert.AreEqual(loanTerm.ToMonths(), 12);
Assert.AreEqual(12, loanTerm.ToMonths());


// xUnit.net
Assert.Equal(loanTerm.ToMonths(), 12);
Assert.Equal(12, loanTerm.ToMonths());
```

An Example of Improved Test Code Readability

```
// NUnit
Assert.AreEqual(loanTerm.ToMonths(), 12);
Assert.That(loanTerm.ToMonths(), Is.EqualTo(12));

// Fluent Assertions
loanTerm.ToMonths().Should().Be(12);
```

Clearer Explanation About Why a Test Failed

```
// NUnit

Assert.That(loanTerm.ToMonths(), Is.EqualTo(12));
```

```
Expected: 12
But was:  11
```

```
loanTerm.ToMonths().Should().Be(12);
```

```
Expected loanTerm.ToMonths() to be 12, but found 11.
```

Clearer Explanation About Why a Test Failed

```
// NUnit

Assert.That(loanTerm.ToMonths(), Is.EqualTo(12));

Expected: 12
But was:  11
```

```
loanTerm.ToMonths().Should().Be(12);

Expected loanTerm.ToMonths() to be 12, but found 11.
```

Better, more descriptive failure messages may prevent the need for debugging through the test.

> “Nothing is more annoying than a unit test that fails without clearly explaining why. More than often, you need to set a breakpoint and start up the debugger to be able to figure out what went wrong.”

Fluent Assertions documentation

https://fluentassertions.com

Summary

Main benefits:

- Better failure messages
- More readable test code

```
sut.CurrencyCode
  .Should()
  .Be("USD");

sliceOfPie
  .Should()
  .BeApproximately(0.33, 0.004);

loanTerm.ToMonths()
  .Should()
  .Be(12, "there are 12 months in a year");

mockIdentityVerifier.Object.LastCheckTime
  .Should()
  .Be(1.January(2000));
```

Summary

```
comparisons
  .Should()
  .Contain(expectedProduct);

act
  .Should()
  .Throw<ArgumentOutOfRangeException>()

using (new AssertionScope())

typeof(IdentityVerifierServiceGateway)
  .Should()
  .Implement<IIdentityVerifier>();
```

Further Learning

Fluent Assertions:

https://fluentassertions.com

https://github.com/fluentassertions/fluentassertions

Extensions:

https://github.com/fluentassertions/fluentassertions.json

https://github.com/fluentassertions/fluentassertions.mvc

https://github.com/fluentassertions/fluentassertions.aspnetcore.mvc