# Unit Testing Basics

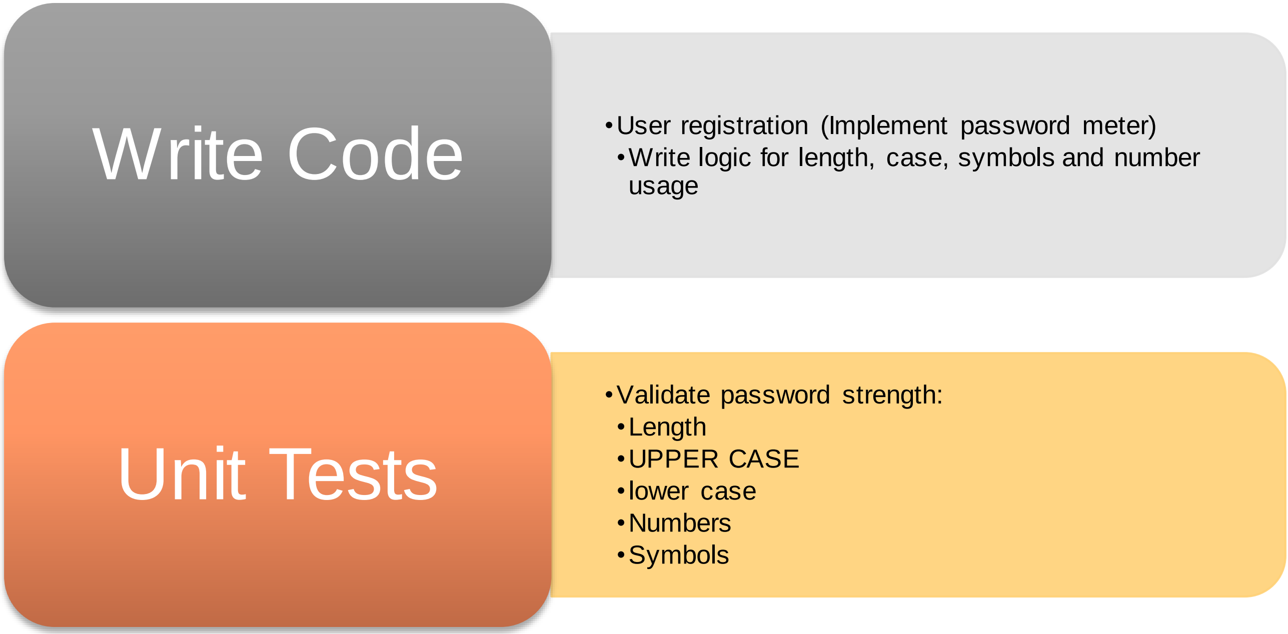
## Visual Studio 2010

http://www.pluralsight.com/

# Outline

- Visual Studio unit testing framework
- Creating your first unit test
- Managing your unit tests
- Debugging unit tests
- “Creating unit tests”- code generation option
- Running unit tests from command line (MSTest.exe)

# Why do you need a Unit Testing Framework?



Write Code

- User registration (Implement password meter)
  - Write logic for length, case, symbols and number usage

Unit Tests

- Validate password strength:
  - Length
  - UPPER CASE
  - lower case
  - Numbers
  - Symbols

# Visual Studio unit test framework

- **Microsoft.VisualStudio.QualityTools.UnitTestFramework.dll**
  - Namespace: Microsoft.VisualStudio.TestTools.UnitTesting

- **Normal code, identified as unit tests by use of attributes**

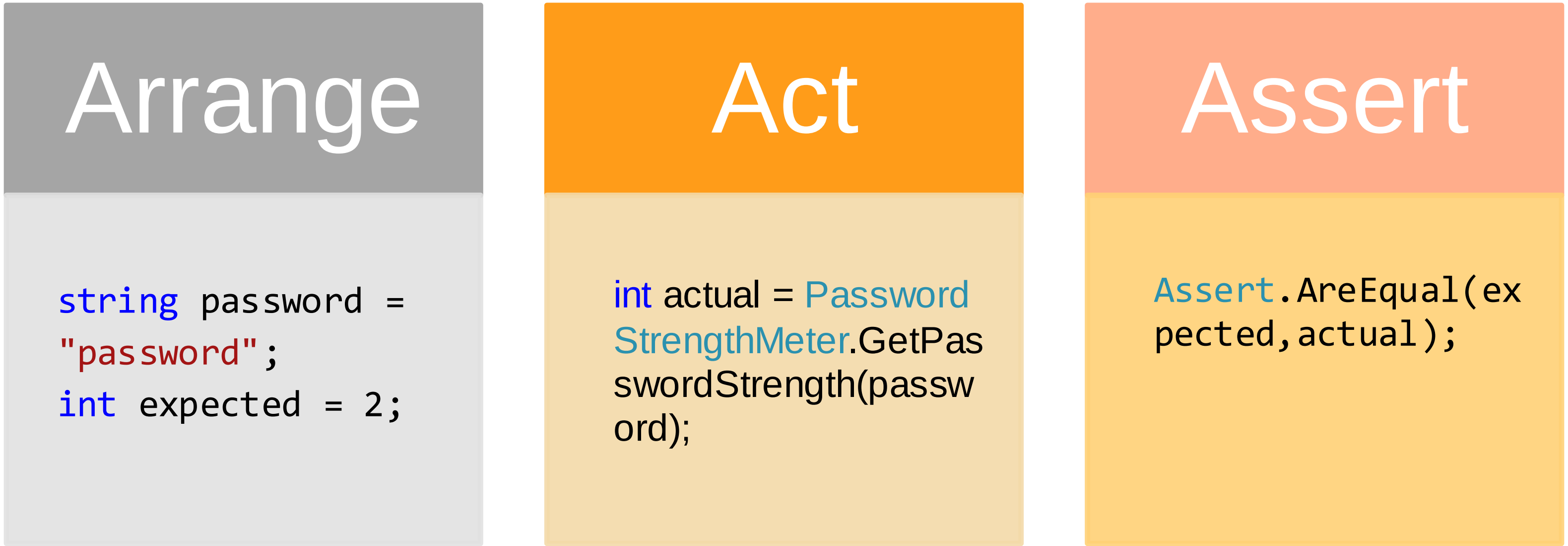
- **Visual Studio uses .NET reflection**

```
 [TestClass]
 public class PasswordStrengthTest
 {
        [TestMethod]
        public void
           PasswordStrengthTest_ForLength()
        {

        }
}
```

# Creating your first unit test

- **Remember the “Three-A’s” pattern**



Arrange

```
string password = "password";
int expected = 2;
```

Act

```
int actual = PasswordStrengthMeter.GetPasswordStrength(password);
```

Assert

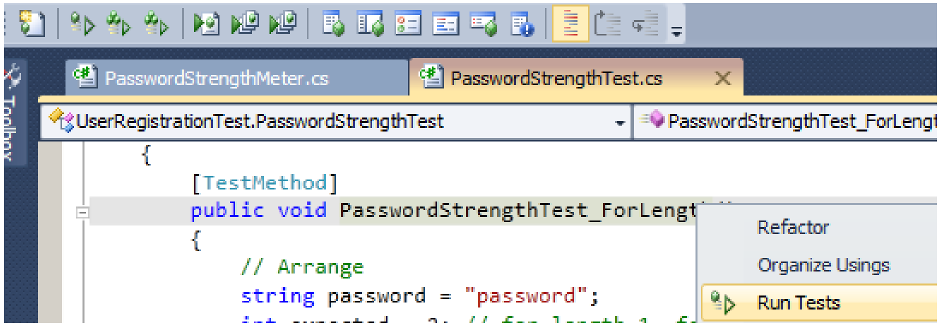
```
Assert.AreEqual(expected,actual);
```

# Managing your unit tests in Visual Studio

- **Test View**
  - All unit tests in flat list
- **Test List Editor**
  - Group and organize related tests
- **Test results**
  - Displays the status of test execution

# Test Run Configuration

- **Test Run**
  - Directly from code
  - “Test tools” tool bar



```
{
    [TestMethod]
    public void PasswordStrengthTest_ForLengt
    {
        // Arrange
        string password = "password";
```

- **Test Run Configuration**
  - Its all about the .testsettings file
- **Two .testsettings created with a test project**
  - You can choose one of them as active for a given test run

# Debugging unit tests

- **Unit tests can be debugged just like any other code**

- **Break points can be set anywhere in your code**
  - To pause at breakpoints you have to explicitly run unit tests in debug mode

# Generating unit tests

- Its Simple
- Avoid if you can

# MSTest.exe

- Used for execution of unit tests from command line
- Located under “\Common7\IDE”
- Command line options:

| Command line options | Description |
|---|---|
| /testcontainer:[file name] | All tests in the test assembly (container) |
| /test:[test name] | A single test |
| /testlist:[test list path] | A test list |
| /testsettings | To specify a test settings file |

- Many more options → [http://msdn.microsoft.com/en-us/library/ms182489.aspx#test](http://msdn.microsoft.com/en-us/library/ms182489.aspx#test)

# Summary

- **Writing your unit tests**
  - Remember the Three-As

- **Managing your unit tests**
  - Use “Test List Editor” and
  - “Test View”

- **Generating unit tests**

- **Executing unit tests from command line**