Testing the Application Components

Gill Cleeren
CTO Xpirit Belgium

@gillcleeren – xpirit.com/gill

Module overview

Understanding unit tests

Writing unit tests

Understanding Unit Tests

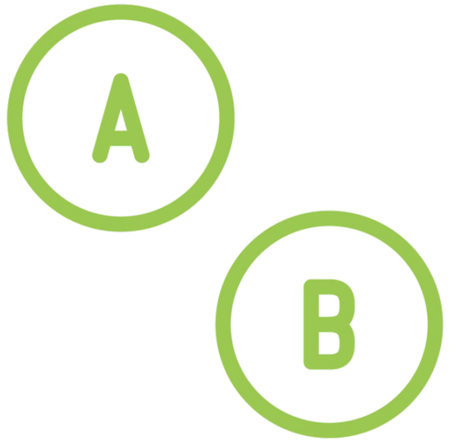
A unit test is an automated piece of code that invokes a unit of work in the system and then checks a single assumption about the behavior of that unit of work.

From The Art of Unit Testing, Roy Osherove

Unit Tests

Block of code

Public methods



Isolated and independent

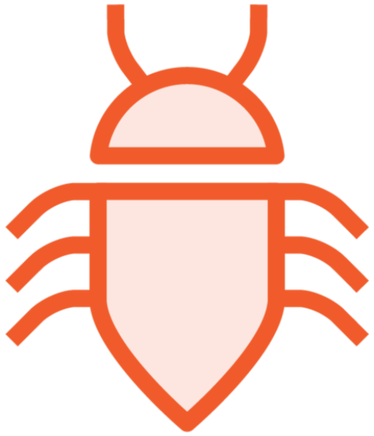
Using Unit Tests

Consistent

Automated

Fast

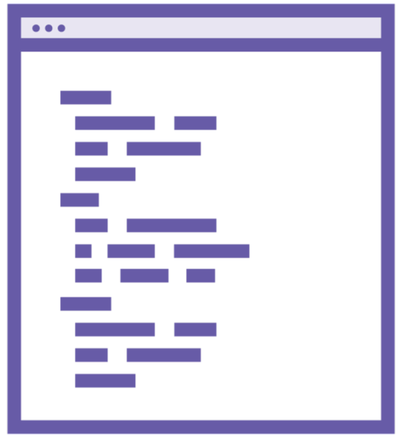
Why Do We Need Unit Tests?



Find bugs

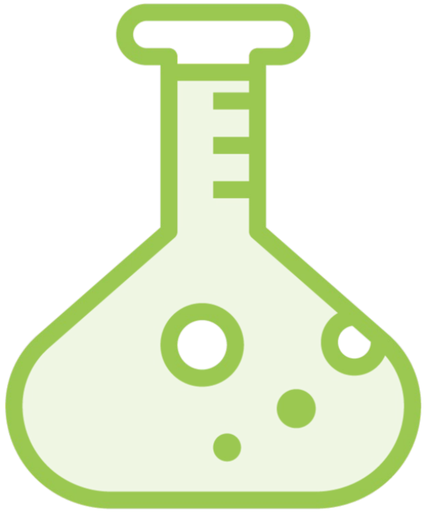
Change without fear of breaking something

Improve quality



Documentation of code

Writing Unit Tests



Parts of a unit test

- Arrange
- Act
- Assert

Used frameworks

- xUnit
- Moq

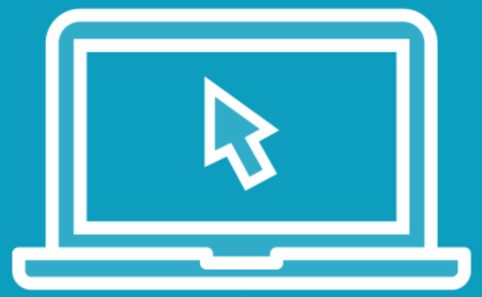
A Simple Unit Test

```
public void CanUpdatePiePrice()
{
    // Arrange
    var pie =
        new Pie { Name = "Sample pie", Price = 12.95M };

    // Act
    pie.Price = 20M;

    //Assert
    Assert.Equal(20M, pie.Price);
}
```

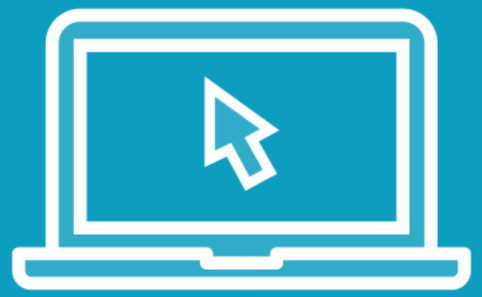
Demo



Adding the correct project and packages

Writing a unit test for a controller

Demo



Creating a unit test for a tag helper

Summary

Unit tests create a safety net around our code

Unit tests can be added on multiple levels in our code

Up next:
Adding interactivity to our site