Telerik Academy

Mocking with Moq and JustMock

Mocking tools for easier unit testing

Introducing...

JustMock

Telerik's newest developer productivity tool for creating better unit tests

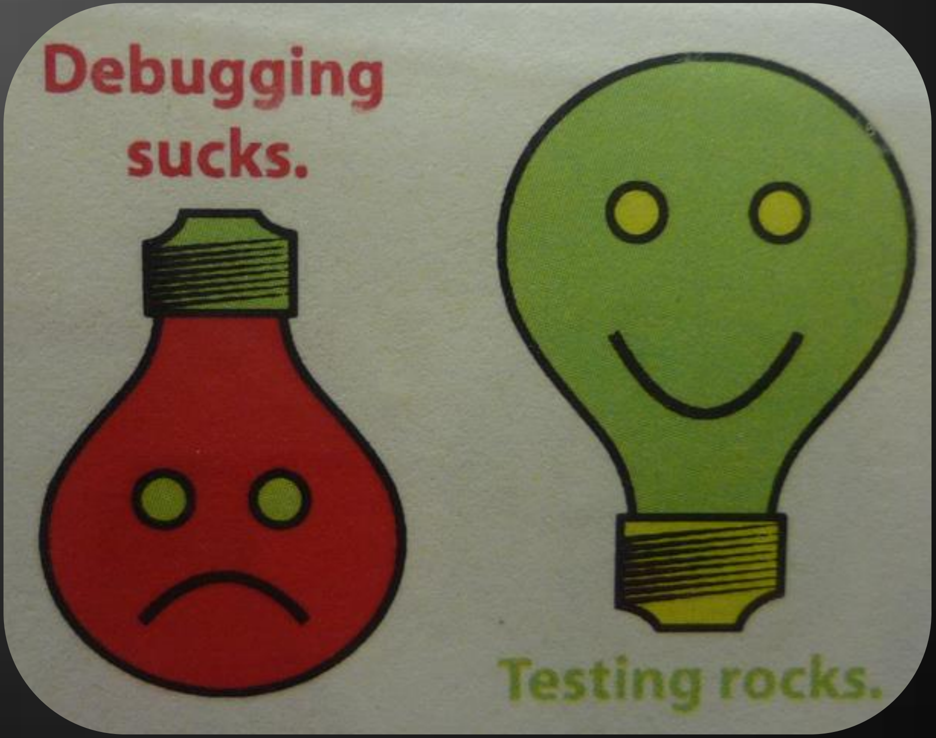
Telerik Software Academy

Learning & Development

http://academy.telerik.com

Table of Contents

Testable Code



How to Write Testable Code

- Inversion of Control Pattern
  - There is a decoupling of the execution of a certain task from implementation
  - Every module can focus on what it is designed for
  - Modules make no assumptions about what other systems do but rely on their contracts
  - Replacing modules has no side effect on other modules
  - More info at http://en.wikipedia.org/wiki/Inversion_of_control

Telerik Academy

How to Write Testable Code

- Public API should work with interfaces, not implementation classes (IEnumerable vs. List)
- Bad code:

```
public Card[] Cards { get; private set; }
```

- Good code:

```
public IList<ICard> Cards { get; private set; }
```

How to Write Testable Code

- Dependency Injection
  - Ninject – http://www.ninject.org/
- Consists of:
  - A dependent consumer
  - A declaration of a component's dependencies, defined as interface contracts
  - An injector (sometimes referred to as a provider or container) that creates instances of classes that implement a given dependency interface on request

How to Write Testable Code

- Bad:

```
public interface IViewBase {}
public interface IPresenterBase {}
public class MemoryLayoutView: IViewBase {}

public class MemoryLayoutPresenter: IPresenterBase
{
    private MemoryLayoutView view = new MemoryLayoutView();
    public MemoryLayoutPresenter() { }
}
```

Telerik Academy

How to Write Testable Code

- Good:

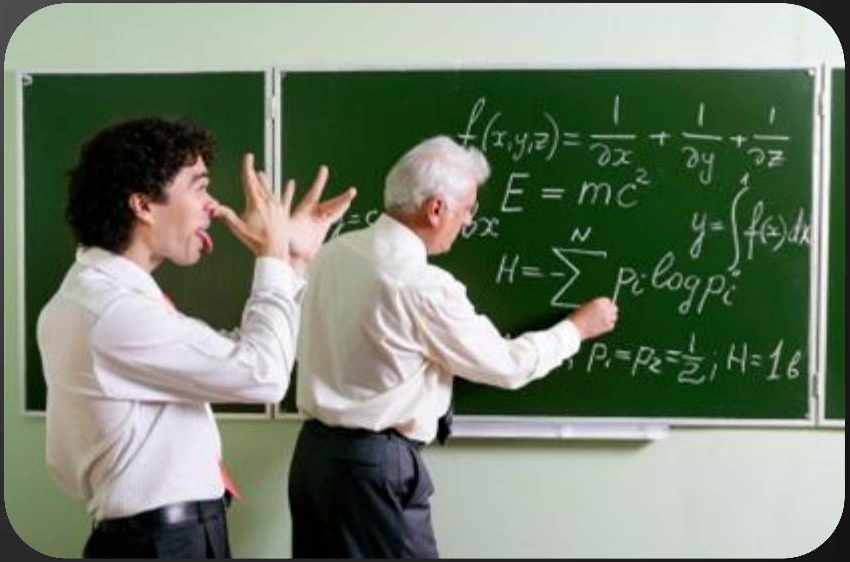
```
public interface IViewBase {}
public interface IPresenterBase {}
public class MemoryLayoutView: IViewBase {}

public class MemoryLayoutPresenter : IPresenterBase
{
    private IViewBase view;

    public MemoryLayoutPresenter(IViewBase myView)
    {
        this.view = myView;
    }
}

public class Program {
    static void Main() {
InjectionContainer.Create<typeof(MemoryLayoutPresenter)>();
    }
}
```

Mocking

Mocking

- Makes Unit Testing more effective
  - Avoid writing boring boilerplate code
- Isolate dependencies among units
- Asserts expectations for code quality
  - Ex: Checks that a method is called only once

Moq


$f(x,y,z) = \frac{1}{\partial x} + \frac{1}{\partial y} + \frac{1}{\partial z}$
$E = mc^2$
$y = \int f(x)dx$
$H = -\sum^{N} p_i \log p_i$
$p_1 = p_2 = \frac{1}{2}; H = 1b$

Moq

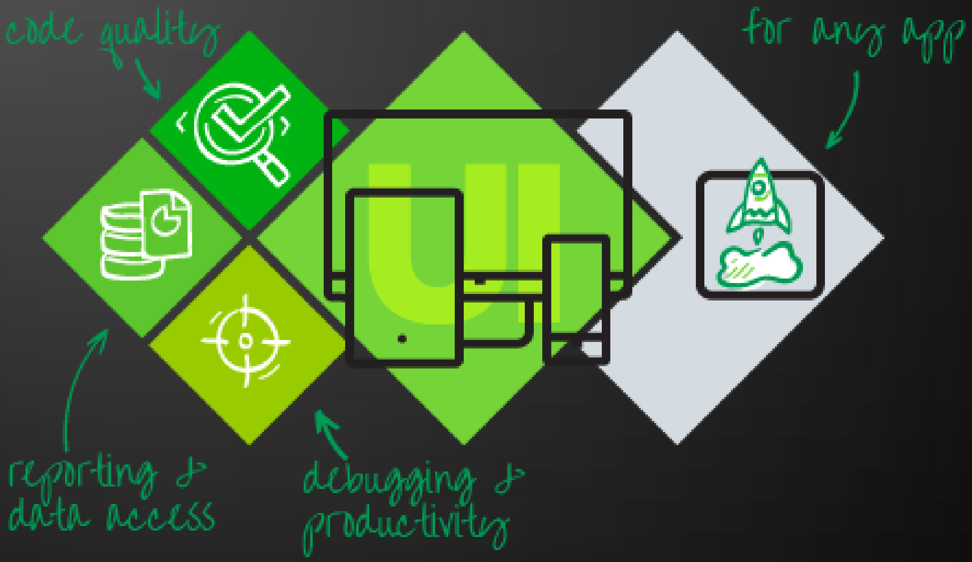
- Install from the NuGet package manager
- Refer the library
- Use its API
- https://github.com/Moq/moq4

```
var mock = new Mock<ICarsRepository>();
mock.Setup(r => r.Add(It.IsAny<Car>())).Verifiable();
mock.Setup(r => r.All()).Returns(this.FakeCarCollection);
```

Moq

- The most often used APIs:
  - .Setup()
  - .Verifiable()
  - .Callback()
  - .Returns()
  - .Throws()
  - It.Is<type>(x => condition)

Telerik JustMock



Telerik JustMock

- Install from the Telerik account
  - http://www.telerik.com/products/mocking.aspx
- Use the Visual Studio NuGet package manager

```
CarsData = Mock.Create<ICarsRepository>();
Mock.Arrange(() => CarsData.Add(Arg.IsAny<Car>())).DoNothing();
Mock.Arrange(() => CarsData.All()).Returns(FakeCarCollection);
Mock.Arrange(() => CarsData.Search(Arg.AnyString))
             .Returns(FakeCarCollection.Where(
                                  c => c.Make == "BMW").ToList());
Mock.Arrange(() => CarsData.GetById(Arg.AnyInt))
             .Returns(FakeCarCollection.First());
```

Telerik JustMock

- Two versions:
  - Free version – excellent when the code is written with testability in mind
  - Paid version – mocks everything (ex: mscorlib, EF, SQL), mocks legacy code base which is not written to be testable, statics, privates
- More information here:
  - http://www.telerik.com/help/justmock/getting-started-commercial-vs-free-version.html

Telerik JustMock

- The most often used APIs:
  - .CallOriginal()
  - .Returns()
  - .DoInstead()
  - .DoNothing()
  - .Throw()
  - Arg.Matches<type>(x => condition)

Telerik Academy

Mocking

Live Demo

Telerik Academy

Mocking with Moq and JustMock

Questions?

http://academy.telerik.com

Free Trainings @ Telerik Academy

- C# Programming @ Telerik Academy
  - csharpfundamentals.telerik.com
- Telerik Software Academy
  - academy.telerik.com
- Telerik Academy @ Facebook
  - facebook.com/TelerikAcademy
- Telerik Software Academy Forums
  - forums.academy.telerik.com

Homework

- Finish the unit testing for CarsController
  - Write mocks for the rest of ICarsRepository interface (sorting)
  - Write missing unit test so that the controller functionality is fully tested