Using Additional Moq Mocking Techniques

Jason Roberts
.NET DEVELOPER

@robertsjason dontcodetired.com

Overview

Throw exceptions from mock objects

Raise events from mock objects

- Manually
- Automatically

Return different results for sequential calls

Mock virtual members of concrete types

Mock virtual protected members

Improve mock setup readability with LINQ

Refactor the test class

Generic type argument matching

Mocking async method return values

Matching Generic Type Arguments

```
public interface IDemoInterface
{
    bool IsOdd<T>(T number);
}


var mock = new Mock<IDemoInterface>();
```

Specific Value (Inferred Generic Type)

```
mock.Setup(x => x.IsOdd(1)).Returns(true);


mock.Object.IsOdd(1);   // Returns true
mock.Object.IsOdd(2);   // Returns false
mock.Object.IsOdd(1.0); // Returns false
```

Any Value for a Specified Generic Type

```
mock.Setup(x => x.IsOdd(It.IsAny<int>())).Returns(true);


mock.Object.IsOdd(1));  // Returns true
mock.Object.IsOdd(2));  // Returns true
mock.Object.IsOdd(1.0); // Returns false
```

Any Value for Any Generic Type

```
mock.Setup(x => x.IsOdd(It.IsAny<It.IsAnyType>()))
    .Returns(true);

mock.Object.IsOdd(1);       // Returns true

mock.Object.IsOdd(2);       // Returns true

mock.Object.IsOdd(1.0);     // Returns true

mock.Object.IsOdd("hello"); // Returns true
```

Any Generic Type or Subtype

```
mock.Setup(
m => m.IsOdd(It.IsAny<It.IsSubtype<ApplicationException>>()))
    .Returns(true);

mock.Object.IsOdd(new ApplicationException()); // True


mock.Object.IsOdd(new Exception()); // False

mock.Object.IsOdd(new WaitHandleCannotBeOpenedException());
// True
```

Mocking Async Method Return Values

```
public interface IDemoInterfaceAsync
{
    Task StartAsync();
    Task<int> StopAsync();
}

var mock = new Mock<IDemoInterfaceAsync>();

mock.Setup(x => x.StartAsync()).Returns(Task.CompletedTask);


mock.Setup(x => x.StopAsync()).Returns(Task.FromResult(42));

mock.Setup(x => x.StopAsync()).ReturnsAsync(42);
```

Summary

.Throws<Exception>();

mockValidator.Raise(...)

.Raises(...)

.SetupSequence(...)

Mocked virtual members of concrete types

Mocked virtual protected members

- using Moq.Protected;
- .Protected()

Mock.Of<IFrequentFlyerNumberValidator>

Refactored the test class

It.IsAny<It.IsAnyType>()

ReturnsAsync(42);

Resources and Further Learning

https://github.com/moq

“Testing .NET Code with xUnit.net: Getting Started” Pluralsight course.