Cheatography

Moq Cheat Sheet

by Sérgio Ferreira (AlienEngineer) via cheatography.com/89520/cs/20375/

Stub - Just a replacement

```
// Create a stub
var mock = new Mock<IMyDependency>();
// Get the Stub
var myDep = mock.Object;
Assert.That(myDep .GetValue(), Is.EqualTo(0));
```

By default: Methods with returning types return default values. (e.g. `int GetValue()` returns 0;

Spy - What happened?

```
// Create a spy
var mock = new Mock<IMyDependency>();
// Spy on executions
mock.Verify(myDep => myDep.GetValue());
```

Optional:
`Verify can be called with optional argument that asserts how many times the method as called: AtLeastOnce, AtLeast, AtMost, AtMostOnce, BetweenExclusive, BetweenInclusive, Exactly, Once, Never`

Tip: Always strive to use concrete expectations. Avoid AtMost, AtLeast and Between.

Spy via Callbacks

```
// Create a spy
var mock = new Mock<IMyDependency>();
string usedArg;
mock
    // Setup to capture any call to done to DoSomething method.
    .Setup(myDep => myDep.DoSomething(It.IsAny<String>())
    // Callback function is called when DoSomething was called.
    .Callback<String>(arg => usedArg = arg);

// Executing with "test" argument
mock.Object.DoSomething("test");

Assert.That(usedArg, Is.EqualTo("test"));
```

Mock - Do what I say!

```
// Create a mock
var mock = new Mock<IMyDependency>();

mock
    // What behavior is going to be mocked
    .Setup(myDep => myDep.GetValue())
    // What's going to be returned by GetValue()
    .Returns(1);

// Using the object
var myDep = mock.Object;

Assert.That(myDep .GetValue(), Is.EqualTo(1));
```

Mock - Setup - Matching Arguments

Expected Argument	Setup
1	`Setup(e => e.Get(1))`
Any	`Setup(e => e.Get(It.IsAny<int>())`
> 10	`Setup(e => e.Get(It.Is<int>(i => i > 10))`
Range [0..10]	`Setup(e => e.Get(It.IsInRange<int>(0, 10, Range.Inclusive))`

An instance of a mock can have more than 1 setup for a given method provided we setup the method with different arguments.

Tip: Avoid using non specific values on capture, except when it's possible to validate what was provided.

Mock - Setup - Returns

value	`Setup(...).Returns(value)`
access argument	`Setup(...).Returns(arg => arg + 10)`
Throw exception	`Setup(...).Throws<Exception>()`
Throw exception with message	`Setup(...).Throws(new Exception("My Message"))`
Lazy access	`Setup(...).Returns(() => value)`

By **Sérgio Ferreira** (AlienEngineer)

cheatography.com/alienengineer/

Published 26th August, 2019.
Last updated 3rd September, 2019.