nmock2 Cheat Sheet

Setting Up

```
Mockery mockery = new Mockery();
```

Creating mocks

```
interfaceMock = mockery.NewMock<InterfaceToBeMocked>();
classMock = mockery.NewMock<ClassToBeMocked>([mock style], [constructor arguments]);
namedMock = mockery.NewNamedMock<TypeToBeMocked>("name". [mock style], [ctr args]);
```

Mock Styles

`MockStyle.Default`	Calls to members that do not have expectations set will result in ExpectationExceptions.
`MockStyle.Transparent (classes only)`	Calls to members that do not have expectations set will pass through to the underlying implementation on the class being mocked.

Using Default Expectation (same as AtLeastOnce)

```
Expect.On(aMock).Method( … )
       .With( … )
       .Will(Return.Value( … ));
```

Setting Basic Expectations

```
Expect.Once.On(aMock).Method( … )
       .With( … )
       .Will(Return.Value( … ));
```

Setting Expectation That Throws Exception

```
Expect.Once.On(aMock).Will(Throw.Exception( … ));
```

Setting Expectation On a Getter

```
Expect.Once.On(aMock)
       .GetProperty( … )
       .Will(Return.Value( … ));
```

Setting Expectation On a Setter

```
Expect.Once.On(aMock)
       .SetProperty( … )
       .To( … );
```

Setting Expectation On Out Parameters

```
Expect.Once.On(aMock).Method( … )
       .With(Is.Anything, Is.Out)  // Is.Out -> paramname
       .Will(Return.Value( … ), Return.OutValue("paramname", value));
```

Setting Expectation On Generic Method Type Parameters (e.g. `aMock.GenericMethod<int, string>();`)

```
Expect.Once.On(aMock).Method("GenericMethod", typeof(int), typeof(string));
```

Stubs

`Expect` can be replaced with `Stub` which essentially means 'zero or more'. Behavior of the stub will be invoked if called, but the stub will not cause the test to fail.

```
Stub.On(aMock)
     .Method( … )
     .With( … )
     .Will(Return.Value( … );
```

Constraining Order

Mocks by default can be in any order. To constrain the order of a set of expectations, wrap the expectations with a using block.

```
using (mockery.Ordered) {
      Expect.Once.On( …
      Expect.Once.On( …
}
```

Event Addition / Removal

```
Expect.Once.On(aMock).EventAdd("eventname");

Expect.Once.On(aMock).EventRemove("eventname");
```

Fire Events

```
Fire.Event("eventname")
      .On(aMock)
      .With(sender, eventargs);
```

Possible Method Call Expectations

`Expect.Once`	`Expect.Never`	`Expect.AtLeastOnce`
`Expect.AtLeast(<# times>)`	`Expect.AtMost(<# times>)`	`Expect.Exactly(<# times>)`
`Expect.Between(<# times>, <# times>)`		

Thread Synchronization – signal an EventWaitHandle

```
Expect.Once.On(aMock).Method(…).Will(Signal.EventWaitHandle(signal));
```

Verification

```
mockery.VerifyAllExpectationsHaveBeenMet();
```

Verification Alternative

```
using (Mockery mockery = new Mockery())
{ … } // Dispose calls Verify…()
```

Add Comments That Are Shown In Error Message

```
Expect.Once.On(aMock).Method(…).Comment("Comment explaining why this is expected");
```

Go to http://nmock2.sf.net.   Powered by bbv behind things.   www.bbv.ch