Mockoholics Anonymous
The Five-step Program

github.com/jeremyarr

My name is Jeremy

and I’m a mockoholic

STEP 1

Admit You Have a Problem

```
def orchestrate():
    do_one()
    do_two()
    do_three()
```

```
class Mockoholism(unittest.TestCase):
    def setUp(self):
        self.do_one_patcher = patch('pycon.do_one')
        self.mock_do_one = self.do_one_patcher.start()

        self.do_two_patcher = patch('pycon.do_two')
        self.mock_do_two = self.do_two_patcher.start()

        self.do_three_patcher = patch('pycon.do_three')
        self.mock_do_three = self.do_three_patcher.start()

    def tearDown(self):
        self.do_one_patcher.stop()
        self.do_two_patcher.stop()
        self.do_three_patcher.stop()
```

One mock is never enough

```
def test_orchestrate(self):
    manager = Mock()

    manager.attach_mock(self.mock_do_one,'mock_do_one')
    manager.attach_mock(self.mock_do_two,'mock_do_two')
    manager.attach_mock(self.mock_do_three,'mock_do_three')

    orchestrate()

    self.assertEqual(manager.mock_calls[0],call.mock_do_one)
    self.assertEqual(manager.mock_calls[1],call.mock_do_two)
    self.assertEqual(manager.mock_calls[2],call.mock_do_three)
```

STEP 2

Make a list of your shortcomings

Fragile Tests

Waste of Time

Doesn’t fix bad architecture

STEP 3

Believe in a higher power to restore your sanity_


Bob
Martin


Kent Beck

STEP 4

Go to rehab


THE CITY OF DETROIT
SPERAMUS MELIORA
RESURGET CINERIBUS
MICHIGAN

Favour state verification over behaviour verification

Limit mocking to architecturally significant boundaries

Test at a higher level of abstraction

Design to well defined interfaces

STEP 5

Preach to other mockoholics so they may also see the light

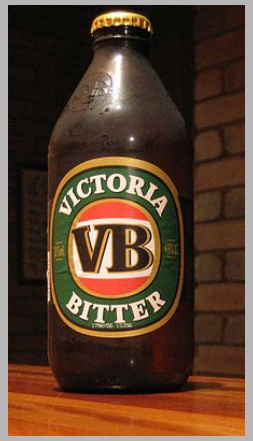
Less tests

More loosely coupled

Coverage neutral

Tests are still fast

Refactoring is fun!



Give it a try!