

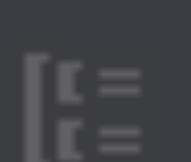
Unit & Integration Tests

Django Development II

What is Automated Tests?



Test Explorer



Search



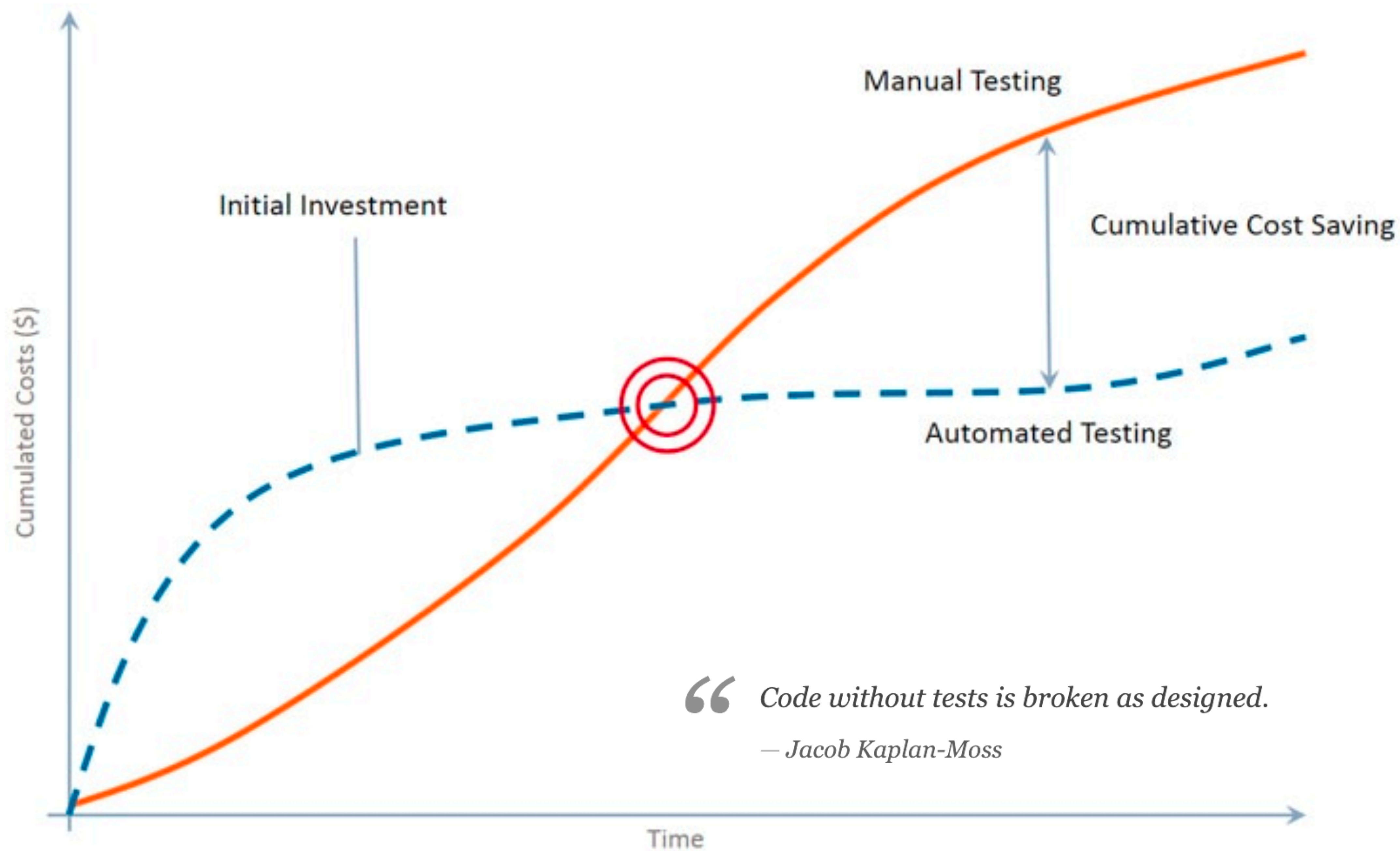
Run All

Run...

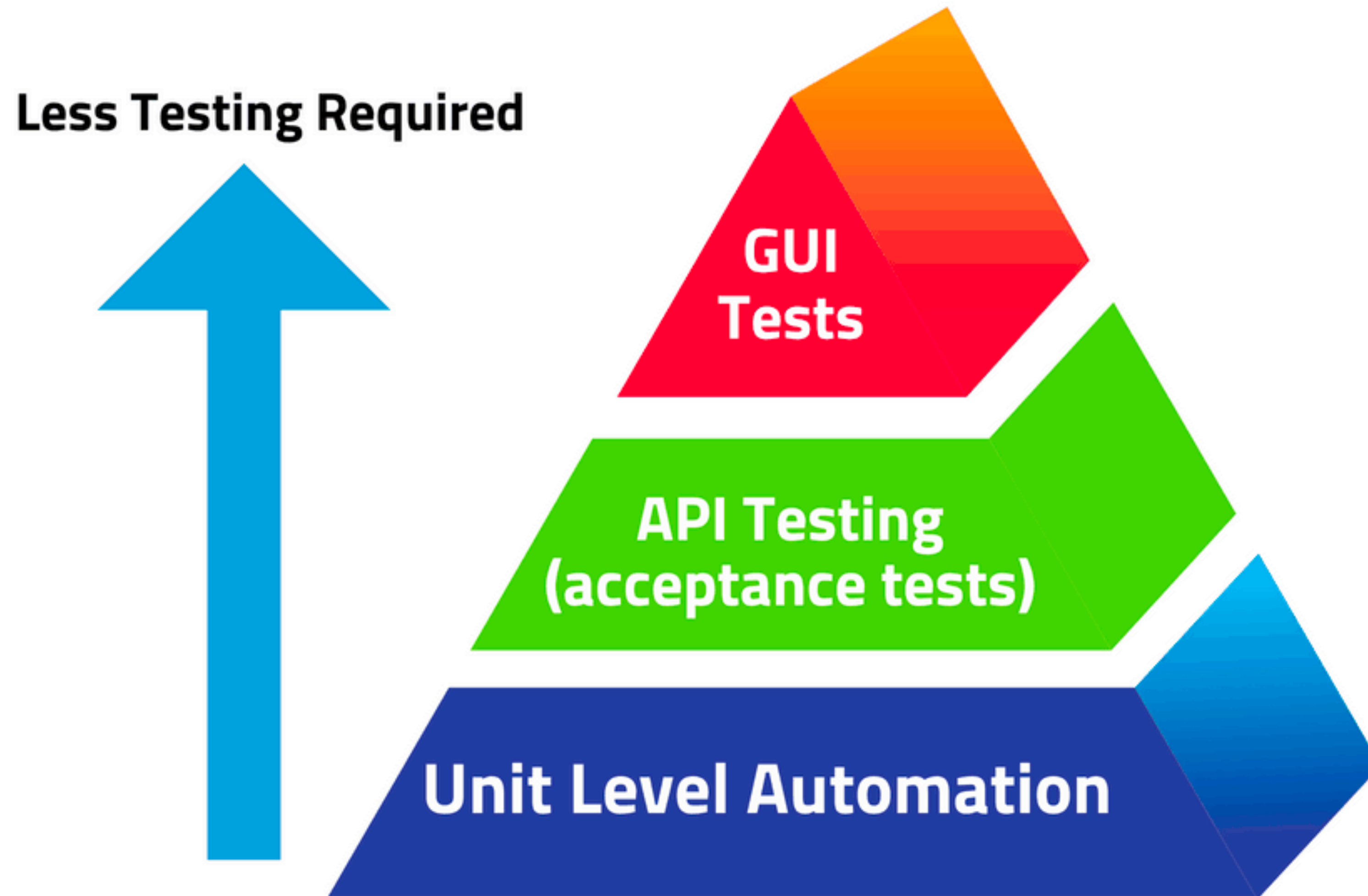
Playlist : All Tests

UnitTestBestPracticeDemoProject (4 tests)

Utils.Tests (4)	283 ms
Utils.Tests.Word (4)	283 ms
WordUtilsTests (4)	283 ms
Reverse_ShouldBeWordInReverse_IfWordsValid	243 ms
Reverse_ShouldInvokeOnce_LogInformationMethod	37 ms
Reverse_ShouldThrowArgumentException_IfWordsEmpty	< 1 ms
Reverse_ShouldThrowArgumentException_IfWordsNull	2 ms



Test Automation Pyramid



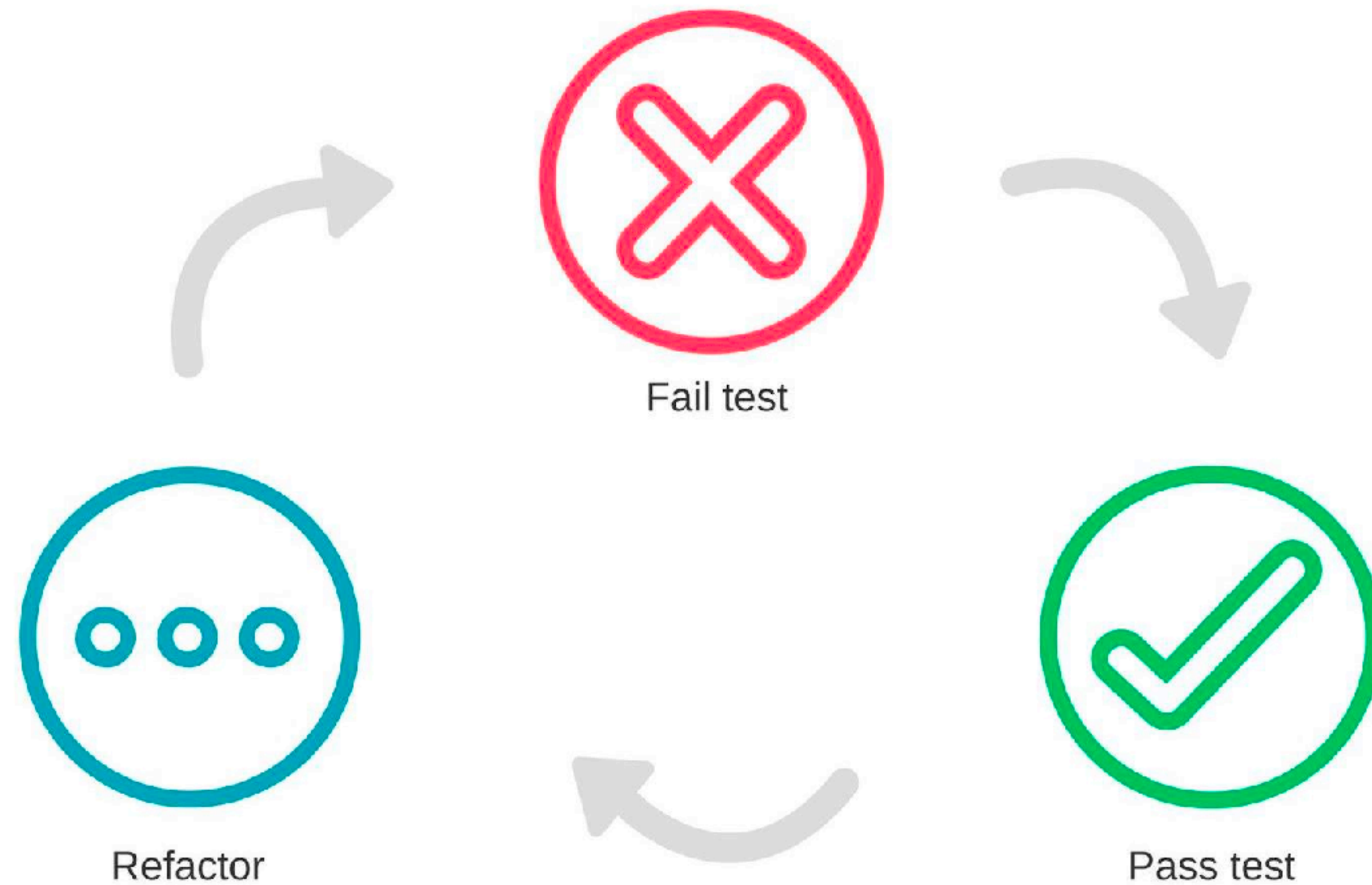
Unit Testing



Good Unit Test Characteristics



Refactoring Code with Tests



Testing with Django

```
from django.test import TestCase
from myapp.models import Animal

class AnimalTestCase(TestCase):
    def setUp(self):
        Animal.objects.create(name="lion", sound="roar")
        Animal.objects.create(name="cat", sound="meow")

    def test_animals_can_speak(self):
        """Animals that can speak are correctly identified"""
        lion = Animal.objects.get(name="lion")
        cat = Animal.objects.get(name="cat")
        self.assertEqual(lion.speak(), 'The lion says "roar"')
        self.assertEqual(cat.speak(), 'The cat says "meow"')
```

pytest testing framework



```
demo py.test
Test session starts (platform: linux2, Python 2.7.6, pytest 2.5.2, pytest-sugar 0.3.3)
plugins: sugar

controller/test_leisure.py ..... 9%
controller/test_nutritious.py ..... 17%
controller/test_youthful.py ..... 21%
model/test_best.py ... 22%
model/test_goodness.py ..... 32%
model/test_lingering.py ..... 36%
model/test_paradise.py ..... 38%
model/test_slick.py ..... 47%
model/test_splendid.py ..... 58%
model/test_stunning.py ..... 70%
view/test_enhance.py ..... 79%
view/test_maxi.py ..... 84%
view/test_sensational.py ..... 96%
view/test_thriving.py ..... 100%

Results (39.80s):
 306 passed
demo
```

pytest tests

```
def test_empty():  
    p = Portfolio()  
    assert p.cost() == 0.0  
  
def test_buy_one_stock():  
    p = Portfolio()  
    p.buy("IBM", 100, 176.48)  
    assert p.cost() == 17648.0  
  
def test_buy_two_stocks():  
    p = Portfolio()  
    p.buy("IBM", 100, 176.48)  
    p.buy("HPQ", 100, 36.15)  
    assert p.cost() == 21263.0
```

test_port2_pytest.py

```
$ pytest -q test_port2_pytest.py
```

```
...
```

[100%]

```
3 passed in 0.01s
```


passing unit tests

