TEST YOUR TESTS WITH PIT FRAMEWORK

dspoljaric@croz.net

@darko_spoljaric

Set up

(given)

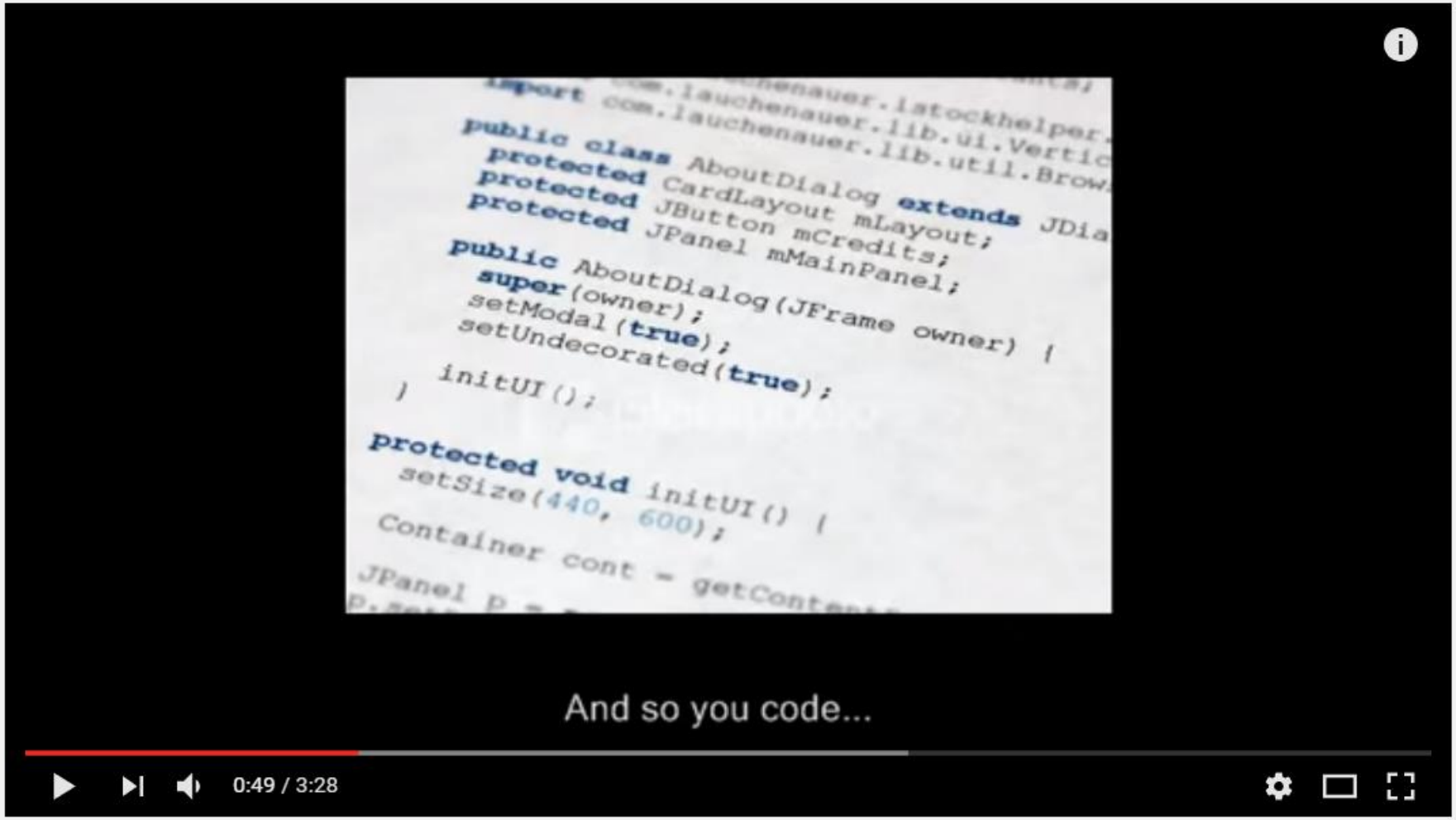
Data, mocks, class under test, environment etc.

Call method under test

(when)

Assert results

(then)



And So You Code - Rap Cover (Stromae Alors on Danse)

M

Mishu

Subscribe 274

1,026,778 views

Add to Share More

8,411 187

```
public class SomeValidatorTest {


    @Test
    public void testValidateWithSuccesfullScenario() {
    }
    @Test
    public void testValidateWithAnotherSuccesfullScenario() {
    }
    @Test
    public void testValidateWithFailingScenario() {
    }
}
```

COVERAGE – DOES IT COVER:

All posible input variations?

What the test returns?

How you've written your flow control code?

(ifs, switches, loops, exceptions, etc.)

When you say your code is covered by tests you're basically saying nothing of value.

```java
@Test
public void increaseCoverage() throws ClassNotFoundException, IllegalAccessException {
    // https://github.com/ronmamo/reflections in use
    Set<Class<?>> classes = getClasses("parser");

    for (Class<?> clazz : classes) {
        Method[] methods = clazz.getDeclaredMethods();
        for (int j = 0; j < methods.length; j++) {
            Method method = methods[j];
            if (!Modifier.isPublic(method.getModifiers())) {
                method.setAccessible(true);
            }

            Object[] args = createArguments(method);

            try {
                Object instance = Class.forName(clazz.getName()).newInstance();
                method.invoke(instance, args);
            } catch (Exception e) {
                // who cares?
            }
        }
    }
}
```

TDD

„Write the minimal amount of code to make the test pass”

- yes, but how do you know you wrote the bare minimum? ☺

- One test per line? Per branch? Per catch block?

WELCOME PIT!

Pit takes the class under test, makes multiple changes on it which result in many new classes. Then it runs existing tests on all of those new classes, to see if any of them fail.

New class = Mutant

Change = Mutation

If a test (suite) fails on changed code (mutant) then the test is well written.

„the mutant is killed”

EXAMPLE #1

```
if (returnValue.startsWith("\n")) {
    returnValue = code.substring(1, code.length());
}
```

NegateConditionalsMutator

```
if (!returnValue.startsWith("\n")) {
    returnValue = code.substring(1, code.length());
}
```

EXAMPLE #2

```
if (returnValue.endsWith("\n")) {
    returnValue = returnValue.substring(0, returnValue.length() - 1);
}
```

MathMutator

```
if (returnValue.endsWith("\n")) {
    returnValue = returnValue.substring(0, returnValue.length() + 1);
}
```

EXAMPLE #3

```
return returnValue;
```

ReturnValsMutator

```
if (returnValue != null) {
    return null;
} else {
    throw new RuntimeException();
}
```

Branch: master ▾ | Create new file | Upload files | Find file | History

pitest / pitest / src / main / java / org / pitest / mutationtest / engine / gregor / **mutators** /

henry Remove redundant modifiers — Latest commit 89c576e on Nov 30, 2015

..		
experimental	Remove redundant modifiers	2 years ago
ArgumentPropagationMutator.java	Apply eclipse cleanup to add java 6 compliant override annotations	2 years ago
ArgumentPropagationVisitor.java	Remove redundant modifiers	2 years ago
ConditionalsBoundaryMutator.java	Remove redundant modifiers	2 years ago
ConstructorCallMutator.java	Apply eclipse cleanup to add java 6 compliant override annotations	2 years ago
IncrementsMutator.java	Remove redundant modifiers	2 years ago
InlineConstantMutator.java	Remove redundant modifiers	2 years ago
InvertNegsMutator.java	Remove redundant modifiers	2 years ago
MathMutator.java	Remove redundant modifiers	2 years ago
MethodCallMethodVisitor.java	Remove redundant modifiers	2 years ago
NegateConditionalsMutator.java	Remove redundant modifiers	2 years ago
NonVoidMethodCallMutator.java	Apply eclipse cleanup to add java 6 compliant override annotations	2 years ago
RemoveConditionalMutator.java	Remove redundant modifiers	2 years ago
ReturnValsMutator.java	Remove redundant modifiers	2 years ago
VoidMethodCallMutator.java	Apply eclipse cleanup to add java 6 compliant override annotations	2 years ago

MUTANTS CAN

Be **killed** by a test

Survive tests

Be **uncovered** by tests

Cause infinite loops to appear thus leading to a timeout

(think about for loops, iterators and MathMutator)

Cause errors

(pit tries to minimise number of such mutants)

HOW MANY MUTANTS ARE TIPICALY CREATED?

Somewhere around the number of lines of code

HOW FAST IS IT?

Bytecode manipulation = no compilation!

(hundreds of thousands of mutants can be created in under a second!)

Shorter tests run first

Extremely paralelizable

WHERE AND WHEN TO USE IT?

Personal hygiene?

Build?

(Should survived mutants cause a build to fail?)

TOOLS SUPPORT

- Eclipse plugin
- IDEA plugin
- Maven plugin
- Gradle plugin
- Sonar plugin

ALTERNATIVES EXIST

- Jester
- Simple Jester
- Jumble
- μJava
- javaLanche

REFERENCES

http://pitest.org/

https://github.com/devopsfolks/podam

https://github.com/ronmamo/reflections

https://github.com/darkospoljaric/pitest-example