SPEC BDD IN PHP

@CAKPER

@CAKPER

SOFTWARE ENGINEER @SENSIOLABSUK

SILESIA PHP USER GROUP // SPUG.PL

#SYMFONY-PL // SYMFONYLAB.PL

WHAT IS TEST?

WHY DO WE TEST?

HOW DO WE TEST?

WE MAKE MISTAKES

SO WE HAVE TO TEST

TEST DRIVEN DEVELOPMENT

KENT BECK // 2003

EXTREME PROGRAMMING // 1999

PROBLEM #1

HOW TO TEST SOMETHING THAT DOES NOT EXIST?

PROBLEM #1

TEST == SPECIFICATION DRIVEN DEVELOPMENT

PROBLEM #1

COMMUNICATION IS OUR PROBLEM

NOT TESTING ITSELF

BEHAVIOUR DRIVEN DEVELOPMENT

DAN NORTH // 2006

BDD

BETTER { NAMING CONVENTIONS TOOLS

BDD

= TDD V2.0

'TDD DONE RIGHT'

PROBLEM #2

WRITING SPECIFICATIONS TAKES TIME

TIME IS A COST

YOU CAN LEARN IT

SPEC BDD IN PHP

13/06/13 GLIWICE

DELIBERATE PRACTICE

PAIR PROGRAMMING
CODE KATA
CODE CLUB
CODE RETREAT

PROBLEM #3

WHY ARE YOU SURE THAT YOUR TEST IS CORRECT?

TDD CYCLE

RED

REFACTOR

GREEN

3 LAWS OF TDD

1. YOU ARE NOT ALLOWED TO WRITE ANY PRODUCTION CODE UNLESS IT IS TO MAKE A FAILING UNIT TEST PASS.

2. YOU ARE NOT ALLOWED TO WRITE ANY MORE OF A UNIT TEST THAN IS SUFFICIENT TO FAIL; AND COMPILATION FAILURES ARE FAILURES.

3. YOU ARE NOT ALLOWED TO WRITE ANY MORE PRODUCTION CODE THAN IS SUFFICIENT TO PASS THE ONE FAILING UNIT TEST.

PROBLEM #4

WHEN AND HOW TO REFACTOR?

PROBLEM #4

‘REFACTORING W/O TESTS = REFUCKTORING’

4 RULES OF SIMPLE DESING

1. PASSES ALL THE TESTS.

2. EXPRESS EVERY IDEA WE NEED TO EXPRESS.

3. CONTAINS NO DUPLICATION.

4. MINIMIZED THE NUMBER OF CLASSES, METHODS AND OTHER MOVING PARTS.

PROBLEM #5

HOW TO TEST DEPENDENCIES?

DO NOT TEST...

TEST DOUBLES

- DUMMIES
- STUBS
- MOCKS
- SPIES

MOCKING SHOULD BE EASY

VERY EASY

PROBLEM #6

CODE COVERAGE

PROBLEM #6

USEFUL TO FIND DEAD CODE ;)

PROBLEM #7

STORY BDD

VS

SPEC BDD

STORY BDD

DESCRIPTION OF BUSINESS-TARGETED APPLICATION BEHAVIOR

SPEC BDD

SPECIFICATION FOR LOW-LEVEL IMPLEMENTATION

PROBLEM #8

PHP UNIT

BY SEBASTIAN BERGMANN©

PHP SPEC 2

PHP SPEC 2

FRAMEWORK SPEC BDD

@_MD & @EVERZET

PROPHECY

PHP SPEC 2

EASY TO USE TDD-CYCLE ORIENTED BEHAVIOUR FOCUSED

PHP SPEC 2

TEST CASE

SPECIFICATION

PHP SPEC 2

TEST

EXAMPLE

PHP SPEC 2

ASSERT

EXPECTATION

PHP SPEC 2

```
class MarkdownSpec extends ObjectBehavior
{
    function it_is_initializable()
    {
        $this->shouldHaveType('Markdown');
    }
}
```

MATCHERS

IDENTITY (===):

```
shouldReturn()
shouldBe()
shouldEqual()
shouldBeEqualTo()
```

MATCHERS

COMPARISON (==):

```
shouldBeLike()
```

MATCHERS

THROW:

```
shouldThrow('\Exception')
->duringSomeMethod()
```

MATCHERS

TYPE:

```
shouldBeAnInstanceOf()
shouldReturnAnInstanceOf()
shouldHaveType()
```

MATCHERS

OBJECT STATE:

```
shouldHave**()
```

TEST DOUBLES

```
/**
 * @param Markdown\Stream $stream
 */
function it_adds_a_end_of_list_to_markup($stream)
{
    $stream->getNextLine()->willReturn("");
    $this->format(" * Hi, there", $stream)
        ->shouldReturn("</li></ul>");
}
```

LET & LETGO

```
function let($die)
{
    $die->beADoubleOf('Die');
    $this->beConstructedWith($die);
}
```

```
function it_live_and_let_die($die)
{
    $this->liveAndLet()->shouldReturn($die);
}

function letgo(){}
```

PHP SPEC 2

```
{
    "require-dev": {
        "phpspec/phpspec": "2.0.*@dev"
    },
    "config": {
        "bin-dir": "bin"
    },
    "autoload": {"psr-0": {"": "src"}}
}
```

DEMO

QUESTIONS?

WE ARE HIRING!

SOFTWARE ENGINEER
FRONT-END ENGINEER
SOFTWARE ENGINEER IN TEST
PROJECT MANAGER
BUSINESS ANALYST

WWW.SENSIOLABS.CO.UK
WWW.INVIQA.COM

THANK YOU!

@CAKPER