

# EXERCISE 0

**Checkout and do composer install**

<https://github.com/lmuzinic/phpdd-pragmatic-tdd>

```
git clone git@github.com:lmuzinic/phpdd-pragmatic-tdd.git
cd phpdd-pragmatic-tdd
composer install
vendor/bin/phpunit
```

OK (1 test, 1 assertion)

**There is <10 MB of packages to download, so you can use your mobile data if WIFI does not work.**



# PRAGMATIC TDD

# HELLO



**Luka Muzinic**  
**@lmuzinic**

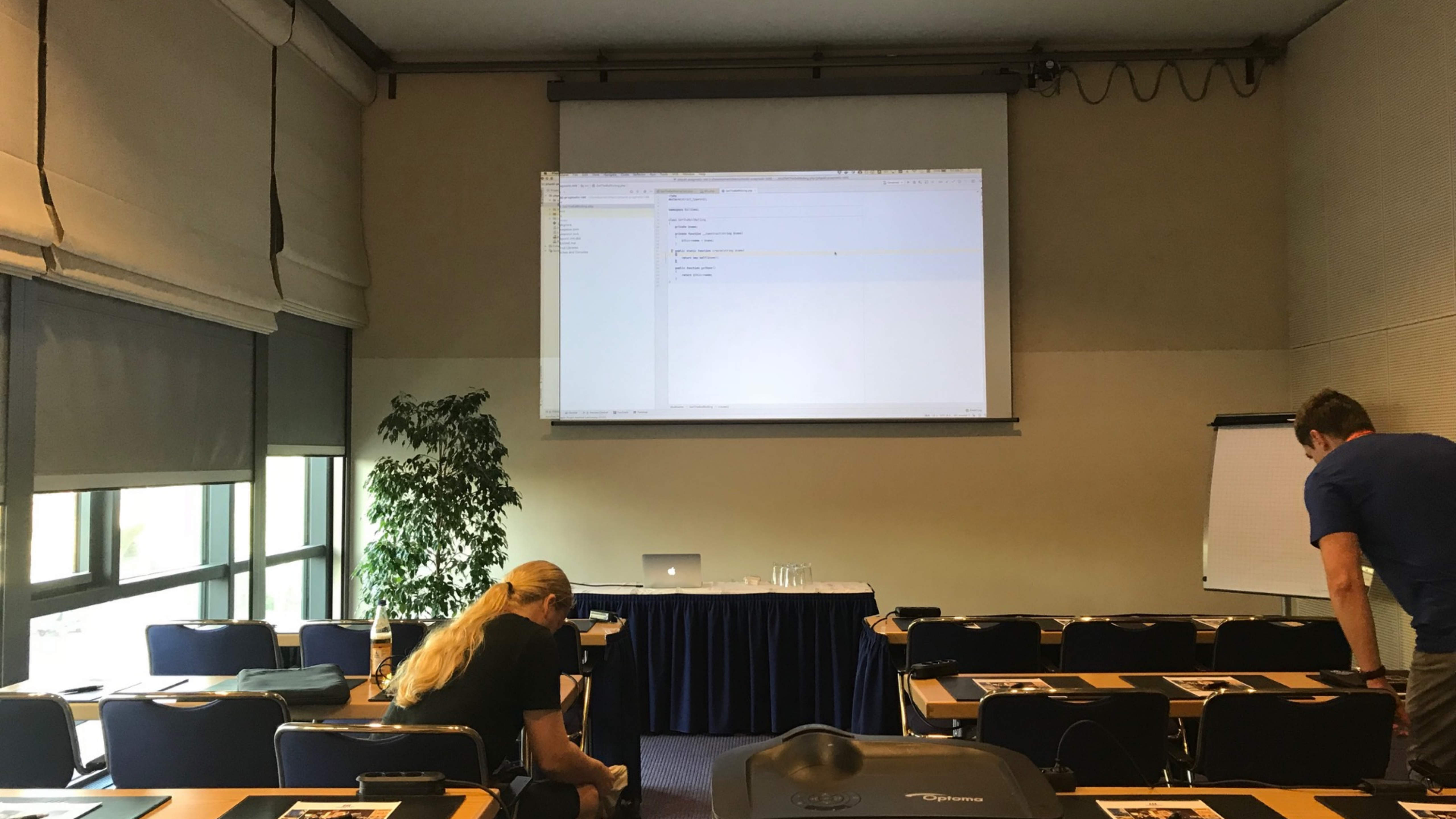
# WORKSHOP RULES

**ASK QUESTIONS**

**IF YOU STILL DO NOT UNDERSTAND, ASK QUESTIONS AGAIN**

**DISCUSS RIGHT NOW, DO NOT WAIT FOR THE “RIGHT MOMENT”**

# WHY WE NEED TESTING?



**WHY AM I  
HERE?**

**ARE WE  
SOFTWARE TESTERS?**

**AND YET  
WE KEEP ON SAYING  
WE WRITE TESTS...**

**WHERE CAN I GET  
MORE OF THOSE  
TESTS?**



# EXCUSES, EXCUSES

TESTS SLOW US DOWN

WE WILL NEVER GET TO 100% CODE COVERAGE

WE DO NOT HAVE TIME TO LEARN TESTING, WE'RE TOO BUSY SHIPPING CODE

# **EXCUSES, EXCUSES**

## **TESTS SLOW US DOWN?**

# STOP TESTFILEING

**DO YOU OFTEN DO THIS?**

~ `php test.php`

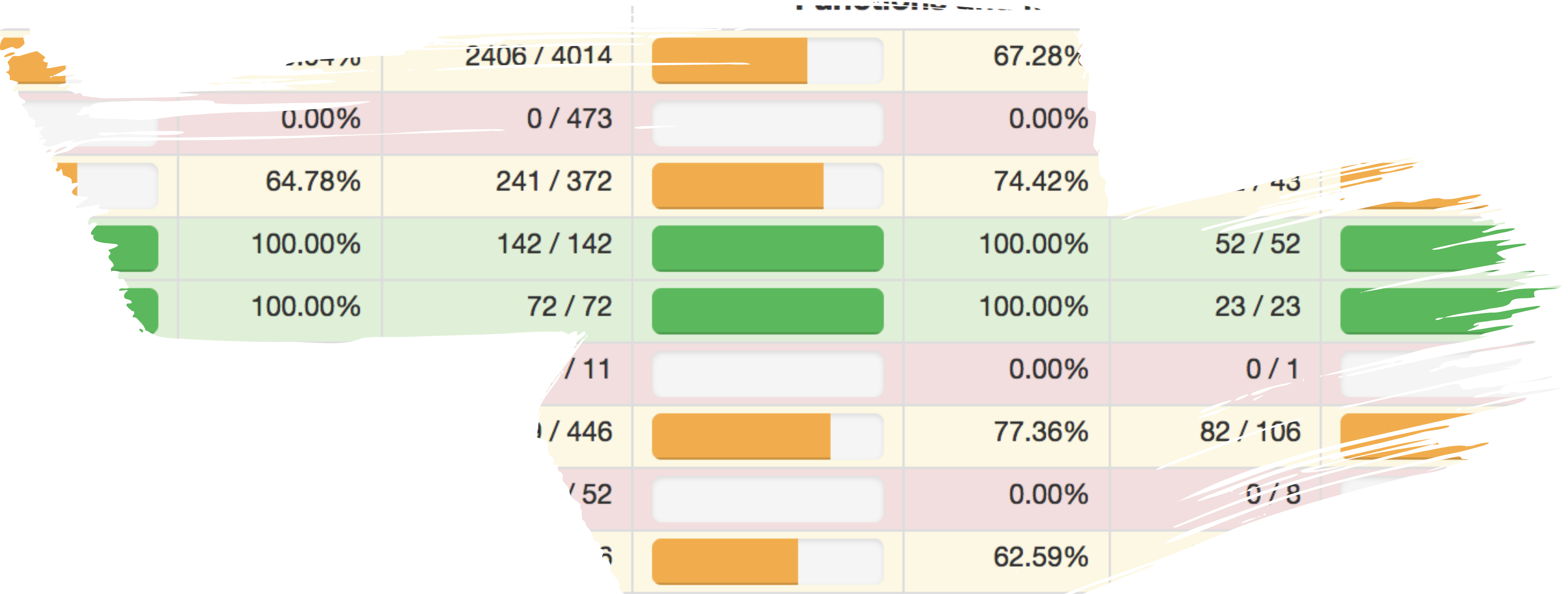
**OR THIS**

`http://localhost/test_problem.php`

# **EXCUSES, EXCUSES**

**WE WILL NEVER GET TO 100%  
CODE COVERAGE**

# CODE COVERAGE






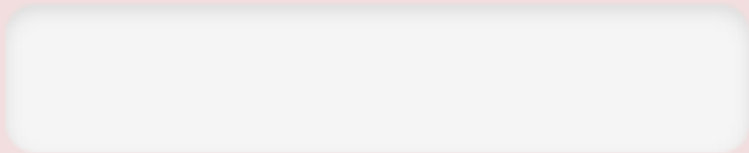
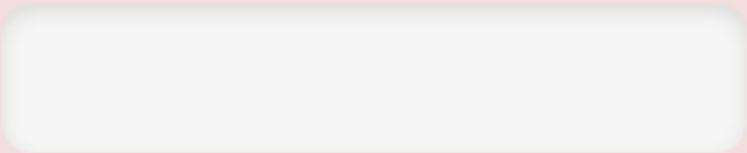
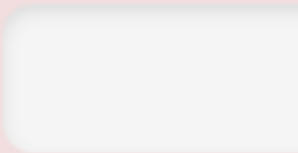















# CODE COVERAGE

Classes and Traits		
<div><div></div></div>	56.88%	62 / 109
<div><div></div></div>	0.00%	0 / 8
<div><div></div></div>	64.29%	9 / 14
<div><div></div></div>	100.00%	9 / 9
<div><div></div></div>	100.00%	5 / 5
<div><div></div></div>	76.47%	13 / 17
<div><div></div></div>	48.15%	26 / 54
	n/a	0 / 0








# CODE COVERAGE

Code Coverage					
Functions and Methods			Classes and Traits		
<div><div></div></div>	67.28%	368 / 547	<div><div></div></div>	56.88%	62 / 109
<div><div></div></div>	0.00%	0 / 28	<div><div></div></div>	0.00%	0 / 8
<div><div></div></div>	74.42%	32 / 43	<div><div></div></div>	64.29%	9 / 14
<div><div></div></div>	100.00%	52 / 52	<div><div></div></div>	100.00%	9 / 9
<div><div></div></div>	100.00%	23 / 23	<div><div></div></div>	100.00%	5 / 5
<div><div></div></div>	77.36%	82 / 106	<div><div></div></div>	76.47%	13 / 17
<div><div></div></div>	62.59%	179 / 286	<div><div></div></div>	48.15%	26 / 54
	n/a	0 / 0		n/a	0 / 0

# CODE COVERAGE

Code Coverage						
Lines			Functions and Methods			
	59.94%	2406 / 4014		67.28%	368 / 547	
	0.00%	0 / 473		0.00%	0 / 28	
	64.78%	241 / 372		74.42%	32 / 43	
	100.00%	142 / 142		100.00%	52 / 52	
	100.00%	72 / 72		100.00%	23 / 23	
	67.04%	299 / 446		77.36%	82 / 106	
	67.54%	1652 / 2446		62.59%	179 / 286	
	n/a	0 / 0		n/a	0 / 0	

20 > 80

	Code Coverage				Cod
		Lines		Function	
Total	<div><div></div></div>	59.94%	2406 / 4014	<div><div></div></div>	
 Command	<div><div></div></div>	0.00%	0 / 473	<div><div></div></div>	
 Controller	<div><div></div></div>	64.78%	241 / 372	<div><div></div></div>	
 Entity	<div><div></div></div>	100.00%	142 / 142	<div><div></div></div>	
 Model	<div><div></div></div>	100.00%	72 / 72	<div><div></div></div>	
 Repository	<div><div></div></div>	67.04%	299 / 446	<div><div></div></div>	
 Service	<div><div></div></div>	67.54%	1652 / 2446	<div><div></div></div>	
 AppBundle.php		n/a	0 / 0		

# **EXCUSES, EXCUSES**

**WE DO NOT HAVE TIME TO LEARN TESTING,  
WE'RE TOO BUSY SHIPPING CODE**

**//@TODO: STANDSTILL**

# DON'T BE SCARED OF PHPUNIT\*

IT IS JUST A CODE RUNNER

UNIT, INTEGRATION OR ACCEPTANCE TESTS

SMOKE TESTS

WEBSITE SCRAPER

...

# TESTS ANSWER QUESTIONS

IS MY CODE WORKING CORRECTLY?

WILL MY CODE WORK CORRECTLY?

HOW IS MY CODE SUPPOSE TO WORK?

IS MY CODE DESIGNED WELL?

CAN I REFACTOR MY CODE?

# TESTS ANSWER QUESTIONS

**IS MY CODE WORKING CORRECTLY?**

WILL MY CODE WORK CORRECTLY?

HOW IS MY CODE SUPPOSE TO WORK?

IS MY CODE DESIGNED WELL?

CAN I REFACTOR MY CODE?

# TESTS ANSWER QUESTIONS

IS MY CODE WORKING CORRECTLY?

**WILL MY CODE WORK CORRECTLY?**

HOW IS MY CODE SUPPOSE TO WORK?

IS MY CODE DESIGNED WELL?

CAN I REFACTOR MY CODE?

# TESTS ANSWER QUESTIONS

IS MY CODE WORKING CORRECTLY?

WILL MY CODE WORK CORRECTLY?

**HOW IS MY CODE SUPPOSE TO WORK?**

IS MY CODE DESIGNED WELL?

CAN I REFACTOR MY CODE?

# TESTS ANSWER QUESTIONS

IS MY CODE WORKING CORRECTLY?

WILL MY CODE WORK CORRECTLY?

HOW IS MY CODE SUPPOSE TO WORK?

**IS MY CODE DESIGNED WELL?**

CAN I REFACTOR MY CODE?

# TESTS ANSWER QUESTIONS

IS MY CODE WORKING CORRECTLY?

WILL MY CODE WORK CORRECTLY?

HOW IS MY CODE SUPPOSE TO WORK?

IS MY CODE DESIGNED WELL?

**CAN I REFACTOR MY CODE?**

# HOW TO WRITE TESTS?

**GIVEN WHEN THEN**

**SETUP EXERCISE VERIFY TEARDOWN**

# ANATOMY OF PHPUNIT TEST CASE

```
class TeamTest extends PHPUnit\Framework\TestCase;
{
    private $team;

    public function setUp()
    {
        $this->team = Team::create('Hellas Verona');
    }

    public function testGetName()
    {
        $name = $this->team->getName();

        $this->assertEquals('Hellas Verona', $name);
    }
}
```

# WORKSHOP RULES

**ASK QUESTIONS**

**IF YOU STILL DO NOT UNDERSTAND, ASK QUESTIONS AGAIN**

**DISCUSS RIGHT NOW, DO NOT WAIT FOR THE “RIGHT MOMENT”**



# EXERCISE 0

**Checkout and do composer install**

<https://github.com/lmuzinic/phpdd-pragmatic-tdd>

```
git clone git@github.com:lmuzinic/phpdd-pragmatic-tdd.git
cd phpdd-pragmatic-tdd
composer install
vendor/bin/phpunit
```

OK (1 test, 1 assertion)

**There is <10 MB of packages to download, so you can use your mobile data if WIFI does not work.**

# EXERCISE 1

## BINARY GAP

[https://app.codility.com/programmers/lessons/1-iterations/binary\\_gap/](https://app.codility.com/programmers/lessons/1-iterations/binary_gap/)

Find longest sequence of zeros in binary representation of an integer.

A binary gap within a positive integer N is any maximal sequence of consecutive zeros that is surrounded by ones at both ends in the binary representation of N.

For example, number 9 has binary representation 1001 and contains a binary gap of length 2. The number 529 has binary representation 1000010001 and contains two binary gaps: one of length 4 and one of length 3. The number 20 has binary representation 10100 and contains one binary gap of length 1. The number 15 has binary representation 1111 and has no binary gaps. The number 32 has binary representation 100000 and has no binary gaps.

# EXERCISE 2

## ODD OCCURRENCES IN ARRAY

[https://app.codility.com/programmers/lessons/2-arrays/odd\\_occurrences\\_in\\_array/](https://app.codility.com/programmers/lessons/2-arrays/odd_occurrences_in_array/)

A non-empty array  $A$  consisting of  $N$  integers is given. The array contains an odd number of elements, and each element of the array can be paired with another element that has the same value, except for one element that is left unpaired.

$A[0] = 9$   $A[1] = 3$   $A[2] = 9$   $A[3] = 3$   $A[4] = 9$   $A[5] = 7$   $A[6] = 9$

# DISCUSSION

**IMPLEMENT A FOOTBALL LEAGUE MANAGEMENT APP**

**– We want to display standings table on our website**

**WHAT IS YOUR BIGGEST CONCERN?**

# EXERCISE 3

**START IMPLEMENTING STANDINGS**

**Write a test for getting sorted standings**

**Talk about domain**

**Implementation**

# EXERCISE 4

## IMPLEMENT TEAM POSITION

An object that will hold position inside the league table

Focus just on this class, use `--filter`

# EXERCISE 5

## IMPLEMENT STANDINGS

Using a TeamPosition that we just created

Do not worry about ugly code, we will refactor it later, just create a thing that works!

# EXERCISE 6

## REFACTOR

Replace spl\_object\_hash with sha1 from team name

Move checking, creating and returning an TeamPosition into separate method

# EXERCISE 7

## NEXT YEAR

League manager wants to change the rules for scoring – teams that have equal number of points, sort by points scored (higher points scored moves up)

Keep the possibility to show scoring for last year as well.

Extract logic into separate class.

# EXERCISE 8

## TEST DUMMIES

Refactor the test using stubs

# EXERCISE 9

**ADVANCED RULEBOOK**

**Write the test.**

**Write the implementation.**

**Push the both rulebooks into standings tests.**

# EXERCISE 10

## CODECOVERAGE & CRAP

### Run codecoverage

```
~ vendor/bin/phpunit --coverage-html var/coverage
```

### Explain CRAP

$CRAP = CC^2 \times U^3 + CC$

# EXERCISE 11

## EXCEPTIONS

Last year I have entered a match with same teams

Last year I have created a team with no name

# EXERCISE 12

## REPOSITORIES

It makes sense that Standings should use a Repository

Implement such repository, add `sleep(1)` to each method.

Figure out how to make test suite fast again.

# EXTRA/HOMEWORK

## GAMES WON

- implement the feature, given a scenario, where standings table displays game won

## TIES

- implement the feature, given a scenario, where two teams have played a tie

## TWO POINTS

- given a scenario where RuleBook also defines how much points should each team get after match win, write additional test case where each win gets you 2 points

## ANYTHING GOES

- have an idea how to make this whole thing better?
- write tests and demonstrate :)

# QUESTIONS? /r/AMA?



**Luka Muzinic**

**@lmuzinic**

**[luka.muzinic.net/talks](http://luka.muzinic.net/talks)**

# **HOMEWORK**

## **READING LIST**

### **Reading list**

<https://www.devmynd.com/blog/five-factor-testing/>

<https://martinfowler.com/articles/practical-test-pyramid.html>

<https://dev.to/theobendixson/the-problem-that-unit-tests-solve-b2l>

<https://blog.liplex.de/testing-private-and-protected-methods-with-phpunit/>

### **Libraries**

<https://github.com/sebastianbergmann/phpunit>

<https://github.com/phpspec/phpspec>

<https://github.com/Codeception/Codeception>

<https://github.com/phpstan/phpstan>

<https://github.com/infection/infection>

# KTHXBAI

Photos by Les Anderson, Joshua Earle, Ian Espinosa and Tom Roberts on Unsplash