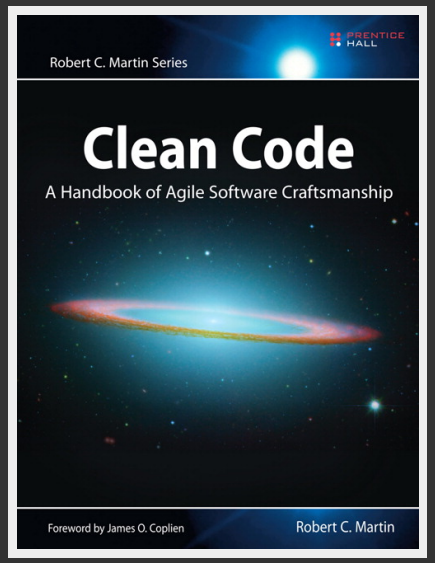
PHPUNIT BEST PRACTICES

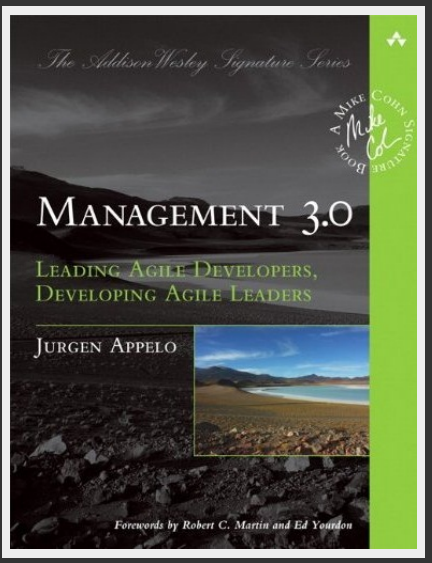
Volker Dusch / @__edorian

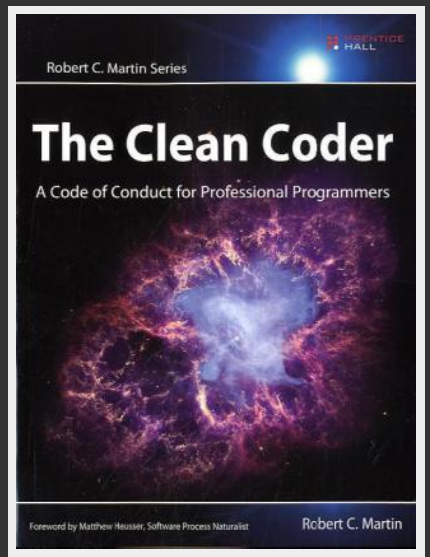
ABOUT ME

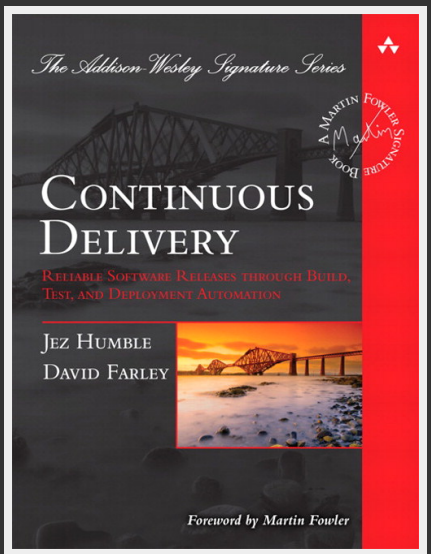
- Software Engineer
- PHP since 11 years
- CI
- CleanCode
- DevOps
- TDD
- Shipping
- Bullet points

INSTEAD OF ME









WORKING FOR

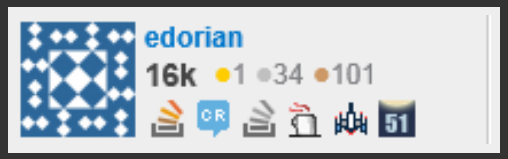
ResearchGate gives science back to the people who make it happen.

We help researchers build reputation and accelerate scientific progress.

On their terms.

GET IN TOUCH

- stack**overflow**:



- Twitter: @__edorian
- g+: Volker Dusch
- IRC: edorian
- Mail: php@wallbash.com

AGENDA

- Some practices I value
- Your mileage may vary
- By no means complete

WRITE TESTS

It's sounds obvious but getting started sometimes is the hardest part!

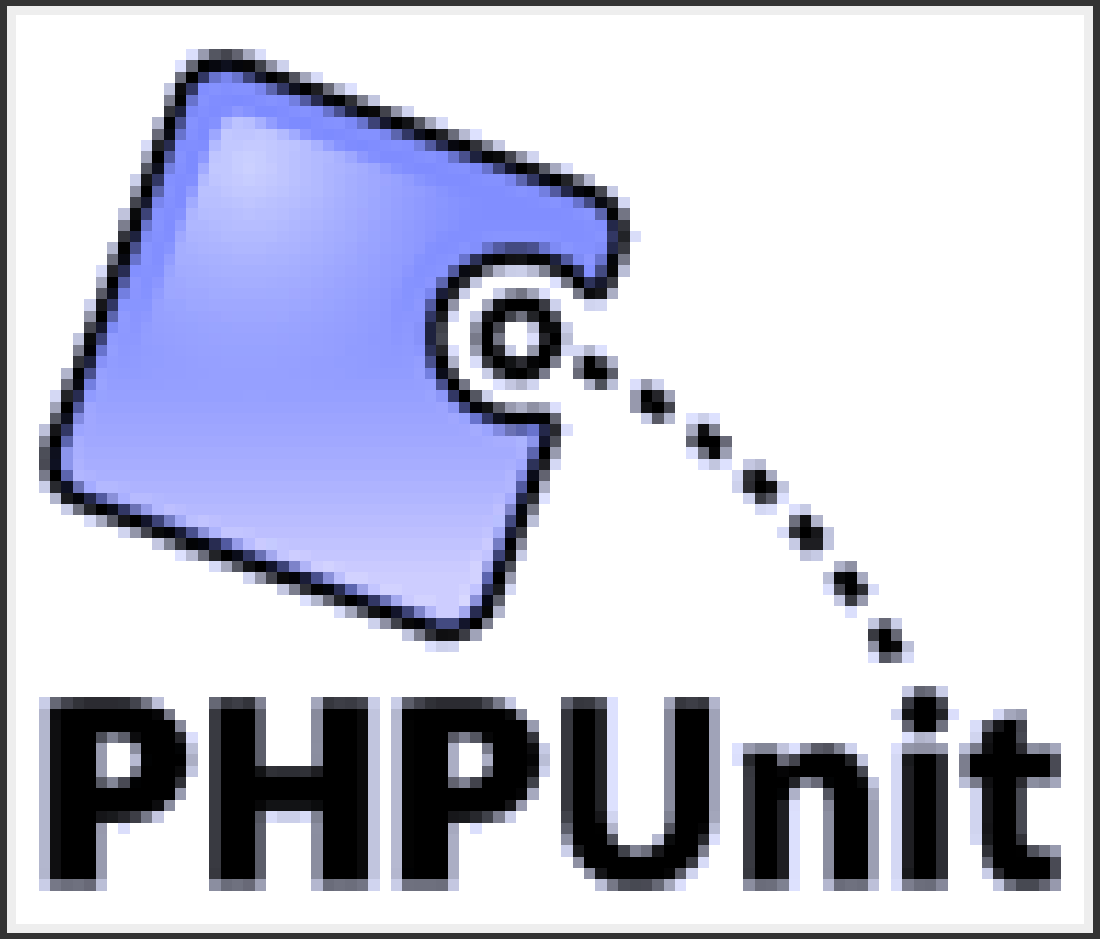
THE FASTEST THING YOU CAN DO

```
hits=`curl -s staging.project.com | grep 'Login:' | wc -l`;
test $hits -eq 1 || echo "Frontpage error!"
```

- Staging server
- Testing your builds
- All without even touching PHPUnit

```
data="login=test&passwort=secure&csrf="$csrfToken
hits=`curl -X POST -d staging.project.com | grep 'Hello, testuser' | wc -l`;
test $hits -eq 1 || echo "Login error!"
```

LET'S GO



UPGRADE TO PHPUNIT 3.7

EASE INSTALLTION

PHAR

```
wget http://pear.phpunit.de/get/phpunit.phar
chmod +x phpunit.phar
./phpunit.phar --version
```

or

```
wget http://pear.phpunit.de/get/phpunit.phar
chmod +x phpunit.phar
mv phpunit.phar /usr/local/bin/phpunit
phpunit --version
```

COMPOSER

The Dependency Manager for PHP

With the best from zypper, bundler, pip, gem and npm

PHPUNIT PER PROJECT

```
composer.json
{
   "require-dev": {
      "phpunit/phpunit": "3.7.*"
   }
}
```

```
composer install
./vendor/bin/phpunit --version
```

PHPUNIT GLOBAL INSTALL

```
{
   "require": {
      "phpunit/phpunit": "3.7.*"
   },
   "config": {
      "bin-dir": "/usr/local/bin/"
   }
}
```

```
sudo php composer installl
phpunuit --version
```

PEAR

- pear config-set auto_discover 1
- pear install pear.phpunit.de/PHPUnit
- phpunit --version

USE SPECIFIC ASSERTIONS

PHPUnit ships with over 90 assertions.

http://www.phpunit.de/manual/current/en/appendixes.assertions.html

Use them to get pretty and helpful error messages.

assertTrue vs assertInstanceOf

```
$foo = new StdClass();
$this->assertTrue($foo instanceOf Countable);
```

“Failed asserting that false is true.”

```
$foo = new StdClass();
$this->assertInstanceOf('Countable', $foo);
```

“Failed asserting that
stdClass() is an instance of interface 'Countable'.”

assertEquals vs assertJsonStringEqualsJsonFile

assertEquals

```
Failed asserting that two strings are equal.
--- Expected
+++ Actual
@@ @@
-'{ "Conference": "FOSDEM", "Talk": "PHPUnit", "JSON": "Apparently", "Shoutout"
: "Jenkins" }'
+'{ "Conference": "FOSDEM", "Talk": "PHPUnit", "JSON": "Apparently", "Shoutout"
: "Hudson" }'
```

assertEquals vs assertJsonStringEqualsJsonFile

assertJsonStringEqualsJsonFile

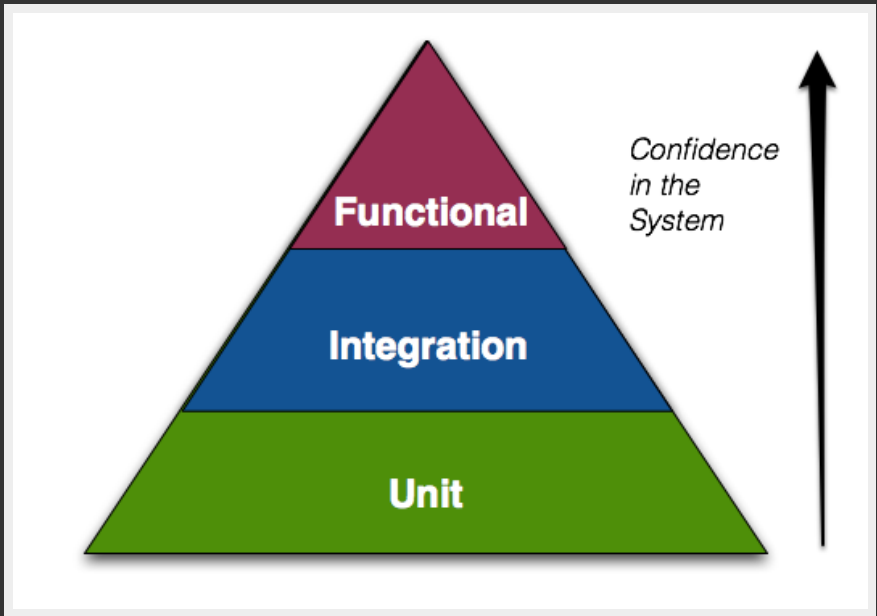
```
Failed asserting that two objects are equal.
--- Expected
+++ Actual
@@ @@
 stdClass Object (
     'Conference' => 'FOSDEM'
     'Talk' => 'PHPUnit'
     'JSON' => 'Apparently'
-    'Shoutout' => 'Jenkins'
+    'Shoutout' => 'Hudson'
 )
```

HAVE A FAST TEST SUITE

If it takes to long to run your tests you won't do it

SEPERATE YOUR TESTS

http://elblinkin.info/2012/03/goldilocks-on-test-sizes/



BY FOLDER STRUCTURE

```
.
|-- src
|   `-- foo
|       `-- bar
|           `-- Baz.php
`-- tests
    |-- functional
    |-- integration
    |-- unit
    |   `-- foo
    |       `-- bar
    |           `-- BazTest.php
    `-- web
```

```
phpunit tests/unit
```

BY CONFIG FILE

```
<testsuites>
  <testsuite name="Unit" suffix="Test.php">
    <directory>tests</directory>
  </testsuite>
  <testsuite name="Integration" suffix="Test.Integration.php">
    <directory>tests</directory>
  </testsuite>
</testsuites>
```

```
phpunit --testsuite Unit
```

OR HOWEVER YOU SEE FIT

- Use @group
- Use @filter and naming conventions

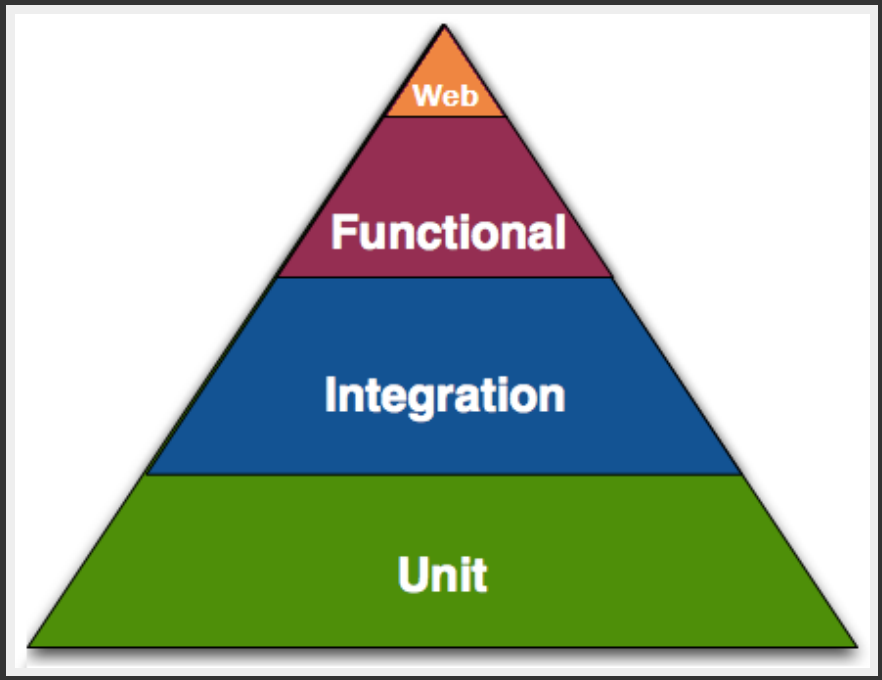
BOOTSTRAP ONLY WHAT YOU NEED

You can use a test listener:

http://www.phpunit.de/manual/current/en/extending-phpunit.html#extending-phpunit.PHPUnit*Framework*TestListener

```
public function startTestSuite(PHPUnit_Framework_TestSuite $suite)
{
    // Just an example of what is possible
    require __DIR__ . $suite->getName() . 'Bootstrap.php';
}
```

HOW MANY TESTS?



- Web: 7
- Funtional: One per feature
- Integration: One per 3 classes
- Unit: Find a balance

---

*totally made up numbers to drive home the point I'm trying to make

WEB TESTS?

- behat (mink) for js-through-the-server testing - Great for testing your whole stack
  - Really hard to maintain
  - Mink relives some of the pain
- Test through your front controller instead of the webserver with behat or phpunit
  - Faster, easier once set up

TEST CLASSES, NOT METHODS

> Unit testing, in PHP, is about testing the observable behaviors of a class!

Observable from the outside! Nobody cares about the internal state of a class if it never changes the outcome of a method call.

SAMPLE

What should we test there?

```
public function setValue($value) {
   $this->value = $value;
}

public function execute() {
   if (!$this->value) {
      throw new Exception("No Value, no good");
   }
   return $value * 10; // business logic
}
```

- **If** we **don't** call setValue calling execute **will** throw an exception
- **If** we **do** call setValue calling execute **will** return the computed result.
- So we are testing **two behaviors** of your class and not the methods in isolation!

RELEVANT BEHAVIORS

What to test then?

return values

```
public fuction celciusToFarenheit($degreesFarenheit) {
    return ($degreesFarenheit - 32) * 5 / 9;
}
```

method calls to other objects

```
public fuction stopCar() {
    $this->handbreak->engage();
    $this->engine->shutdown();
}
```

Global state

```
public fuction avoidThisWherePossible($logMessage) {
    file_put_contents(static::$LOGFILE, $logMessage, FILE_APPEND);
    $_SESSION['logcalls']++;
}
```

DON’T TEST GETTERS AND SETTERS

One test case per behavior

- You waste time
- Your code coverage reports won't tell you about dead code
- If they don't impact the outcome delete them

QUESTIONS?

> *"The secret in testing is in writing testable code"*
>
> - Miško Hevery

Additional resources

- "The Clean Code Talks -- Unit Testing
- How to Write Clean, Testable Code
- The Clean Code Talks - Don't Look For Things!
- Flaw: Brittle Global State & Singletons
- static considered harmful
- The UNIT in unit testing
- An introduction to PHPUnits @covers annotation

THANK YOU