
# Introduction to Unit Testing with PHPUnit

by Michelangelo van Dam

# Contents


- ✓Who am I ?
- ✓What is Unit Testing ?
- ✓Unit Testing in short
- ✓Why do Testing ?
- ✓SimpleTest
- ✓PHPUnit
- ✓Starting w/ PHPUnit
- ✓Example
- ✓More Testing
- ✓Data Provider
- ✓Expected Exceptions
- ✓Fixtures
- ✓Doubles
- ✓Stubs
- ✓Mocks
- ✓Database Testing
- ✓Zend_Test
- ✓Interesting Readings
- ✓Questions ?

# Who am I ?

Michelangelo van Dam

Independent Enterprise PHP consultant
Co-founder PHPBelgium

Mail me at dragonbe [at] gmail [dot] com
Follow me on http://twitter.com/DragonBe
Read my articles on http://dragonbe.com
See my profile on http://linkedin.com/in/michelangelovandam

# What is Unit Testing ?

Wikipedia: "*is a method of testing that verifies the individual units of source code are working properly*"

# Unit Testing in short

- unit: the smallest testable code of an app
  - procedural: function or procedure
  - OOP: a method
- test: code that checks code on
  - functional behavior
    - ✓ expected results
    - ✓ unexpected failures

# Why do (Unit) Testing ?

- automated testing
- test code on functionality
- detect issues that break existing code
- progress indication of the project
- alerts generation for monitoring tools

# SimpleTest

- comparable to JUnit/PHPUnit
- created by Marcus Baker
- popular for testing web pages at browser level

# PHPUnit

- Part of xUnit familiy (JUnit, SUnit,...)
- created by Sebastian Bergmann
- integrated/supported
  - Zend Studio
  - Zend Framework

# Starting with PHPUnit

Installation is done with the PEAR installer

```
# pear channel-discover pear.phpunit.de
# pear install phpunit/PHPUnit
```

Upgrading is as simple

```
# pear upgrade phpunit/PHPUnit
```

# Hello World

```php
<?php
class HelloWorld
{
    public $helloWorld;

    public function __construct($string = 'Hello World!')
    {
        $this->helloWorld = $string;
    }

    public function sayHello()
    {
        return $this->helloWorld;
    }
}
```

# Test HelloWorld class

```
<?php
require_once 'HelloWorld.php';
require_once 'PHPUnit/Framework.php';

class HelloWorldTest extends PHPUnit_Framework_TestCase
{
    public function test__construct()
    {
        $hw = new HelloWorld();
        $this->assertType('HelloWorld', $hw);
    }

    public function testSayHello()
    {
        $hw = new HelloWorld();
        $string = $hw->sayHello();
        $this->assertEquals('Hello World!', $string);
    }
}
```

# Testing HelloWorld

```
# phpunit HelloWorldTest HelloWorldTest.php
PHPUnit 3.3.2 by Sebastian Bergmann.

..

Time: 0 seconds

OK (2 tests, 2 assertions)
```

# More testing

- data providers (@dataProvider)
- exception (@expectedException)
- fixtures (setUp() and tearDown())
- doubles (mocks and stubs)
- database testing

# Data Provider

- provides arbitrary arguments
  - array
  - object (that implements Iterator)
- annotated by @dataProvider provider
- multiple arguments

# CombineTest

```php
<?php
class CombineTest extends PHPUnit_Framework_TestCase
{
    /**
     * @dataProvider provider
     */
    public function testCombine($a, $b, $c)
    {
        $this->assertEquals($c, $a . ' ' . $b);
    }

    public function provider()
    {
        return array (
             array ('Hello','World','Hello World'),
             array ('Go','PHP','Go PHP'),
             array ('This','Fails','This succeeds')
        );
    }
}
```

# Testing CombineTest

```
# phpunit CombineTest CombineTest.php
PHPUnit 3.3.2 by Sebastian Bergmann.

..F

Time: 0 seconds

There was 1 failure:

1) testCombine(CombineTest) with data set #2 ('This', 'Fails',
'This succeeds')
Failed asserting that two strings are equal.
expected string <This succeeds>
difference      <     xxxxx???>
got string      <This Fails>
/root/dev/phpunittutorial/CombineTest.php:9

FAILURES!
Tests: 3, Assertions: 3, Failures: 1.
```

# Expected Exception

- testing exceptions
  - that they are thrown
  - are properly catched

# OopsTest

```
<?php
class OopsTest extends PHPUnit_Framework_TestCase
{
    public function testOops()
    {
        try {
            throw new Exception('I just made a booboo');
        } catch (Exception $expected) {
            return;
        }
        $this->fail('An expected Exception was not thrown');
    }
}
```

# Testing OopsTest

```
# phpunit OopsTest OopsTest.php
PHPUnit 3.3.2 by Sebastian Bergmann.

.

Time: 0 seconds

OK (1 test, 0 assertions)
```

# Fixtures

- is a "known state" of an application
  - needs to be 'set up' at start of test
  - needs to be 'torn down' at end of test
  - shares "states" over test methods

# FixmeTest

```php
<?php
class FixmeTest extends PHPUnit_Framework_TestCase
{
    protected $fixme;

    public function setUp()
    {
        $this->fixme = array ();
    }

    public function testFixmeEmpty()
    {
        $this->assertEquals(0, sizeof($this->fixme));
    }

    public function testFixmeHasOne()
    {
        array_push($this->fixme, 'element');
        $this->assertEquals(1, sizeof($this->fixme));
    }
}
```

# Testing FixmeTest

```
# phpunit FixmeTest FixmeTest.php
PHPUnit 3.3.2 by Sebastian Bergmann.

..

Time: 0 seconds

OK (2 tests, 2 assertions)
```

# Doubles

- stub objects
- mock objects

# Stubs

- isolates tests from external influences
  - slow connections
  - expensive and complex resources
- replaces a "system under test" (SUT)
  - for the purpose of testing

# StubTest

```php
<?php
// example taken from phpunit.de
class StubTest extends PHPUnit_Framework_TestCase
{
    public function testStub()
    {
        $stub = $this->getMock('SomeClass');
        $stub->expects($this->any())
             ->method('doSometing')
             ->will($this->returnValue('foo'));
    }

    // Calling $stub->doSomething() will now return 'foo'
}
```

# Testing StubTest

```
# phpunit StubTest StubTest.php
PHPUnit 3.3.2 by Sebastian Bergmann.

.

Time: 0 seconds

OK (1 test, 1 assertion)
```

# Mocks

- simulated objects
- mimics API or behaviour
- in a controlled way
- to test a real object

# ObserverTest

```php
<?php
// example taken from Sebastian Bergmann's slides on
// slideshare.net/sebastian_bergmann/advanced-phpunit-topics

class ObserverTest extends PHPUnit_Framework_TestCase
{
    public function testUpdateIsCalledOnce()
    {
        $observer = $this->getMock('Observer', array('update'));

        $observer->expects($this->once())
                 ->method('update')
                 ->with($this->equalTo('something'));

        $subject = new Subject;
        $subject->attach($observer)
                ->doSomething();
    }
}
```

# Database Testing

- Ported by Mike Lively from DBUnit
- PHPUnit_Extensions_Database_TestCase
- for database-driven projects
- puts DB in know state between tests
- imports and exports DB data from/to XML
- easily added to existing tests

# BankAccount Example

**BankAccount example by Mike Lively**

*http://www.ds-o.com/archives/63-PHPUnit-Database-Extension-DBUnit-Port.html*
*http://www.ds-o.com/archives/64-Adding-Database-Tests-to-Existing-PHPUnit-Test-Cases.html*

**BankAccount class by Sebastian Bergmann**

*http://www.slideshare.net/sebastian_bergmann/testing-phpweb-applications-with-phpunit-and-selenium*

**The full BankAccount class**

*http://www.phpunit.de/browser/phpunit/branches/release/3.3/PHPUnit/Samples/BankAccount/BankAccount.php*

# BankAccount

```php
<?php
require_once 'BankAccountException.php';

class BankAccount
{
    private $balance = 0;

    public function getBalance()
    {
        return $this->balance;
    }

    public function setBalance($balance)
    {
        if ($balance >= 0) {
            $this->balance = $balance;
        } else {
            throw new BankAccountException;
        }
    }

    ...
```

# BankAccount (2)

```
    ...

    public function depositMoney($balance)
    {
        $this->setBalance($this->getBalance() + $balance);
        return $this->getBalance();
    }

    public function withdrawMoney($balance)
    {
        $this->setBalance($this->getBalance() - $balance);
        return $this->getBalance();
    }
}

<?php
class BankAccountException extends RuntimeException { }
```

# BankAccountTest

```
<?php
require_once 'PHPUnit/Extensions/Database/TestCase.php';
require_once 'BankAccount.php';

class BankAccountDBTest extends PHPUnit_Extensions_Database_TestCase
{
    protected $pdo;

    public function __construct()
    {
        $this->pdo = new PDO('sqlite::memory:');
        BankAccount::createTable($this->pdo);
    }

    protected function getConnection()
    {
        return $this->createDefaultDBConnection($this->pdo, 'sqlite');
    }

    protected function getDataSet()
    {
        return $this->createFlatXMLDataSet(
            dirname(__FILE__) . '/BankAccounts.xml');
    }

    ...
```

# BankAccountTest (2)

```
    ...

    public function testNewAccountCreation()
    {
        $bank_account = new BankAccount('123456789123456789', $this->pdo);
        $xml_dataset = $this->createFlatXMLDataSet(
            dirname(__FILE__) . '/NewBankAccounts.xml');
        $this->assertDataSetsEqual(
                $xml_dataset, $this->getConnection()->createDataSet()
        );
    }
}
```

# Testing BankAccount

```
# phpunit BankAccountDbTest BankAccountDbTest.php PHPUnit 3.3.2 by
Sebastian Bergmann.

F

Time: 0 seconds

There was 1 failure:

...
```

# Testing BA (2)

```
...
1) testNewAccountCreation(BankAccountDBTest)
Failed asserting that actual
+----------------------+----------------------+
| bank_account                                |
+----------------------+----------------------+
|    account_number    |       balance        |
+----------------------+----------------------+
|  1234567891234567    |          0           |
+----------------------+----------------------+
|  1234861235723618    |          89          |
+----------------------+----------------------+
|  1593490364962048    |         100          |
+----------------------+----------------------+
|  1593648723021506    |         1216         |
+----------------------+----------------------+
```

# Testing BA (3)

```
...

is equal to expected
+----------------------+----------------------+
| bank_account                                |
+----------------------+----------------------+
|    account_number    |       balance        |
+----------------------+----------------------+
|  159349036496204 86  |        100.00        |
+----------------------+----------------------+
|  159364872302150 67  |       1216.00        |
+----------------------+----------------------+
|  123486123572361 85  |        89.00         |
+----------------------+----------------------+

 Reason: Expected row count of 3, has a row count of 4
/root/dev/phpunittutorial/BankAccountDbTest.php:33

FAILURES!
Tests: 1, Assertions: 1, Failures: 1.
```

# BankAccount Dataset

```
<!-- file: BankAccounts.xml -->
<dataset>
    <bank_account account_number="15934903649620486" balance="100.00" />
    <bank_account account_number="15936487230215067" balance="1216.00" />
    <bank_account account_number="12348612357236185" balance="89.00" />
</dataset>
```

# Testing MVC ZF apps

- Available from Zend Framework 1.6
- Using Zend_Test
  *http://framework.zend.com/manual/en/zend.test.html*
- Requests and Responses are mocked

# Hints & Tips

- Make sure auto loading is set up
  - your tests might fail on not finding classes
- Move bootstrap to a plugin
  - allows to PHP callback the bootstrap
  - allows to specify environment succinctly
  - allows to bootstrap application in a 1 line

# Defining the bootstrap

```
/**
 * default way to approach the bootstrap
 */
class IndexControllerTest extends Zend_Test_PHPUnit_ControllerTestCase
{
    public $bootstrap = '/path/to/bootstrap.php';

    // ...
}


/**
 * Using PHP callback
 */
class IndexControllerTest extends Zend_Test_PHPUnit_ControllerTestCase
{
    public $bootstrap = array('App', 'bootstrap');

    // ...
}
```

# Testing homepage

```
class IndexControllerTest extends Zend_Test_PHPUnit_ControllerTestCase
{
    // ...

    public function testHomePage()
    {
        $this->dispatch('/');
        // ...
    }
}
```

# Testing GET params

```
class IndexControllerTest extends Zend_Test_PHPUnit_ControllerTestCase
{
    // ...

    public function testGetActionShouldReceiveGetParams()
    {
        // Set GET variables:
        $this->request->setQuery(array(
            'foo' => 'bar',
            'bar' => 'baz',
        ));
    }

    // ...
}
```

# Testing POST params

```
class IndexControllerTest extends Zend_Test_PHPUnit_ControllerTestCase
{
    // ...

    public function testPostActionShouldReceivePostParams()
    {
        // Set POST variables:
        $this->request->setPost(array(
            'foo' => 'bar',
            'bar' => 'baz',
        ));
    }

    // ...
}
```

# Testing COOKIE params

```
class IndexControllerTest extends Zend_Test_PHPUnit_ControllerTestCase
{
    // ...

    public function testCookieActionShouldReceiveCookieParams()
    {
        // First set a cookie value
        $this->request->setCookie('username', 'DragonBe');

        // Or set multiple cookies at once
        $this->request->setCookies(array(
            'last_seen' => time(),
            'userlevel' => 'Admin',
        ));
    }

    // ...
}
```

# Let's dispatch it

```
class IndexControllerTest extends Zend_Test_PHPUnit_ControllerTestCase
{
    // ...

    public function testCookieActionShouldReceiveCookieParams()
    {
        // First set a cookie value
        $this->request->setCookie('username', 'DragonBe');

        // Or set multiple cookies at once
        $this->request->setCookies(array(
            'last_seen' => time(),
            'userlevel' => 'Admin',
        ));

        // Let's define the request method
        $this->request->setMethod('POST');

        // Dispatch the homepage
        $this->dispatch('/');
    }

    // ...
}
```

# Demo

- Using Zend Framework "QuickStart" app
  *http://framework.zend.com/docs/quickstart*
  - modified with
    - ✓ detail entry
- Downloads provided on
  *http://mvandam.com/demos/zftest.zip*

# Interesting Readings

- PHPUnit by Sebastian Bergmann
  *http://phpunit.de*
- Art of Unit Testing by Roy Osherove
  *http://artofunittesting.com*
- Mike Lively's blog
  *http://www.ds-o.com/archives/63-PHPUnit-Database-Extension-DBUnit-Port.html*
- Zend Framework Manual: Zend_Test
  *http://framework.zend.com/manual/en/zend.test.phpunit.html*

# Questions ?

Thank you.
This presentation will be available on
http://slideshare.com/DragonBe