INTRODUCTION TO UNIT TESTING

WITH PHPUNIT

Fraser Reed / @fraser_reed

ABOUT ME: FRASER REED

Server Side Engineer - WORKSHOPX

databases scaling

testing automating

canvaspop dna 11 crated. PopKey

TOPICS

WHAT IS UNIT TESTING?

WHAT IS UNIT TESTING?

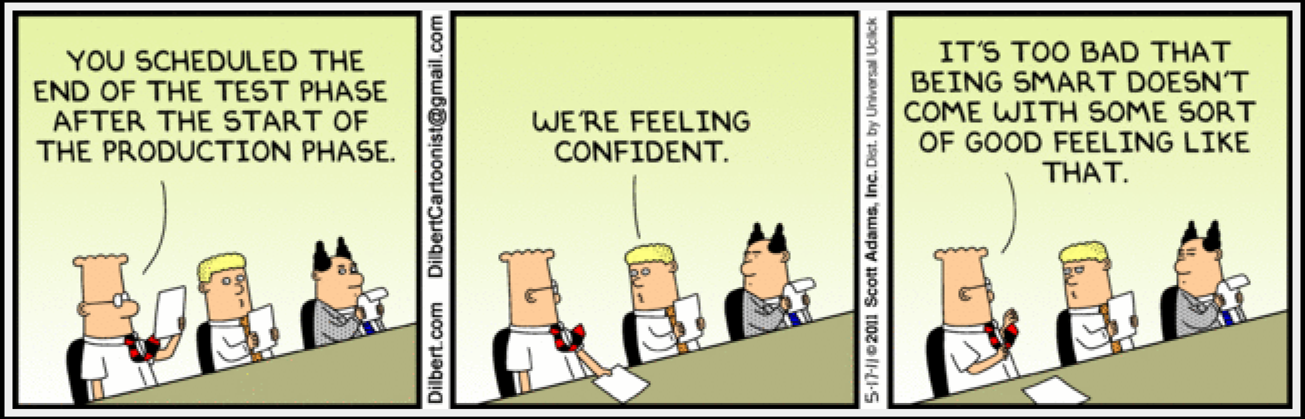
Testing individual units of code to determine if they are fit to use.

The smallest testable part of an application: functions, classes and methods

Prove that code performs within a set of guidelines

Unit testing is a fundamental building block of writing professional, maintainable code

WHY SHOULD I TEST MY CODE?


YOU SCHEDULED THE END OF THE TEST PHASE AFTER THE START OF THE PRODUCTION PHASE.
WE'RE FEELING CONFIDENT.
IT'S TOO BAD THAT BEING SMART DOESN'T COME WITH SOME SORT OF GOOD FEELING LIKE THAT.
Dilbert.com DilbertCartoonist@gmail.com
5-17-11 © 2011 Scott Adams, Inc. Dist. by Universal Uclick

WHY SHOULD I TEST MY CODE?

Confidence in your code

Prevent breaking changes

Speeds up future development

Helps write clean code

BACKGROUND ON PHPUNIT

Written by Sebastian Bergmann

https://phpunit.de/

https://github.com/sebastianbergmann/phpunit

Current stable version: 4.5.1

Many optional packages:
code coverage, lines of code, dbunit, selenium, etc.

INSTALLATION

INSTALLATION VIA PHAR

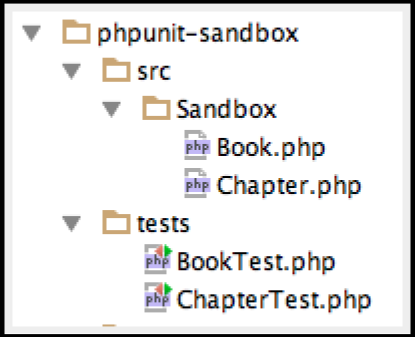
```
wget https://phar.phpunit.de/phpunit.phar
php phpunit.phar
```

INSTALLATION VIA COMPOSER

```
{
    "require-dev": {
        "phpunit/phpunit": "4.5.*"
    }
}
```

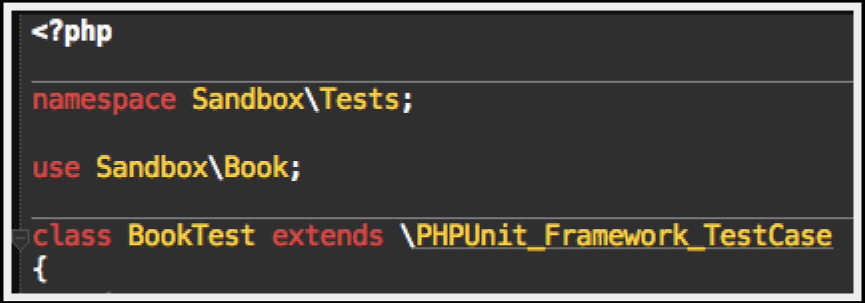
THE BASICS

The tests for a class Class go into a class ClassTest.



```
phpunit-sandbox
  src
    Sandbox
      Book.php
      Chapter.php
  tests
    BookTest.php
    ChapterTest.php
```

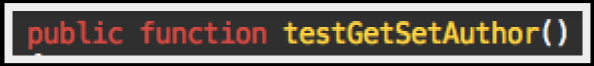
ClassTest inherits (most of the time) from PHPUnit_Framework_TestCase.



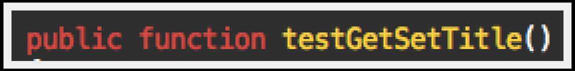
```php
<?php

namespace Sandbox\Tests;

use Sandbox\Book;

class BookTest extends \PHPUnit_Framework_TestCase
{
```

Alternatively, you can extend PHPUnit_Framework_TestCase to override or customize basic functionality.

The tests are public methods that are named test*.

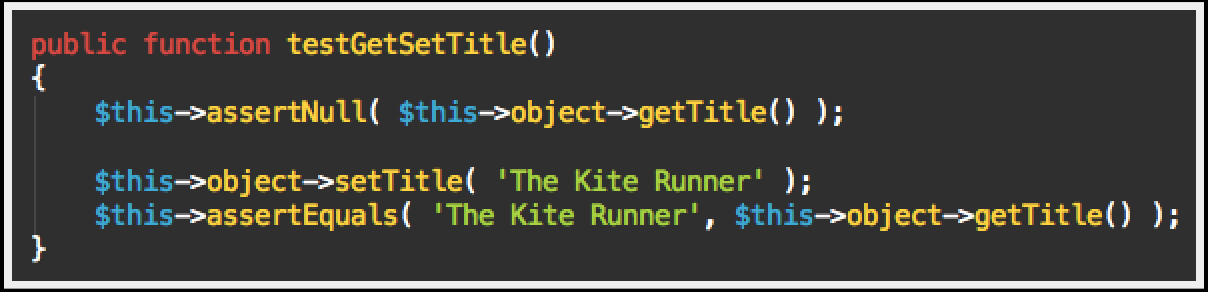


```
public function testGetSetAuthor()
```



```
public function testGetSetTitle()
```

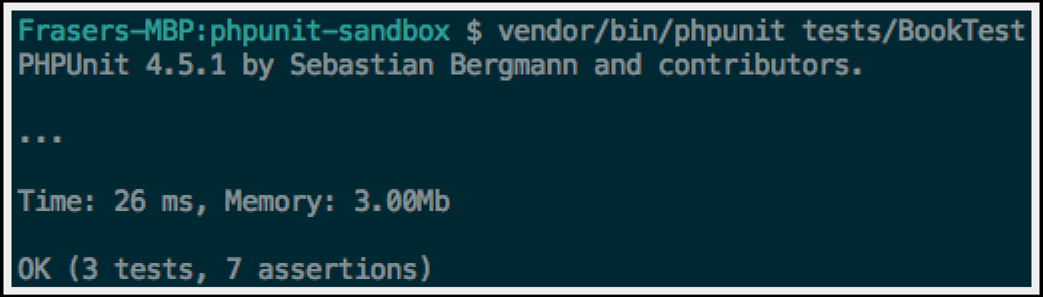
Inside the test methods, assertion methods are used to assert that an actual value matches an expected value.



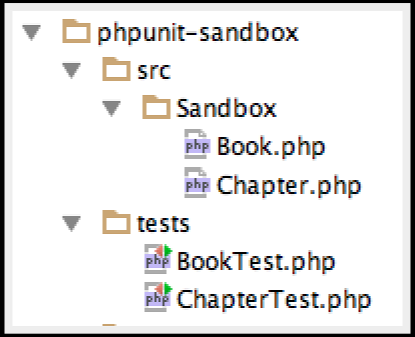
```php
public function testGetSetTitle()
{
    $this->assertNull( $this->object->getTitle() );

    $this->object->setTitle( 'The Kite Runner' );
    $this->assertEquals( 'The Kite Runner', $this->object->getTitle() );
}
```

RUNNING THE TESTS

Direct execution

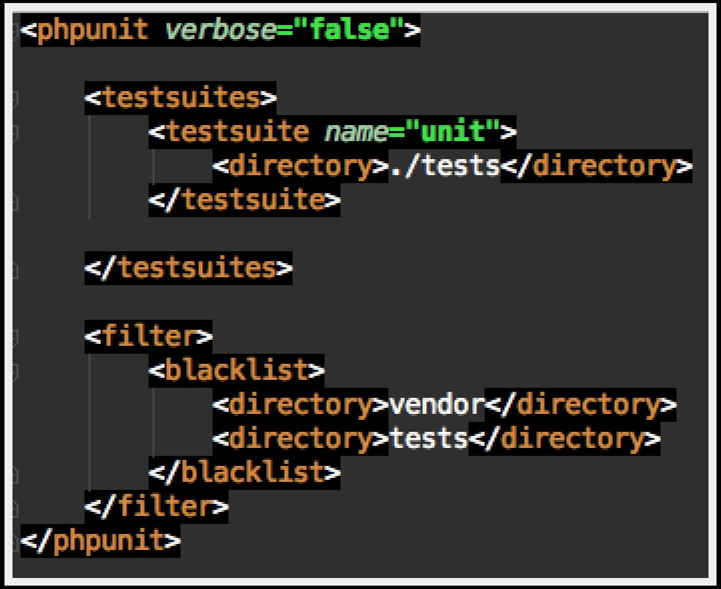


```
Frasers-MBP:phpunit-sandbox $ vendor/bin/phpunit tests/BookTest
PHPUnit 4.5.1 by Sebastian Bergmann and contributors.

...

Time: 26 ms, Memory: 3.00Mb

OK (3 tests, 7 assertions)
```

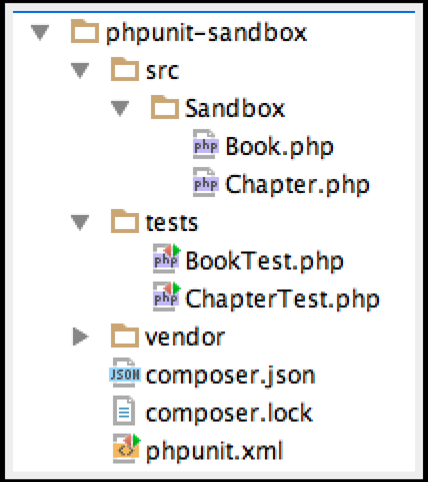


```
▼ phpunit-sandbox
  ▼ src
    ▼ Sandbox
        Book.php
        Chapter.php
  ▼ tests
      BookTest.php
      ChapterTest.php
```

Execution via phpunit.xml

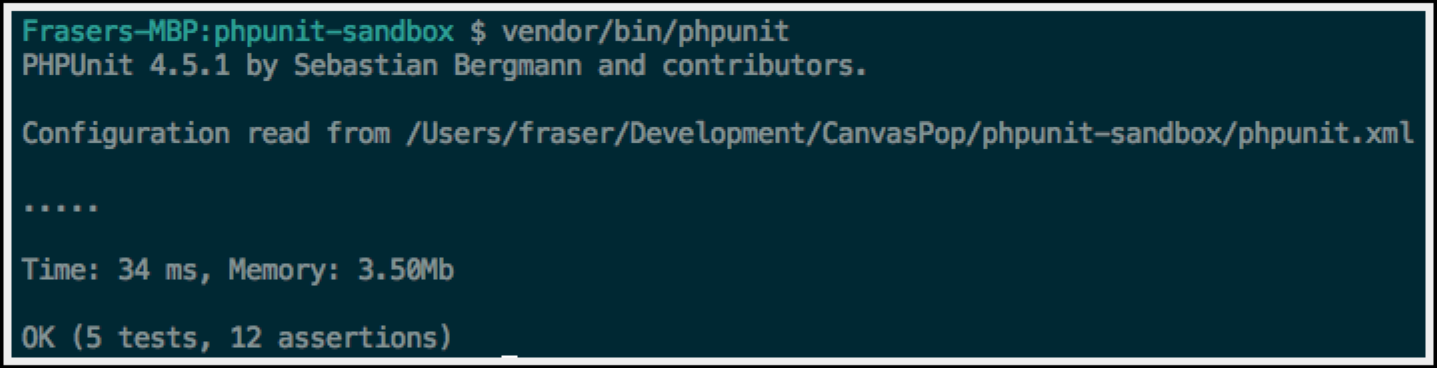


```xml
<phpunit verbose="false">

    <testsuites>
        <testsuite name="unit">
            <directory>./tests</directory>
        </testsuite>

    </testsuites>

    <filter>
        <blacklist>
            <directory>vendor</directory>
            <directory>tests</directory>
        </blacklist>
    </filter>
</phpunit>
```



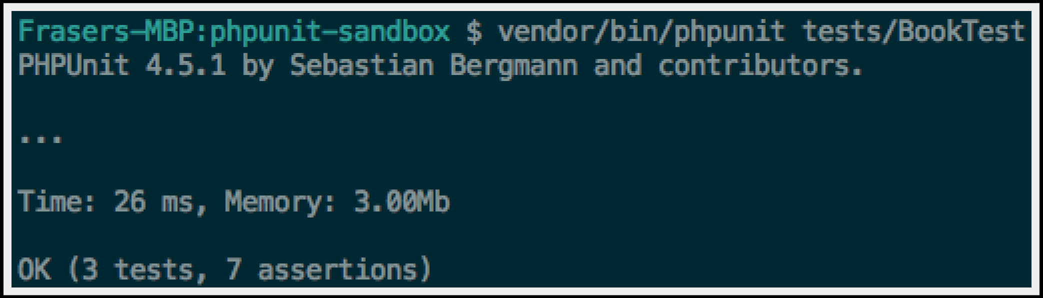
```
▼ phpunit-sandbox
    ▼ src
        ▼ Sandbox
            Book.php
            Chapter.php
    ▼ tests
        BookTest.php
        ChapterTest.php
    ▶ vendor
    composer.json
    composer.lock
    phpunit.xml
```

Execution via phpunit.xml



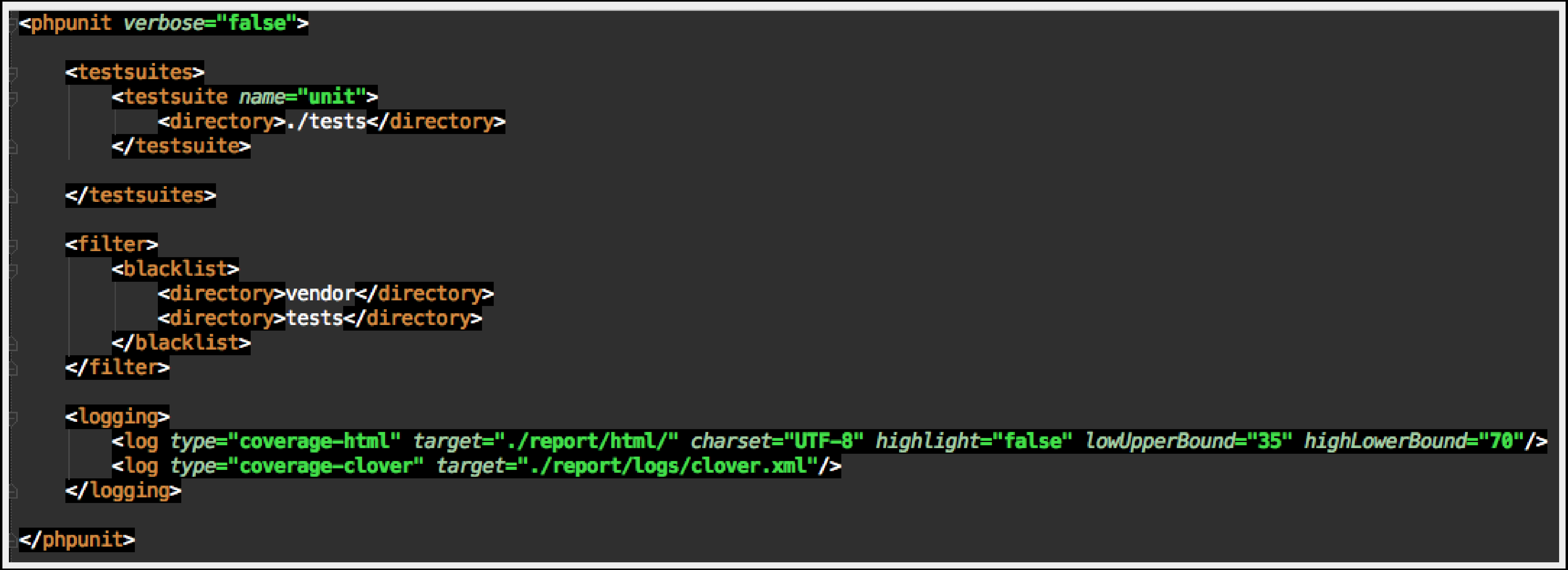
```
Frasers-MBP:phpunit-sandbox $ vendor/bin/phpunit
PHPUnit 4.5.1 by Sebastian Bergmann and contributors.

Configuration read from /Users/fraser/Development/CanvasPop/phpunit-sandbox/phpunit.xml

.....

Time: 34 ms, Memory: 3.50Mb

OK (5 tests, 12 assertions)
```

vs.



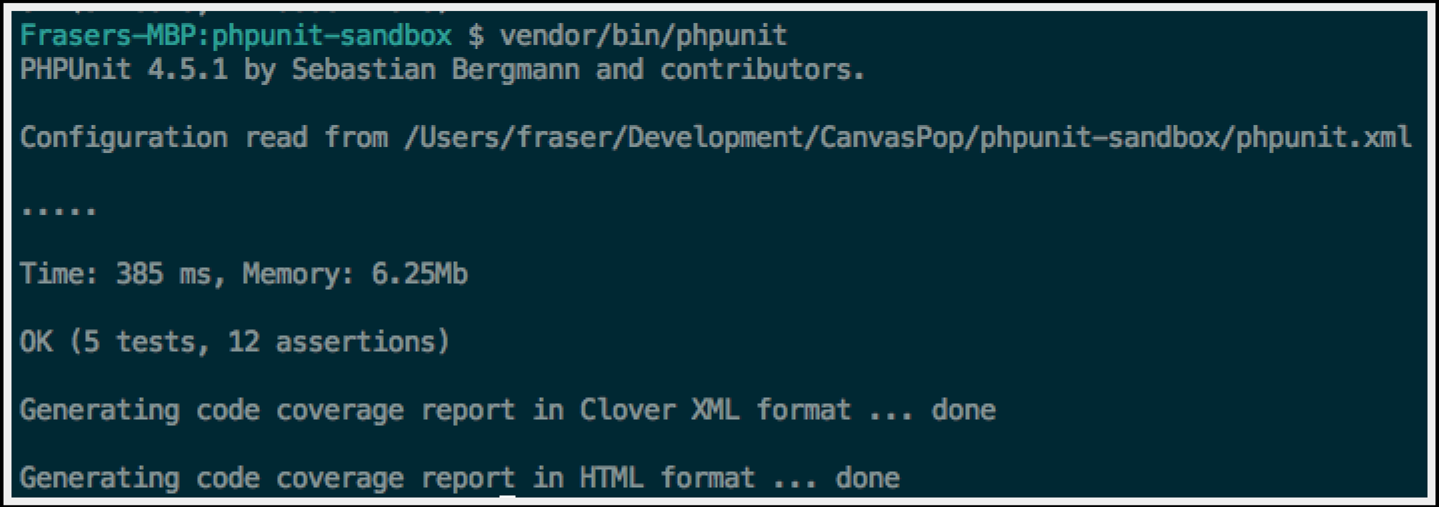
```
Frasers-MBP:phpunit-sandbox $ vendor/bin/phpunit tests/BookTest
PHPUnit 4.5.1 by Sebastian Bergmann and contributors.

...

Time: 26 ms, Memory: 3.00Mb

OK (3 tests, 7 assertions)
```

Execution via phpunit.xml with code coverage



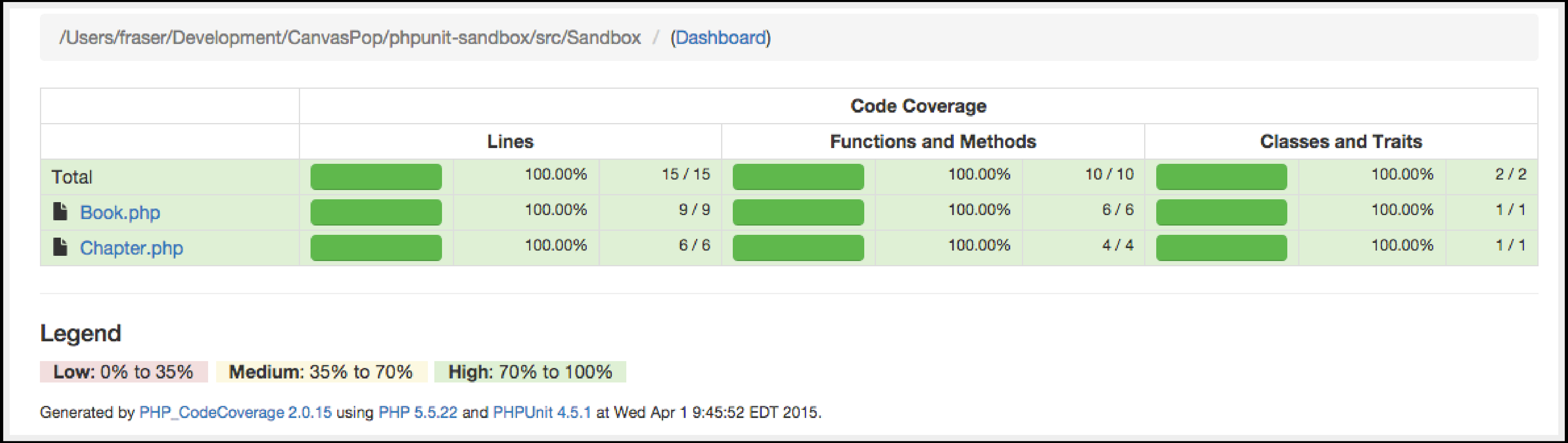
```
<phpunit verbose="false">

    <testsuites>
        <testsuite name="unit">
            <directory>./tests</directory>
        </testsuite>

    </testsuites>

    <filter>
        <blacklist>
            <directory>vendor</directory>
            <directory>tests</directory>
        </blacklist>
    </filter>

    <logging>
        <log type="coverage-html" target="./report/html/" charset="UTF-8" highlight="false" lowUpperBound="35" highLowerBound="70"/>
        <log type="coverage-clover" target="./report/logs/clover.xml"/>
    </logging>

</phpunit>
```

Execution via phpunit.xml with code coverage



```
Frasers-MBP:phpunit-sandbox $ vendor/bin/phpunit
PHPUnit 4.5.1 by Sebastian Bergmann and contributors.

Configuration read from /Users/fraser/Development/CanvasPop/phpunit-sandbox/phpunit.xml

.....

Time: 385 ms, Memory: 6.25Mb

OK (5 tests, 12 assertions)

Generating code coverage report in Clover XML format ... done

Generating code coverage report in HTML format ... done
```

Execution via phpunit.xml with code coverage



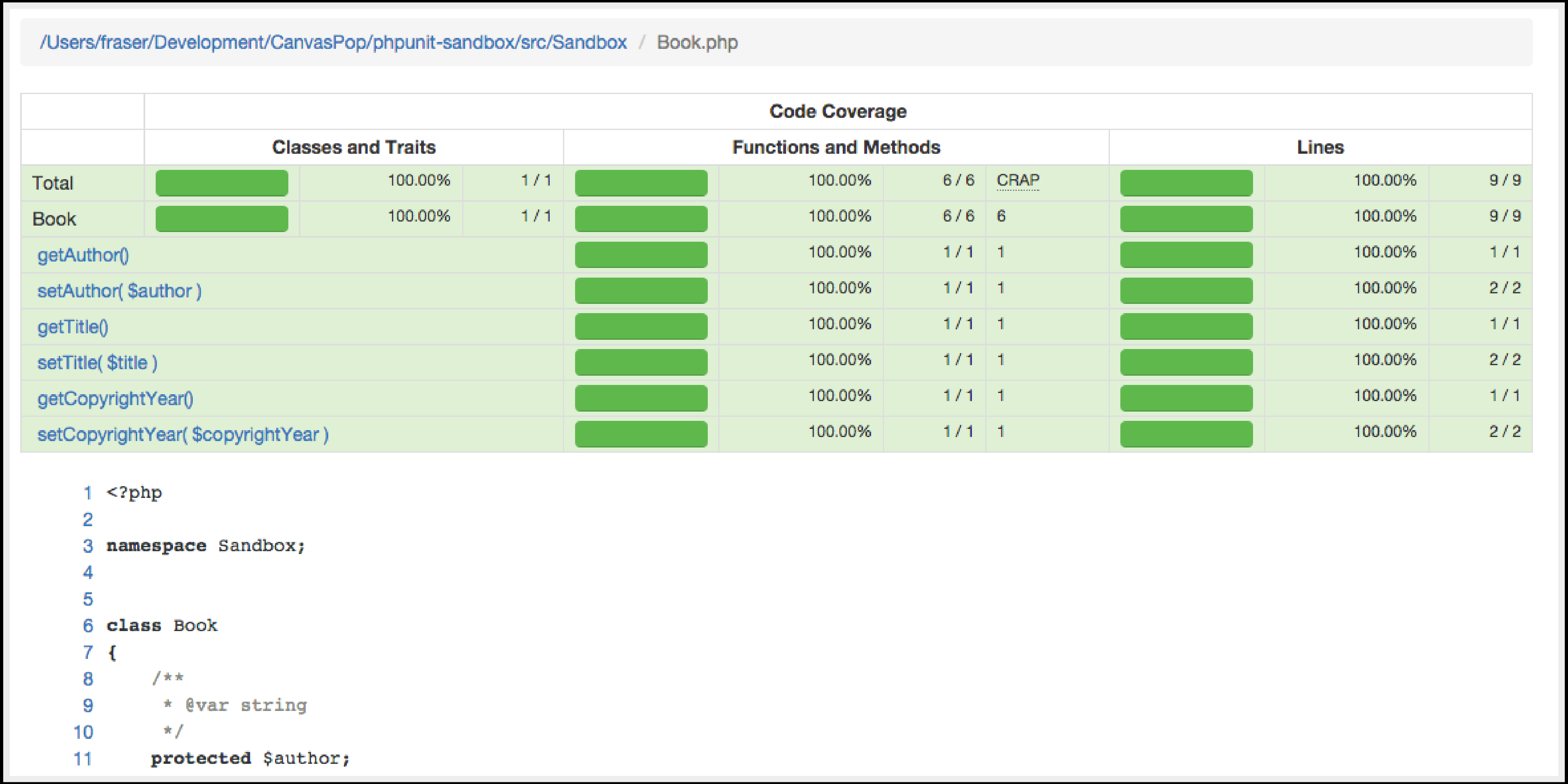
/Users/fraser/Development/CanvasPop/phpunit-sandbox/src/Sandbox / (Dashboard)

	Code Coverage								
	Lines			Functions and Methods			Classes and Traits		
Total	🟩	100.00%	15 / 15	🟩	100.00%	10 / 10	🟩	100.00%	2 / 2
Book.php	🟩	100.00%	9 / 9	🟩	100.00%	6 / 6	🟩	100.00%	1 / 1
Chapter.php	🟩	100.00%	6 / 6	🟩	100.00%	4 / 4	🟩	100.00%	1 / 1

Legend

Low: 0% to 35% **Medium:** 35% to 70% **High:** 70% to 100%

Generated by PHP_CodeCoverage 2.0.15 using PHP 5.5.22 and PHPUnit 4.5.1 at Wed Apr 1 9:45:52 EDT 2015.

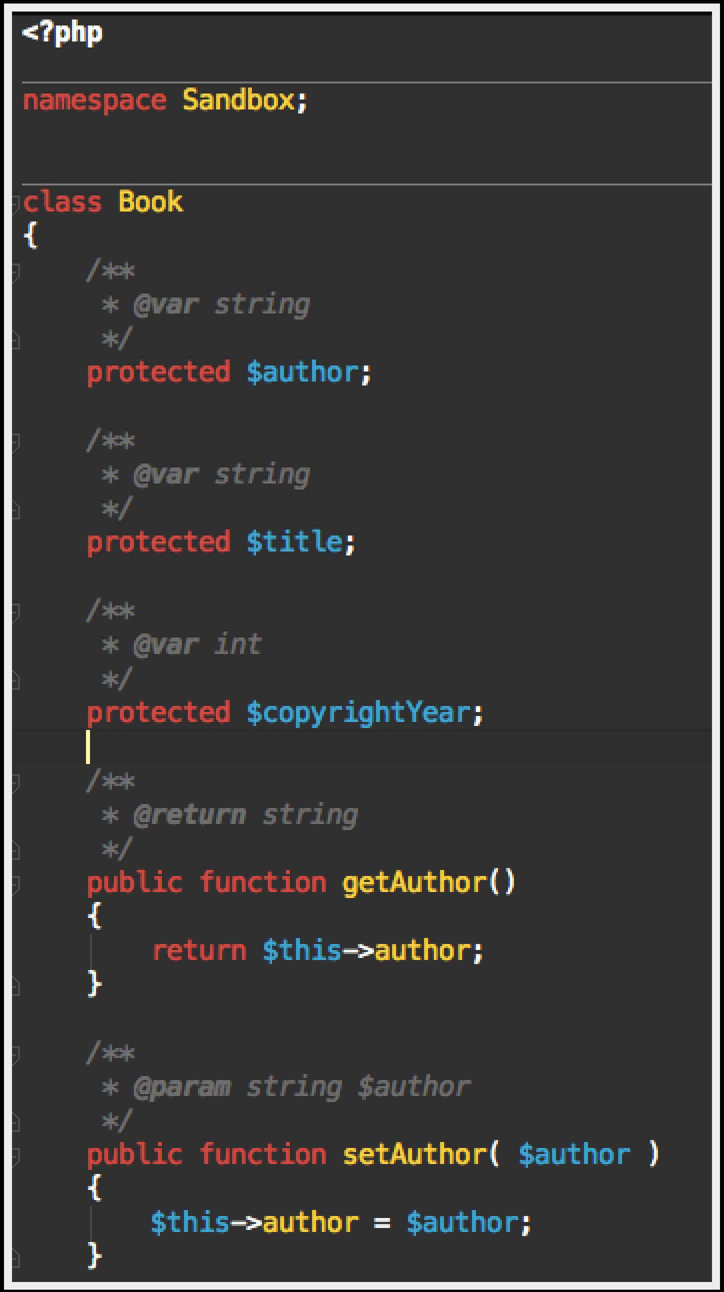


/Users/fraser/Development/CanvasPop/phpunit-sandbox/src/Sandbox / Book.php

	Code Coverage									
	Classes and Traits			Functions and Methods				Lines		
Total	🟩	100.00%	1 / 1	🟩	100.00%	6 / 6	CRAP	🟩	100.00%	9 / 9
Book	🟩	100.00%	1 / 1	🟩	100.00%	6 / 6	6	🟩	100.00%	9 / 9
getAuthor()				🟩	100.00%	1 / 1	1	🟩	100.00%	1 / 1
setAuthor($author)				🟩	100.00%	1 / 1	1	🟩	100.00%	2 / 2
getTitle()				🟩	100.00%	1 / 1	1	🟩	100.00%	1 / 1
setTitle($title)				🟩	100.00%	1 / 1	1	🟩	100.00%	2 / 2
getCopyrightYear()				🟩	100.00%	1 / 1	1	🟩	100.00%	1 / 1
setCopyrightYear($copyrightYear)				🟩	100.00%	1 / 1	1	🟩	100.00%	2 / 2

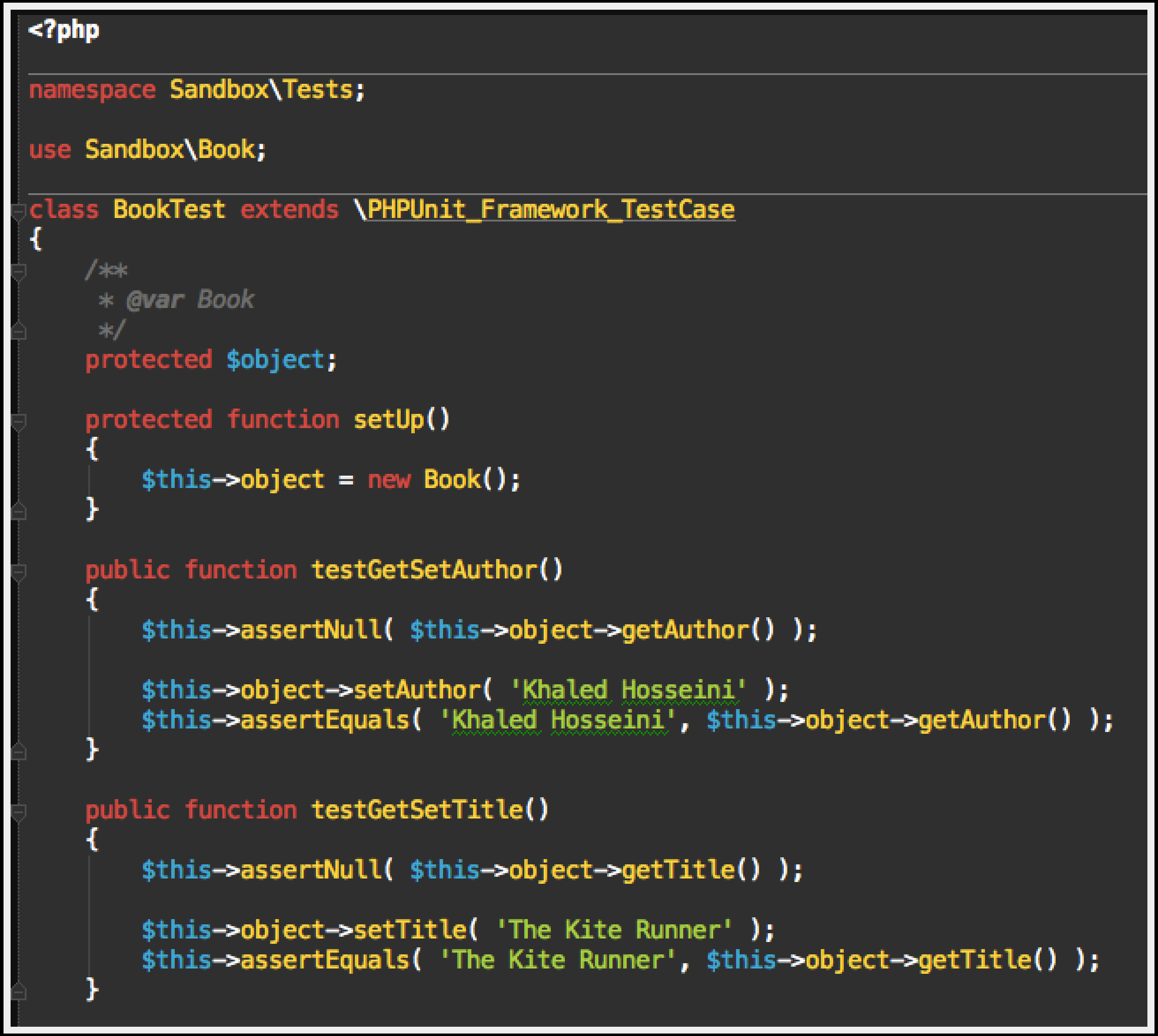
```
<?php

namespace Sandbox;


class Book
{
    /**
     * @var string
     */
    protected $author;
```

SIMPLE EXAMPLE

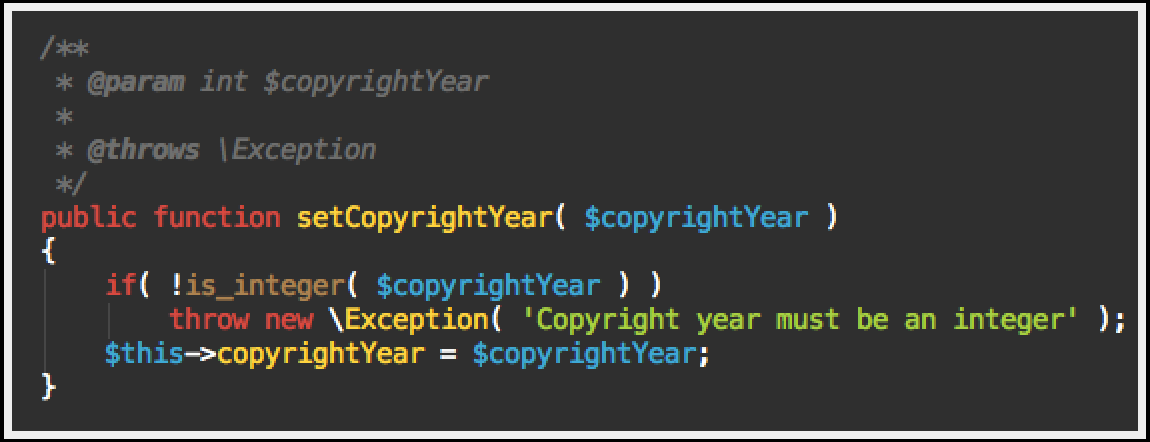


```php
<?php

namespace Sandbox;


class Book
{
    /**
     * @var string
     */
    protected $author;

    /**
     * @var string
     */
    protected $title;

    /**
     * @var int
     */
    protected $copyrightYear;

    /**
     * @return string
     */
    public function getAuthor()
    {
        return $this->author;
    }

    /**
     * @param string $author
     */
    public function setAuthor( $author )
    {
        $this->author = $author;
    }
```

Assertions

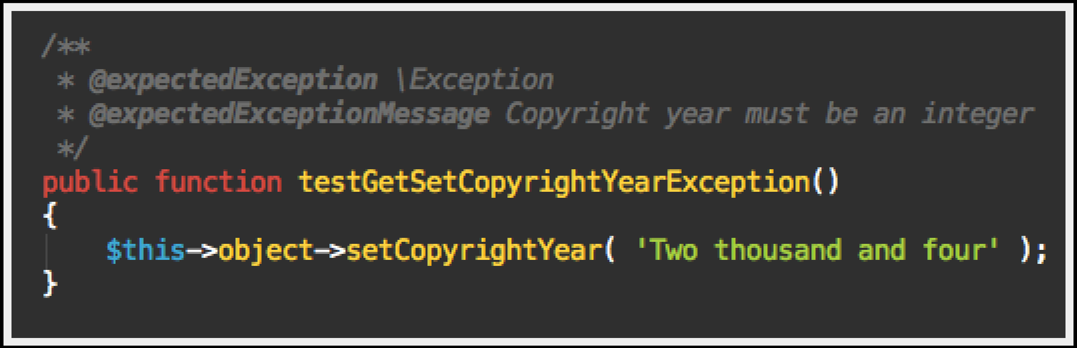


```php
<?php

namespace Sandbox\Tests;

use Sandbox\Book;

class BookTest extends \PHPUnit_Framework_TestCase
{
    /**
     * @var Book
     */
    protected $object;

    protected function setUp()
    {
        $this->object = new Book();
    }

    public function testGetSetAuthor()
    {
        $this->assertNull( $this->object->getAuthor() );

        $this->object->setAuthor( 'Khaled Hosseini' );
        $this->assertEquals( 'Khaled Hosseini', $this->object->getAuthor() );
    }

    public function testGetSetTitle()
    {
        $this->assertNull( $this->object->getTitle() );

        $this->object->setTitle( 'The Kite Runner' );
        $this->assertEquals( 'The Kite Runner', $this->object->getTitle() );
    }
```



```php
/**
 * @param int $copyrightYear
 *
 * @throws \Exception
 */
public function setCopyrightYear( $copyrightYear )
{
    if( !is_integer( $copyrightYear ) )
        throw new \Exception( 'Copyright year must be an integer' );
    $this->copyrightYear = $copyrightYear;
}
```

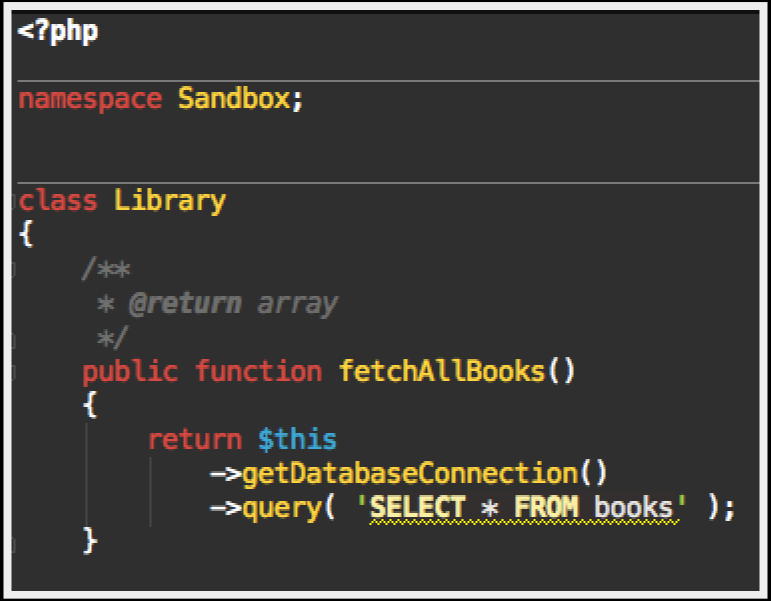
Testing for exceptions



```
/**
 * @expectedException \Exception
 * @expectedExceptionMessage Copyright year must be an integer
 */
public function testGetSetCopyrightYearException()
{
    $this->object->setCopyrightYear( 'Two thousand and four' );
}
```

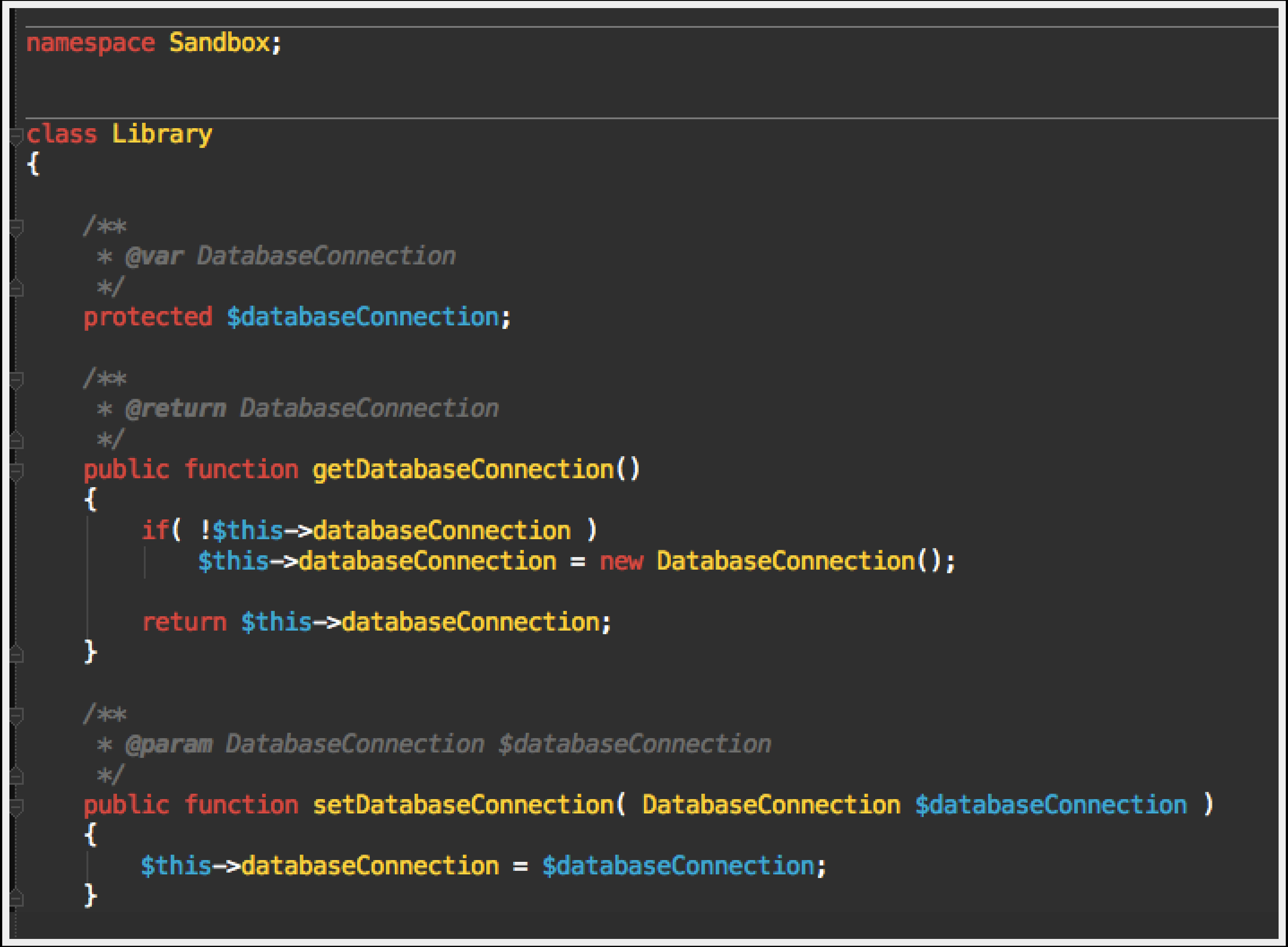
WORKING WITH RESOURCES

Class requires a database connection



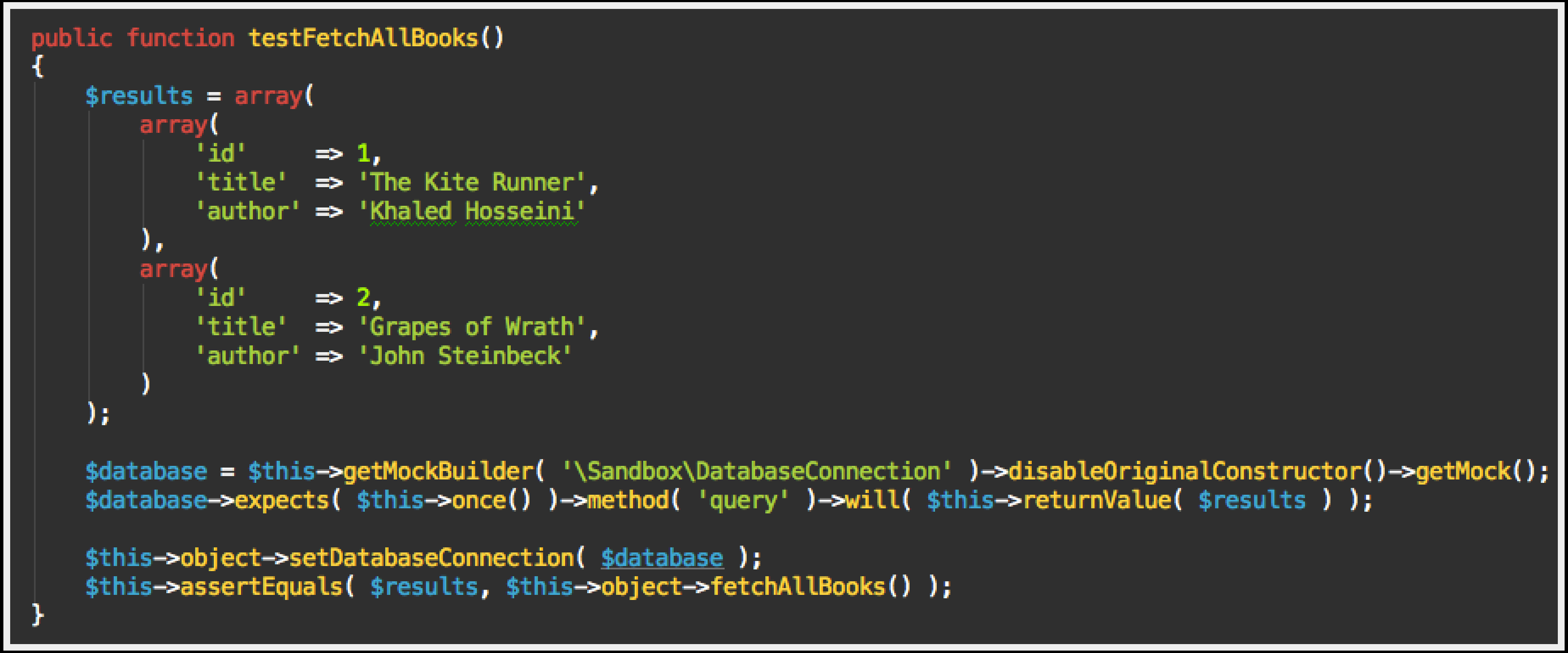
```php
<?php

namespace Sandbox;


class Library
{
    /**
     * @return array
     */
    public function fetchAllBooks()
    {
        return $this
            ->getDatabaseConnection()
            ->query( 'SELECT * FROM books' );
    }
```

Use dependency injection



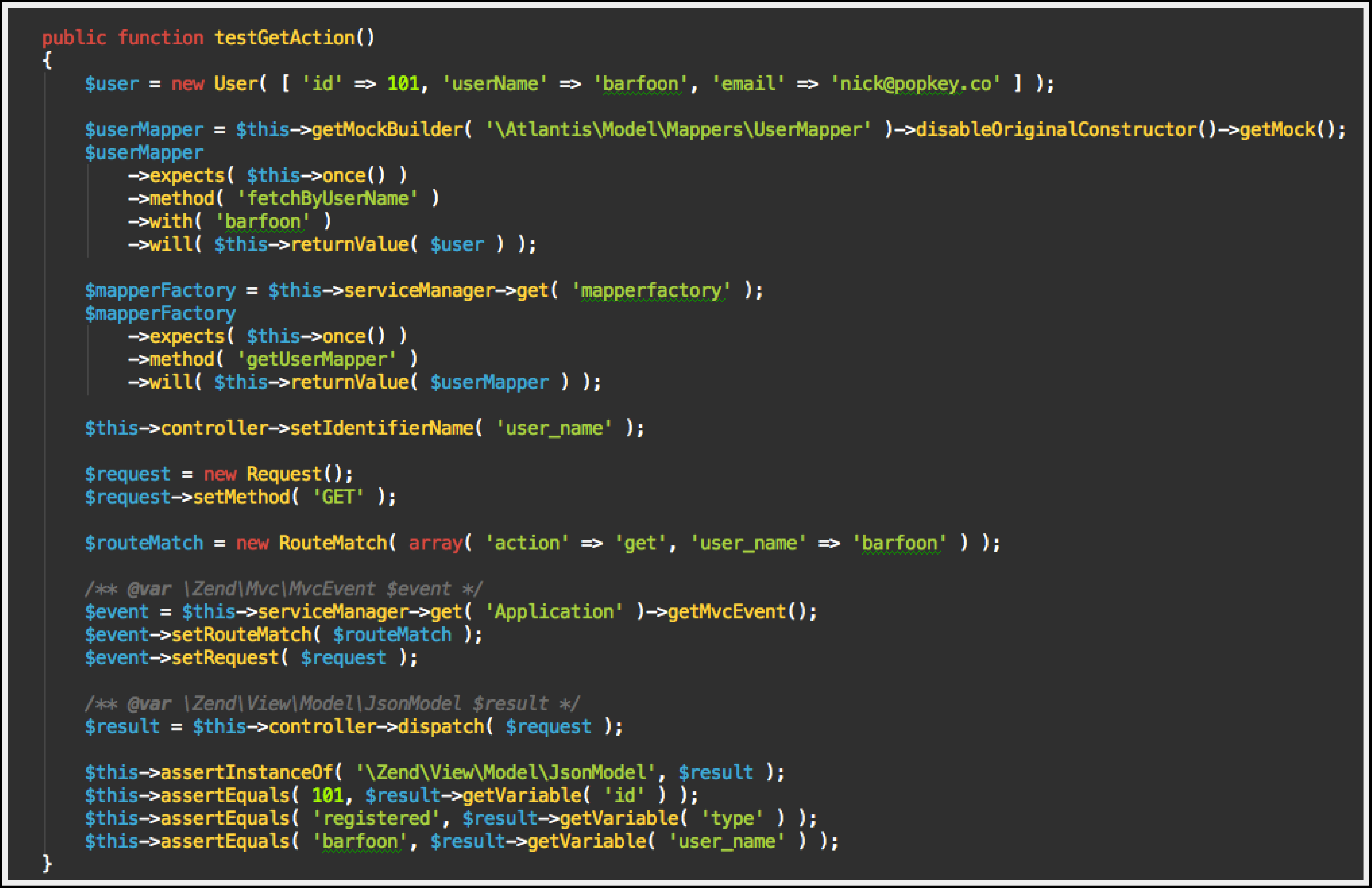
```php
namespace Sandbox;


class Library
{

    /**
     * @var DatabaseConnection
     */
    protected $databaseConnection;

    /**
     * @return DatabaseConnection
     */
    public function getDatabaseConnection()
    {
        if( !$this->databaseConnection )
            $this->databaseConnection = new DatabaseConnection();

        return $this->databaseConnection;
    }

    /**
     * @param DatabaseConnection $databaseConnection
     */
    public function setDatabaseConnection( DatabaseConnection $databaseConnection )
    {
        $this->databaseConnection = $databaseConnection;
    }
```

Mock resources



```php
public function testFetchAllBooks()
{
    $results = array(
        array(
            'id'     => 1,
            'title'  => 'The Kite Runner',
            'author' => 'Khaled Hosseini'
        ),
        array(
            'id'     => 2,
            'title'  => 'Grapes of Wrath',
            'author' => 'John Steinbeck'
        )
    );

    $database = $this->getMockBuilder( '\Sandbox\DatabaseConnection' )->disableOriginalConstructor()->getMock();
    $database->expects( $this->once() )->method( 'query' )->will( $this->returnValue( $results ) );

    $this->object->setDatabaseConnection( $database );
    $this->assertEquals( $results, $this->object->fetchAllBooks() );
}
```

EXTENDING THE CONCEPTS



```
public function testGetAction()
{
    $user = new User( [ 'id' => 101, 'userName' => 'barfoon', 'email' => 'nick@popkey.co' ] );

    $userMapper = $this->getMockBuilder( '\Atlantis\Model\Mappers\UserMapper' )->disableOriginalConstructor()->getMock();
    $userMapper
        ->expects( $this->once() )
        ->method( 'fetchByUserName' )
        ->with( 'barfoon' )
        ->will( $this->returnValue( $user ) );

    $mapperFactory = $this->serviceManager->get( 'mapperfactory' );
    $mapperFactory
        ->expects( $this->once() )
        ->method( 'getUserMapper' )
        ->will( $this->returnValue( $userMapper ) );

    $this->controller->setIdentifierName( 'user_name' );

    $request = new Request();
    $request->setMethod( 'GET' );

    $routeMatch = new RouteMatch( array( 'action' => 'get', 'user_name' => 'barfoon' ) );

    /** @var \Zend\Mvc\MvcEvent $event */
    $event = $this->serviceManager->get( 'Application' )->getMvcEvent();
    $event->setRouteMatch( $routeMatch );
    $event->setRequest( $request );

    /** @var \Zend\View\Model\JsonModel $result */
    $result = $this->controller->dispatch( $request );

    $this->assertInstanceOf( '\Zend\View\Model\JsonModel', $result );
    $this->assertEquals( 101, $result->getVariable( 'id' ) );
    $this->assertEquals( 'registered', $result->getVariable( 'type' ) );
    $this->assertEquals( 'barfoon', $result->getVariable( 'user_name' ) );
}
```

TAKEAWAYS

Start small, incrementally improve

Keep tests simple

Test for successes and failures

Strive for quality

QUESTIONS?

Fraser Reed

@fraser_reed

https://github.com/fraserreed