Testing your DB code with PHPUnit

Ade Slade

Why bother writing tests?

Why bother testing database interaction?

Overview

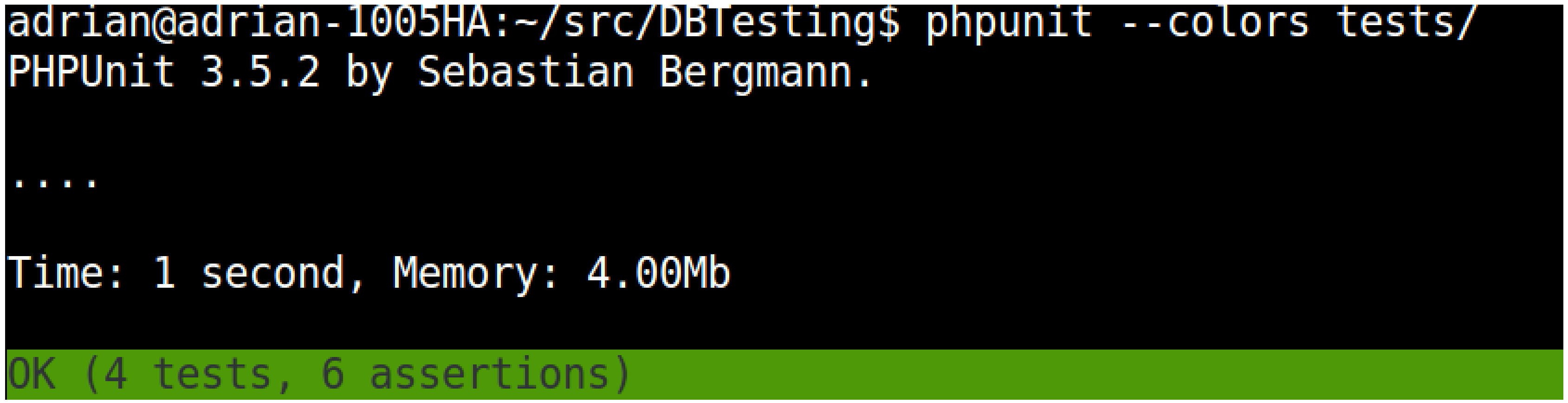
- Introduction
- Installation
- Getting started
- DataSets & DataTables
- Assertions
- Examples

Introduction

- DbUnit extension for PHPUnit
- Easy way to test code which interacts with the database by:
  - Putting the database into a known state before each test
  - Asserting the contents of the database are equal to the expected contents
- Internally uses PDO

PHPUnit

- Test Framework
- Member of the xUnit family of test frameworks
- Very marvellous
- > SimpleTest?



```
adrian@adrian-1005HA:~/src/DBTesting$ phpunit --colors tests/
PHPUnit 3.5.2 by Sebastian Bergmann.

....

Time: 1 second, Memory: 4.00Mb

OK (4 tests, 6 assertions)
```

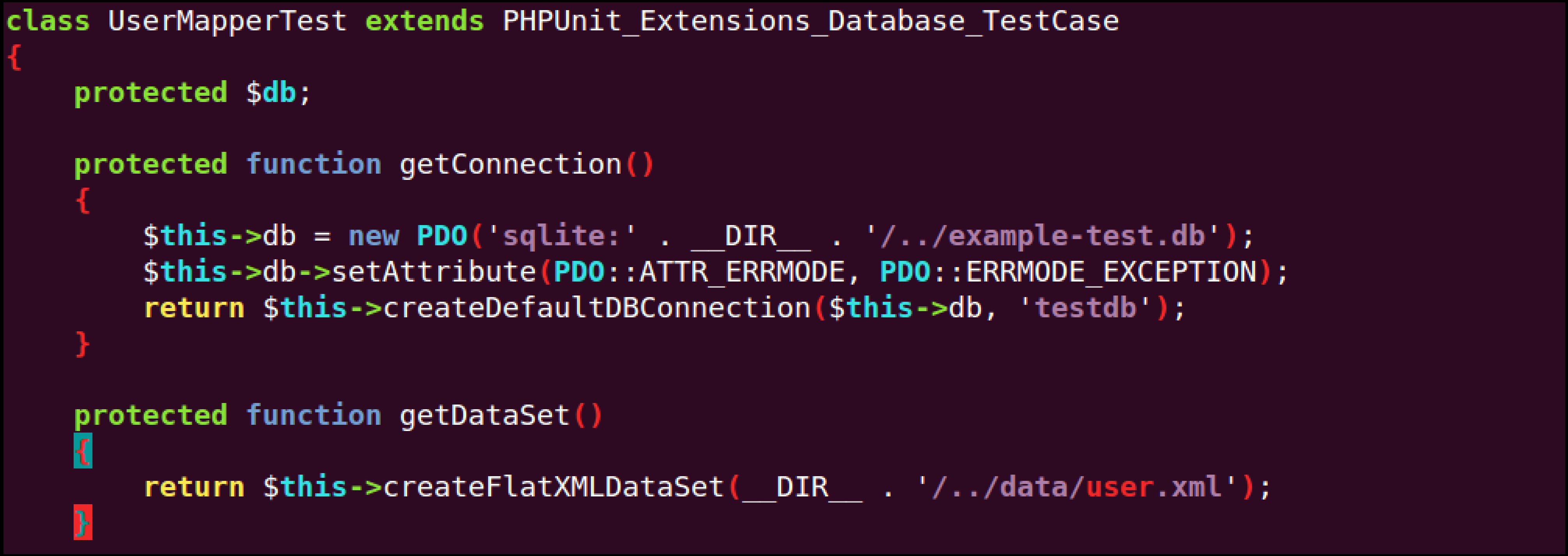
Installation

- pear channel-discover pear.phpunit.de
- pear install phpunit/phpunit
- pear install phpunit/dbunit

Getting started

- Extend `PHPUnit_Extensions_Database_TestCase`
- Requires two methods to be implemented
  - `getDataSet`
  - `getConnection`

Test Case

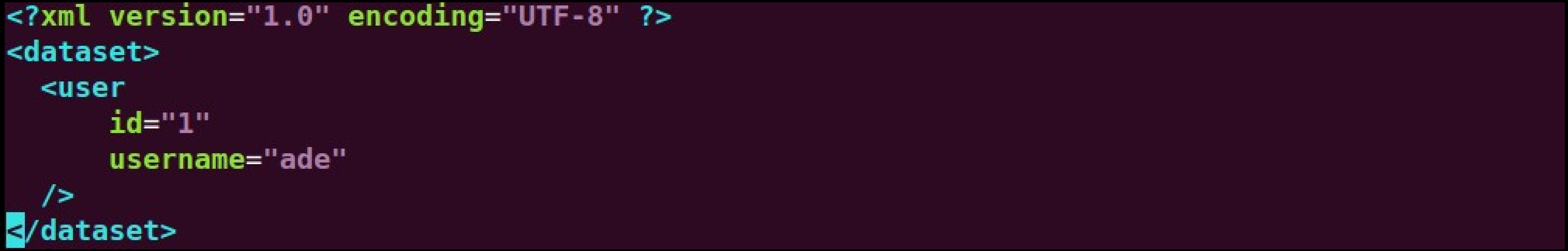


```php
class UserMapperTest extends PHPUnit_Extensions_Database_TestCase
{
    protected $db;

    protected function getConnection()
    {
        $this->db = new PDO('sqlite:' . __DIR__ . '/../example-test.db');
        $this->db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        return $this->createDefaultDBConnection($this->db, 'testdb');
    }

    protected function getDataSet()
    {
        return $this->createFlatXMLDataSet(__DIR__ . '/../data/user.xml');
    }
```

Flat XML DataSet



```xml
<?xml version="1.0" encoding="UTF-8" ?>
<dataset>
  <user
      id="1"
      username="ade"
  />
</dataset>
```

DataSets

- Can be in a variety of formats:
  - Flat XML
  - XML
  - Yaml
  - CSV
  - PHP array based data set
  - Or in whatever format you like...

More fun with DataSets:

- Composite
  - Aggregating datasets
- Replacement
  - Replace values in the DataSet
- Filter
  - Include and exclude tables and columns in a DataSet

DataSets and DataTables

- DataSet
  - Used to define known state
  - Set of tables
- DataTable
  - Individual table

Retrieving table state

- `$this->getConnection()->createDataSet()`
  - Optionally pass in an array of tablenames
- `$this->getConnection()->createQueryTable()`
- `PHPUnit_Extensions_Database_DataSet_QueryDataSet`

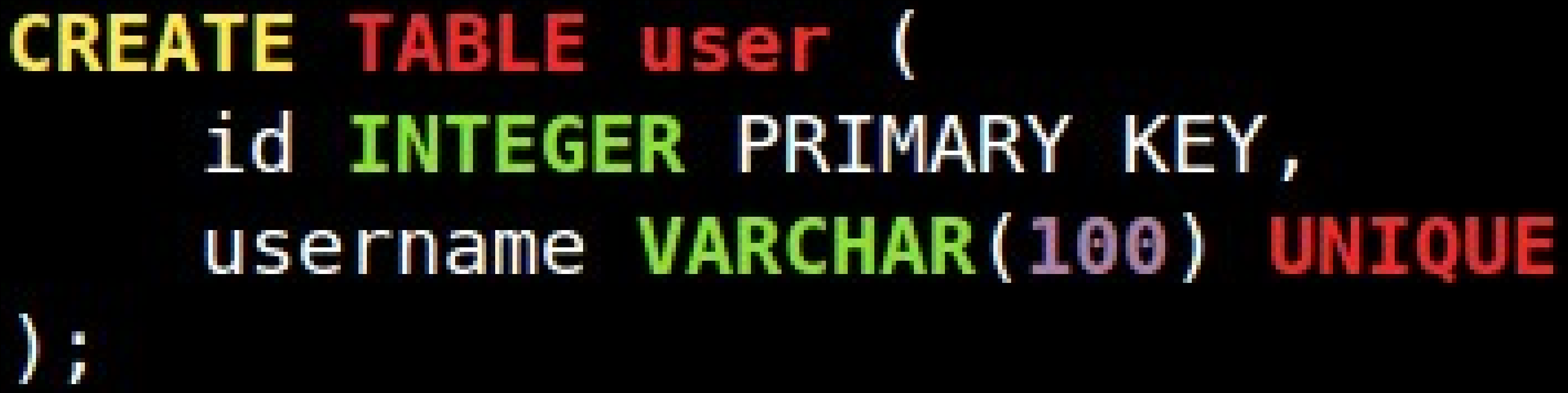
Assertions

- Verify state of the database
  - assertDataSetsEqual
- Verify state of a table
  - assertTablesEqual
- Asserting table row count
  - getRowCount

Writing some tests

- user table
- User object
- UserMapper to persist the User object

Schema



```sql
CREATE TABLE user (
    id INTEGER PRIMARY KEY,
    username VARCHAR(100) UNIQUE
);
```

User insertion test

```php
public function testInsertingUserInsertsUser()
{
    $userMapper = new UserMapper($this->db);
    $user = new User();
    $user->setUsername('terry_tibbs');
    $userMapper->insert($user);
    $this->assertEquals(2, $this->getConnection()->getRowCount('user'));

    $expected = $this->createFlatXmlDataSet(__DIR__ . "/../data/user-insert.xml");

    $actual = new PHPUnit_Extensions_Database_DataSet_QueryDataSet($this->getConnection());
    $actual->addTable('user');

    $this->assertDataSetsEqual($expected, $actual);
}
```

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<dataset>
  <user
      id="1"
      username="ade"
  />
  <user
      id="2"
      username="terry_tibbs"
  />
</dataset>
```

User update test

```php
public function testUpdateUser()
{
    $userMapper = new UserMapper($this->db);
    $user = $userMapper->findById(1);
    $user->setUsername('jeff');
    $userMapper->update($user);

    $expectedTable = $this->createFlatXmlDataSet(__DIR__ . "/../data/user-update.xml")->getTable('user');
    $actualTable = $this->getConnection()->createQueryTable('user', 'SELECT * FROM user');

    $this->assertTablesEqual($expectedTable, $actualTable);
}
```

```xml
?xml version="1.0" encoding="UTF-8" ?>
dataset>
  <user
      id="1"
      username="jeff"
  />
/dataset>
```

User removal test

```php
public function testRemovingUserRemovesUser()
{
    $userMapper = new UserMapper($this->db);
    $user = $userMapper->findById(1);
    $userMapper->delete($user);

    $this->assertEquals(0, $this->getConnection()->getRowCount('user'));
}
```

Asserting the result of a query

```php
public function testComplexQuery()
{
    $queryTable = $this->getConnection()->getQueryTable(
        'myComplexQuery', 'SELECT complexQuery ... '
    );

    $expectedTable = $this->createFlatXmlDataSet('complexQueryAssertion.xml')
                          ->getTable('myComplexQuery');
    $this->assertTablesEqual($expectedTable, $queryTable);
}
```

Summary

Resources

- PHPUnit manual
  - http://www.phpunit.de/manual/current/en/database.html
- Benjamin Eberlei's tutorial
  - http://www.beberlei.de/dbunit.html
- Mike Lively's (author's) blog
  - http://www.digitalsandwich.com
- Sebastian Bergmann's slides on “Testing PHP/MySQL Applications with PHPUnit/DbUnit”
  - On slideshare
- Sebastian Bergmann's slides on “Testing LAMP applications”
  - On slideshare

Yay! Finished

- Thanks for listening
- Appreciate any and all feedback :)

- email: adeslade@gmail.com
- twitter: @adeslade
- web: http://adeslade.co.uk
- github: https://github.com/adeslade/DBTesting