

# Testen State of the Art

IPC Spring 2012

Manuel Pichler (@manuelp)  
Tobias Schlitt (@tobySen)  
2012-06-04

- ▶ **Manuel & Toby**
- ▶ Degree in computer science
- ▶ More than 10 years of professional PHP
- ▶ Open source enthusiasts
- ▶ Contributing to various FLOSS projects

Co-founders of

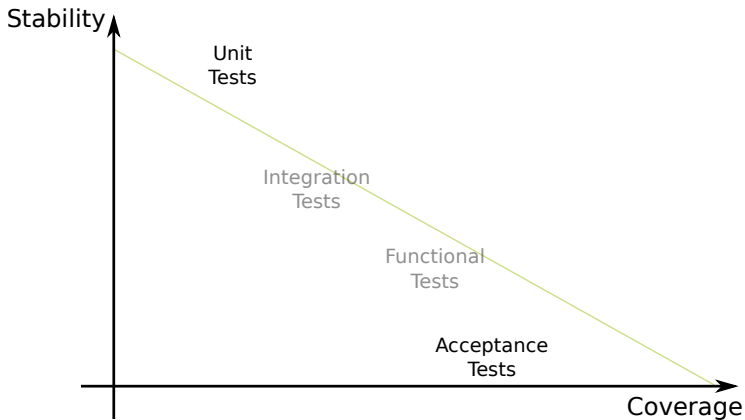


**Helping people to create  
high quality web  
applications.**

<http://qafoo.com>

# Instability vs. Coverage

---



# Outline

---

## Unit Testing

Testing File Uploads

Behavior Driven Development

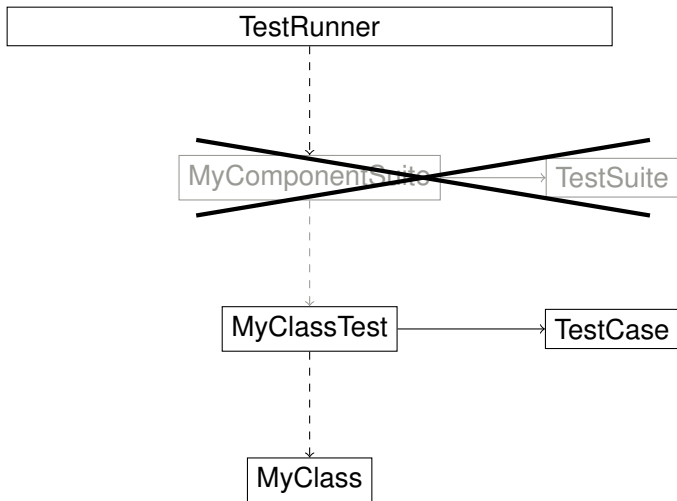
Automated Acceptance Tests

Testing JavaScript

Outlook

# PHPUnit schematics

---



# Test suites

---

- ▶ File system based
- ▶ phpunit.dist.xml / phpunit.xml
- ▶ @group annotations

- ▶ In the dir you execute `$ phpunit`
  - ▶ `phpunit.xml` available?
  - ▶ `phpunit.dist.xml` available?
- ▶ `$ phpunit -c my-phpunit.xml`

# phpunit.xml

---

```
1 <phpunit bootstrap="libraries/autoload.php"  
2     colors="true"  
3     convertErrorsToExceptions="true"  
4     convertNoticesToExceptions="true"  
5     convertWarningsToExceptions="true"  
6     >  
7 <groups>  
8     <exclude><group>integration</group></exclude>  
9 </groups>  
10  
11 <testsuites>  
12     <testsuite name="All_Tests">  
13         <directory suffix=".php">tests</directory>  
14     </testsuite>  
15 </testsuites>  
16  
17 <filter>  
18     <whitelist>  
19         <directory suffix=".php">../src</directory>  
20     <exclude>  
21         <file>../src/bootstrap.php</file>  
22     </exclude>  
23     </whitelist>  
24 </filter>  
25 </phpunit>
```



# Useful Annotations

---

- ▶ `@covers`
- ▶ `@group`
- ▶ `@ticket`
- ▶ `@dataProvider`
- ▶ `@expectedException*`
- ▶ `@codeCoverageIgnore*`
- ▶ `@depends`

# Outline

---

Unit Testing

**Testing File Uploads**

Behavior Driven Development

Automated Acceptance Tests

Testing JavaScript

Outlook

# A Minimal File Upload

---

```
1 <?php
2 class UploadExample
3 {
4     protected $dest;
5     public function __construct($dest)
6     {
7         $this->dest = rtrim($dest, '/') . '/';
8     }
9     public function handle($name)
10    {
11        if (is_uploaded_file($_FILES[$name]['tmp_name'])) {
12            move_uploaded_file(
13                $_FILES[$name]['tmp_name'],
14                $this->dest . $_FILES[$name]['name']
15            );
16        }
17    }
18 }
```

- ▶ PHP Testing Framework (PHPT)
- ▶ Used to test PHP core
- ▶ `$ pear run-tests example-test-case.phpt`
- ▶ Integrates into PHPUnit
- ▶ Details: [http://qa.php.net/phpt\\_details.php](http://qa.php.net/phpt_details.php)

# A Simple PHPT Test

---

```
1  --TEST--
2  Example test case
3  --FILE--
4  <?php
5  var_dump(strpos('Manuel_Pichler', 'P'));
6  var_dump(strpos('Manuel_Pichler', 'Z'));
7  --EXPECT--
8  int(7)
9  bool(false)
```

# File Uploads with PHPT

---

```
1  --TEST--
2  Example test emulating a file upload
3  --POST_RAW--
4  Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryfywL8UCjFtqUBTQn
5
6  -----WebKitFormBoundaryfywL8UCjFtqUBTQn
7  Content-Disposition: form-data; name="file"; filename="example.txt"
8  Content-Type: text/plain
9
10  Qafoo provides quality assurance support and consulting
11
12  -----WebKitFormBoundaryfywL8UCjFtqUBTQn
13  Content-Disposition: form-data; name="submit"
14
15  Upload
16  -----WebKitFormBoundaryfywL8UCjFtqUBTQn--
17  --FILE--
18  <?php
19  require __DIR__ . '/UploadExample.php';
20
21  $upload = new UploadExample('/tmp');
22  $upload->handle('file');
23
24  var_dump(file_exists('/tmp/example.txt'));
25  ?>
26  --EXPECT--
27  bool(true)
```

# PHPT within PHPUnit

---

```
1 <phpunit>
2   <testsuites>
3     <testsuite name="PHPT">
4       <directory suffix=".pht" >./</directory>
5     </testsuite>
6   </testsuites>
7 </phpunit>
```

# Outline

---

Unit Testing

Testing File Uploads

**Behavior Driven Development**

Automated Acceptance Tests

Testing JavaScript

Outlook



# Behavior Driven Development

---

- ▶ Behavior Driven Development (BDD)
- ▶ Deeper coupling between
  - ▶ Business Analysis
  - ▶ Quality Assurance
- ▶ Basic ideas:
  - ▶ Use cases as requirements
  - ▶ Make use cases executable
  - ▶ Verify functionality
- ▶ Advantages:
  - ▶ Communication
  - ▶ Documentation
- ▶ Short **Human readable tests**

# The BDD process

---

1. Define a Domain Specific Language (DSL)
2. Describe Use Cases
3. Implement against Use Case execution

# Behat

---

- ▶ BDD framework for PHP
- ▶ <http://behat.org/>
- ▶ Inspired by Ruby's Cucumber
- ▶ Getting started:  
[http://docs.behat.org/quick\\_intro.html](http://docs.behat.org/quick_intro.html)

# Defining Features

---

```
1 Feature:
2   BDD tests for the eZ Publish Section API
3
4   Scenario: Admin user can delete a section
5     Given I have a section with name "MySection" and identifier "MySectionKey"
6       And I am logged in as user "admin"
7       When I delete section with identifier "MySectionKey"
8       Then I expect no section for identifier "MySectionKey" exists
9
10  Scenario: Anonymous user cannot delete a section
11    Given I have a section with name "MySection" and identifier "MySectionKey"
12      And I am logged in as user "anonymous"
13      When I delete section with identifier "MySectionKey"
14      Then I expect delete failes with an exception of type "unauthorized"
```

# Implementing the DSL

---

```
16 class FeatureContext extends BehatContext
17 {
48 /**
49  * @Given /^ I have a section with name "([^"]+)" and identifier "([^"]+)"$/
50  */
51 public function iHaveASectionWithNameAndIdentifier( $name, $identifier )
52 {
53     $sectionService = $this->repository->getSectionService();
54
55     $sectionCreate      = $sectionService->newSectionCreateStruct();
56     $sectionCreate->name      = $name;
57     $sectionCreate->identifier = $identifier;
58
59     $this->runtimeData[$identifier] = $sectionService->createSection( $sectionCreate
60     );
61 }
112 }
```

# When ...

---

```
1 Feature:
2   BDD tests for the eZ Publish Section API
3
4   Scenario: Admin user can delete a section
5     Given I have a section with name "MySection" and identifier "MySectionKey"
6       And I am logged in as user "admin"
7       When I delete section with identifier "MySectionKey"
8       Then I expect no section for identifier "MySectionKey" exists
9
10  Scenario: Anonymous user cannot delete a section
11    Given I have a section with name "MySection" and identifier "MySectionKey"
12      And I am logged in as user "anonymous"
13      When I delete section with identifier "MySectionKey"
14      Then I expect delete failes with an exception of type "unauthorized"
```

# Implementing the DSL

---

```
16 class FeatureContext extends BehatContext
17 {
    69 /**
    70  * @When /^I delete section with identifier "([^"]+)"/
    71  */
    72 public function iDeleteSectionWithIdentifier( $identifier )
    73 {
    74     $sectionService = $this->repository->getSectionService();
    75
    76     try
    77     {
    78         $section = $sectionService->loadSectionByIdentifier( $identifier );
    79         $sectionService->deleteSection( $section );
    80     }
    81     catch ( \Exception $e )
    82     {
    83         $this->runtimeData[ '--exception ' ] = $e;
    84     }
    85 }
    112 }
```

# Then ...

---

```
1 Feature:
2   BDD tests for the eZ Publish Section API
3
4   Scenario: Admin user can delete a section
5     Given I have a section with name "MySection" and identifier "MySectionKey"
6       And I am logged in as user "admin"
7       When I delete section with identifier "MySectionKey"
8       Then I expect no section for identifier "MySectionKey" exists
9
10  Scenario: Anonymous user cannot delete a section
11    Given I have a section with name "MySection" and identifier "MySectionKey"
12      And I am logged in as user "anonymous"
13      When I delete section with identifier "MySectionKey"
14      Then I expect delete failes with an exception of type "unauthorized"
```



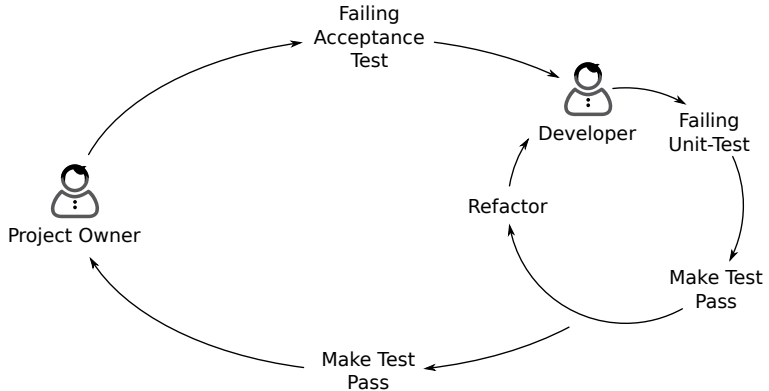
# Implementing the DSL

---

```
16 class FeatureContext extends BehatContext
17 {
    87 /**
    88  * @Then /^I expect no section for identifier "[^"]+" exists$/
    89  */
    90 public function iExpectNoSectionForIdentifierExists( $identifier )
    91 {
    92     $sectionService = $this->repository->getSectionService();
    93     try
    94     {
    95         $sectionService->loadSectionByIdentifier( $identifier );
    96         throw new \Behat\Behat\Exception\PendingException( "Identifier _{$identifier}_
    still _exists._" );
    97     }
    98     catch ( \eZ\Publish\API\Repository\Exceptions\NotFoundException $e )
    99     {
    100     }
    101 }
    112 }
```

# Double Feedback Test Cycle

---



# Outline

---

Unit Testing

Testing File Uploads

Behavior Driven Development

**Automated Acceptance Tests**

Testing JavaScript

Outlook

# Automated Acceptance Tests

---

- ▶ High level system tests
- ▶ Assert user expectations
- ▶ Good start for refactoring
- ▶ Approaches
  - ▶ HTTP spider (e.g. Goutte)
  - ▶ Browser controller (e.g. Selenium, Sahi)
  - ▶ Browser emulator (e.g. Zombie.js)

# Mink

---

- ▶ Acceptance Test Framework
- ▶ <http://mink.behat.org>
- ▶ Abstraction for underlying tools
- ▶ Full integration into Behat

# Testing a Registration Form

---

```
1 Feature: Registration
2   In order to perform actions like an order
3   As a normal user
4   I must perform a registration
5
6   Scenario: Test a default registration
7     Given I am on "/register"
8     When I select "private" from "register_personal_customer_type"
9       And I select "mr" from "register_personal_salutation_mr"
10      And I fill in "firstname" with "Tobias"
11      And I fill in "lastname" with "Schlitt"
12      And I fill in "register_personal_email" with "ts01@qafoo.com"
13      And I fill in "register_personal_password" with "sindelfingen"
14      And I fill in "register_personal_passwordConfirmation" with "sindelfingen"
15      And I fill in "phone" with "+49 (0)209 40501252"
16      And I fill in "street" with "Bochumer Strasse"
17      And I fill in "streetnumber" with "226"
18      And I fill in "zipcode" with "45886"
19      And I fill in "city" with "Gelsenkirchen"
20      And I select "2" from "country"
21      And I press "registerbutton"
22      Then I should see "Willkommen, Tobias Schlitt"
```

# Outline

---

Unit Testing

Testing File Uploads

Behavior Driven Development

Automated Acceptance Tests

**Testing JavaScript**

Outlook

# Testing JavaScript

---

- ▶ You test PHP ...
- ▶ ...do you test JavaScript?



# js-test-driver

---

- ▶ Test runner for JavaScript
- ▶ <https://code.google.com/p/js-test-driver/>
- ▶ Test JS inside browsers
- ▶ Run JS tests from command line
- ▶ No HTML required

# Simple JS Code

---

```
1 var Hello = function() {};  
2 Hello.prototype = {  
3   say: function( who ) {  
4     if ( who === undefined ) {  
5       who = "World";  
6     }  
7     return "Hello_" + who + "!";  
8   }  
9 }
```

# A Simple Test Case

---

```
1  TestCase( "Hello_Object_tests", {
2    "setUp": function () {
3      this.hello = new Hello();
4    },
5    "testCallableWithoutArgument": function() {
6      assertEquals( "Hello_World!", this.hello.say() );
7    },
8    "testCallableWithArgument": function() {
9      assertEquals( "Hello_Jakob!", this.hello.say( "Jakob" ) );
10   }
11 });
```

# Configuring js-test-driver

---

```
1 server: http://localhost:4223
2
3 load:
4   - src/hello.js
5   - tests/hello.js
```

# Running the Tests

---

- ▶ First shell

```
1 $ java -jar JsTestDriver-1.3.4.b.jar --port 4223
```

- ▶ Capture browser

- ▶ `http://localhost:4223/capture`

- ▶ Second shell

```
1 $ java -jar JsTestDriver-1.3.4.b.jar --tests all
2 setting runnermode QUIET
3 ....
4 Total 4 tests (Passed: 4; Fails: 0; Errors: 0) (1.00 ms)
5 Firefox 12.0 Linux: Run 2 tests (Passed: 2; Fails: 0; Errors 0) (1.00 ms)
6 Chrome 18.0.1025.168 Linux: Run 2 tests (Passed: 2; Fails: 0; Errors 0) (0.00 ms)
```

# Outline

---

Unit Testing

Testing File Uploads

Behavior Driven Development

Automated Acceptance Tests

Testing JavaScript

Outlook

# Unified Build Process

---

- ▶ <https://github.com/manuelpichler/build-commons>
- ▶ Framework for build scripts
- ▶ Unified targets, like
  - ▶ test
  - ▶ package
  - ▶ verify
  - ▶ deploy
- ▶ 39 supported tool extensions
  - ▶ PHPUnit, CodeSniffer, PDepend, PHPMD, PHPCPD, ...
  - ▶ DB, DBDeploy, PHAR, Pirum, ...
  - ▶ js-test-driver, JSLint, JSHint, ...
  - ▶ ...



THANK YOU

Rent a quality expert  
[qafoo.com](http://qafoo.com)