

# Einfaches Testen von Webapplikationen mit SimpleTest

# Agenda

- Unit Tests/Web-Tests
- Durchführung von Web-Tests
- HTML Elemente
- Ergebnisdarstellung
- Gruppierung
- Grenzen von SimpleTest
- Alternativen

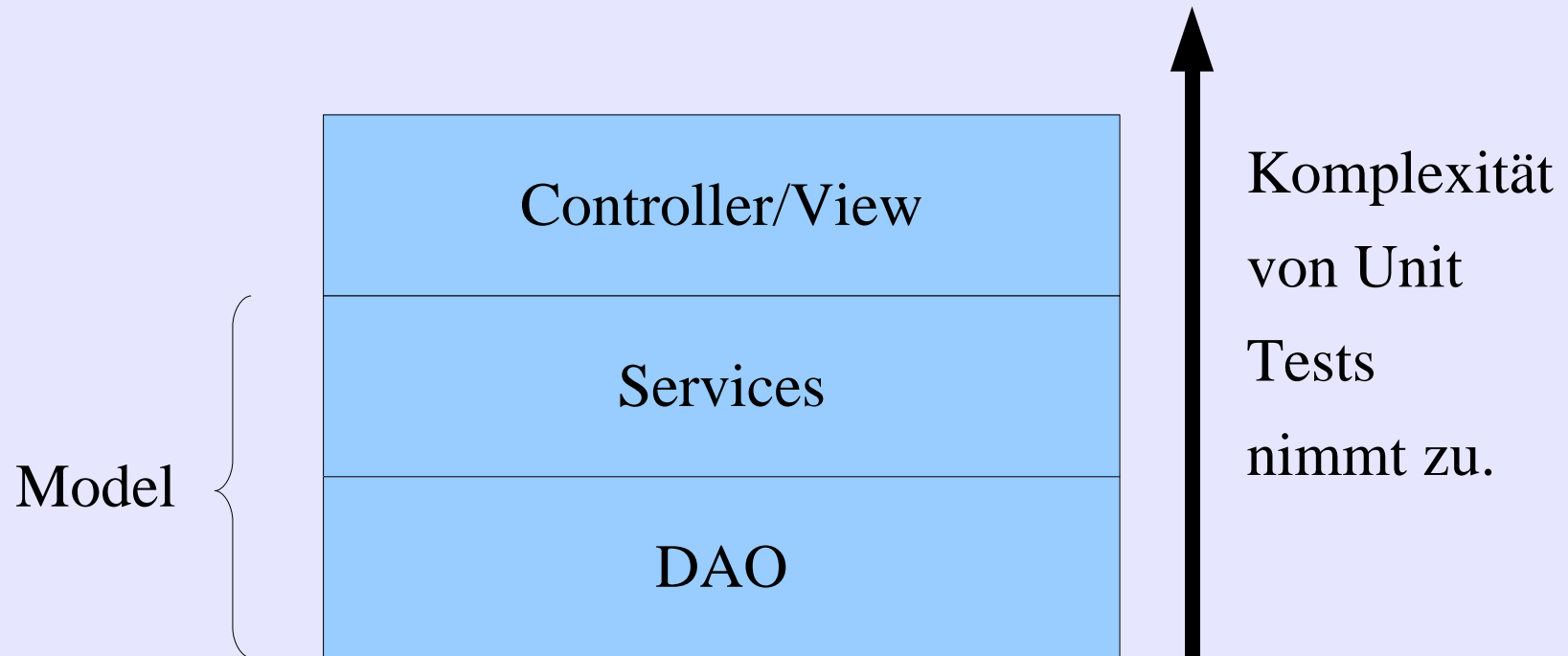
# Warum überhaupt testen?

- Es funktioniert nicht alles so wie man es erwartet.
  - Konsequenz:
    - Es muss geprüft werden!

# Unit Tests

- Aufgabe von Unit Tests:
  - Entwickler Tests
  - Extreme Programming Ansatz
  - Zuerst den Test schreiben, dann die Implementation

# Unit Tests



# Warum Web Tests?

- Unit Tests haben Grenzen
  - Grafische Benutzer Oberflächen (GUI)
    - Model-View-Controller (MVC) Strukturen
      - Model Ok
      - View + Controller mithilfe anderer Frameworks

# Warum Web Tests?

- Unit Tests haben Grenzen
  - Nebenläufiger Code
  - Mehrschichten Architektur
    - Mock Objects (Test Pattern)

# Warum Web Tests?

- Logik Testen, die mit Unit Tests nicht abgedeckt werden kann.
  - z.B. Workflow etc. der durch Controller/View dargestellt wird.
  - Test aus Sicht des Anwenders.
  - Eventuell einfacher als C/V Test Framework.

# Durchführung Web Tests

- Von Hand
  - Vorteil:
    - Einfach.
  - Nachteil:
    - Nicht Reproduzierbar.
    - Sehr aufwendig.

# Durchführung Web Tests

- Mithilfe eines Werkzeuges
  - Vorteil:
    - Reproduzierbar
    - Ablaufzeit ist kalkulierbar
    - Integration in den Entwicklungsprozess

# Durchführung Web Tests

- Mithilfe eines Werkzeuges
  - Nachteil:
    - Je nach Entwicklungsprozess aufwendiger zu erstellen.
    - Einarbeitung.

# Durchführung Web Tests

- Kommerzielle Werkzeuge
  - eTest Suite von empirix
  - SilkTest von segue
  - WinRunner mercury
  - etc.

# Durchführung Web Tests

- Kommerzielle Werkzeuge
  - Vorteile:
    - GUI
      - Nutzung durch nicht technisches Personal
    - Je nach Projektgröße/Struktur
      - Kosteneinsparung

# Durchführung Web Tests

- Kommerzielle Werkzeuge
  - Nachteile:
    - Anschaffungskosten (Investitionskosten)
    - Einarbeitungs- bzw. Schulungsaufwand
    - Portabilität

# Durchführung Web Tests

- OpenSource Werkzeuge
  - HTMLUnit
  - JWebUnit
  - HTTPUnit
  - CanooWebTest
  - FIT
  - etc.

# Durchführung Web Tests

- OpenSource Werkzeuge
  - Vorteile:
    - Meist eine einfache Integration in den Entwicklungsprozess.
    - Frühere Rückmeldung bzgl. der fachlichen Anforderungen.

# Durchführung Web Tests

- OpenSource Werkzeuge
  - Nachteile:
    - Erstellung der Testfälle ist je nach Tool mehr technisch als fachlich orientiert (teilweise programmiert).
    - Teilweise keine GUI.

# Testplan

- 1. Test
  - Seite aufrufen
    - Prüfen, ob auch alle notwendigen Angaben auf der Seite vorhanden sind.
  - ...
- ...

# 1. Test mit SimpleTest

**PAwAs**

**Projekt Aufwanderfassungs- und Abrechnungssystem**

Anmeldung

Mandant-Nr:   
Anmeldung:   
Passwort:

PAwAs V0.50.1 Revision \$Rev: 25 \$ © Karl Heinz Marbaise

# 1. Test mit SimpleTest

```
require_once('simpletest/web_tester.php');
require_once('simpletest/reporter.php');
class LoginTest extends WebTestCase {
    function testTitle() {
        $this->get('http://pawas.wanderer');
        $this->assertTitle('» PAwAs « ::: Projekt Aufwanderfassungs- und
            Abrechnungssystem ::: PAwAs V0.50.1');
    }
}
$test = &new LoginTest();
$test->run(new HtmlReporter());
```

# 1. Test mit SimpleTest

- Das Ergebnis:

**LoginTest**|

1/1 test cases complete: 1 passes, 0 fails and 0 exceptions.

# 2. Test mit SimpleTest

- Sind alle Eingabe Felder vorhanden?

```
class LoginUserPasswordTest extends WebTestCase {  
    function testUserPassword() {  
        $this->get('http://pawas.wanderer');  
        $this->assertField('iClient', '');  
        $this->assertField('sUserName', '');  
        $this->assertField('sPassword', '');  
    }  
}
```

# 3. Test mit SimpleTest

## Eingabe Felder füllen und „Clicken“

```
class LoginMitUserPasswordTest extends WebTestCase {  
    function testUserPassword() {  
        $this->get('http://pawas.wanderer');  
        $this->setField('iClient', 1);  
        $this->setField('sUserName', 'kama');  
        $this->setField('sPassword', 'egon');  
        $this->click('Anmelden');  
    }  
}
```

# 4. Test mit SimpleTest

## Kombination prüfen und füllen

```
class LoginTestKombination extends WebTestCase {
    function testUserPassword() {
        $this->get('http://pawas.wanderer');
        $this->assertTrue($this->setField('iClient', 1));
        $this->assertTrue($this->setField('sUserName', 'kama'));
        $this->assertTrue($this->setField('sPassword', 'egon'));
        $this->click('Anmelden');
    }
}
```

# Unit Tests / Web Tests

- Welche Methoden werden als Testmethoden angesehen und ausgeführt?
  - Alle Methoden, die mit „test“ beginnen.

# Unit Tests / Web Tests

- Welche Methoden werden als Testmethoden angesehen und ausgeführt?
  - Alle Methoden, die mit „test“ beginnen.

# 5. Test mit SimpleTest

```
class LoginPageTest extends WebTestCase {  
    function setUp() {  
        $this->get('http://pawas.wanderer');  
    }  
    function testPageTitle() {  
        $this->assertTitle('» PAwAs « ::: Projekt ...');  
    }  
    function testLoginFields() {  
        $this->assertTrue($this->setField('iClient', 1));  
        $this->assertTrue($this->setField('sUserName', 'kama'));  
        $this->assertTrue($this->setField('sPassword', 'egon'));  
        $this->click('Anmelden');  
    }  
}
```

# Testplan

- ...
- 2. Test
  - Anmeldung durchführen
    - Prüfen, ob auch alle notwendigen Angaben auf der Seite vorhanden sind.
  - ...
- ...

# Anmeldung und Prüfung

```
class LoginPageTest extends WebTestCase {  
    ....  
    function testLoginFields() {  
        $this->assertTrue($this->setField('iClient', 1));  
        $this->assertTrue($this->setField('sUserName', 'kama'));  
        $this->assertTrue($this->setField('sPassword', 'egon'));  
        $this->click('Anmelden');  
        $this->assertLink('Abmelden');  
        $this->assertLink('Kunden');  
        $this->assertLink('Projektliste');  
        $this->assertLink('Aufgaben');  
        $this->assertLink('Arbeitszeit');  
        ...  
    }  
}
```

# Testplan

- ...
- 3. Test
  - Anmeldung mit falschen Daten
    - Prüfen, ob auch alle notwendigen Angaben auf der Seite vorhanden sind. Sprich Fehlermeldungen etc.
  - ...

# Formen

- Mehrere Formen auf einer Seite:
  - `SubmitFormById('FormId');`
- Keine direkte Auswahl der Formen möglich.

# HTML: DropDown Listen

...

```
<select name="auswahl">  
  <option value="0">Heino</option>  
  <option value="1">Michael Jackson</option>  
  <option value="2">Tom Waits</option>  
  <option value="3">Nina Hagen</option>  
  <option value="4">Marianne Rosenberg</option>  
</select>
```

...

# HTML: DropDown Listen

```
class DropDownBoxTest extends WebTestCase {  
    ....  
    function testUserPassword() {  
        $this->get('http://simpletest.wanderer/dropdownbox.php');  
        $this->assertTrue($this->setField('auswahl', 1));  
        $this->clickSubmit('Favorit');  
        $this->assertText('Es wurde der Favorit: Michael Jackson  
            gewählt!');  
    }  
}  
}
```

# HTML: Checkboxes

```
<form class="demo">
  <strong>Create privileges allowed:</strong>
  <input type="checkbox" name="crud[]" value="c" checked><br>
  <strong>Retrieve privileges allowed:</strong>
  <input type="checkbox" name="crud[]" value="r" checked><br>
  <strong>Update privileges allowed:</strong>
  <input type="checkbox" name="crud[]" value="u" checked><br>
  <strong>Destroy privileges allowed:</strong>
  <input type="checkbox" name="crud[]" value="d" checked><br>
  <input type="submit" value="Enable Privileges">
</form>
```

# HTML: Checkboxes

```
$this->get('http://simpletest.wanderer/checkbox.php');  
$this->assertField('crud[]', array('c', 'r', 'u', 'd'));  
$this->setField('crud', array('r'));  
$this->click('Enable Privileges');
```

...Text prüfen...

# HTML: Radiobutton

```
<form action="radiobutton.php" method="get">
<p>Geben Sie Ihre Zahlungsweise an:</p>
<p>
<input type="radio" name="Zahlungsmethode" value="Mastercard">
  Mastercard<br>
<input type="radio" name="Zahlungsmethode" value="Visa"> Visa<br>
<input type="radio" name="Zahlungsmethode" value="AmericanExpress">
  American Express
</p>
<input type="submit" name="senden" value="Abkassieren">
</form>
```

# HTML: Radiobutton

```
function testRadiobutton() {  
    $this->get('http://simpletest.wanderer/radiobutton.php');  
    $this->setField('Zahlmethode', 'Visa');  
    $this->click('Abkassieren');  
    $this->assertText('Es wurde die Zahlungsart Visa ausgewählt!');  
}
```

# HTML: MultiSelect

```
<form action="multiselectbox.php" method="get"
  name="multiselect">
  <select name="auswahl[]" multiple="multiple">
    <option value="1">Heino</option>
    <option value="2">Michael Jackson</option>
    <option value="3">Tom Waits</option>
    <option value="4">Nina Hagen</option>
    <option value="5">Marianne Rosenberg</option>
  </select>
  <input type="submit" name="senden" value="Favorit">
</form>
```

# HTML: MultiSelect

```
function testMultiSelect() {  
    $this->get('http://simpletest.wanderer/multiselectbox.php');  
    $this->setField('auswahl[]', array(1, 2));  
    $this->click('Favorit');  
    $this->assertText('Es wurde die Favoriten: 1,2 gewählt!');  
}
```

# Ergebnisdarstellung

- Reporter wird angegeben, oft wird der HtmlReporter verwendet.

```
$test = &new LoginTest();  
$test->run(new HtmlReporter());
```

# Ergebnisdarstellung

- Im Zusammenhang mit der Kommandozeile kann auch die Verwendung des TextReporter sinnvoll sein (Automatisierung).

# Gruppierung der Tests

- Bisher in jeder Datei ein Test mit Ergebnisdarstellung (HtmlReporter).
- Es besteht die einfache Möglichkeit, eine Menge von Tests zu kombinieren.

# Gruppierung der Tests

```
require_once('simpletest/web_tester.php');
```

```
require_once('simpletest/reporter.php');
```

```
$group_test = &new GroupTest('All Tests');
```

```
$group_test->addTestFile('checkboxtest.php');
```

```
$group_test->addTestFile('radiobuttontest.php');
```

```
$group_test->addTestFile('multiselectboxtest.php');
```

```
$group_test->run(new HtmlReporter());
```

# Grenzen von SimpleTest

- Es ist nicht möglich, Tabelleninhalte bzw. Tabellen komfortabel zu prüfen.
- JavaScript
  - Derzeit keinerlei Unterstützung vorhanden (eventuell in 3.0 ;-)

# Alternativen

- Seiten mit JavaScript
  - SeleniumRC
- Andere Test Frameworks
  - z.B. JWebUnit (Java ;-)
    - Ebenfalls Probleme mit JavaScript.
    - Bessere Integration in Ant.

# Noch Fragen?

Vielen Dank für Ihre Aufmerksamkeit.

Kontakt:

[phpcon@soebes.de](mailto:phpcon@soebes.de)

# Online Quellen

- SimpleTest Homepage  
<http://www.lastcraft.com/overview.php>
- SimpleTest API Doc  
<http://simpletest.org/>
- Unit Test  
<http://www.phpunit.de>
- SeleniumRC  
<http://www.openqa.org/selenium-rc/>

# Online Quellen

- Phing Homepage  
<http://phing.info>
- Ant Homepage  
<http://ant.apache.org>
- Unit Test  
<http://www.junit.org>

# Online Quellen

- Java Web Test Tools
  - JWebUnit  
<http://jwebunit.sourceforge.net>
  - HTTPUnit  
<http://httpunit.sourceforge.net>
  - Canoo WebTest  
<http://webtest.canoo.com>