Agenda

- **Tag 1 – Einführung**
  - Installation
  - Erste Anwendung
  - Architektur
- **Tag 2 – Entwicklung**
  - Models
  - Darstellung
  - Formulare
  - Ajax
- **Tag 3 – Fortgeschrittene Themen**
  - Tests
  - Security & Deployment
  - Lokalisierung & Internationalisierung
  - Performance
  - Best Practices

Testing

Tests

anderScore trust in competence

GFU Cyrus AG

Allgemeines

- Fokussierung des Frameworks auf Java
  - JUnit
  - TestNG
  - Mockito
  - …
- Unterstützung durch Wicket
  - WicketTester
  - FormTester
  - TagTester

Tests

anderScore trust in competence

GFU Cyrus AG

WicketTester

- Utility-Klasse zur Unterstützung von Tests
- Kommunikation mit dem Framework
  - Rendern von Pages und Komponenten
  - Prüfung des Status von Komponenten
  - Isoliertes Testen von Komponenten
  - Klicken auf Links
  - …

WicketTester

```
public class TestHomePage {

    private WicketTester tester;

    @BeforeEach
    public void setUp() {
        tester = new WicketTester(new WicketApplication());
    }

    @Test
    public void testComponents() {
        // Rendern der Homepage
        tester.startPage(HomePage.class);
        // Rendering der Homepage überprüfen
        tester.assertRenderedPage(HomePage.class);
        // Überprüfung auf Vorhandensein von Label
        tester.assertLabel("label", "First label");
        // Klick auf Link simulieren
        tester.clickLink("reload");
        // Überprüfung von Vorhandensein von Label
        tester.assertLabel("label", "Second label");
        // Überprüfung der Aktivierung eines DatePickers
        tester.assertEnabled("form:datepicker");
    }
}
```

Tests

FormTester

- Testen von Formularen im Speziellen
- Überprüfung von Feedback Messages
- Erzeugung durch WicketTester

```
protected void insertUsernamePassword(String username, String password) {
    // Rendern der Homepage
    tester.startPage(HomePage.class);
    // FormTester für einzelnes Formular erstellen
    FormTester formTester = tester.newFormTester("form");
    // Eingabewerte setzen
    formTester.setValue("username", username);
    formTester.setValue("password", password);
    // Formular abschicken
    formTester.submit();
}
```

Tests

FormTester

- Testen von Formularen im Speziellen
- Überprüfung von Feedback Messages
- Erzeugung durch WicketTester

```
@Test
public void testMessageForSuccessfulLogin() {
    insertUsernamePassword("Wicket", "anderScore");
    tester.assertInfoMessages("Benutzername und Passwort korrekt!");
}

@Test
public void testMessageForFailedLogin () {
    insertUsernamePassword("Wicket", "Apache");
    tester.assertErrorMessages("Benutzername oder Passwort falsch!");
}
```

Tests

TagTester

- Feingranulare Überprüfung des gerenderten Markups
- Erstellung von TagTestern
  - Repräsentieren DOM-Elemente
  - Statische Factory-Methoden
  - Angabe von Selektionskriterien

Tests

TagTester

- Erwartetes Markup

```
<html xmlns:wicket="http://wicket.apache.org">

<head>
    <title></title>
</head>

<body>

<span class="myClass"></span>
<div class="myClass"></div>

</body>

</html>
```

TagTester

- Test

```
@Test
public void homePageMarkupTest() {
    // Rendern der Page
    tester.startPage(HomePage.class);
    // Extraktion des Markups
    String responseTxt = tester.getLastResponse().getDocument();
    // TagTester mit Attributselektor erstellen
    TagTester tagTester = TagTester.createTagByAttribute(responseTxt, "class", "myClass");

    // Überprüfung des ersten Elementes
    assertNotNull(tagTester);
    assertEquals("span", tagTester.getName());

    // Überprüfung beider Elemente
    List<TagTester> tagTesterList = TagTester.createTagsByAttribute(responseTxt,
            "class", "myClass", false);

    assertEquals(2, tagTesterList.size());
}
```