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

COMPONENTS

Darstellung

anderScore trust in competence

GFU Cyrus AG

Labels

- Dynamische Ausgabe von Zeichenketten
- Convenience-Konstruktor ohne Model

- HTML

```
<p>Anbieter des Trainings: <span wicket:id="trainer"></span></p>
```

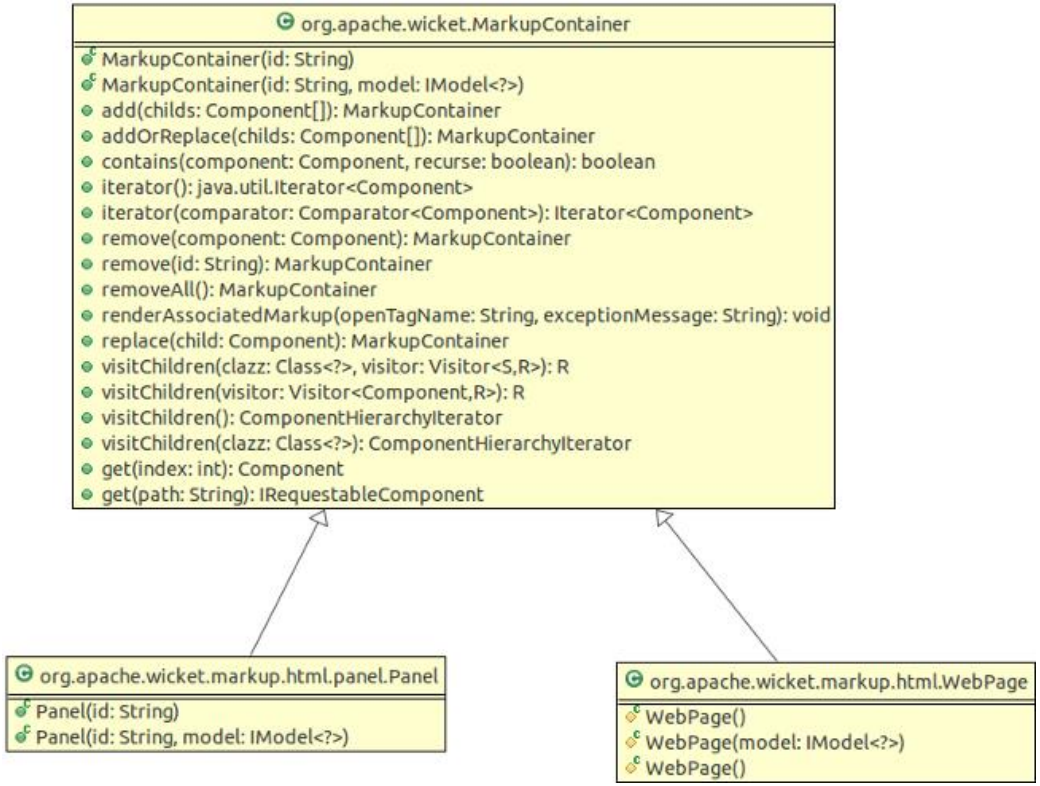
- Java

```
add(new Label("trainer", "anderScore"));
```

Darstellung

Panels

- Wiederverwendbare Container für Komponenten
- HTML + Java



Darstellung

Panels

- HTML: Panel

```
<html>

<head></head>

<body>

<wicket:panel>
   <!-- Diverse Labels. Äußeres Gerüst wird ignoriert! -->
</wicket:panel>

</body>

</html>
```

Darstellung

anderScore trust in competence

GFU Cyrus AG

Panels

- Java: Panel

```
public class PersonPanel extends Panel {

    public PersonPanel(String id, IModel<Person> person) {
        super(id);
        setDefaultModel(new CompoundPropertyModel(person));

        add(new Label("name"));
        add(new Label("surname"));
        add(new Label("address"));
        add(new Label("email"));
        add(new Label("spouse.name"))
    }
}
```

Darstellung

Panels

- Java: Page

```
public class PersonPage extends WebPage {

    public PersonPage() {
        Person john = new Person("John", "Doe");
        IModel<Person> person = new Model<>(john);

        add(new PersonPanel("person", person));
    }
}
```

Darstellung

anderScore trust in competence

GFU Cyrus AG

Templating

- Vorgabe eines gemeinsamen Rahmens
  - Header
  - Navigation
  - Content
  - Footer
- Strategie
  - Panels für gemeinsame Bereiche konstruieren
  - Panels zu Template Page zusammenbauen
  - Subpages: Nutzung von Java und Markup Inheritance
  - Subpages: Spezifischen Content definieren
- Spezielle Tags
  - wicket:child
  - wicket:extend

Darstellung

anderScore trust in competence

GFU Cyrus AG

Templating

- HTML: Template Page

```
<html>

<head></head>

<body>

<div id="header" wicket:id="headerPanel">header</div>
<div id="body">
    <div id="menu" wicket:id="menuPanel">menu</div>
    <wicket:child/>
</div>
<div id="footer" wicket:id="footerPanel">footer</div>

</body>

</html>
```

Darstellung

anderScore trust in competence

GFU Cyrus AG

Templating

- Java: Template Page

```
public class GFUTemplate extends WebPage {

    private Panel headerPanel;
    private Panel menuPanel;
    private Panel footerPanel;

    public GFUTemplate() {
        add(headerPanel = new HeaderPanel("headerPanel"));
        add(menuPanel = new MenuPanel("menuPanel"));
        add(footerPanel = new FooterPanel("footerPanel"));
    }
}
```

Darstellung

anderScore trust in competence

GFU Cyrus AG

Templating

- HTML: Page

```
<html>

<head></head>

<body>

<wicket:extend>
    <!-- Tags für die gewünschten Komponenten einfügen -->
</wicket:extend>

</body>
</html>
```

Darstellung

anderScore trust in competence

GFU Cyrus AG

Templating

- Java: Page

```
public class PersonPage extends GFUTemplate {

    public PersonPage() {
        super();

        // Gewünschte Komponenten einfügen
    }
}
```

Darstellung

Repeater

- Mehrfaches Anzeigen von Elementen
- Beispiel: Listen
- Alternativen in Wicket
  - RepeatingView
  - ListView
  - DataView

Darstellung

RepeatingView

- Wiederholung eines einfachen Markup-Fragments

- HTML

```
<ul>
    <li wicket:id="listItems"></li>
</ul>
```

- Java

```
RepeatingView listItems = new RepeatingView("listItems");

listItems.add(new Label(listItems.newChildId(), Model.of("green")));
listItems.add(new Label(listItems.newChildId(), Model.of("blue")));
listItems.add(new Label(listItems.newChildId(), Model.of("red")));
```

Darstellung

ListView

- Darstellung komplexeren Markups
- Jedes Element wird zu eigenem ListItem

- HTML

```
<div wicket:id="persons">
    <div><b>Full name: </b></div>
    <div wicket:id="fullName"></div>
</div>
```

Darstellung

ListView

- Java

```
public HomePage(final PageParameters parameters) {
    List<Person> persons = Arrays.asList(new Person("John", "Doe"));

    add(new ListView<Person>("persons", persons) {

        @Override
        protected void populateItem(ListItem<Person> item) {
            item.add(new Label("fullName",
                    new PropertyModel(item.getModel(), "fullName")));
        }
    });
}
```

Darstellung

DataView

- Effiziente Darstellung großer Datenmengen
- DataProvider
  - Paging, Sorting
  - ListDataProvider
  - SortableDataProvider

- HTML

```
<table>
    <tr>
        <th>Name</th><th>Vorname</th>
    </tr>
    <tr wicket:id="rows">
        <td wicket:id="dataRow"></td>
    </tr>
</table>
```

Darstellung

DataView

- Java

```
//Methode außerhalb definiert
List<Person> persons = loadPersons();

ListDataProvider<Person> listDataProvider = new ListDataProvider<>(persons);

DataView<Person> dataView = new DataView<>("rows", listDataProvider) {

   @Override
   protected void populateItem(Item<Person> item) {
      Person person = item.getModelObject();
      RepeatingView repeatingView = new RepeatingView("dataRow");

      repeatingView.add(new Label(repeatingView.newChildId(), person.getName()));
      repeatingView.add(new Label(repeatingView.newChildId(), person.getSurname()));
      item.add(repeatingView);
   }
};

add(dataView);
```