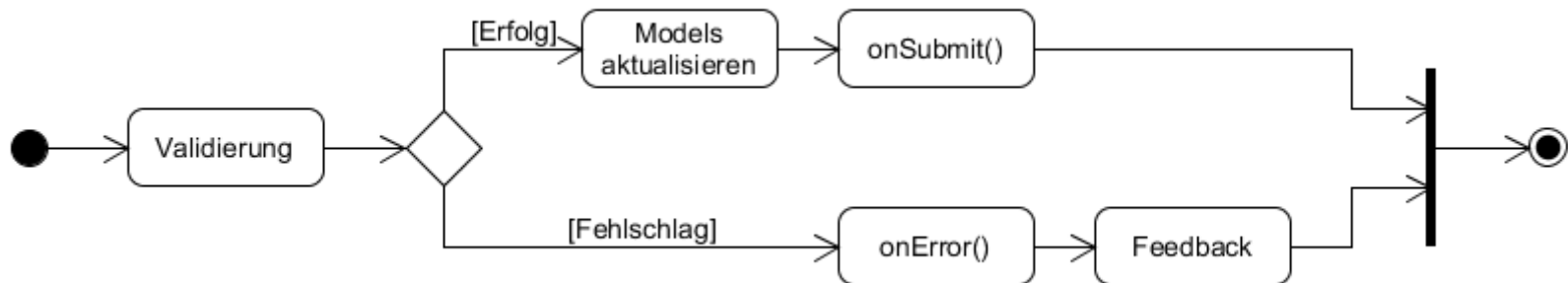


Formulare

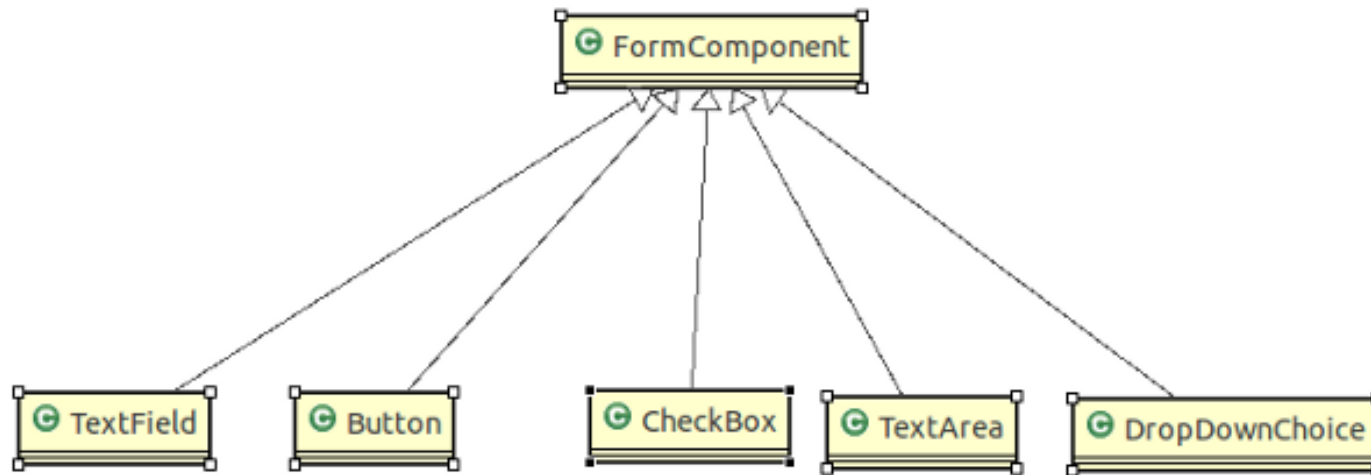
Grundlagen

- Eigene Komponenten
- Handler auf Serverseite
 - `onSubmit()`
 - `onError()`
- Verarbeitung



Grundlagen

- Vorgefertigte Eingabekomponenten
 - Textfelder
 - CheckBoxen
 - DropDowns
 - Buttons
 - ...



Grundlagen

- HTML: Page

```
<html>
```

```
<head><title>Person</title></head>
```

```
<body>
```

```
<form id="personForm" method="get" wicket:id="personForm">  
  <span>Name: </span><input wicket:id="name" type="text" id="name"/>  
  <span>Nachname: </span><input wicket:id="surname" type="text" id="surname"/>  
  <input type="submit" name="Save" value="Speichern"/>  
</form>
```

```
</body>
```

```
</html>
```

Grundlagen

- Java: Page

```
public class PersonPage extends WebPage {  
  
    public PersonPage() {  
        super();  
  
        add(new PersonForm("personForm"));  
    }  
}
```

Grundlagen

- Java: Formular

```
public class PersonForm extends Form {

    private TextField nameField;
    private TextField surnameField;

    public PersonForm(String id) {
        super(id);

        nameField = new TextField("name", Model.of(""));
        surnameField = new TextField("surname", Model.of(""));

        add(nameField);
        add(surnameField);
    }

    public final void onSubmit() {
        String name = (String) nameField.getDefaultModelObject();
        String surname = (String) surnameField.getDefaultModelObject();

        System.out.println("Hallo " + name + " " + surname);
    }
}
```

Validierung

- Interface *IValidator*
- Vorgefertigte Validatoren
 - EmailAddressValidator
 - URLValidator
 - DateValidator
 - RangeValidator
 - ...

Validierung

- Eigene Validatoren

```
public class AgeValidator implements IValidator<Integer> {  
  
    public void validate(IValidatable<Integer> validatable) {  
        int age = validatable.getValue();  
  
        if(age < 18 || age > 100){  
            ValidationError error = new ValidationError(this);  
            validatable.error(error);  
        }  
    }  
}
```


JSR 303

- Definiert Annotationen zur Bean-Validierung
 - @Range
 - @Past
 - @NotNull
 - @Pattern
 - ...
- Nutzung in Wicket
 - Annotation der Beans
 - Aktivierung in Application-Klasse
 - Hinzufügen von PropertyValidator zu Feldern

JSR 303

- Bean

```
public class Person implements Serializable {  
  
    @NotNull  
    private String name;  
  
    @Pattern(regexp = "^...$")  
    private String email;  
  
    @Range(min = 18, max = 150)  
    private int age;  
  
    @Past @NotNull  
    private Date birthDay;  
  
    @NotNull  
    private Address address;  
  
}
```

JSR 303

- Application-Klasse

```
@Override  
public void init(){  
    super.init();  
  
    new BeanValidationConfiguration().configure(this);  
}
```

- PropertyValidator

```
Form<Person> form = new Form<>("personForm");  
  
form.add(new TextField("name").add(new PropertyValidator()));  
form.add(new TextField("email").add(new PropertyValidator()));  
form.add(new TextField("age").add(new PropertyValidator()));
```

Feedback

- Rückmeldung an den Nutzer
- Überschreibbare Standardtexte
- Internationalisierung
- Speicherung in Resource Bundles

Feedback Panel

- Auflistung von Messages
- Interface *IFeedbackMessageFilter*

- HTML

```
<div wicket:id="successMessage" ></div>  
<div wicket:id="feedbackMessage" ></div>
```

- Java

```
add(new FeedbackPanel("feedbackMessage",  
    new ExactErrorLevelFilter(FeedbackMessage.ERROR)));
```

```
add(new FeedbackPanel("successMessage",  
    new ExactErrorLevelFilter(FeedbackMessage.SUCCESS)));
```

Flash Messages

- Nachrichten von Komponenten
- Levels (vgl. Logging)
 - debug
 - info
 - success
 - warn
 - error
 - fatal
- Definition eigener FeedbackMessageFilter

Flash Messages

```
public class CustomFeedbackMessageFilter implements IFeedbackMessageFilter {  
  
    private int errorLevel;  
  
    public CustomFeedbackMessageFilter(int errorLevel){  
        this.errorLevel = errorLevel;  
    }  
  
    public boolean accept(FeedbackMessage message) {  
        boolean accept = false;  
  
        // Überprüfen, ob eine Message ausgegeben werden soll...  
  
        return accept;  
    }  
  
    // Eventuell Berücksichtigung weiterer Level  
}
```