

CONSULTING  
**enpit**

# SESSION 4

## FORMULARE, LAYOUT & VALIDIERUNG

# AGENDA

1. Formulare
2. Layout
3. Validierung
4. Hands-On #4

# ZIEL

[< Back](#)

CONSULTING  
**enpit** JET Spotify Explorer

## Interpret hinzufügen

Name: \*

Genre: \*

Rock ▼

Gründungsjahr:

 ▼ ▲

Geben Sie eine Zahl zwischen 1800 und 2050 ein.

Speichern

# HTML VS. JET FORMULARE

HTML	JET
<code>&lt;form&gt;</code>	<code>&lt;div class="oj-form"&gt;</code>
<code>&lt;input type="text"&gt;</code>	<code>&lt;input data-bind=„ojComponentW=ôcomponentW=, <b>ojInputText</b> `I===valueW=value}`"&gt;</code>
<code>&lt;input type="submit"&gt;...</code>	<code>&lt;button data-bind="click: <b>submitFunc</b>, ojComponent: { component: 'ojButton', label: 'Submit' }"&gt;</code>

# VIELE EINGEBAUTE FORM CONTROLS

- `ojInputText`
- `ojInputNumber`
- `ojInputDateTime`
- `ojInputSearch`
- `ojSelect`
- `ojSlider`
- `ojSwitch`
- ...

# BEISPIEL

```
self.name = ko.observable('');
self.genre = ko.observableArray(['rock']);
self.year = ko.observable(new Date().getFullYear());

self.tracker = ko.observable();

self.save = function () {
    var trackerObj = ko.utils.unwrapObservable(self.tracker);
    if (isValid(trackerObj)) {
        ko.postbox.publish('add-artist', {
            name: self.name(),
            genre: self.genre()[0],
            year: self.year()
        });
        window.history.back();
    }
};
```

# AGENDA

1. Formulare
2. Layout
3. Validierung
4. Hands-On #4



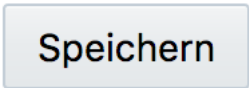
# LAYOUT PER CSS-KLASSEN

- JET implementiert Alta-UI
- CSS-Klassen zur weiteren Gestaltung
- Bsp. Button-Styling:

```
<button class="oj-button oj-button-confirm" ...>
```

Speichern

vs.

Speichern

# FLEXBOX

- JET nutzt Flexbox zum Ausrichten der Elemente
  - “Flex-Layout”
- In eigenen CSS-Klassen implementiert
  - `oj-flex`
  - `oj-flex-auto`
  - ...

# WEITERE LAYOUT-STRATEGIEN

- **Grid**-Layout (Erweiterung von Flex-Layout)
  - oj-sm-6, oj-lg-3, ...
- **Form**-Layout
  - oj-form-layout, oj-form, oj-form-control-group
- **Panel, Popup, Tabs, Accordion, ...**

# BEST PRACTICES FÜR LAYOUT?

- Cookbook referenzieren
- So wenig **eigenes CSS** wie möglich
- [Alta-UI](#) einhalten
  - **Ausnahmen** möglich
- Flexbox verstehen ([Flexbox-Froggy](#))

# AGENDA

1. Formulare
2. Layout
3. Validierung
4. Hands-On #4

# EINGEBAUTE VALIDATOREN

- Required Validator
  - Erzeugt \* am Label
- Typ der Komponente (bspw. *oJInputNumber*)
- Min-/Max-Validator

# VALIDIERUNG VERWENDEN

- **Status** der Validierung per *InvalidComponentTracker* überwachen
  - Reguläres Knockout Observable
- Vorzeitige Formular-Absendung verhindern per **disabled** Funktion im Button
  - Stolperstein: Funktionsaufruf statt –referenz!

# BEISPIEL

```
<div class="oj-flex">
  <div class="oj-flex-item">
    <label for="nameControl">Name:</label>
    <input id="nameControl"
      placeholder="Name des Interpreten"
      data-bind="ojComponent: {
        component: 'ojInputText',
        value: name,
        required: true,
        invalidComponentTracker: tracker,
        rootAttributes: {style:'max-width:100%'}
      }">
  </div>
</div>
```



# AGENDA

1. Formulare
2. Layout
3. Validierung
4. Hands-On #4



Hands On #4

# LAYOUT, FORMULARE, VALIDIERUNG

08.05.2017

18

# HANDS-ON #4

- [exercises/4\\_forms](#)
- README öffnen ([Browser](#))

```
$ cd exercises/4_forms
```

```
$ grunt serve
```

- **Optionale Bonusaufgabe**

**HABEN SIE NOCH FRAGEN?**

**VIELEN DANK FÜR IHRE  
AUFMERKSAMKEIT**

CONSULTING  
**enpit**