Kolibri

AUSWAHLKOMPONENTE

IP6 Midterm Presentation
Ramona Marti, Lea Burki

fhnw.ch

Hallo

Wir sinds...

Lea Burki

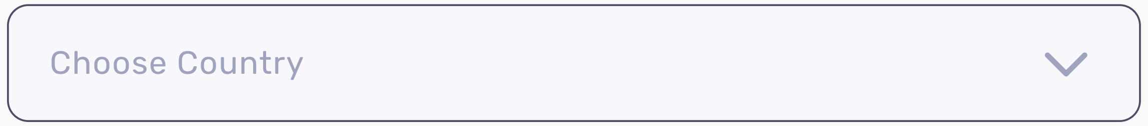
lea.burki@students.fhnw.ch

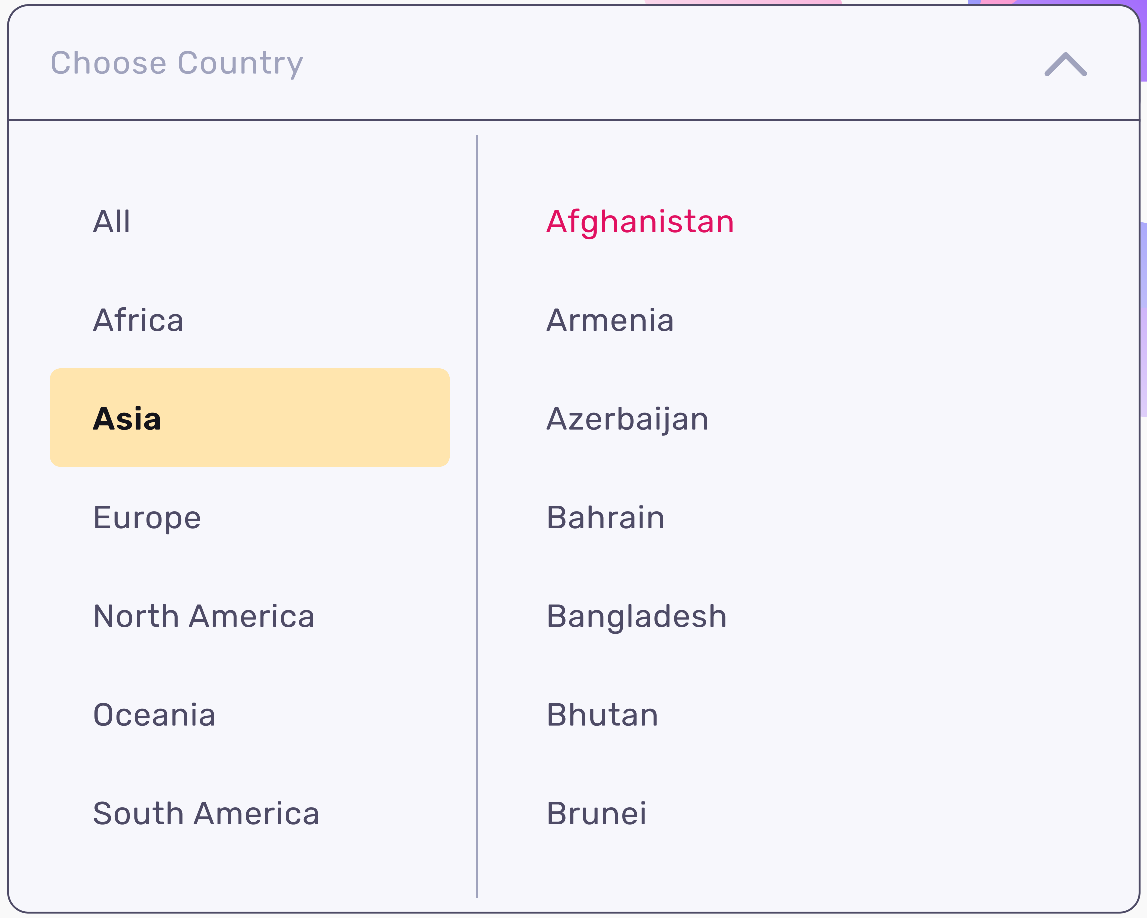
Ramona Marti

ramona.marti@students.fhnw.ch

Rückblick IP5

Select | Datalist





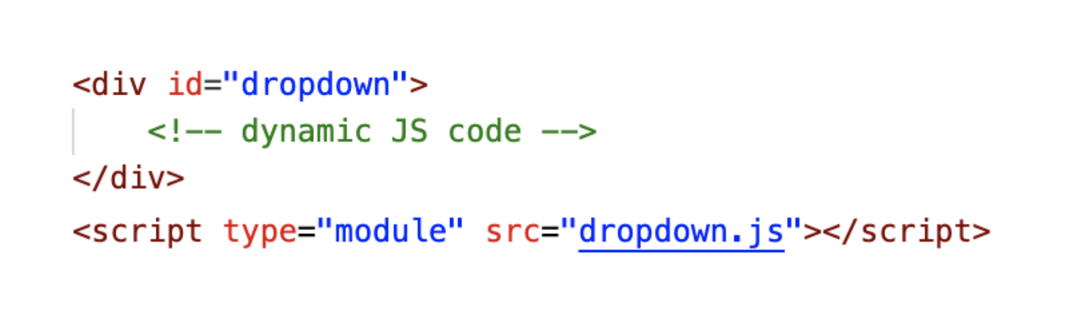
LIVE DEMO

Komponente

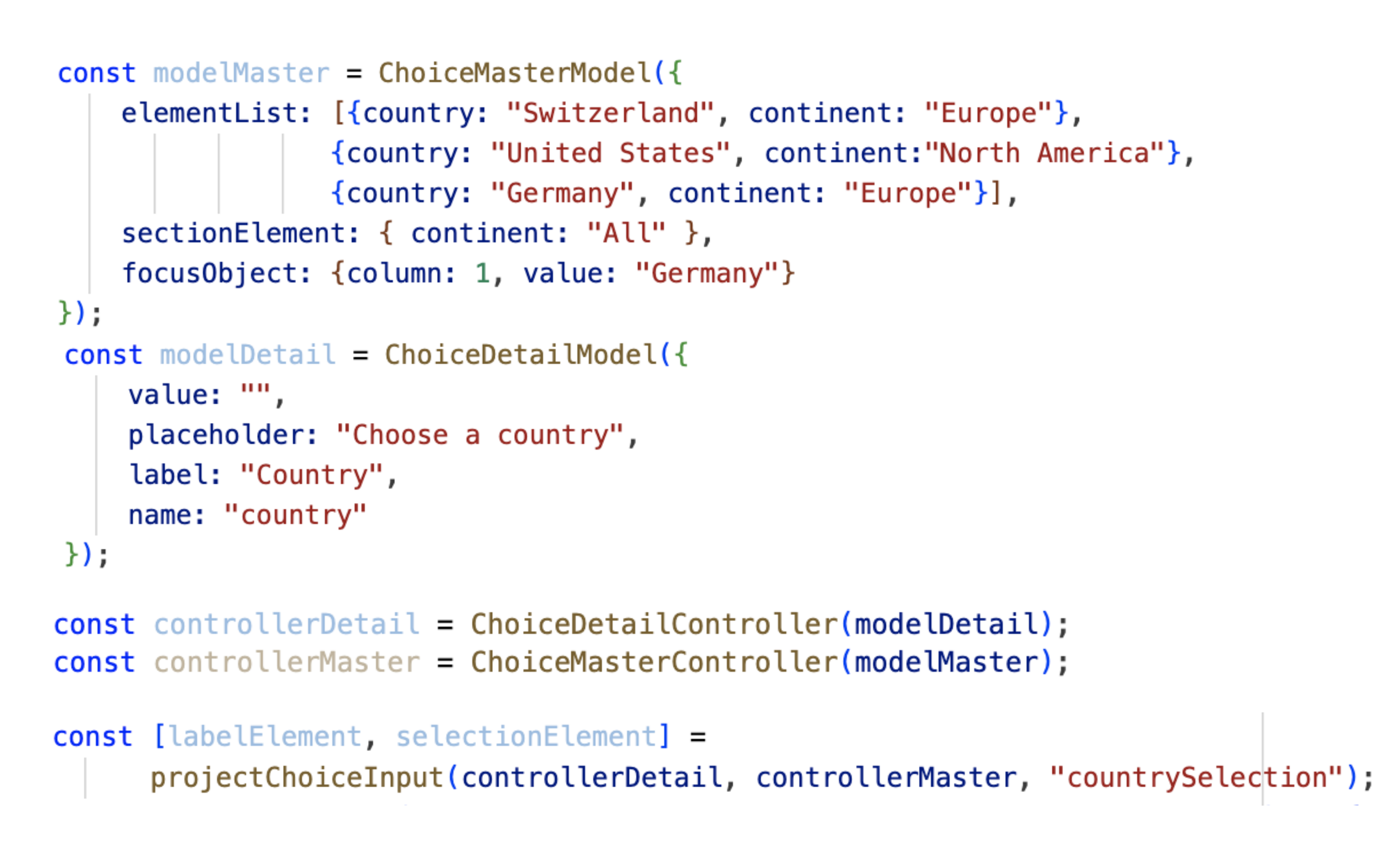
Rückblick IP5

Code

- Projector Pattern
- Master Detail View



```html
<div id="dropdown">
    <!-- dynamic JS code -->
</div>
<script type="module" src="dropdown.js"></script>
```

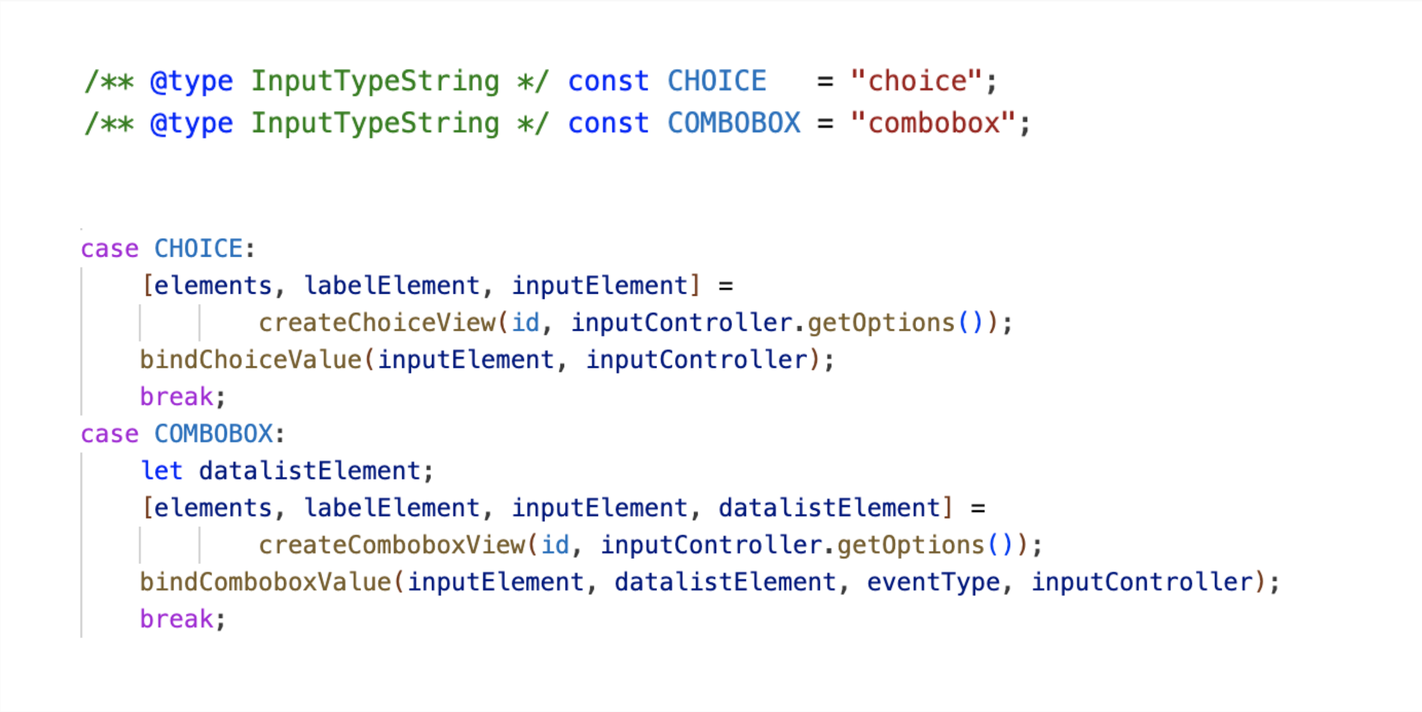


```javascript
const modelMaster = ChoiceMasterModel({
    elementList: [{country: "Switzerland", continent: "Europe"},
                  {country: "United States", continent:"North America"},
                  {country: "Germany", continent: "Europe"}],
    sectionElement: { continent: "All" },
    focusObject: {column: 1, value: "Germany"}
});
const modelDetail = ChoiceDetailModel({
    value: "",
    placeholder: "Choose a country",
    label: "Country",
    name: "country"
});

const controllerDetail = ChoiceDetailController(modelDetail);
const controllerMaster = ChoiceMasterController(modelMaster);

const [labelElement, selectionElement] =
    projectChoiceInput(controllerDetail, controllerMaster, "countrySelection");
```

Integration

Select | Datalist

- Erweiterungen
- Projector Views
- Projector Bindings

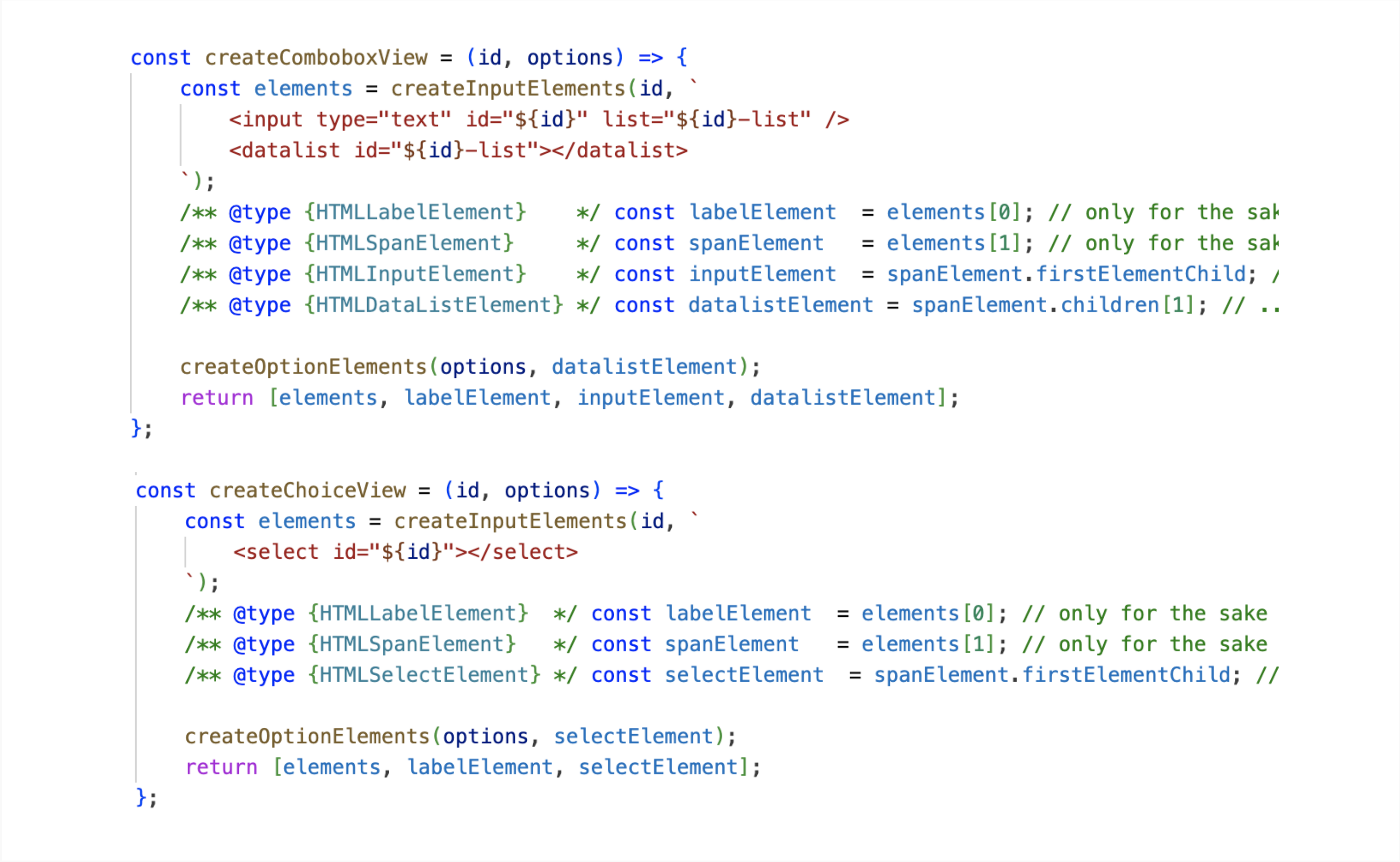


```
/** @type InputTypeString */ const CHOICE   = "choice";
/** @type InputTypeString */ const COMBOBOX = "combobox";


case CHOICE:
    [elements, labelElement, inputElement] =
            createChoiceView(id, inputController.getOptions());
    bindChoiceValue(inputElement, inputController);
    break;
case COMBOBOX:
    let datalistElement;
    [elements, labelElement, inputElement, datalistElement] =
            createComboboxView(id, inputController.getOptions());
    bindComboboxValue(inputElement, datalistElement, eventType, inputController);
    break;
```

Integration

Select | Datalist

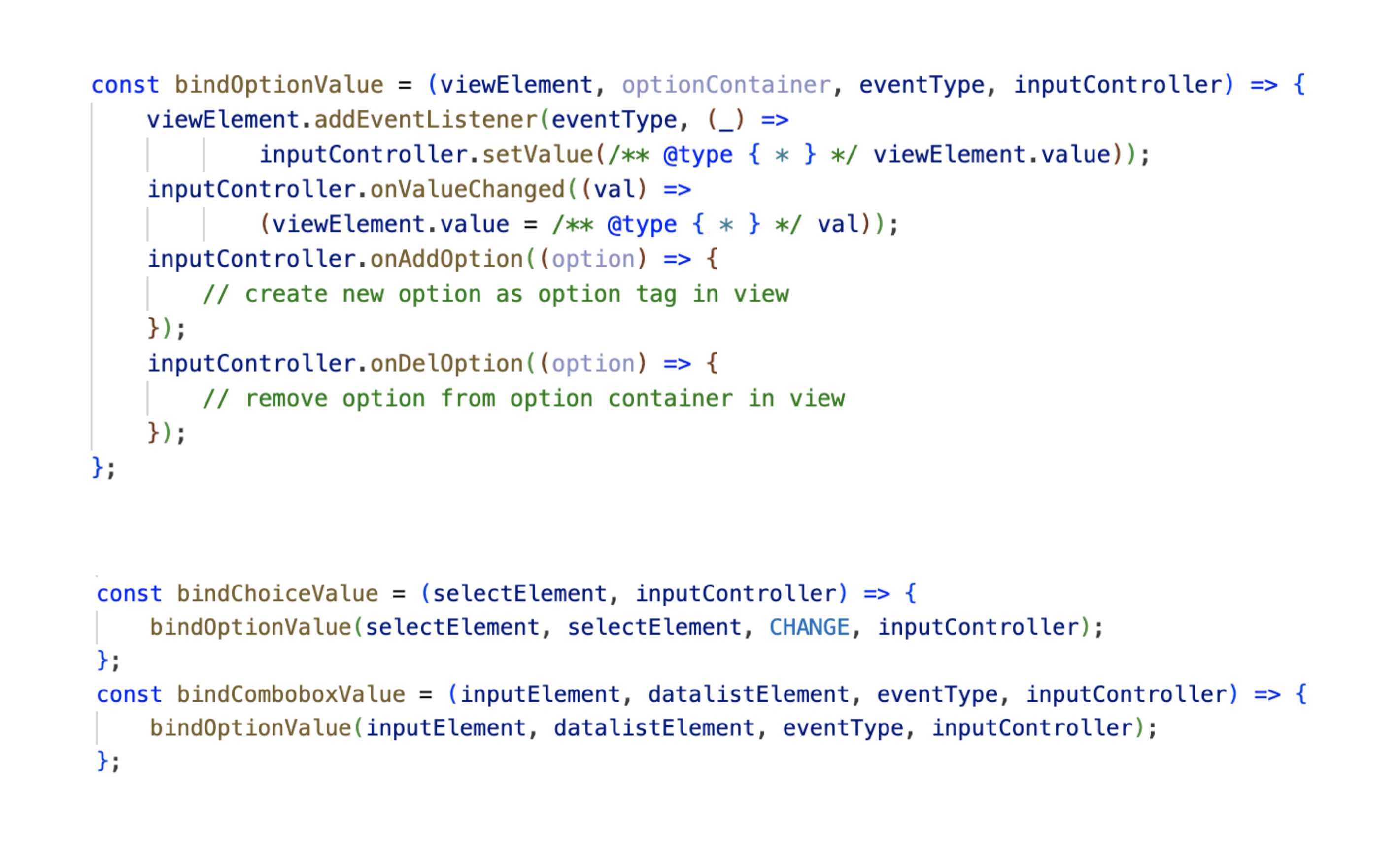
- Erweiterungen
- Projector Views
- Projector Bindings



```
const createComboboxView = (id, options) => {
    const elements = createInputElements(id, `
        <input type="text" id="${id}" list="${id}-list" />
        <datalist id="${id}-list"></datalist>
    `);
    /** @type {HTMLLabelElement}    */ const labelElement    = elements[0]; // only for the sak
    /** @type {HTMLSpanElement}     */ const spanElement     = elements[1]; // only for the sak
    /** @type {HTMLInputElement}    */ const inputElement    = spanElement.firstElementChild; /
    /** @type {HTMLDataListElement} */ const datalistElement = spanElement.children[1]; // ..

    createOptionElements(options, datalistElement);
    return [elements, labelElement, inputElement, datalistElement];
};

const createChoiceView = (id, options) => {
    const elements = createInputElements(id, `
        <select id="${id}"></select>
    `);
    /** @type {HTMLLabelElement}  */ const labelElement  = elements[0]; // only for the sake
    /** @type {HTMLSpanElement}   */ const spanElement   = elements[1]; // only for the sake
    /** @type {HTMLSelectElement} */ const selectElement  = spanElement.firstElementChild; //

    createOptionElements(options, selectElement);
    return [elements, labelElement, selectElement];
};
```

Integration
Select | Datalist

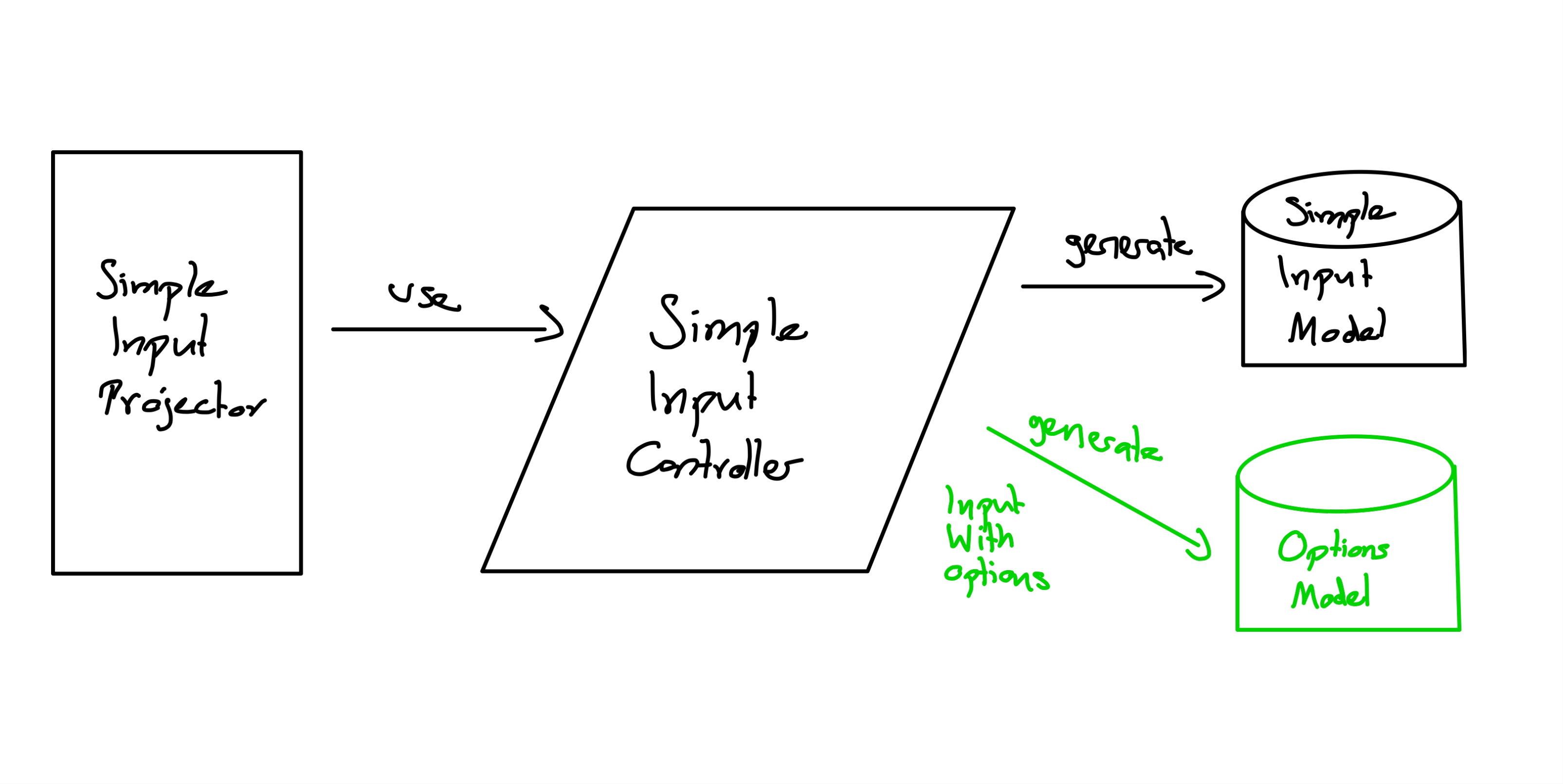
- Erweiterungen
- Projector Views
- Projector Bindings

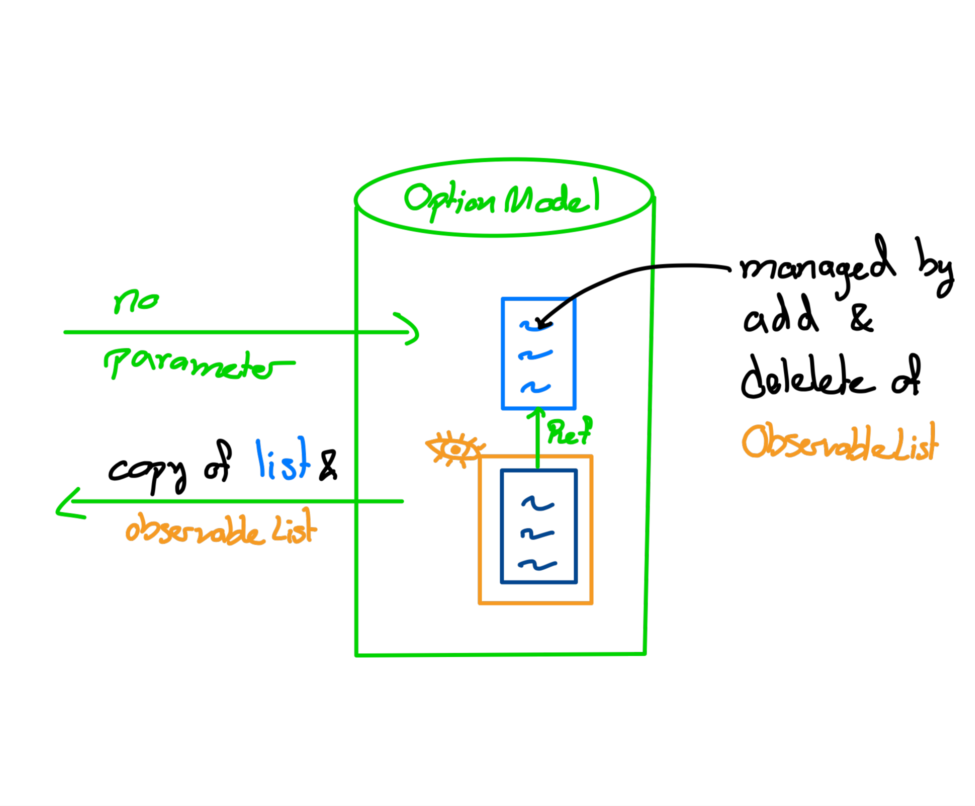


```
const bindOptionValue = (viewElement, optionContainer, eventType, inputController) => {
    viewElement.addEventListener(eventType, (_) =>
            inputController.setValue(/** @type { * } */ viewElement.value));
    inputController.onValueChanged((val) =>
            (viewElement.value = /** @type { * } */ val));
    inputController.onAddOption((option) => {
        // create new option as option tag in view
    });
    inputController.onDelOption((option) => {
        // remove option from option container in view
    });
};


const bindChoiceValue = (selectElement, inputController) => {
    bindOptionValue(selectElement, selectElement, CHANGE, inputController);
};
const bindComboboxValue = (inputElement, datalistElement, eventType, inputController) => {
    bindOptionValue(inputElement, datalistElement, eventType, inputController);
};
```

Update

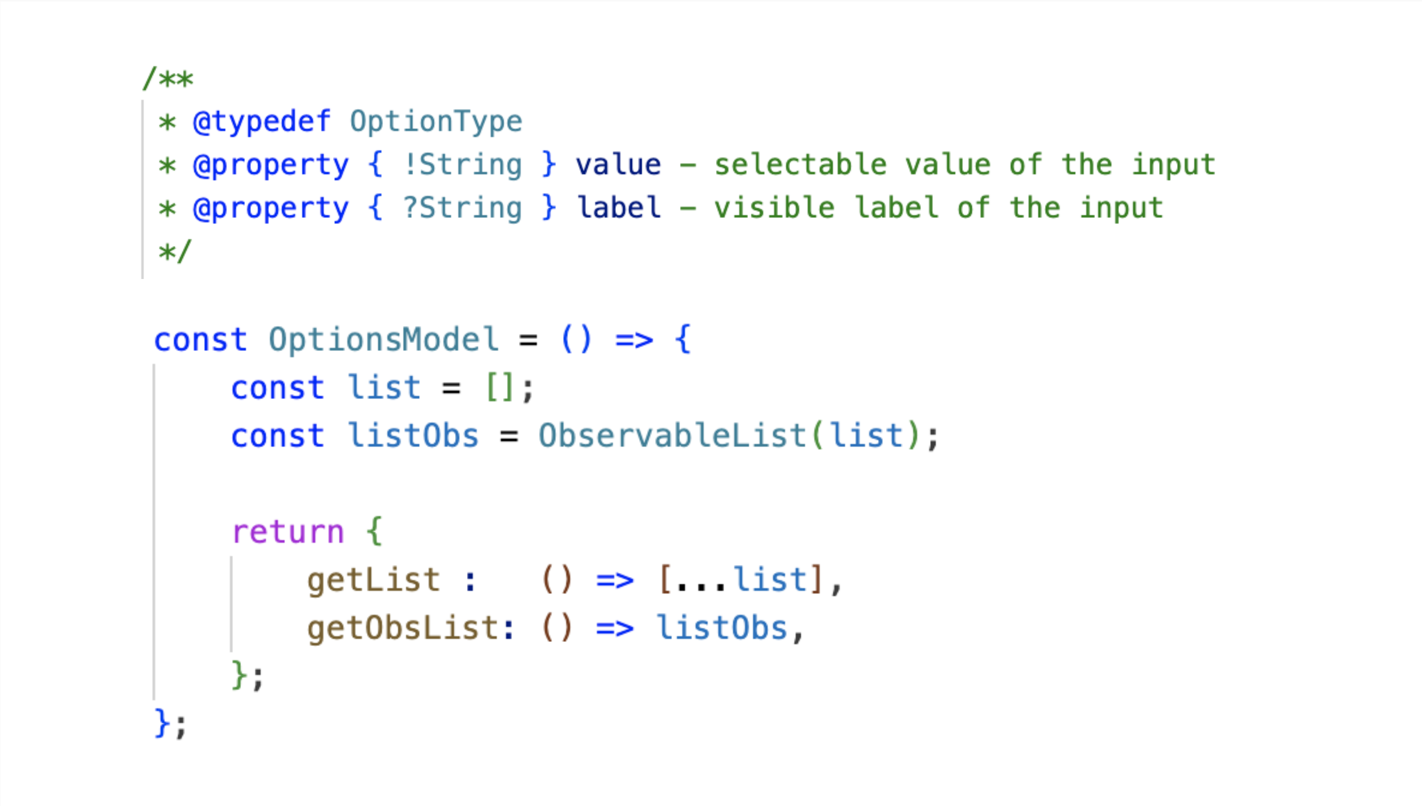
Model | Controller





Update

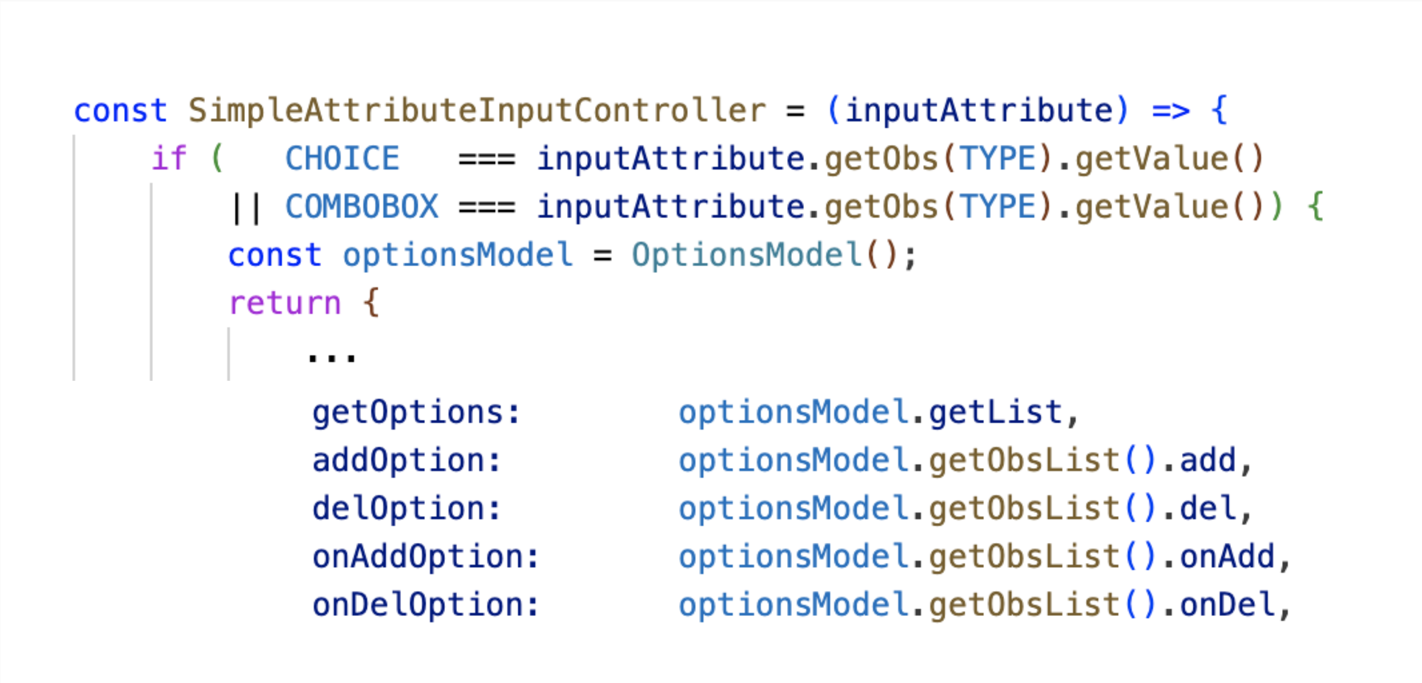
Model | Controller



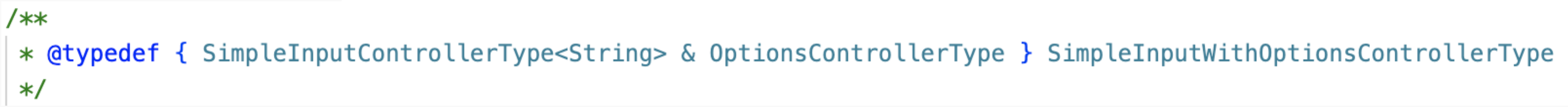
```
/**
 * @typedef OptionType
 * @property { !String } value - selectable value of the input
 * @property { ?String } label - visible label of the input
 */

const OptionsModel = () => {
    const list = [];
    const listObs = ObservableList(list);

    return {
        getList :   () => [...list],
        getObsList: () => listObs,
    };
};
```

Update

Model | Controller



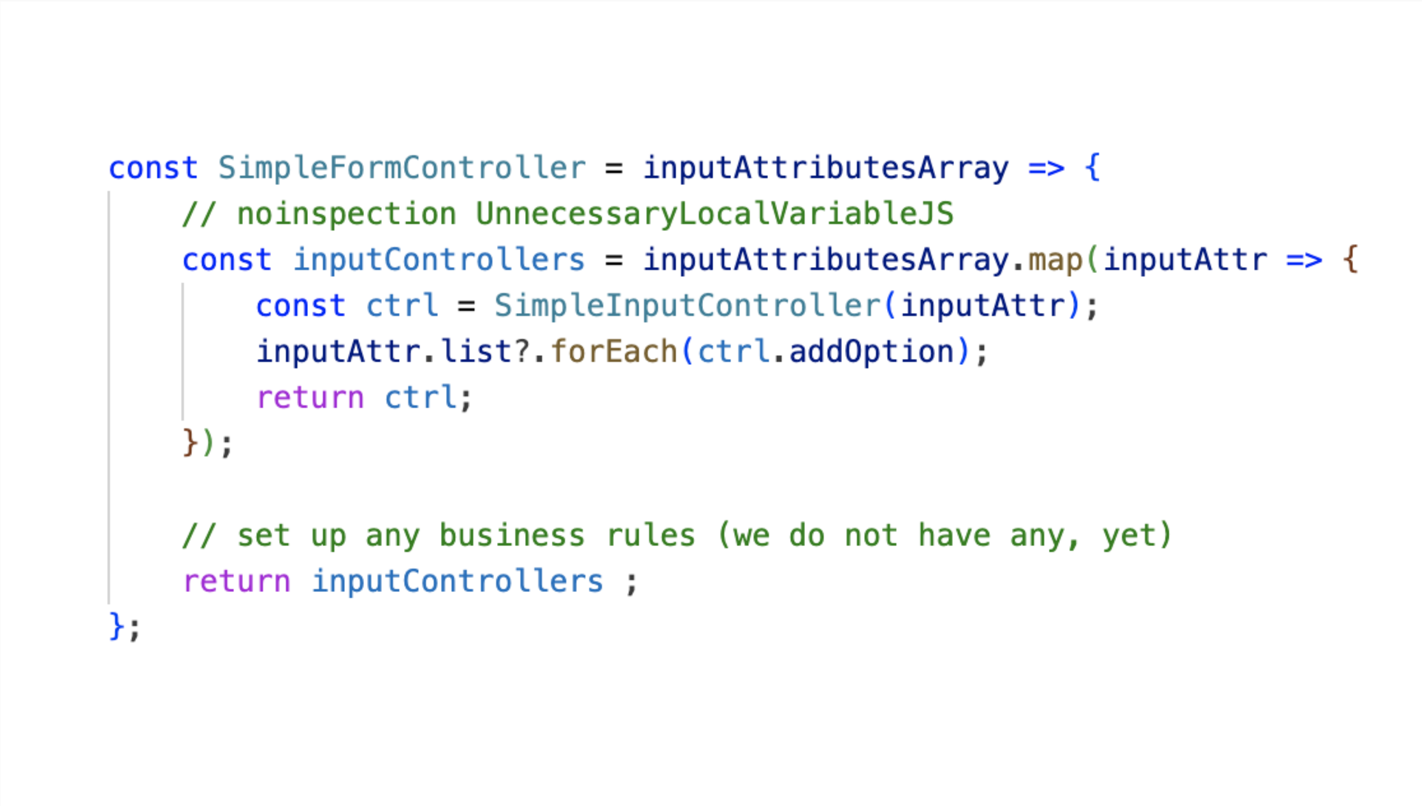
```
const SimpleAttributeInputController = (inputAttribute) => {
    if (   CHOICE   === inputAttribute.getObs(TYPE).getValue()
        || COMBOBOX === inputAttribute.getObs(TYPE).getValue()) {
        const optionsModel = OptionsModel();
        return {
            ...
            getOptions:        optionsModel.getList,
            addOption:         optionsModel.getObsList().add,
            delOption:         optionsModel.getObsList().del,
            onAddOption:       optionsModel.getObsList().onAdd,
            onDelOption:       optionsModel.getObsList().onDel,
```



```
/**
 * @typedef { SimpleInputControllerType<String> & OptionsControllerType } SimpleInputWithOptionsControllerType
 */
```

Update

Model | Controller

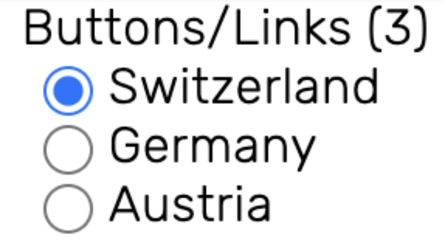


```
const SimpleFormController = inputAttributesArray => {
    // noinspection UnnecessaryLocalVariableJS
    const inputControllers = inputAttributesArray.map(inputAttr => {
        const ctrl = SimpleInputController(inputAttr);
        inputAttr.list?.forEach(ctrl.addOption);
        return ctrl;
    });

    // set up any business rules (we do not have any, yet)
    return inputControllers ;
};
```

LIVE DEMO

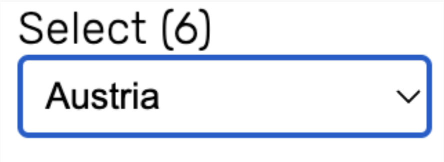
Wo wir sind...

Browser Konsistenz



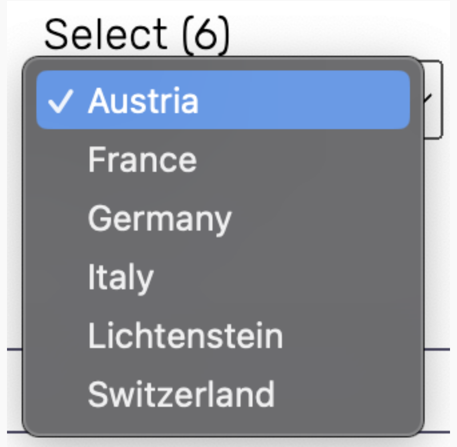
Buttons/Links (3)

● Switzerland
○ Germany
○ Austria



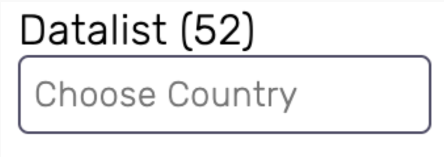
Select (6)

Austria



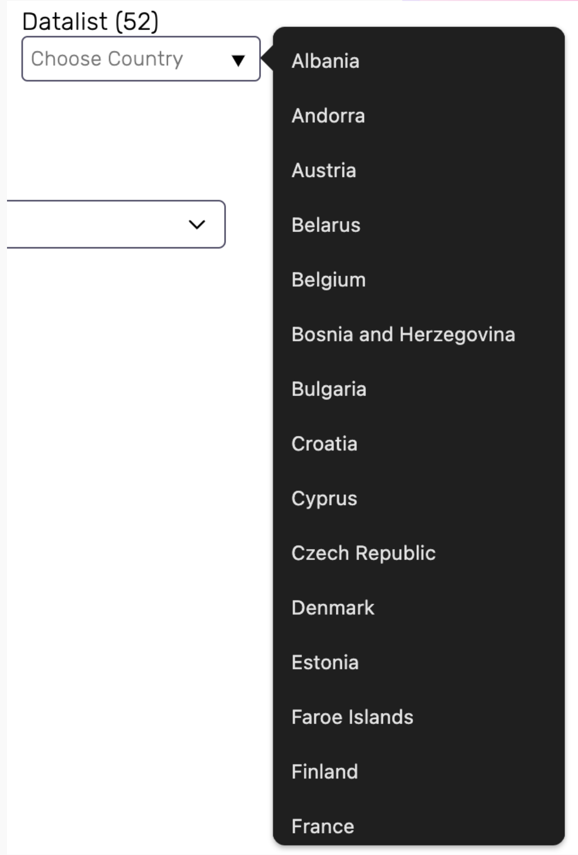
Select (6)

✓ Austria
France
Germany
Italy
Lichtenstein
Switzerland



Datalist (52)

Choose Country



Datalist (52)

Choose Country ▾

Albania
Andorra
Austria
Belarus
Belgium
Bosnia and Herzegovina
Bulgaria
Croatia
Cyprus
Czech Republic
Denmark
Estonia
Faroe Islands
Finland
France

Personas

- Benutzer
- Entwickler

Personas

Planung

- Projector
- Research Browser Inkonsistez
- Design
- Zuschneiden auf Personas
- Testing

Thank you

Diskussion & Fragen