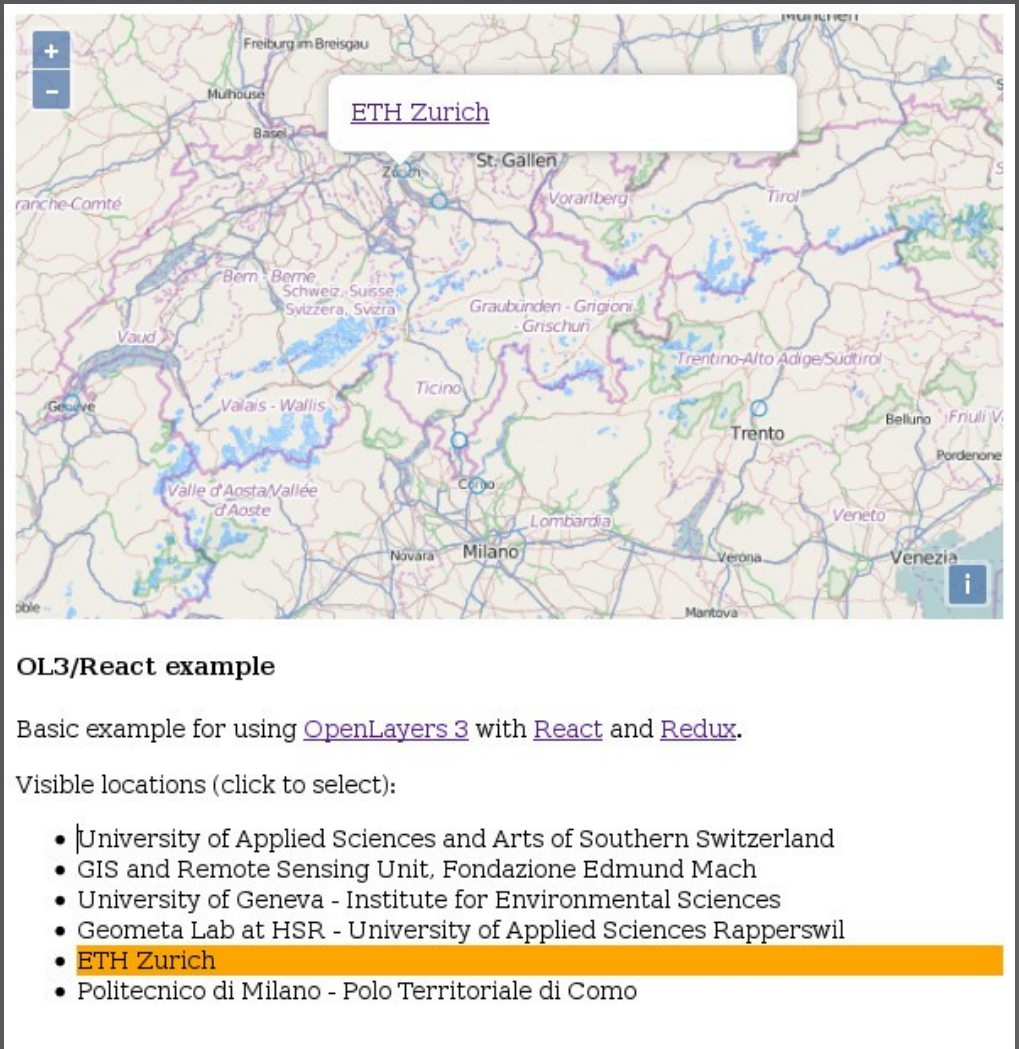
Building an OpenLayers 3 map viewer with React

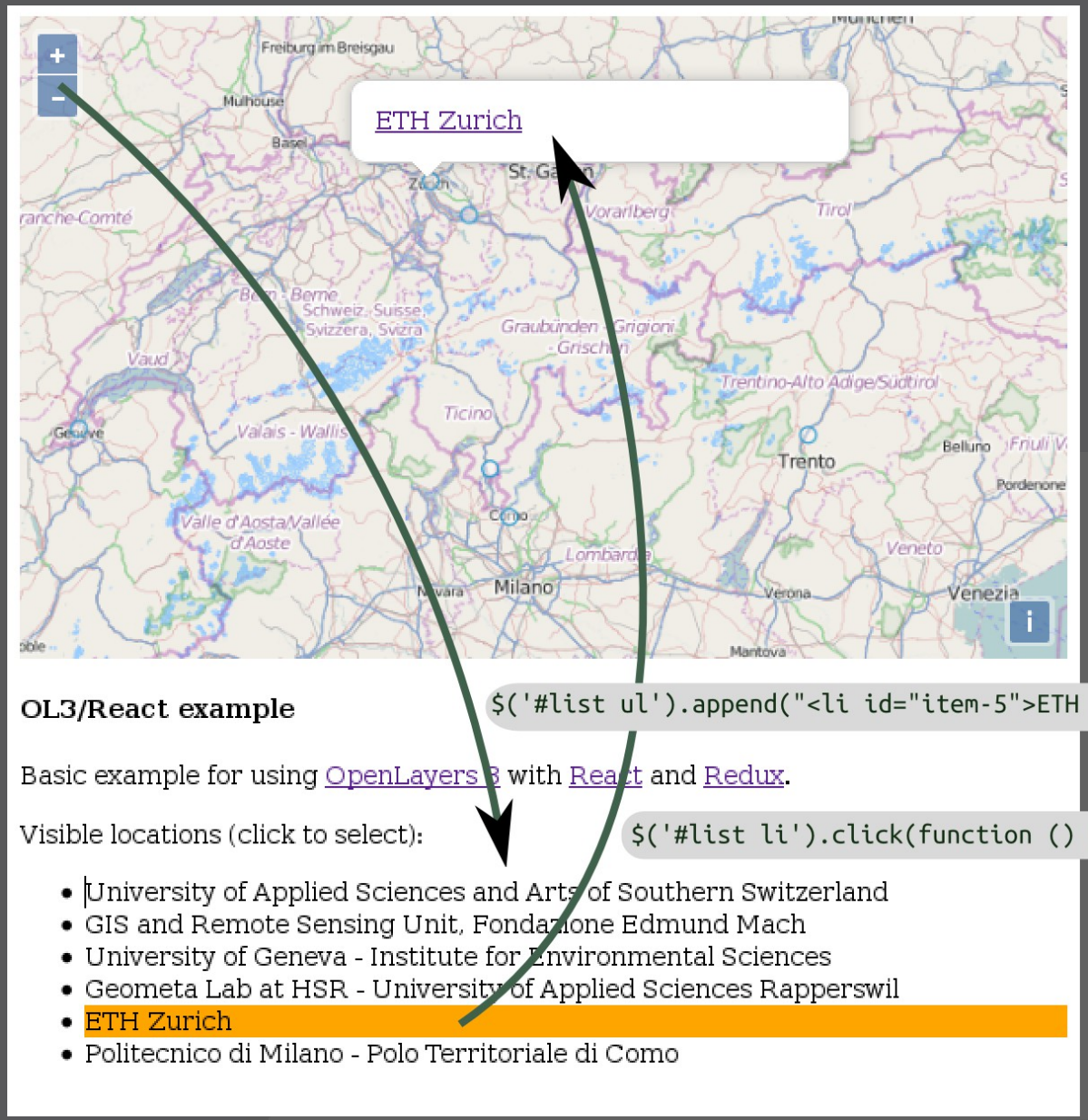
@PirminKalberer
Sourcepole AG, Switzerland
www.sourcepole.com

Map/Table Example

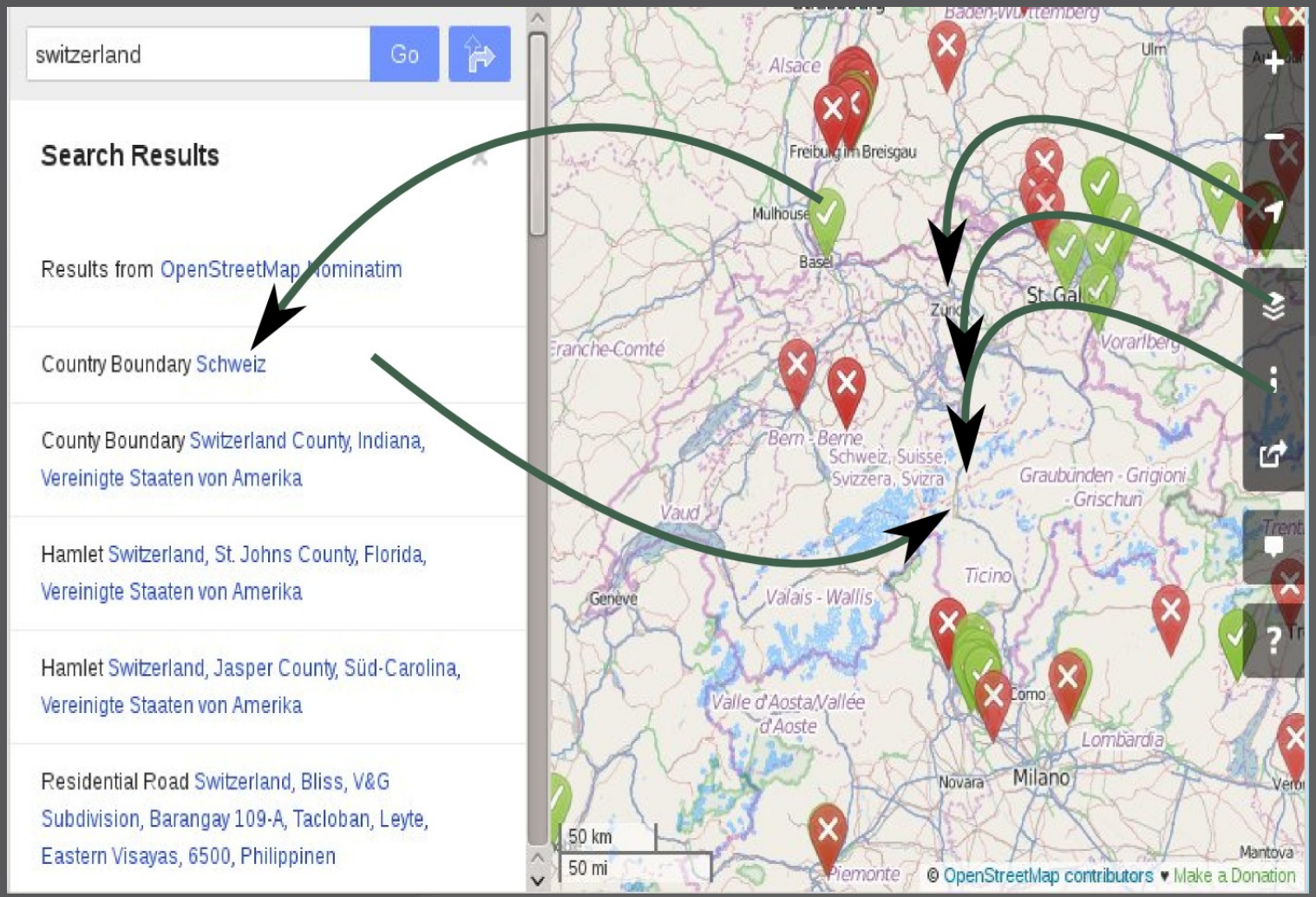


- https://github.com/pka/ol3-react-example

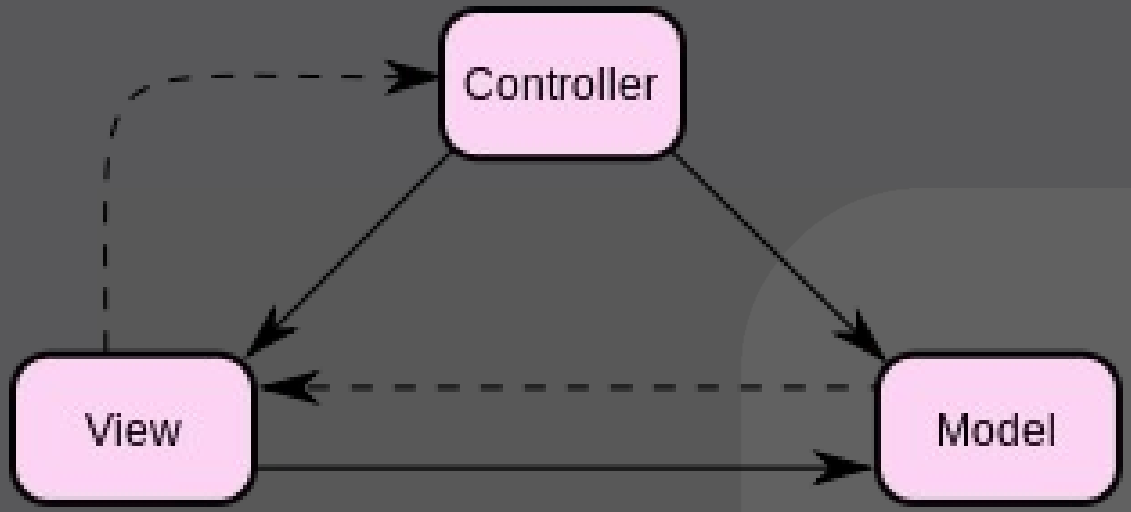
The „JQuery way“



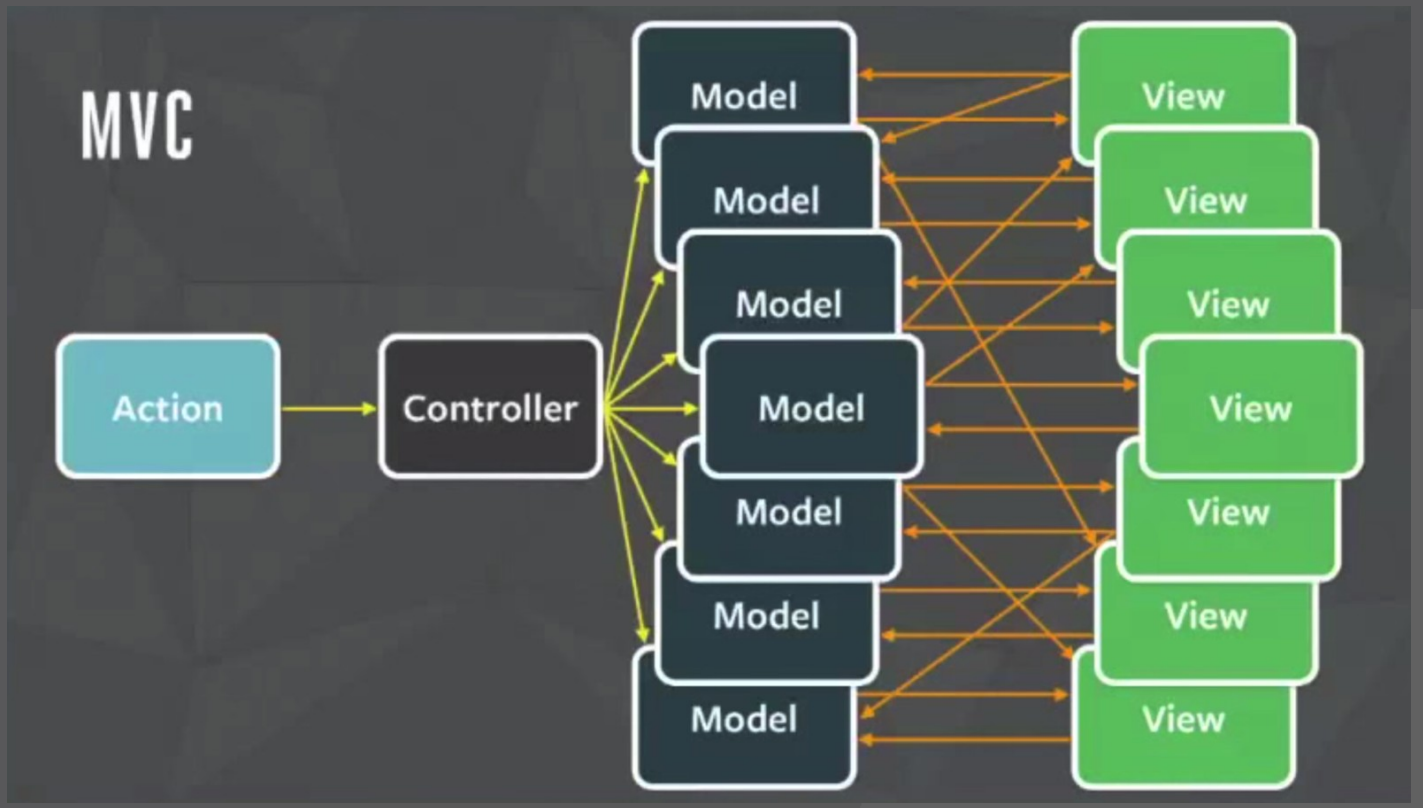
The „JQuery way“ more complex



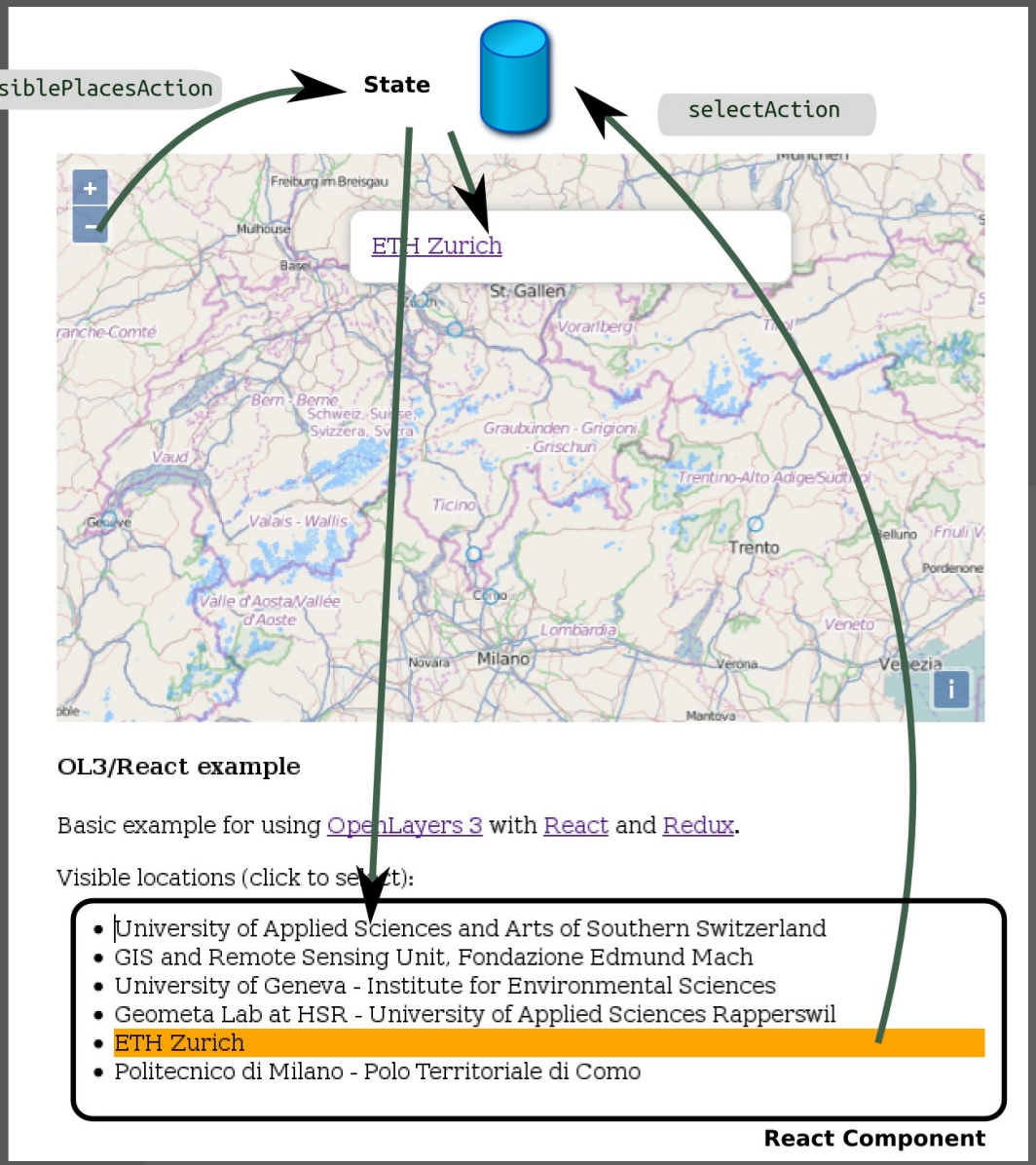
The MVC way



The MVC way – more complex

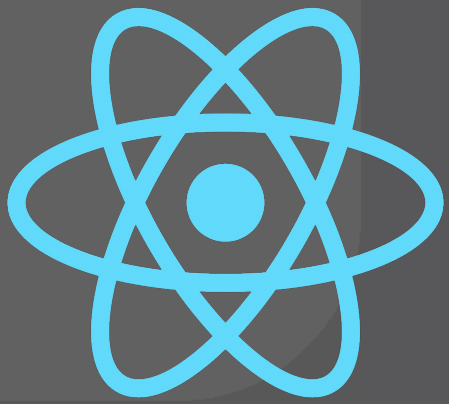


The React way



React

- http://facebook.github.io/react/
- The V in MVC
- Components, not templates
  - Display logic and markup are thighly coupled
- Re-render, don't mutate the DOM
- Fast Virtual DOM
- Components are reusable, composable, unit testable
- Concepts: https://youtu.be/x7cQ3mrcKaY



React Component

```
var MyComponent = React.createClass({
    render: function(){
        return (
            <h1>Hello {this.props.name}!</h1>
        );
    }
});

React.render(<MyComponent name="World" />,
             document.getElementById('myDiv'));
```

JSX

- **JSX:**

```
render () {
    return (
        <h1>Hello {this.props.name}!</h1>
    );
```

- **Javascript:**

```
render () {
    return React.createElement("div", null, "Hello ",
        this.props.name);
  }
```

Lists and events

```
var PlaceList = React.createClass( {
  render: function() {
    var placeNodes = this.props.places.map(function (place) {
      return (
        <li onClick={onSelectClick}>{place}</li>;
    };
    return (
      <ul>
        {placeNodes}
      </ul>
    );
  }
});
```

Composing components

- CommentBox
  - CommentList
    - Comment
  - CommentForm

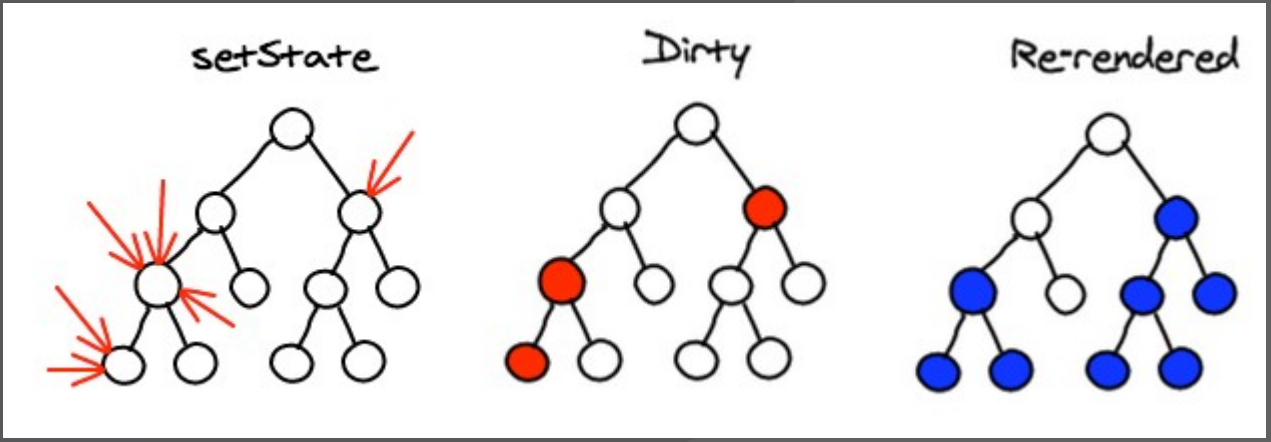
```
var CommentBox = React.createClass({
  render: function() {
    return (
      <div className="commentBox">
        <h1>Comments</h1>
        <CommentList />
        <CommentForm />
      </div>
    );
  }
});
```

Props / State

- **Props**
  - Read-only attributes
- **State**
  - State is set using the setState method. Calling setState triggers UI updates.
- **Re-render the whole app once the state changes**
- **Unidirectional Data Flow**
- **Data is guaranteed up to date**
- **Virtual DOM: makes re-rendering on every change fast**

Virtual DOM

- On every update React builds a new virtual DOM subtree
- … diffs it with the old one
- … computes the minimal set of DOM mutations and puts them in a queue
- … and batch executes all updates



Redux

- **Flux: application architecture**
  - Pattern rather than framework
  - Unidirectional data flow
  - http://facebook.github.io/flux/
- **Redux: “Reduced” Flux implementation**
  - http://rackt.github.io/redux/
- **Single store**
- **Central state → State history, etc.**
- **Middleware (Logging, etc.)**

React with OpenLayers

- **HTML:**

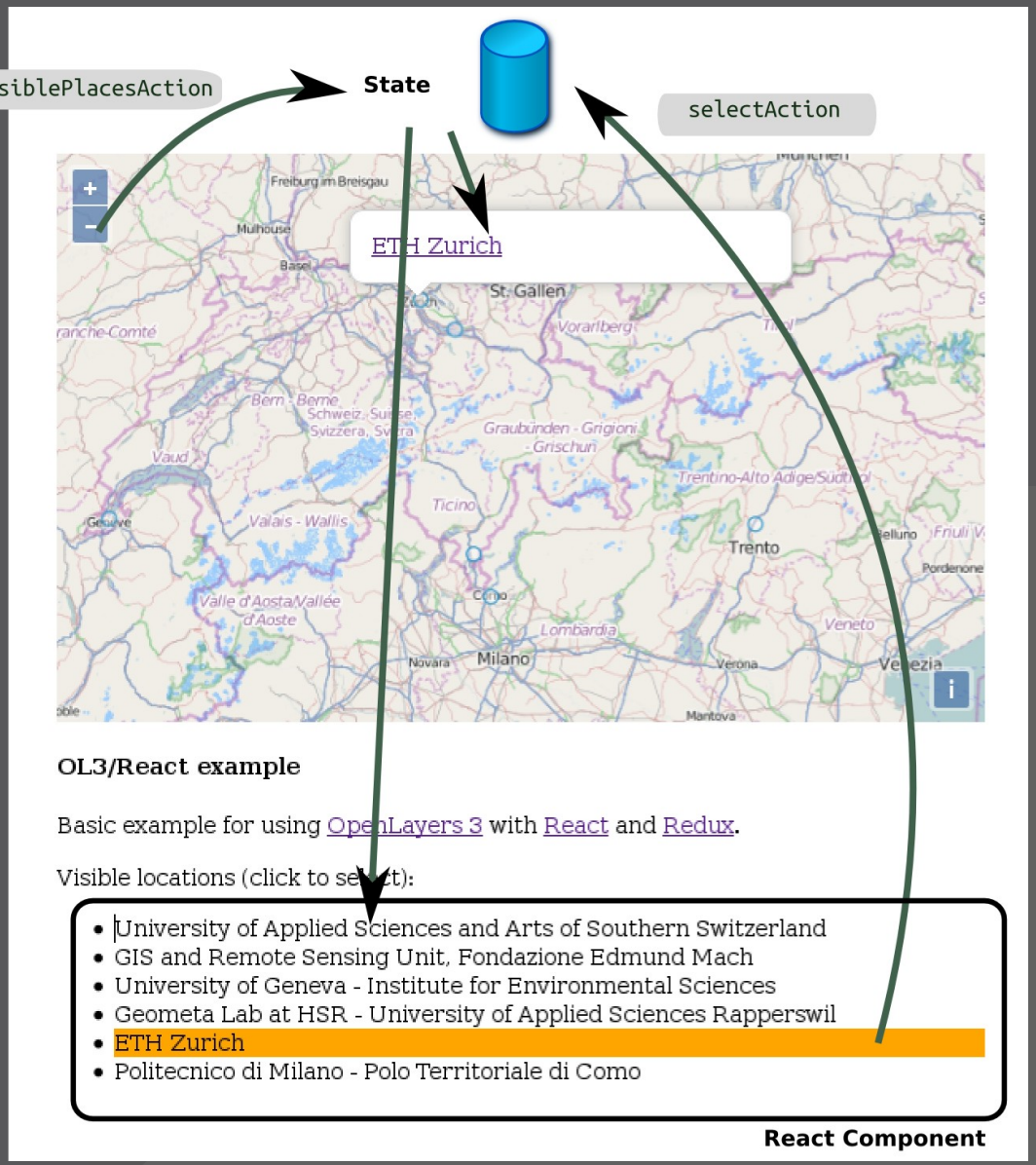
```
<body>
  <div id="olmap"/>
  <div id="reactcomponents"/>
</body>
```

- **React state:**
  - visible places
  - selected place

- **React component:**

```
var PlaceList = React.createClass( {
  render: function() {
    return (
      <ul>{...}</ul>
    );
```

Updating state



Updating State

- **OL → React State**

```
function updateVisiblePlaces() {
  var visiblePlaces =
    placeLayer.getSource().getFeaturesInExtent(extent) ...
  store.dispatch(visiblePlacesAction(visiblePlaces))
}
placeLayer.on('change', updateVisiblePlaces);
```

- **React Component → State + OL update**

```
onSelectClick: function(e) {
  dispatch(selectAction(itemId));
  // Update map
  updateSelection(itemId)
}
```

  - Alternative approach: use Redux middleware for updating state in OL

Hot reloading

- **Presentation at ReactEurope 2015: https://youtu.be/xsSn0QynTHs**
- **Save change in source (JS, CSS, …) → Updated view in browser**
  - Keeps application state
  - State history: Undo/Redo
  - State snapshots for unit testing

Complete code example

- https://github.com/pka/ol3-react-example

Thank you! - Questions?

@PirminKalberer
@sourcepole