AEM SPA Editor - Deep Dive - Part. 1 React

Patrick Fauchere

#AdobeRemix

Jon Noorlander

Agenda

- Project Structure
- AEM Page Component
- Content Services
- SPA Initialization
- Frontend Component
- Container components
- Model Manager and Provider
- App Routing and Model Router


Adobe
Project Structure
#AdobeRemix
Jon Noorlander

Project Structure

- Team and project composition
  - Single Project with modules
    - The frontend project is one of the modules
  - 2 projects
    - A project for the AEM modules
    - A project for the frontend App

Project Structure

- Breakdown
  - AEM area
    - Bundles
      - AEM Component **Sling Model**
      - **Servlets** (e.g. Server-Side Rendering)
    - Content
      - **AEM Component** HTL templates
      - Editable **Templates** and **Content Policies**
      - **Content Structure** and component data of the App
  - Frontend area
    - Frontend components **mapped** to AEM component resource types

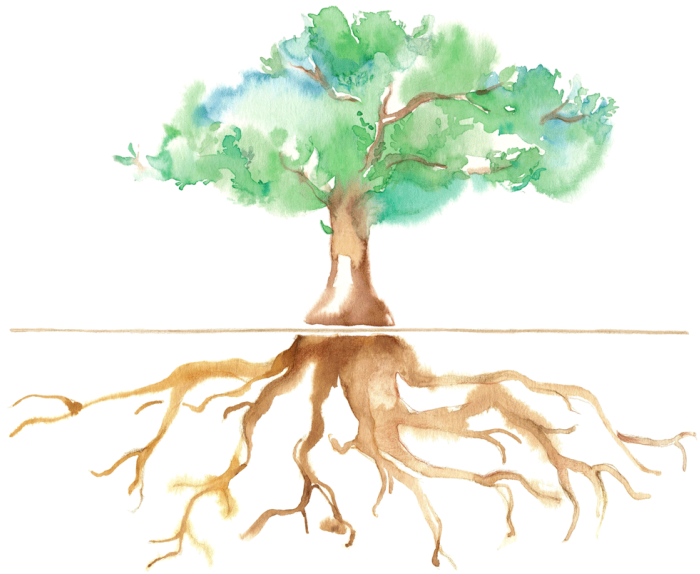
Project Structure

- Client library packaging and delivering strategies
  - Copy in the content directory (full-stack project structure type)
    - npm: *aem-clientlib-generator*
  - Direct Upload (any project structure type)
    - npm: *aemfed*


Adobe
AEM Page Component
#AdobeRemix
Jon Noorlander

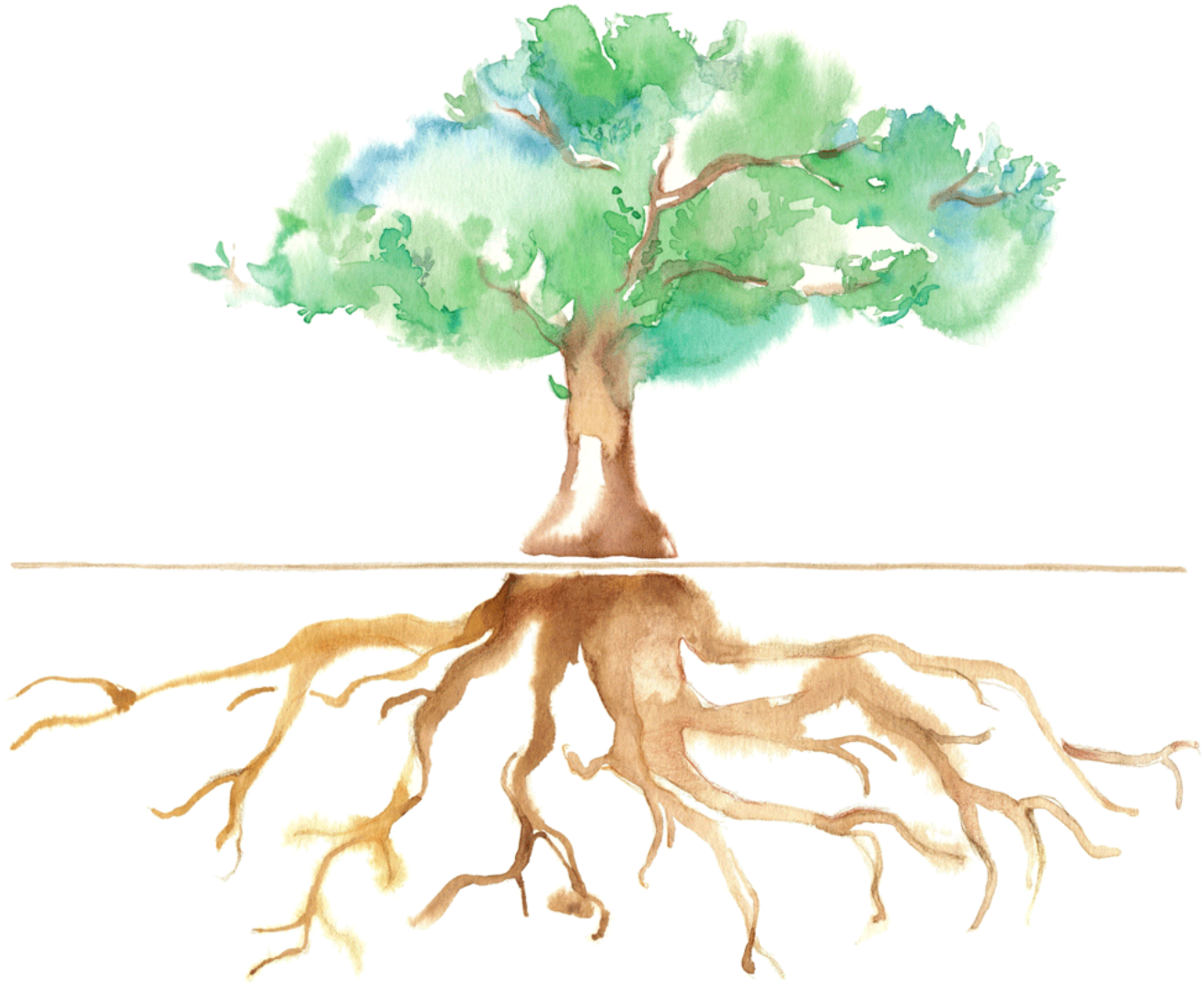
AEM Page Component

- HTL template
  - Meta properties
    - Initialize the PageModelManager
      - cq:datatype -> Format in which to fetch the data and to update the page model (JSON)
      - cq:wcmmode -> WCMMode edit or preview (Author instance only)
      - cq:pagemodel_root_url -> URL to the model of the App root (incl. selectors)



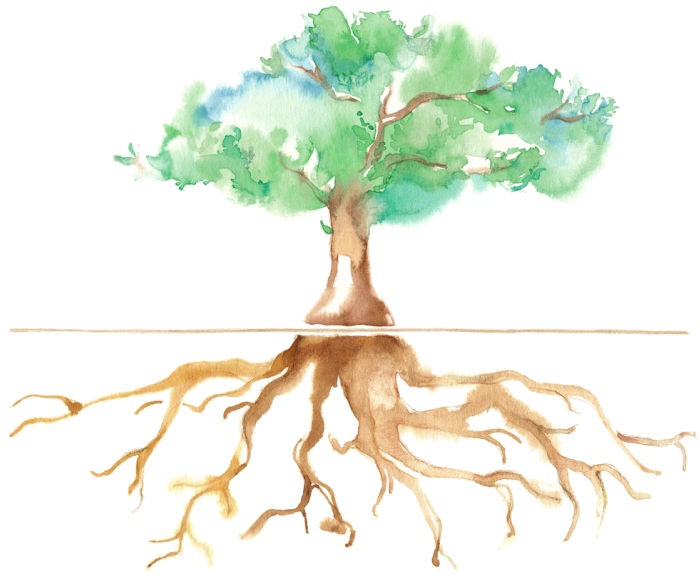
AEM Page Component

- Root model
  - Initial model of the SPA
    - Common content
    - Pre-loaded page models



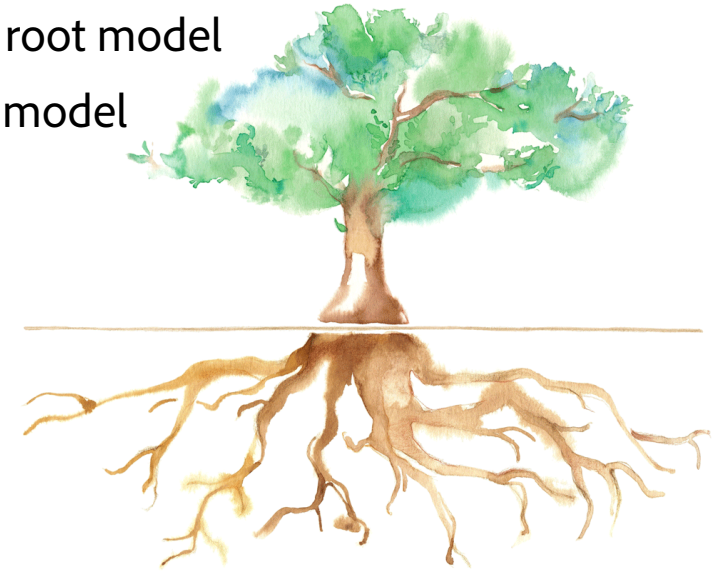
AEM Page Component

- HTL template
  - Client library categories
    - Files: customheaderlibs.html , customfooterlibs.html
    - **data-sly-call="${**clientLib.*js* **@** *categories*=**'cq.authoring.pagemodel.messaging'}"**
    - **data-sly-call="${**clientlib.<js|*css*> **@** *categories*=**'my-spa-category'}"**
  - Body of the page
    - File: body.html
    - <**div id="page"**></**div**>



AEM Page Component

- Hierarchy Page
  - Initial JSON representation of the App
  - Instant and asynchronous model
  - Flat list of child pages (Root model)
  - Content Policy properties
    - isRoot -> Identifies the page as the root of an App
    - structureDepth -> Maximum depth of the lookup for pages to be added to the root model
    - structurePatterns -> List of regular expressions to remove pages from the root model




Adobe
Content Services
#AdobeRemix
Jon Noorlander

Content Services

- Model Selector
  - .model.
    - Multiple selectors?
      - Model selector must come first
    - Default name
    - Also used by the Core Components

/content/my-site/my-page.model.json

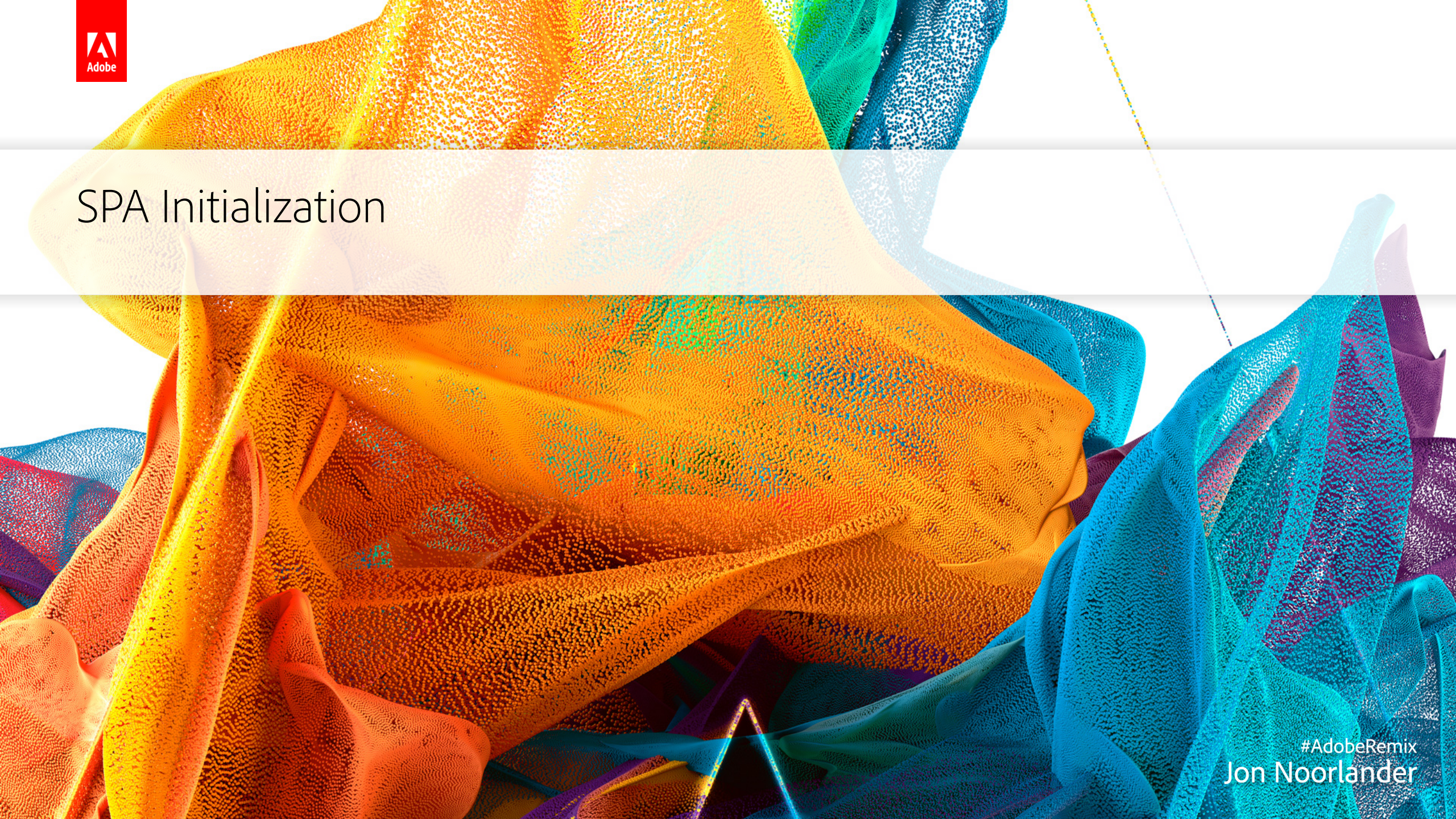
Content Services

- JSON representation of the content
  - Leverage the Sling Model Exporter
- Meta fields (reserved)
  - :type -> Resource type of an AEM component
  - :items -> Map of child components
  - :itemsOrder -> Ordered list of the item keys
  - :children -> Map of child pages
  - :path -> Path to the resource


```
{
  ":hierarchyType": "page",
  ":children": {
    "/content/we-retail-journal/react/home": { … }, // 6 items
    "/content/we-retail-journal/react/blog": { … }, // 6 items
    "/content/we-retail-journal/react/blog/aboutus": { … } // 6 items
  },
  ":path": "/content/we-retail-journal/react",
  "title": "We.Retail Journal",
  ":type": "we-retail-journal/react/components/structure/app",
  ":items": {
    "root": { … }, // 6 items
    "image": { … } // 1 item
  },
  ":itemsOrder": [
    "root",
    "image"
  ]
}
```

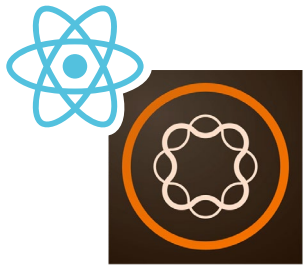
Content Services

- Sling Model
  - **Mandatory** for the SPA SDK
  - **Schema** of the component
  - **Business logic**
  - *export.json* API
    - ComponentExporter.**class**
    - ContainerExporter.**class**
    - HierarchyNodeExporter.**class**
  - Core components
    - Sling models
    - Use the *export.json* API


Adobe
SPA Initialization
#AdobeRemix
Jon Noorlander

SPA Initialization

- Explore the frontend project
- Project bootstrap
  - Initialize the **ModelManager**
    - Meta properties
    - URL of the root model
    - **ModelClient**
  - Instantiate the components



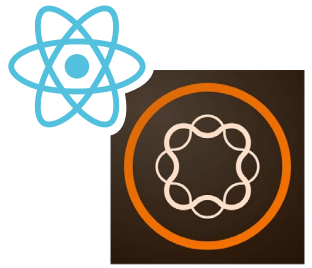
SPA Initialization

- Dynamic instantiation of the complete page

```
ReactDOM.render((<App cqChildren={pageModel[Constants.CHILDREN_PROP]}
cqItems={pageModel[Constants.ITEMS_PROP]}
cqItemsOrder={pageModel[Constants.ITEMS_ORDER_PROP]}
cqPath={pageModel[Constants.PATH_PROP]}/>), document.getElementById('page'));
```

Why do we have to set so many fields?

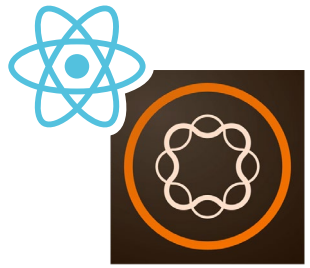
- *Done once when bootsrapping the App*
- *We already receive the model after the initialization of the ModelManager*
- *We are considering the introduction a helper*



SPA Initialization

- Static Instantiation of a component

```
// Component file
let MyEditableComponent = withAsyncModel(MapTo('component/resource/type')(MyComponent,
EditConfig));
export {MyEditableComponent};
...
// Bootsrap file
import {MyEditableComponent } from "./components/EditableComponent";
<MyEditableComponent cqPath={'/absolute/path/to/my/resource'}/>
```

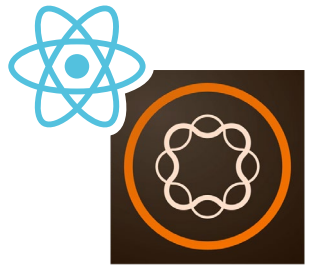



Adobe
Frontend Component
#AdobeRemix
Jon Noorlander

Frontend Component

- Register
  - MapTo
    - Maps an AEM component type to a Frontend component class
    - Store the Class in the ComponentMapping
    - Composition
      - withEditContext -> Authoring context
      - withModel -> Access to the model of a component
      - withEditable -> Decorate the component with the data that enables authoring capabilities

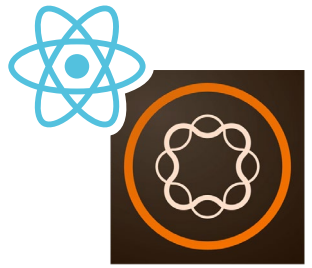
```
export MapTo('component/resource/type')(Component, EditConfig)
```



Frontend Component

- Configure the authoring experience
  - **EditConfig**
    - emptyLabel -> Label for the placeholder of an empty component
    - isEmpty -> Is the (AEM) component empty

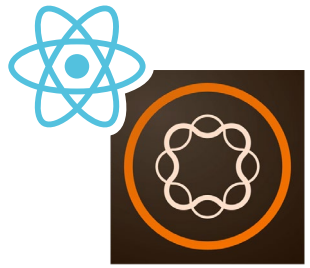
```
export MapTo('component/resource/type')(Component, EditConfig)
```

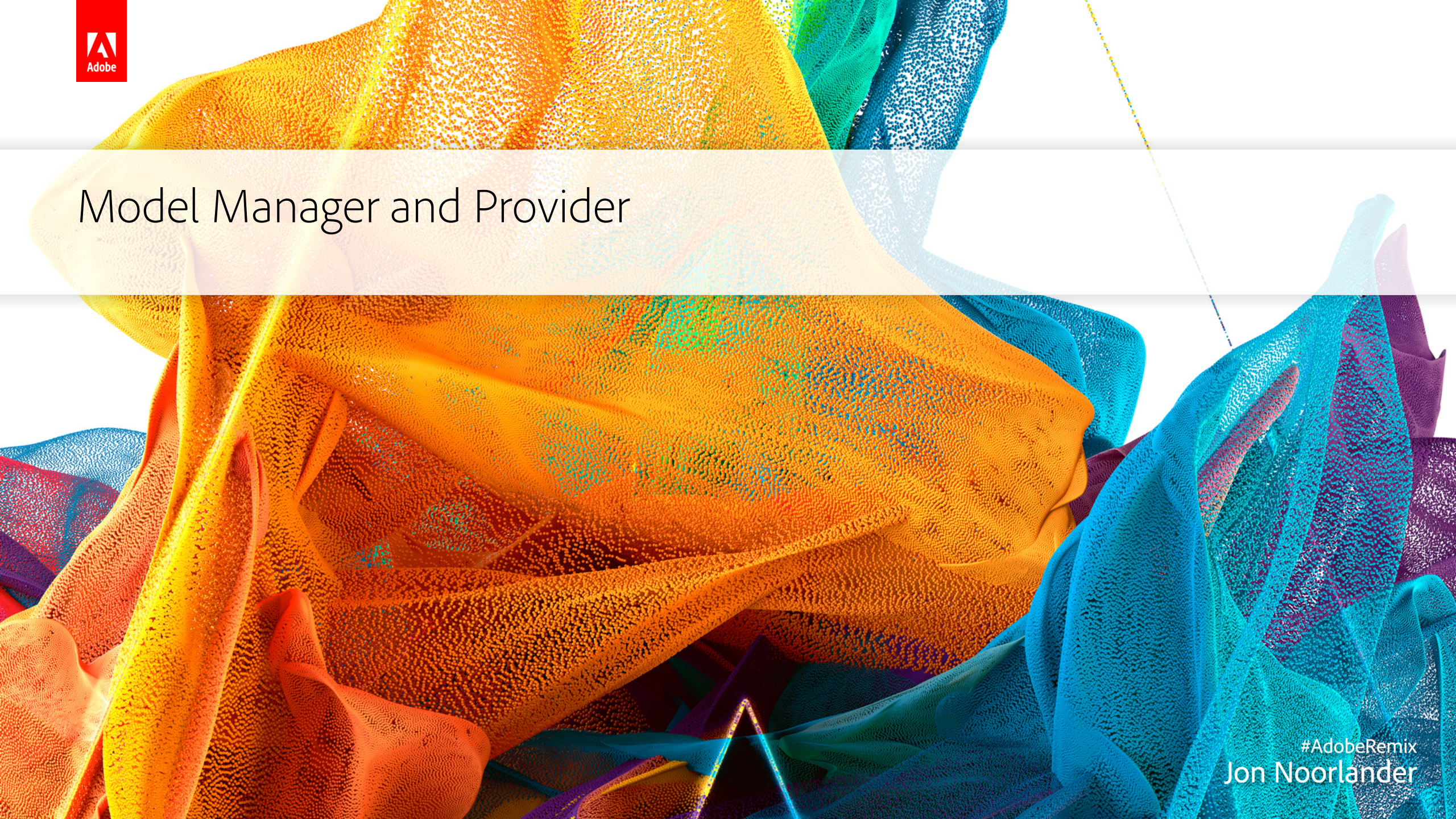



Adobe
Container components
#AdobeRemix
Jon Noorlander

Container components

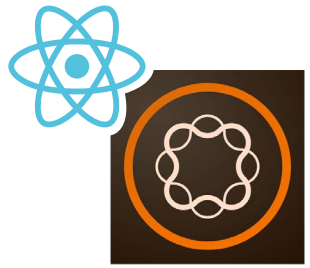
- Extensible components
- Container
  - Dynamic instantiation of components
- ResponsiveGrid
  - Provide access to the layout class names
- Page
  - Dynamic instantiation of child pages
    - Read the :children field of the model




Adobe
Model Manager and Provider
#AdobeRemix
Jon Noorlander

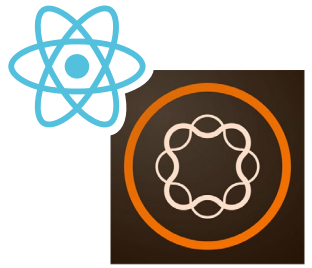
Model Manager and Provider

- ModelManager
  - **Stores** the model
  - **Caches** the model
  - Allows to **asynchronously load** fragments of model
  - Returns the **immutable** model of a component
  - Is updated by the Page Editor
  - Can **force reload** the model of a component



Model Manager and Provider

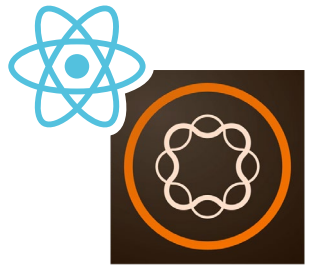
- ModelProvider
  - Listens to model changes by resource path
  - Model fields to component properties transformation
    - Utils.modelToProps(model)
      - *myProp -> this.props.myProp*
      - *:type -> this.props.cqType (reserved)*




Adobe
App Routing and Model Router
#AdobeRemix
Jon Noorlander

App Routing and Model Router

- App **Routing** relates to the **navigation** the App
- The Model Router
  - **Pre-fetch** fragments of model
  - Listen for calls to the **History API**
  - **Enabled** by default
- Meta properties to initialize the *ModelRouter* (Optinal)
  - cq:pagemodel_route_filters -> array of regular expressions to ignore routes
  - cq:pagemodel_router -> to disable the router



Resources

- Weekend Tutorial
  - https://helpx.adobe.com/experience-manager/kt/sites/using/getting-started-spa-wknd-tutorial-develop.html
- Documentation
  - https://helpx.adobe.com/experience-manager/6-4/sites/developing/using/spa-overview.html
- Sample – We retail Journal
  - https://github.com/adobe/aem-sample-we-retail-journal
- Client library packaging
  - https://www.npmjs.com/package/aem-clientlib-generator
  - https://www.npmjs.com/package/aemfed
- Core components delegation pattern
  - https://github.com/Adobe-Marketing-Cloud/aem-core-wcm-components/wiki/Delegation-Pattern-for-Sling-Models


Adobe