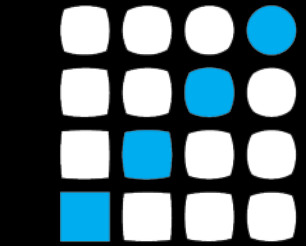


EUROPE'S LEADING AEM DEVELOPER CONFERENCE

28th – 30th SEPTEMBER 2020

Tracking Data with the Adobe Client Data Layer

Jean-Christophe Kautzmann, Laurentiu Magureanu, Benedikt Wedenik

Who are we?

Jean-Christophe Kautzmann

Software engineer at Adobe
AEM Sites

Laurentiu Magureanu

Software engineer at Adobe
AEM Commerce

Benedikt Wedenik

Technical consultant at Adobe
AEP, Analytics, Target and Launch

adaptTo()

Agenda

- The Adobe Client Data Layer
- Integration with the AEM Core Components
- Integration with custom components
- Integration with Adobe Launch
- Q & A

The Adobe Client Data Layer

adaptTo()

What is a data layer?

A data layer consists of a JavaScript client-side event-driven data store that can be used on web pages:

- to collect data about what the visitors experience on the web page;
- to communicate this data to digital analytics and reporting servers.

What does the Adobe Client Data Layer do?

The Adobe Client Data Layer is a JavaScript store for data and events happening on a page within the scope of a request. It provides an API to:

- Register data that is to be merged into the data layer state.
- Trigger events that relate to the data stored in the data layer.
- Get the current data layer state of all merged data.
- Register listeners that are called for specific events or data changes.

adaptTo()

Getting the Data Layer ready

- Loading the data layer script:

```
<script src="adobe-client-data-layer.min.js" defer async></script>
```

- Declaring the adobeDataLayer array:

```
window.adobeDataLayer = window.adobeDataLayer || [];
```

- Pushing a Data Object:

```
window.adobeDataLayer.push({
    "page": {
        "title": "Getting Started"
    }
});
```

adobeDataLayer.push()

- Pushing an Event Object:

```
window.adobeDataLayer.push({
    "event": "show",
    "eventInfo": {
        "reference": "component.accordion-1-item-2"
    },
    "component": {
        "accordion-1": {
            "shownItems": [
                "component.accordion-1-item-1",
                "component.accordion-1-item-2"
            ]
        }
    }
});
```

adobeDataLayer.push()

- Pushing an Object to Delete Data:

```
window.adobeDataLayer.push({
    "component": {
        "map-1": null,
        "accordion-1": {
            "shownItems": [
                undefined,
                "component.accordion-1-item-2"
            ]
        }
    }
});
```

adobeDataLayer.push()

- Pushing an Object to Update an Existing Array:

```
window.adobeDataLayer.push({
    "component": {
        "accordion-1": {
            "shownItems": [
                "component.accordion-1-item-3"
            ]
        }
    }
});
```

This command appends a new item to the shownItems array.

adobeDataLayer.push()

- Pushing a Function:

```
var myHandler = function(event) {
    console.log(event);
};
window.adobeDataLayer.push(function(dl) {
    dl.getState();
    dl.addEventListener("click", myHandler);
});
```

adobeDataLayer.getState()

- Getting the merged state:

```
window.adobeDataLayer.push(function(dl) {
    var state = dl.getState();
    console.log(state);
});
```

```
{
    "page": {
        "title": "Getting Started"
    },
    "component": {
        "hero-1": {
            "title": "Learn More",
            "link": "learn-more.html"
        },
        "accordion-1": {
            "title": "Step by step",
            "shownItems": [
                "component.accordion-1-item-2"
            ]
        },
        "accordion-1-item-1": {…},
        "accordion-1-item-2": {…}
    }
}
```

adobeDataLayer.getState()

- Getting the merged state of a specific part:

```
window.adobeDataLayer.push(function(dl) {
    var state = dl.getState("component.hero-1");
    console.log(state);
});
```

- Console output:

```
{
    "title": "Learn More",
    "link": "learn-more.html"
}
```

adobeDataLayer.addEventListener()

- Registering an Event Listener:

```
var myHandler = function(event) {
    console.log(event);
};
window.adobeDataLayer.push(function(dl) {
    dl.addEventListener(
        "adobeDataLayer:change",
        myHandler,
        {"path": "component.accordion-1"}
    );
});
```

adobeDataLayer.addEventListener()

- Console output:


```
{
    "component": {
        "accordion-1": {
            "title": "Step by step",
            "shownItems": [
                "component.accordion-1-item-1"
            ]
        },
        "accordion-1-item-1": {
            "title": "Step One",
            "parent": "component.accordion-1"
        },
        "accordion-1-item-2": {
            "title": "Step Two",
            "parent": "component.accordion-1"
        }
    }
}
{
    "event": "show",
    "eventInfo": {
        "reference": "component.accordion-1-item-2"
    },
    "component": {
        "accordion-1": {
            "shownItems": [
                "component.accordion-1-item-1",
                "component.accordion-1-item-2"
            ]
        }
    }
}
```

adobeDataLayer.removeEventListener()

- Unregistering all listeners for the adobeDataLayer:change event:

```
window.adobeDataLayer.push(function(dl) {
    dl.removeEventListener("adobeDataLayer:change");
});
```

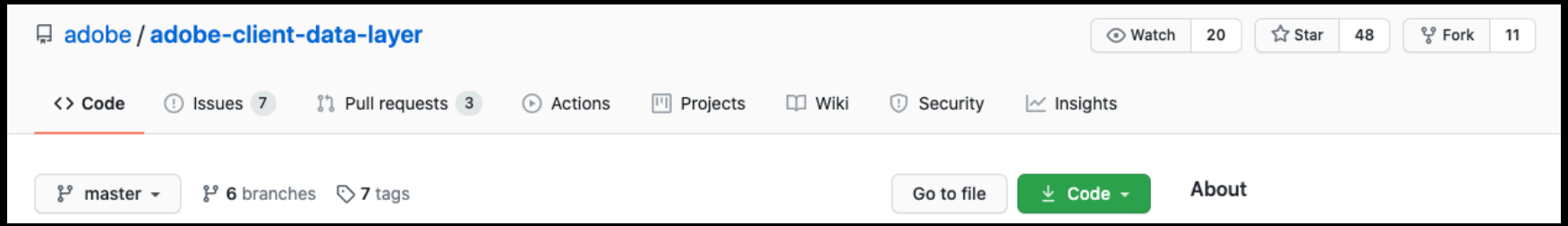
- Unregistering a specific listener for the click event:

```
window.adobeDataLayer.push(function(dl) {
    dl.removeEventListener("click", myHandler);
});
```

Contributions welcome!

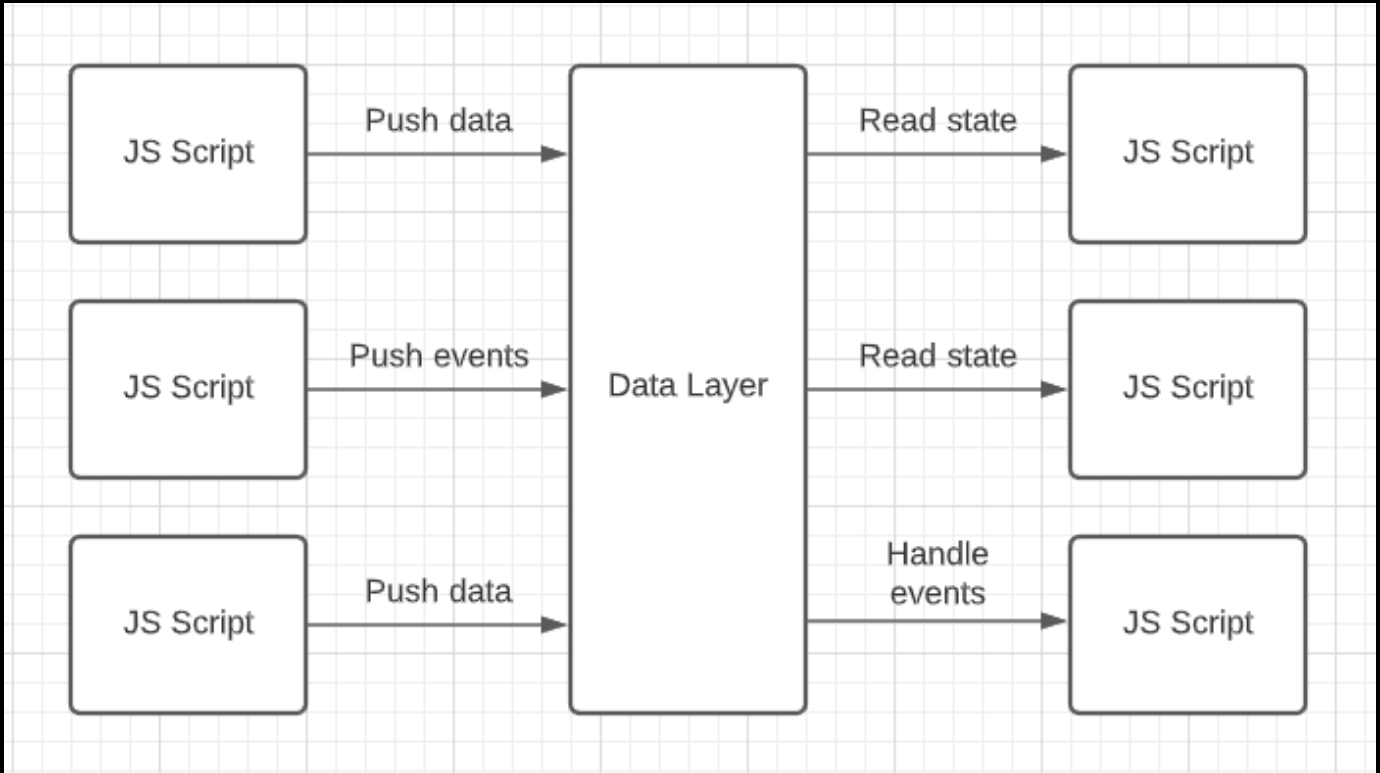
Feel free to contribute to the ACDL project (questions, issues, PRs, feedback, …):

https://github.com/adobe/adobe-client-data-layer

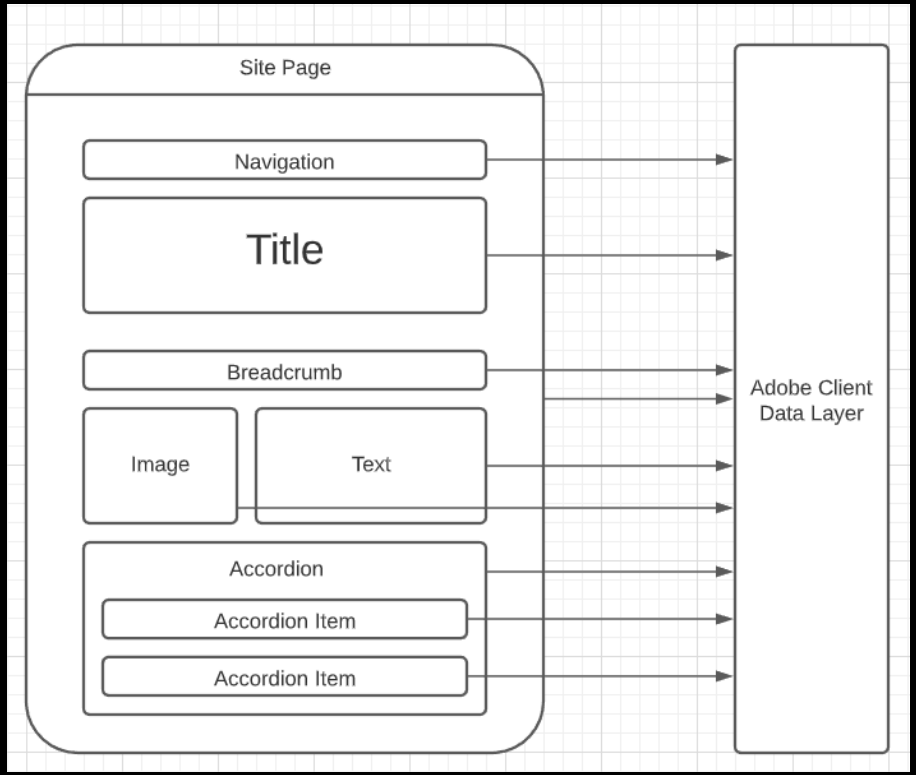


Integration with the AEM Core Components

Data Layer in a nutshell

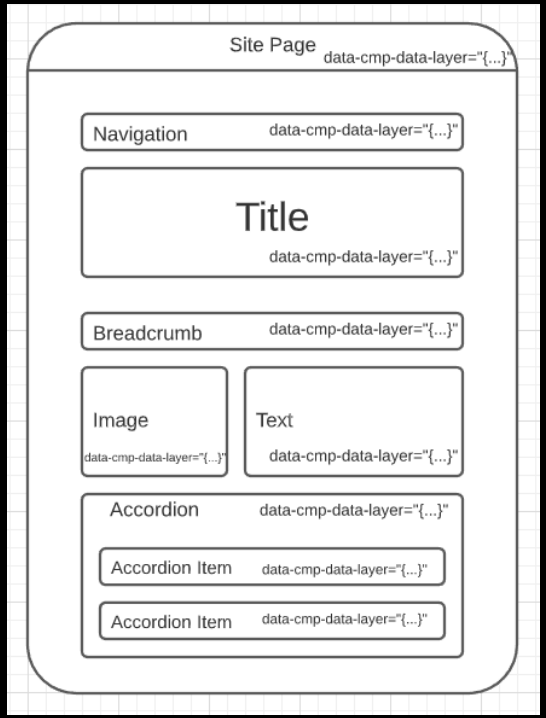


Core Components data push



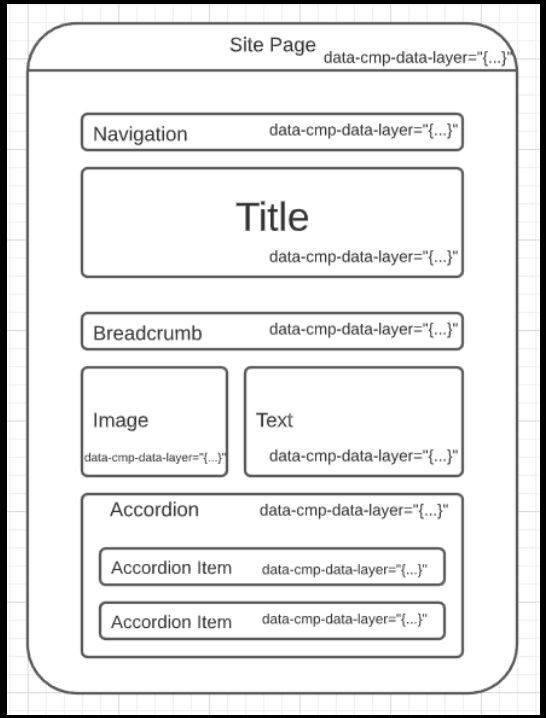
Client-side data generation

```
<div data-sly-use.accordion="com.adobe.cq.wcm.core.components.models.Accordion"
  id="${accordion.id}"
  class="cmp-accordion"
  data-cmp-is="accordion"
  data-cmp-data-layer="${accordion.data.json}">
    ...
</div>
```



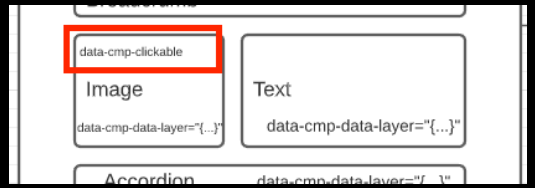
JS pushing component data

```
var components = document.querySelectorAll("[data-cmp-data-layer]");

components.forEach(function(component) {
  addComponentToDataLayer(component);
});

dataLayer.push({
  event: "cmp:loaded"
});
```



Registering event handlers

```
<a data-sly-unwrap="${!image.link}"
  class="cmp-image__link"
  href="${image.link}"
  data-cmp-clickable="${image.data ? true : false}"
  data-cmp-hook-image="link">
    ...
</a>
```




```
var clickableElements = document.querySelectorAll("[data-cmp-clickable]");

clickableElements.forEach(function(element) {
  attachClickEventListener(element);
});
```

Components XDM schemas

```
Page data structure:
{
  @type             // resource type
  repo:modifyDate   // last modified date
  dc:title          // title
  dc:description    // description
  xdm:text          // text
  xdm:linkURL       // link URL
  xdm:tags          // page tags
  repo:path         // page path
  xdm:template      // page template
  xdm:language      // page language
}
```

https://github.com/adobe/xdm/blob/master/docs/reference/content/componentized-page.schema.md

How it’s generated

```
public abstract class AbstractComponentImpl implements Component
{
  ...
  @Nullable
  public ComponentData getData() {
    ...
    return componentData;
  }

  @NotNull
  protected ComponentData getComponentData() {
    return new ComponentDataImpl(this, resource);
  }
  ...
}
```

```
data-cmp-data-layer="${accordion.data.json}"
```

How it’s generated

```
public class ComponentDataImpl implements ComponentData
{
  public ComponentDataImpl(AbstractComponentImpl component, Resource resource)
  {
    this.component = component;
    this.resource = resource;
  }

  public String getType() {
    return resource.getResourceType();
  }

  public String getTitle() {
    return component.getDataLayerTitle();
  }
}
```

Enabling Data Layer integration

The Adobe Client Data Layer is disabled by default. To enable it:

- Create a sling config under */conf/<my-site>/sling:configs/com.adobe.cq.wcm.core.components.internal.DataLayerConfig*.
- Add the enabled *boolean* property and set it to *true*.
- Add a *sling:configRef* property to the *jcr:content* node of your site.

adaptTo()

Integration with Custom Components

Data Layer for Custom Components

In order to add component data to the data layer:

- Leverage the existing Core Components integration.
- Add your own custom logic.

Using existing integration

To automatically add a custom component to the data layer:

- Define the properties of the custom component model that needs to be tracked.
- Add a component *ID* to the the custom component HTL.
- Add the *data-cmp-data-layer* attribute to the custom component HTL.
- Generate the JSON data structure in your model

Custom HTL data generation

```
<div data-sly-use.mycomp="com.example.custom.Model"
  id="${mycomp.id}"
  class="cmp-custom-model"
  data-cmp-is="custom-component"
  data-cmp-data-layer="${mycomp.data.json}">
    ...
</div>
```

Custom component data

```
public class CustomComponentDataImpl implements ComponentData
{
  public CustomComponentDataImpl(Component customComponent, Resource resource) {
    ...
  }
  @Override
  public String getTitle() {
    return customComponent.getTitle();
  }
  ...
  @Override
  public String getJson() {
    // return a JSON like {componentId: { componentDataJson }}
  }
  ...
}
```

Custom component model

```
public class CustomModel implements Component
{
  ...
  @Nullable
  public ComponentData getData() {
    ...
    componentData = getComponentData();
    return componentData;
  }

  @NotNull
  protected ComponentData getComponentData() {
    return new CustomComponentDataImpl(this, resource);
  }
  ...
}
```

Data Layer Events for Custom Components

To leverage existing integration for events:

- In the custom component HTL add the *data-cmp-clickable* attribute to the element to be tracked.
- Make sure the component HTL has an *ID* on the top DOM element.

Custom events (*show, hide, etc*) should be triggered by component *clientlibs*

Integration with Adobe Launch

ACDL Launch Extension

The ACDL Launch Extension enables to:

- Rename the adobeDataLayer object
- Load the Adobe Client Data Layer script
- Listen to Data Layer push events
- Retrieve the Computed State of the Data Layer
- Retrieve a specific element from the Data Layer by path

Live Demo

ACDL Launch Extension

To install the Adobe Client Data Layer Extension, navigate to the Extension catalogue in Launch Extension and select the Adobe Client Data Layer.

Launch Extension: Events

The Launch Extension enables to listen to following events:

- Data changes
- All events pushed to the data layer
- Specific events based on the event name
- Past and/or future events

Launch Extension: Actions

Following actions are supported:

- Reset Data Layer
- Push JSON content to the Data Layer

Launch Extension: Data Elements

Following Data Elements are available:

- Computed State: returns the computed state of either the complete Data Layer or of a part of it defined by its path
- Data Layer Size: returns the number of elements pushed to the Data Layer

adaptTo()

Resources

- Adobe Client Data Layer: https://github.com/adobe/adobe-client-data-layer
- Integration with the Core Components: https://github.com/adobe/aem-core-wcm-components/blob/master/DATA_LAYER_INTEGRATION.md
- ACDL Launch Extension – not officially released yet
- Collect page data with Adobe Analytics https://docs.adobe.com/content/help/en/experience-manager-learn/sites/integrations/analytics/collect-data-analytics.html

Q & A

Thank You!