20/03/2025 - TASKS

Table of Contents

---

1. Develop a Custom Widget Component

AEM supports custom components for dynamic content management. The Widget component enables users to input details such as name, description, and date.

Steps:

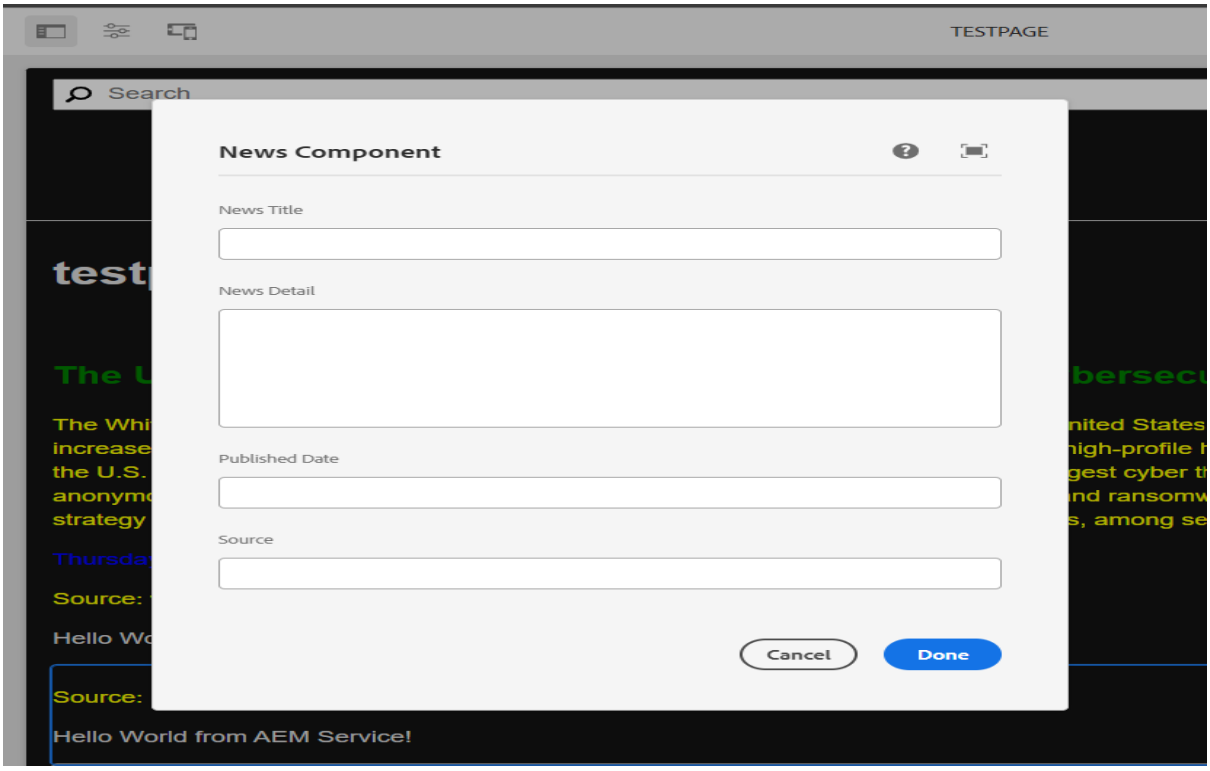
- **Open CRXDE Lite (http://localhost:4502/crx/de).**
- **Navigate to /apps/myProject/components and create a folder named widget.**
- **Inside widget, define component files including widget.html.**
- **Fetch content dynamically using repository properties.**
- **Save and activate for usage.**

Code Snippet:

widget.html

```
<sly data-sly-use.clientLib="/libs/granite/sightly/templates/clientlib.html"/>
<sly data-sly-call="${clientLib.css @ categories='myProject.widget'}"/>
<div class="widget-container">
  <h2 class="widget-title">${properties.name}</h2>
  <p class="widget-desc">${properties.description}</p>
  <p class="widget-date">${properties.date}</p>
```

<sly data-sly-call="${clientLib.js @ categories='myProject.widget'}"/>



---

2. Implement a Multifield Feature

A multifield component allows authors to input multiple data entries dynamically.

Steps:

- **Create a new component named widgetMultifield under /apps/myProject/components.**
- **Define a cq:dialog with a multifield widget.**
- **Configure it to accept dynamic inputs.**
- **Save and activate the component.**

Code Snippet:

WidgetListModel.java

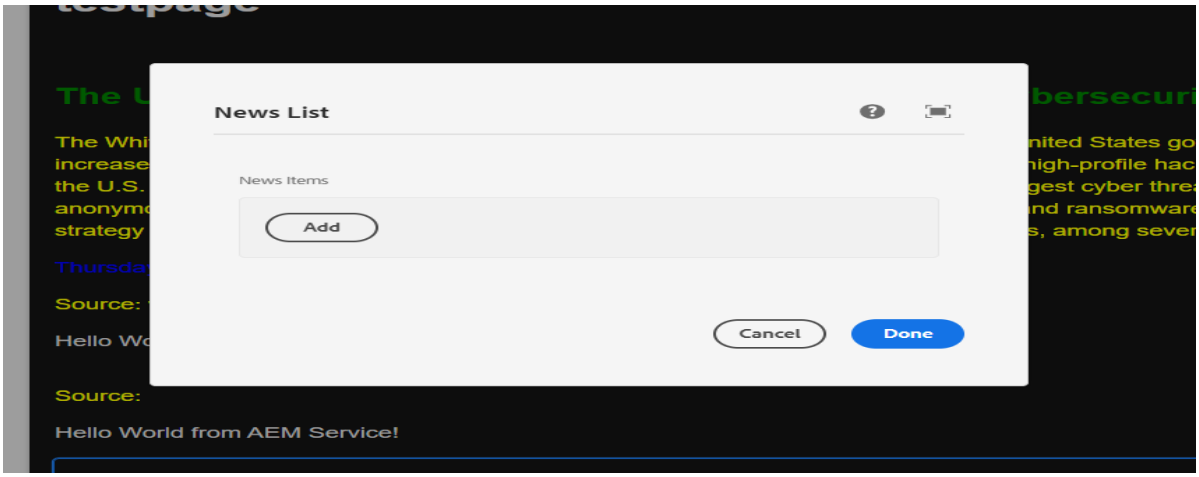
@Model(adaptables = Resource.class, defaultInjectionStrategy = DefaultInjectionStrategy.OPTIONAL)

```
public class WidgetListModel {
    @Inject
    private List<WidgetItem> widgetItems;

    public List<WidgetItem> getWidgetItems() {
        return Optional.ofNullable(widgetItems).orElse(new ArrayList<>());
    }

    public static class WidgetItem {
        @Inject
        private String name;
        @Inject
        private String description;

        public String getName() {
            return name;
        }
        public String getDescription() {
            return description;
        }
    }
}
```



---

3. Add Clientlibs for Styling and Scripts

Clientlibs in AEM help manage styles and scripts efficiently.

Steps:

- **Go to /apps/myProject/clientlibs and create a folder named clientlib-widget.**
- **Define CSS styles for:**
  - **Red headings**
  - **Blue descriptions**
  - **Black date text**
- **Link clientlib to the widget component.**
- **Save and activate.**

Code Snippet:

```
.widget-container h2 {
   color: red;
}


.widget-container p {
   color: blue;
```

}

.widget-container .widget-date {

color: black;

}

---

4. Create a Reusable Page Structure

A Base Page Component allows for a standardized structure across multiple pages.

Steps:

- **Navigate to /apps/myProject/components and create basepage.**
- **Define a cq:dialog for configurable properties.**
- **Structure the template with head, body, and footer sections.**
- **Save and activate.**

Code Snippet:

<jcr:root xmlns:sling="http://sling.apache.org/jcr/sling/1.0"

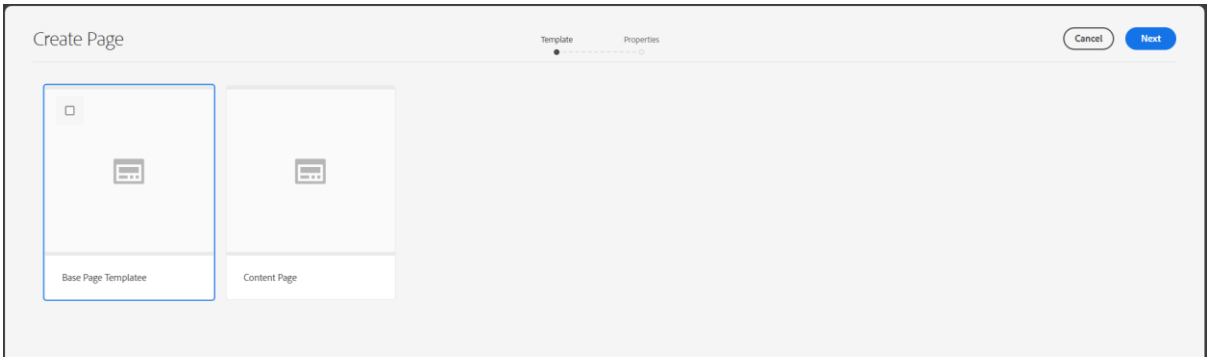
xmlns:cq="http://www.day.com/jcr/cq/1.0"

xmlns:jcr="http://www.jcp.org/jcr/1.0"

jcr:primaryType="cq:Component"

jcr:title="Base Page"

componentGroup="myProject"/>



5. Configure Site-wide Settings

Global Page Properties allow managing branding elements like site titles and logos.

Steps:

- **Create a template-type component for global configurations.**
- **Define a cq:dialog for site-wide settings.**
- **Ensure properties can be accessed dynamically.**
- **Save and activate.**

Code Snippet:

```
<jcr:root xmlns:sling="http://sling.apache.org/jcr/sling/1.0"
        xmlns:cq="http://www.day.com/jcr/cq/1.0"
        xmlns:jcr="http://www.jcp.org/jcr/1.0"
        jcr:primaryType="cq:Component"
        jcr:title="Global Configurations"
        componentGroup="myProject"/>
```

cq:dialog

```
<jcr:root xmlns:sling="http://sling.apache.org/jcr/sling/1.0"
        xmlns:cq="http://www.day.com/jcr/cq/1.0"
        xmlns:jcr="http://www.jcp.org/jcr/1.0"
        jcr:primaryType="nt:unstructured">
   <items jcr:primaryType="nt:unstructured">
      <siteTitle jcr:primaryType="nt:unstructured"
sling:resourceType="granite/ui/components/coral/foundation/form/textfield"
fieldLabel="Site Title" name="./siteTitle"/>
      <siteLogo jcr:primaryType="nt:unstructured"
sling:resourceType="granite/ui/components/coral/foundation/form/pathfield"
fieldLabel="Site Logo Path" name="./siteLogo"/>
```

```
    </items>
</jcr:root>
```

---

6. Understanding extraClientLibs Usage

extraClientLibs is a property in AEM for dynamic loading of additional scripts and styles.

Use Cases:

- **Loading specific styles/scripts only for targeted templates or components.**
- **Extending client libraries without modifying core configurations.**

Implementation Steps:

- **Define extra client libraries inside the component/template.**
- **Configure extraClientLibs to load required CSS/JS.**
- **Ensure dependencies are managed properly.**
- **Save and verify integration.**