AEM

Contents

[**Setup** 1](#_Toc163047073)

[**Setup enviproment** 1](#_Toc163047074)

[**Debuger in aem set up** 2](#_Toc163047075)

[**Add adobe repo tool** 3](#_Toc163047076)

[**Component and template** 6](#_Toc163047077)

[**Component** 6](#_Toc163047078)

[**Template** 8](#_Toc163047079)

[**Aem Concept** 8](#_Toc163047080)

[**Package Manager** 8](#_Toc163047081)

[**Custom Logger** 8](#_Toc163047082)

[**System User** 9](#_Toc163047083)

[**I18n** 9](#_Toc163047084)

[**Custom Error** 11](#_Toc163047085)

[**Workflow** 13](#_Toc163047086)

[**Client Lib** 16](#_Toc163047087)

[**Event** 17](#_Toc163047088)

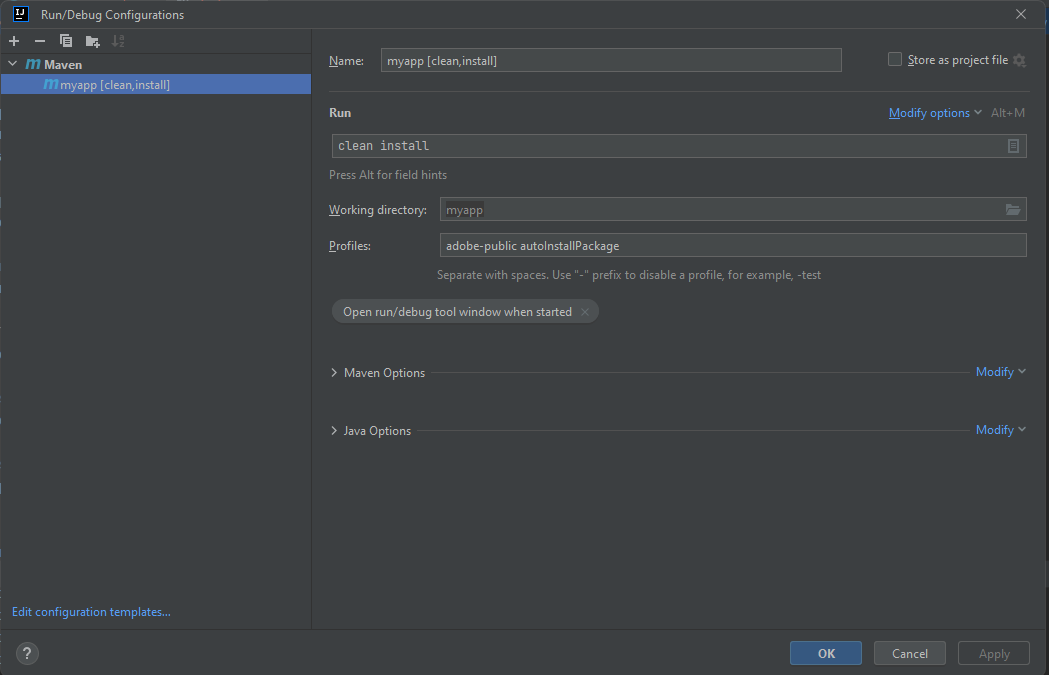
[**HTL** 17](#_Toc163047089)

# **Setup**

## **Setup enviproment**

**Create AEM with maven:**

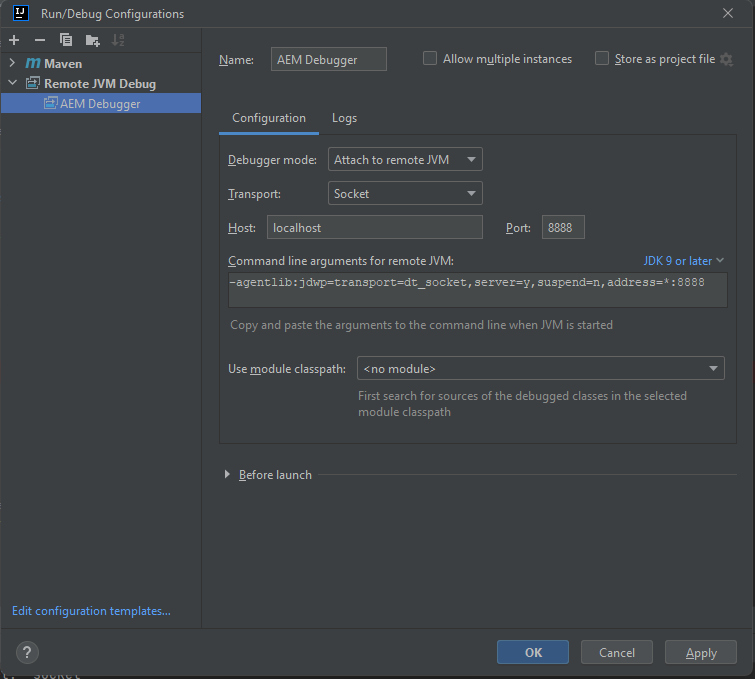
mvn -B archetype:generate -D archetypeGroupId=com.adobe.aem -D archetypeArtifactId=aem-project-archetype -D archetypeVersion=26 -D appTitle="My App Aem Tutorial" -D appId="myapp" -D groupId="com.myapp" -D frontendModule="none" -D aemVersion=6.5.0

after create: mvn install -> set up maven Run 

## **Debuger in aem set up**

Code run debuger in aem : java -Xmx2048M -agentlib:jdwp=transport=dt\_socket,address=8888,server=y,suspend=n -jar aem-author-p4502.jar

Setup intejj Debugger



Take pointer in to there want debug and restart page constain component or service

## **Add adobe repo tool**

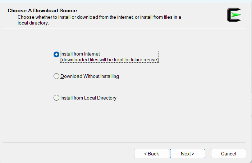
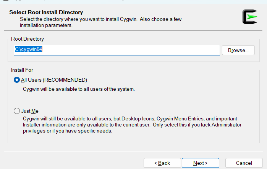
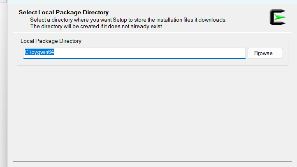
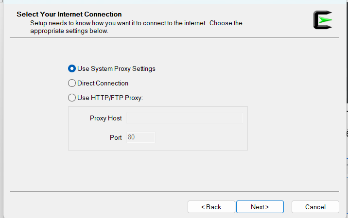
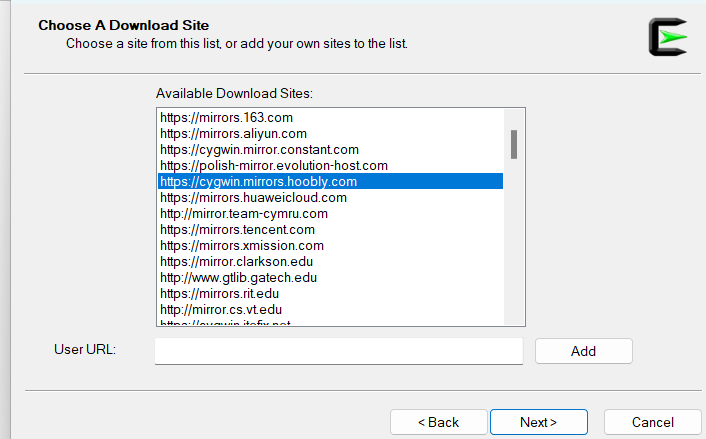
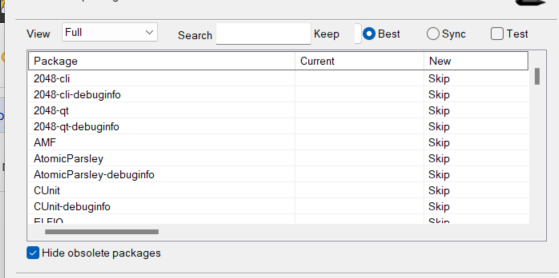
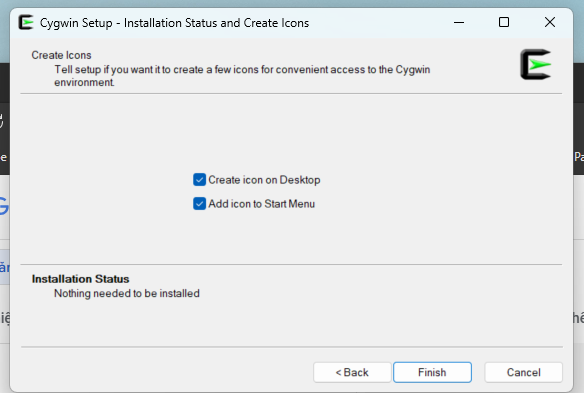
Link download: [tools/repo at master · Adobe-Marketing-Cloud/tools · GitHub](https://github.com/Adobe-Marketing-Cloud/tools/tree/master/repo)

Setup In intelejj:

File -> setting -> tools -> External tools

|  |  |
| --- | --- |
|  | Các method |
|  | Program (Cygwin64): C:\cygwin64\bin\bash.exe  Argument : -l F:\work-space\aem\tools-repo-1.4\repo\repo get -f "$UnixSeparators($FilePath$)$"  Working directory: $ProjectFileDir$ |
|  | Program(Cygwin64): C:\cygwin64\bin\bash.exe  Argument: -l F:\work-space\aem\tools-repo-1.4\repo\repo put -f "$UnixSeparators($FilePath$)$"  Working directory: $ProjectFileDir$ |

Program: đường dẫn của Cygwin64(là environment chứa các thư viện để chạy repo tool) bạn cần tải và cofig các thư viện cần thiết

Set Cygwin64: -> ->->-> chọn <https://cygwin.mirrors.hoobly.com> ->  search thử viện cần tải chọn phiên bản và tích vào next ->finish

Library (thiếu thì khi chạy repo sẽ lỗi có thể add thử viện theo lỗi):

* Zip
* Curl

# **Component and template**

## **Component**

**Example :**

* **package com.myapp.core.models.** **GridImageModel (**List Nested list)(path in to DAM Assest to get link image)
* **package com.myapp.core.models.NavModel (List)**
* Component Content/tab **(dialog tab and list)(to handle you need write model to map with attribute on dialog)**

Key word:

* Sling model ( to create model)
* Resource ( to get Resource on Tree structure of base resouce)
* [Form — Granite UI 1.0 documentation (adobe.com)](https://developer.adobe.com/experience-manager/reference-materials/6-5/granite-ui/api/jcr_root/libs/granite/ui/components/coral/foundation/form/index.html) ( To write Dialog in XML for Component)

|  |  |
| --- | --- |
| **@Model** | This is the primary annotation used to declare a **Sling Model**. It is used at the class level to define the adaptable classes (usually Resource or SlingHttpServletRequest) and the resource types that the model is applicable for |
| **@Inject** | Used to inject a property value from the current resource or request into a field or method. It supports injection of simple types, arrays, collections, and even other Sling Models. |
| **@Optional** | This annotation is used alongside **@Inject** to indicate that the injection is not mandatory. If the property is not found, the field will be left null (or with its default value) without causing an exception. |
| **@Named** | Allows specifying the exact property name to be injected, especially useful when the field name in the Java model does not match the property name in the JCR |
| **@Default** | Provides default values for injected fields if the property is not present in the resource. It can be used with simple types and arrays |
| **@Via** | Specifies an intermediary to use when injecting a property. It can be used to adapt to another type or use a resource superType for injection |
| **@PostConstruct** | Designates a method that should be called after all injection has occurred but before the model is returned for use. It's useful for initialization logic that requires injected fields to be set |
| **@ResourcePath** | Allows injecting a resource into a model field based on a path. This is particularly useful for loading fixed resources. |
| **@ChildResource** | Injects a child resource relative to the current resource. Useful for hierarchical data structures. |
| **@Self** | Injects the adaptable instance itself, often used to re-inject a request or resource with different adaptations. |
|  |  |

Note: SlingHttpServletRequest and Sling.Resource

// not unit test

## **Template**

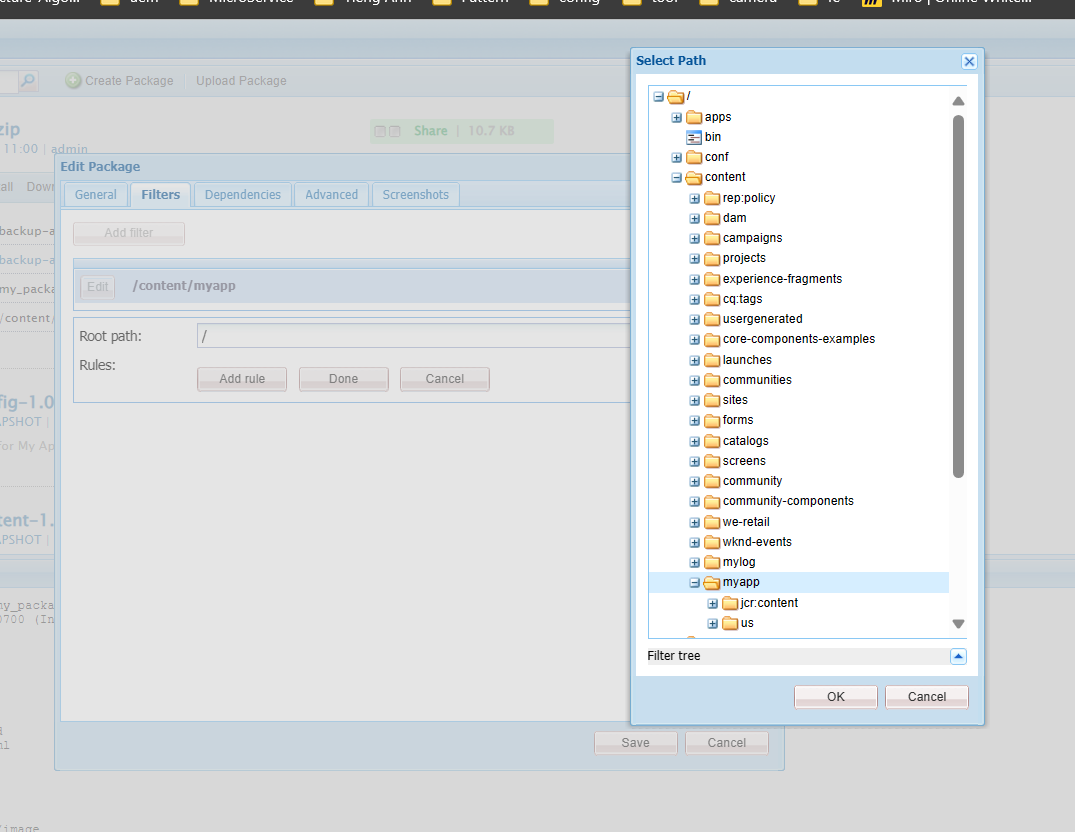
General -> Template -> choose myProject -> can create and edit avaitable template

# **Aem Concept**

## **Package Manager**

Tool -> Deloyment -> package manager -> [CRX Package Manager](http://localhost:4502/crx/packmgr/index.jsp)

Create backup:

Create package -> edit -> filter  -> choose my content backup -> build

* Can dowload and import into when take out content

## **Custom Logger**

Paste file in to Ui.config and deploy



## **System User**

****

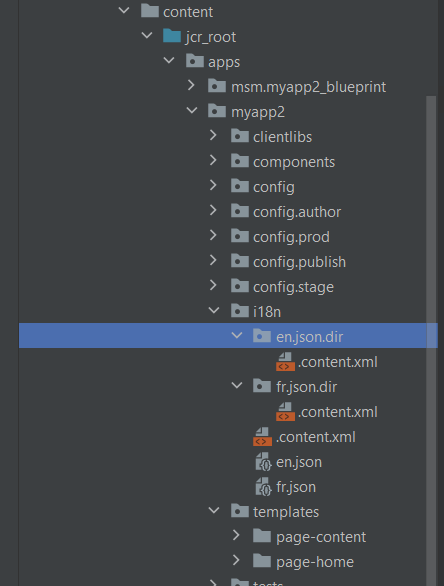
File create account or can into config to create account and permission as code

[Adobe AEM Commons (adobe-consulting-services.github.io)](https://adobe-consulting-services.github.io/acs-aem-commons/) (link dowload)

* Import into package
* After import open [AEM Security](http://localhost:4502/useradmin)
* Can create, control permission for user or create group permission for user

## **I18n**

Demo in HelloWorld Component (title of dialog) and (attribute)



->

en.json.dir->

<?xml version="1.0" encoding="UTF-8"?>

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

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

         jcr:language="en"

         jcr:mixinTypes="[mix:language]"

         jcr:primaryType="nt:file">

   <jcr:content

           jcr:primaryType="nt:resource"/>

</jcr:root>

En.json ->

{

 "&amp;copy; {0} MY APP2 Site. All rights reserved." : "&amp;copy; {0} MY APP2 Site. Tous droits réservés.",

 "promo.title" : "Title",

 "promo.btn.label": "English",

 "poster.language" : "English"

}

Fr.json ->

{

 "&amp;copy; {0} MY APP2 Site. All rights reserved." : "&amp;copy; {0} MY APP2 Site. Tous droits réservés.",

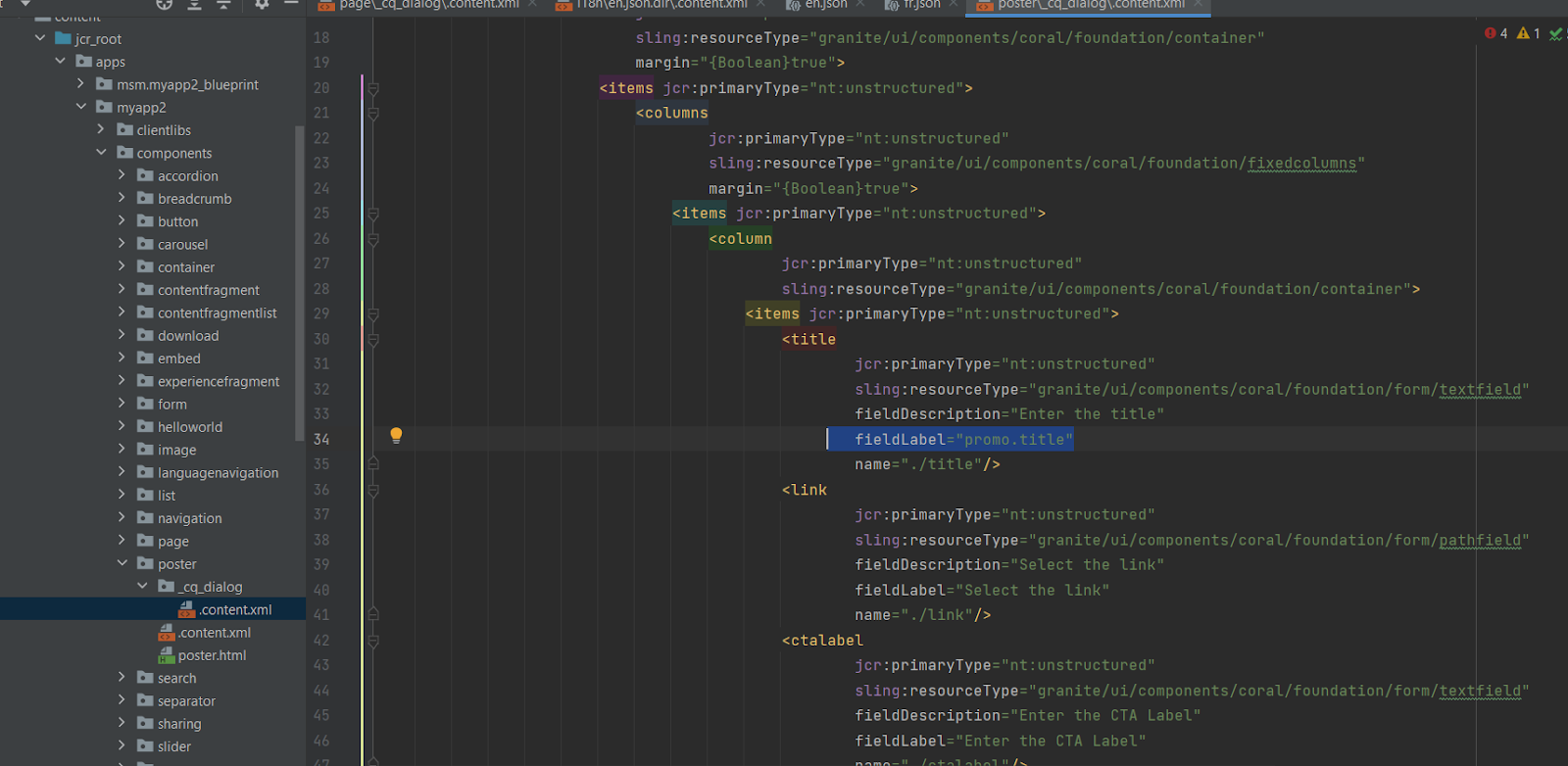
 "promo.title" : "Tire",

 "promo.btn.label": "réservés",

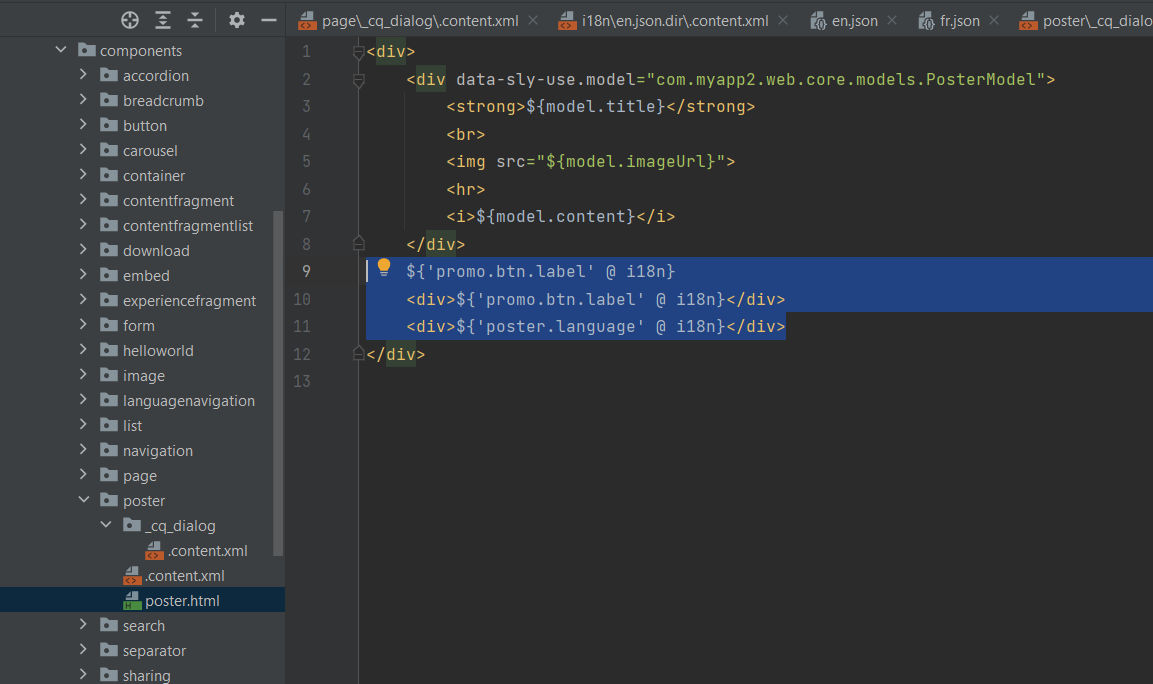
 "poster.language" : "Franch"

}

Cofig i18n on cofig field :



→ nó sẽ chạy theo language trên user admin

I18 trên page :

## **Custom Error**

* Create apps.sling.servlet.errorhandler
* Create 404.jsp
* Create 500.jsp
* META-INF/vault/filter.xml -> add **<filter root="/apps/sling"/>**
* Add Tag <error> in to dialog of page component **(\_cq\_dialog/.content.xml**)
* Into config and add **com.adobe.acs.commons.errorpagehandler.impl.ErrorPageHandlerImpl.xml**
* Open **AEM dashboard** create **errors page** and inside will create **400 page** and **500 page**
* open **properties of en page and** add /content/myapp/us/en/errors (link of errors page)

404.jsp

<%@page session="false"%><%%><%@include file="/apps/acs-commons/components/utilities/errorpagehandler/404.jsp" %>

500.jsp

<%@page session="false"%><%%><%@include file="/apps/acs-commons/components/utilities/errorpagehandler/default.jsp" %>

META-INF/vault/filter.xml

<filter root="/apps/sling"/>

Config com.adobe.acs.commons.errorpagehandler.impl.ErrorPageHandlerImpl.xml

<?xml version="1.0" encoding="UTF-8"?>  
<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" xmlns:nt="http://www.jcp.org/jcr/nt/1.0"  
 jcr:primaryType="sling:OsgiConfig"  
 enabled="{Boolean}true"  
 vanity.dispatch.enabled="{Boolean}true"  
 serve-authenticated-from-cache="{Boolean}true"  
 ttl="{Long}300"  
 error-page.system-path="/content/error"  
 error-page.fallback-name="500"  
/>

**Dialog of Page Component**

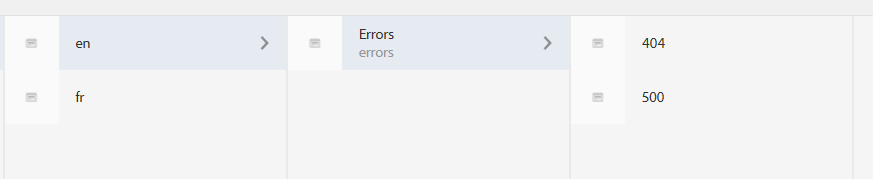
Error tag

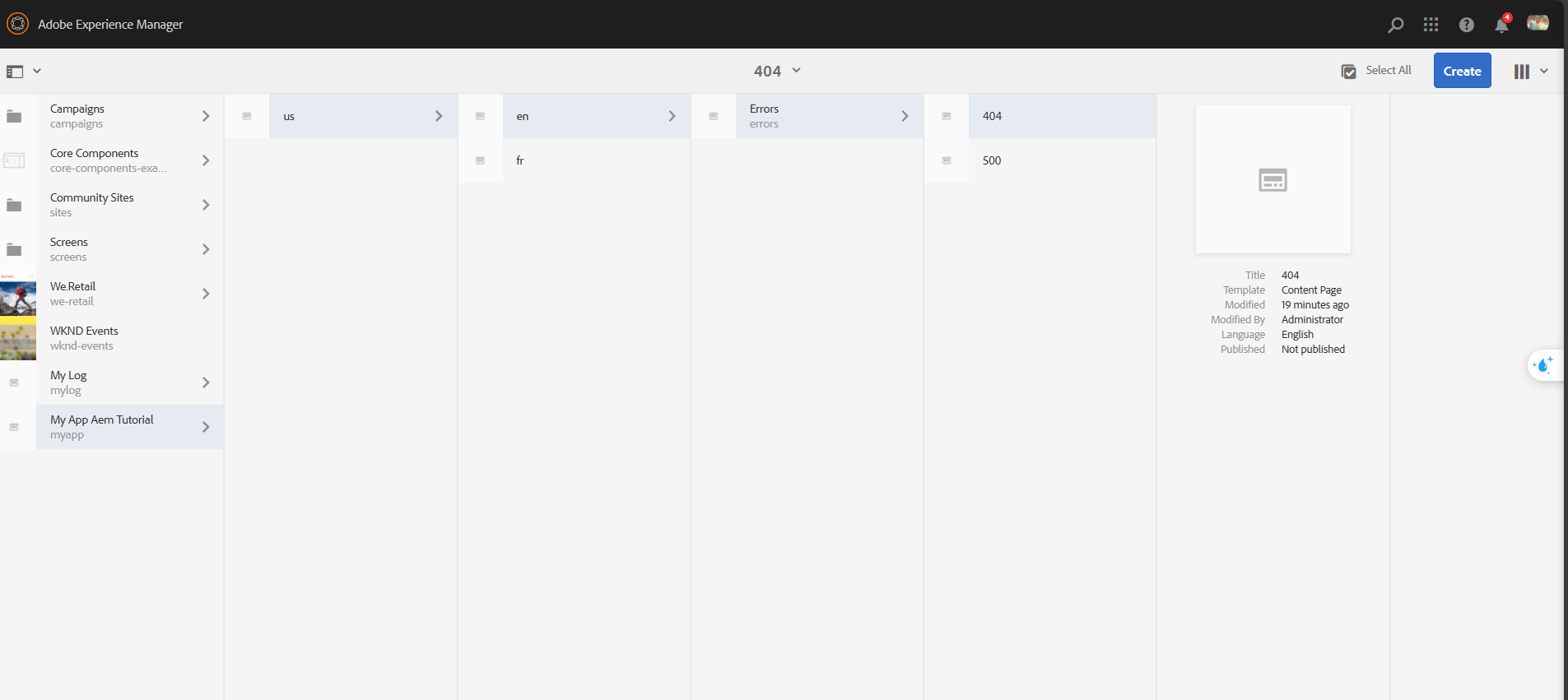
<error  
 jcr:primaryType="nt:unstructured"  
 jcr:title="Error Pages"  
 sling:resourceType="granite/ui/components/coral/foundation/fixedcolumns">  
 <items jcr:primaryType="nt:unstructured">  
 <column  
 jcr:primaryType="nt:unstructured"  
 sling:resourceType="granite/ui/components/coral/foundation/container">  
 <items jcr:primaryType="nt:unstructured">  
 <errorpages  
 jcr:primaryType="nt:unstructured"  
 sling:resourceType="granite/ui/components/coral/foundation/form/pathfield"  
 fieldDescription="Error pages for this content tree"  
 fieldLabel="Error Pages"  
 name="./errorPages"/>  
 </items>  
 </column>  
 </items>  
</error>

**Content file: \_cq\_dialog/.content.xml**

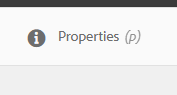
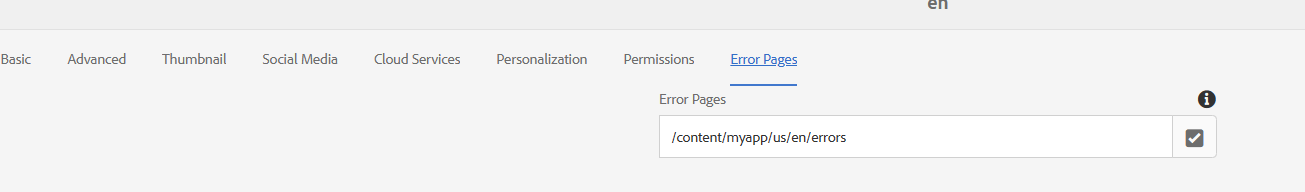


Create 404 and 500 page:





Add Error page into en page

 -> 

**Save and Test**

## **Workflow**

**DashBoard note**:

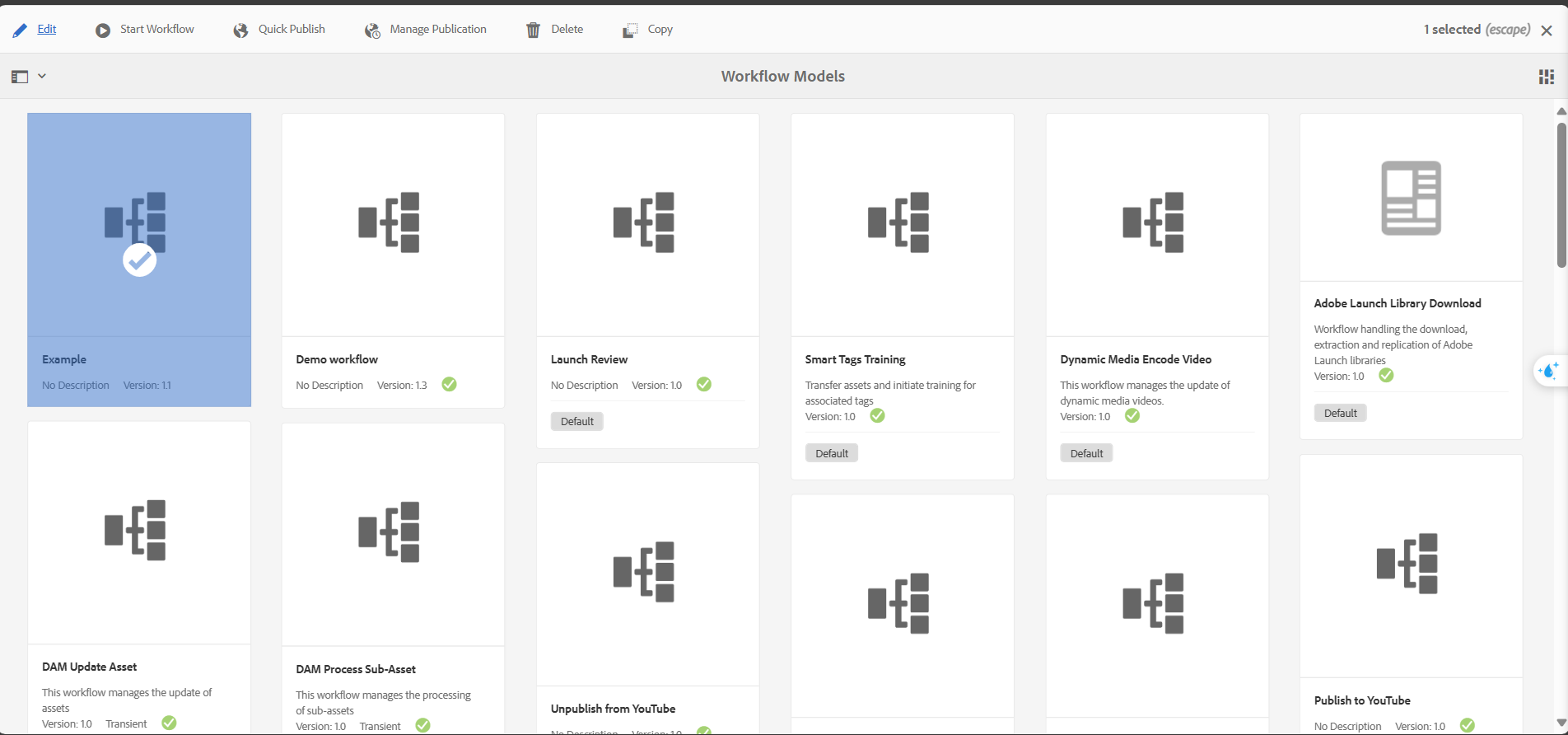
* Workflow Model (define available workflow)
* Instance (Intant is basically a working instance of workflow, you can see workflow is runing)
* Launcher ( are basicaly Cofiguration that can trigger a workflow, you can see information config of workflow)
* Archive ( Archive Store workflow is Complete)
* Failures ( workflows is not complete)

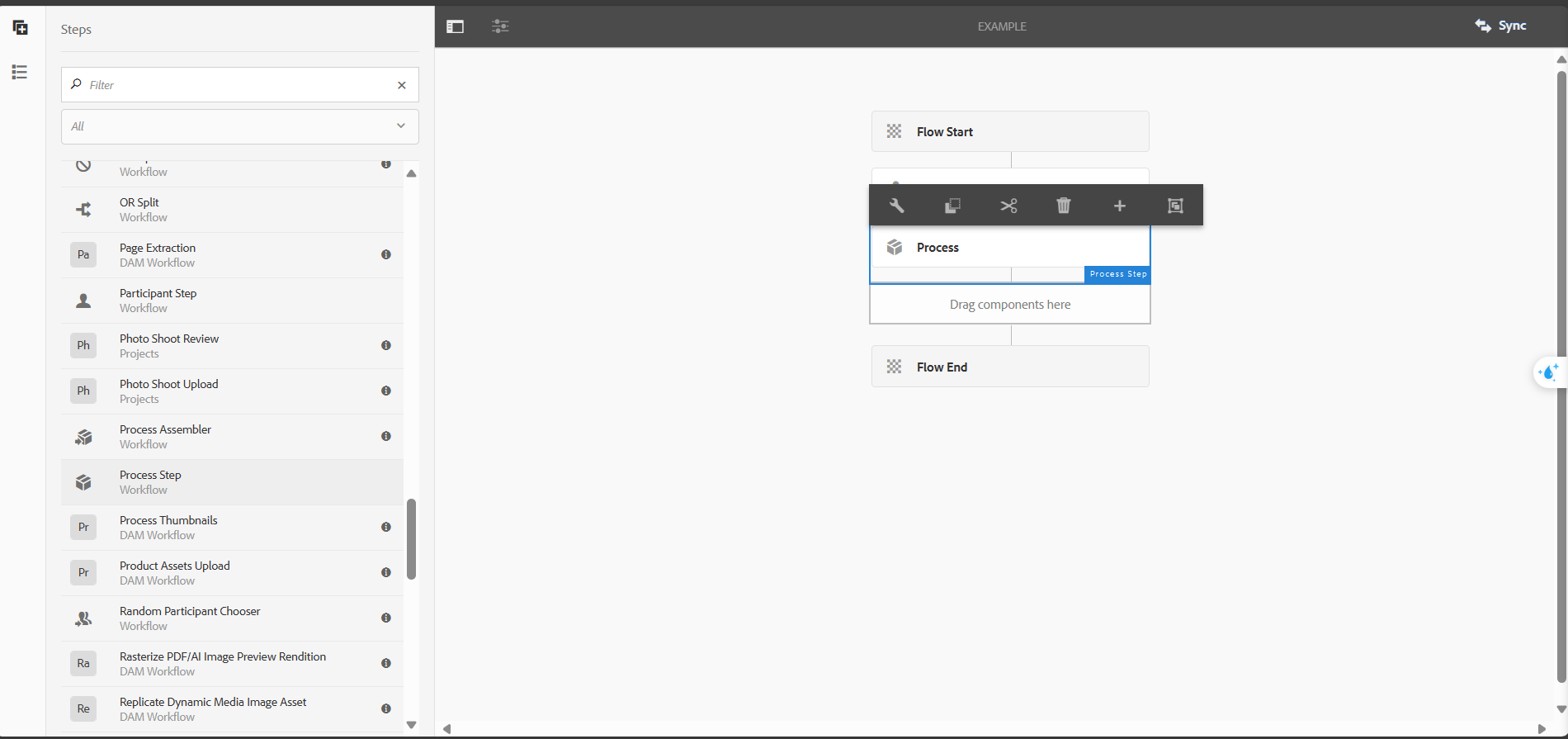
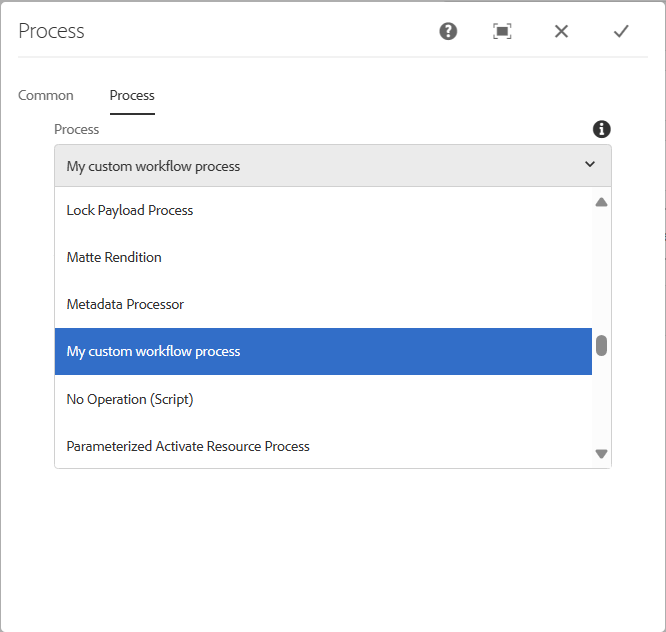
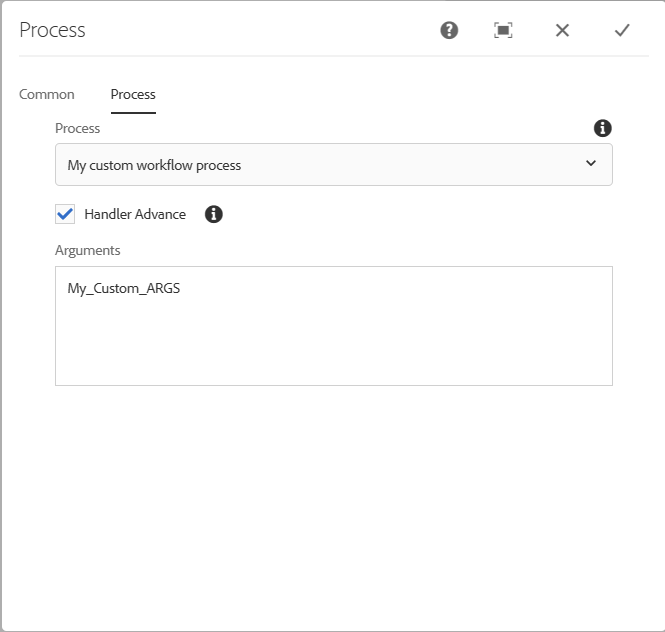
**Custom workflow java class:**

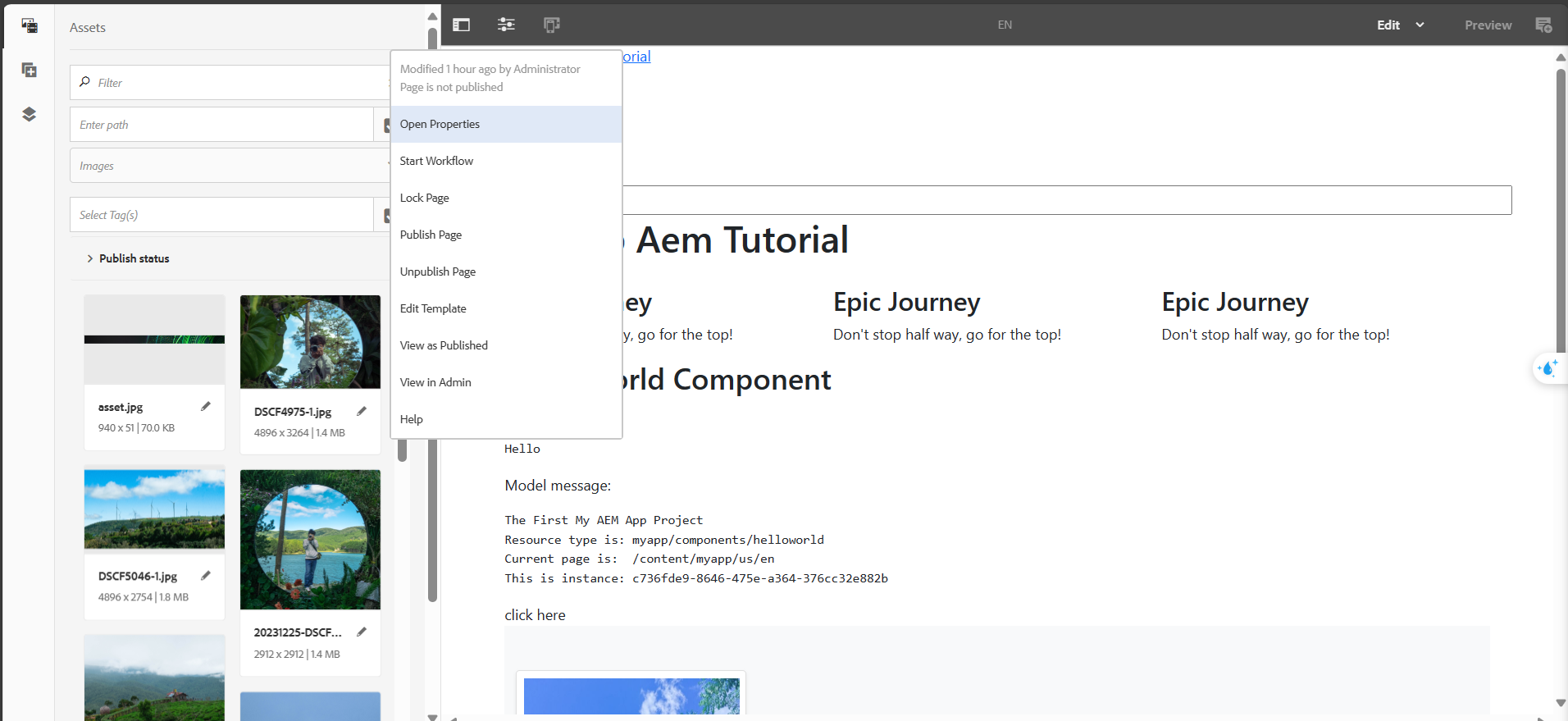
****

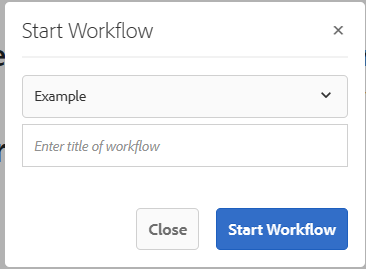
**Create workflow :**

**Models -> choose model -> enter edit button -> step process -> process -> choose My custom process**

****

**** **** 

** start workflow**

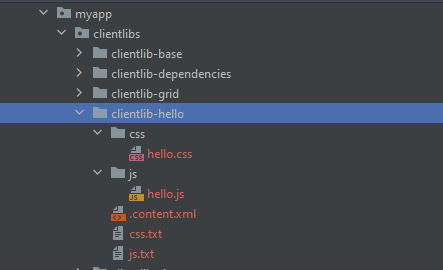
** choose workflow model is example (just create)**

**And finally start**

* **When open page workflow will run**

## **Client Lib**

Create by structure:



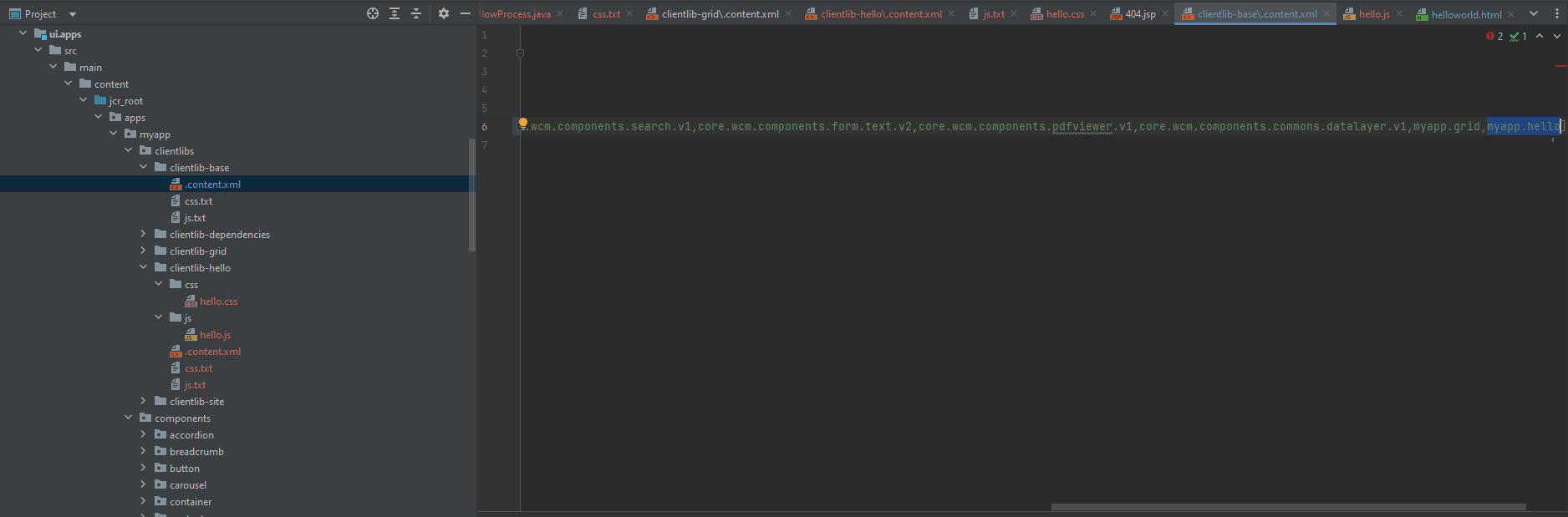
****

**method 1st:**

<sly data-sly-use.clientLib="/libs/granite/sightly/templates/clientlib.html">  
 <sly data-sly-call="${clientlib.all @ categories='myapp.hello'}"/>  
</sly>

**Add to component with tag**

**method 2st: add to base client lib**

****

**Note: But not successs**

## **Event**

* Follow code in pakage
  + package com.myapp.core.listeners
  + use JCREventHandler
  + use OSGiEventHandler

## **HTL**

* note for hello world

|  |  |
| --- | --- |
| data-sly-text="${properties.text}" | Use for properties or model |
| data-sly-test="${properties.text}" | Use for properties or model |
| data-sly-repeat="${currentPage.listChildren}" | For loop |
| data-sly-list.item="${currentPage.listChildren}" | For loop |
| data-sly-include="helloworld1.html" | Embed a html |
| data-sly-resource="${'container' @resourceType='myapp/components/container'}" | Embed a ResourComponent |

****